Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

«Затверджую»
в.о. завідуючого кафедри
комп'ютерних систем та робототехніки
_____ к. ф.-м. н., доцент Максим Хруслов
«___» грудня 2024 р.

Пояснювальна записка

до кваліфікаційної роботи
магістра

на тему: «**МОДЕЛЬ КЛАСИФІКАЦІЇ СТАНУ КОМП'ЮТЕРНИХ МЕРЕЖ**»

Спеціальність 174 – Автоматизація, комп'ютерно-інтегровані технології та робототехніка
Галузь знань 17 – Електроніка, автоматизація та електронні комунікації.
Освітня програма «Комп'ютеризовані системи управління та автоматика»

Захищено на засіданні

Екзаменаційної комісії № 44

протокол № __ від __.12.2024 р.

Оцінка _____ / _____

Голова Екзаменаційної комісії

_____ **СКОБ Ю. О.**

Виконав:
Студент групи КУ– 61
РОМАНОВ Роман Романович

Керівник: в.о. завідуючого кафедри комп'ютерних систем та робототехніки, к.ф.-м.н, доцент, ХРУСЛОВ Максим Михайлович

Рецензент: доцент кафедри математичного моделювання та аналізу даних, к.ф.-м.н. СПОРОВ Олександр Євгенович

Харків – 2024

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи магістра складається зі вступу, трьох розділів, висновків, списку використаних джерел і двох додатків. Загальний обсяг роботи складає 84 сторінки, із яких 65 сторінок основної частини з 29 рисунками, 2 таблицями, списку використаних джерел із 30 найменувань та трьома додатками.

Метою кваліфікаційної роботи є підвищення точності класифікації стану комп'ютерних мереж за допомогою алгоритмів машинного навчання. Для цього було досліджено сучасні підходи до класифікації мережевих даних, а також методи машинного навчання, які дозволяють ефективно аналізувати великі обсяги інформації з мережевих сенсорів та пристроїв.

Об'єкт дослідження – процес класифікації стану комп'ютерних мереж.

Предмет дослідження – розробка та дослідження моделі класифікації стану комп'ютерних мереж.

Актуальність дослідження: комп'ютерні мережі є основою сучасної інформаційної інфраструктури, надаючи доступ до даних, ресурсів і послуг для великої кількості користувачів і пристроїв. У міру збільшення кількості підключених пристроїв і розширення можливостей інтернету речей (IoT), роль комп'ютерних мереж в економіці, державному управлінні та повсякденному житті зростає. Проте складність управління та моніторингу мереж значно зростає, що робить надзвичайно актуальною проблему забезпечення їх надійності, безпеки та стабільності.

Для управління такими викликами існує потреба в ефективних методах моніторингу, діагностики та класифікації стану мережі, що дозволяють своєчасно ідентифікувати аномалії та потенційні загрози. Одним із підходів до розв'язання цієї проблеми є використання автоматизованих систем на основі моделей класифікації, що базуються на методах машинного навчання та обробки великих даних. Такі моделі можуть допомогти у визначенні стану мережі в режимі реального часу, а також у прогнозуванні ймовірності

виникнення проблем, що дозволяє запобігти потенційним інцидентам до їхнього виникнення.

Кваліфікаційна магістерська робота присвячена розробці моделі класифікації стану комп’ютерних мереж, яка здатна точно оцінювати поточний стан мережі та своєчасно виявляти можливі загрози й аномалії.

За результатами дослідження: розроблена модель класифікації стану комп’ютерних мереж, здатна ідентифікувати та класифікувати мережеві стани з високою точністю. Запропонована модель пройшла етапи навчання та тестування на реальних даних мережевого трафіку, що дало змогу оцінити її продуктивність, точність та адаптивність до змінних умов у мережі.

За результатами експериментів модель продемонструвала високу точність у виявленні аномалій, таких як перевантаження, спроби несанкціонованого доступу та збої у зв’язку. Зокрема, середня точність класифікації сягнула понад 80%, що підтверджує її ефективність у завданнях моніторингу мереж. Також вдалося досягти значного зниження кількості хибно-позитивних спрацювань, що є важливим фактором для ефективного управління мережевими інцидентами та забезпечення стабільності системи.

Одержані результати можуть бути використані: розроблена модель класифікації може використовуватися як універсальний інструмент для забезпечення безпеки та стабільності комп'ютерних мереж у різних сферах — від комерційних до державних організацій, що підвищує її цінність для практичного впровадження.

Ключові слова: КОМП’ЮТЕРНІ МЕРЕЖІ, МОДЕЛЬ OSI, МАШИННЕ НАВЧАННЯ, АЛГОРИТМИ КЛАСИФІКАЦІЇ, КОМП’ЮТЕРНА МОДЕЛЬ, БІБЛІОТЕКИ PYTHON.

ABSTRACT

The explanatory note to the master's thesis consists of an introduction, three sections, conclusions, a list of used sources and two appendices. The total volume of the work is 84 pages, of which 65 pages are the main part with 29 figures, 2 table, a list of used sources with 30 items and three appendices.

The purpose of the qualification work is to increase the accuracy of classification of the state of computer networks using machine learning algorithms. For this purpose, modern approaches to the classification of network data, as well as machine learning methods, which allow efficient analysis of large volumes of information from network sensors and devices, were investigated.

The object of research is the process of classifying the state of computer networks.

The subject of the research is the development and research of a model for classifying the state of computer networks.

Relevance of the study: computer networks are the basis of modern information infrastructure, providing access to data, resources and services for a large number of users and devices. As the number of connected devices increases and the Internet of Things (IoT) expands, the role of computer networks in the economy, government, and everyday life is growing. However, the complexity of managing and monitoring networks is growing significantly, which makes the problem of ensuring their reliability, security and stability extremely urgent.

To manage such challenges, there is a need for effective methods of monitoring, diagnosing and classifying the state of the network, allowing for timely identification of anomalies and potential threats. One of the approaches to solving this problem is the use of automated systems based on classification models based on methods of machine learning and big data processing. Such models can help determine the state of the network in real time, as well as predict the likelihood of problems, which allows you to prevent potential incidents before they occur.

The qualifying master's thesis is devoted to the development of a model for classifying the state of computer networks, which is capable of accurately assessing the current state of the network and timely detecting possible threats and anomalies.

According to the research results: a computer network state classification model was developed, capable of identifying and classifying network states with high accuracy. The proposed model passed the stages of training and testing on real network traffic data, which made it possible to evaluate its performance, accuracy and adaptability to changing conditions in the network.

According to the results of the experiments, the model demonstrated high accuracy in detecting anomalies such as congestion, unauthorized access attempts, and communication failures. In particular, the average classification accuracy reached more than 80%, which confirms its effectiveness in network monitoring tasks. It was also possible to significantly reduce the number of false positives, which is an important factor for effective management of network incidents and ensuring system stability.

The obtained results can be used: the developed classification model can be used as a universal tool for ensuring the security and stability of computer networks in various areas — from commercial to government organizations, which increases its value for practical implementation.

Keywords: COMPUTER NETWORKS, OSI MODEL, MACHINE LEARNING, CLASSIFICATION ALGORITHMS, COMPUTER MODEL, PYTHON LIBRARIES.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ СКОРОЧЕНЬ І ТЕРМІНІВ

ATM – Asynchronous Transfer Mode;

IDE – Integrated Development Environment;

IoT – Internet of Things;

KNN – K-Nearest Neighbors;

LAN – Local Area Network;

MAN – Metropolitan Area Network;

MPLS – Multiprotocol Label Switching;

OSI – Open Systems Interconnection;

PAN – Personal Area Network;

SVM – Support Vector Machine;

TCP/IP – Transmission Control Protocol / Internet Protocol;

WAN – Wide Area Network;

Wi-Fi – Wireless Fidelity.

ВСТУП

Комп’ютерні мережі є основою сучасної інформаційної інфраструктури, надаючи доступ до даних, ресурсів і послуг для великої кількості користувачів і пристроїв. У міру збільшення кількості підключених пристроїв і розширення можливостей інтернету речей (IoT), роль комп’ютерних мереж в економіці, державному управлінні та повсякденному житті зростає. Проте складність управління та моніторингу мереж значно зростає, що робить надзвичайно актуальною проблему забезпечення їх надійності, безпеки та стабільності.

Інциденти, пов’язані з мережею, такі як збої у зв’язку, перевантаження каналів, атаки на безпеку та інші аномалії, можуть мати серйозні наслідки. Наприклад, для бізнесу це означає зниження продуктивності, втрату клієнтів і навіть загрозу фінансовій стабільності. У державному секторі, де мережі забезпечують критично важливі послуги, такі проблеми можуть призвести до зриву операцій і виникнення національних загроз. Тому забезпечення безперервного та безпечного функціонування мереж є пріоритетним завданням як для мережевих адміністраторів, так і для дослідників у галузі інформаційних технологій.

Для управління такими викликами існує потреба в ефективних методах моніторингу, діагностики та класифікації стану мережі, що дозволяють своєчасно ідентифікувати аномалії та потенційні загрози. Одним із підходів до розв'язання цієї проблеми є використання автоматизованих систем на основі моделей класифікації, що базуються на методах машинного навчання та обробки великих даних. Такі моделі можуть допомогти у визначенні стану мережі в режимі реального часу, а також у прогнозуванні ймовірності виникнення проблем, що дозволяє запобігти потенційним інцидентам до їхнього виникнення.

Важливим аспектом у розробці моделей класифікації є вибір методів машинного навчання, які здатні працювати з великими обсягами мережевих

даних і можуть ефективно ідентифікувати закономірності та аномалії. Зокрема, широко застосовуються методи кластеризації, дерев рішень, нейронних мереж, а також глибокого навчання для аналізу складних патернів у даних про мережевий трафік. На сьогоднішній день дослідження в галузі мережевої класифікації фокусуються на підвищенні точності прогнозування, швидкості обробки, масштабованості та можливості адаптації до змін у мережевих умовах.

Кваліфікаційна магістерська робота присвячена розробці моделі класифікації стану комп'ютерних мереж, яка здатна точно оцінювати поточний стан мережі та своєчасно виявляти можливі загрози й аномалії.

Метою дослідження є підвищення точності класифікації стану комп'ютерних мереж за допомогою алгоритмів машинного навчання. Для цього було досліджено сучасні підходи до класифікації мережевих даних, а також методи машинного навчання, які дозволяють ефективно аналізувати великі обсяги інформації з мережевих сенсорів та пристроїв.

Для досягнення поставленої мети необхідно розглянути такі **задачі**:

1) аналітичний огляд сучасних технологій комп'ютерних мереж;
2) дослідження методів класифікації;
3) дослідження основних бібліотек Python для машинного навчання;
4) розробка концептуальної моделі класифікації стану комп'ютерних мереж;
5) програмна реалізація комп'ютерної моделі класифікації стану комп'ютерних мереж;
6) проведення тестування розробленої комп'ютерної моделі класифікації стану комп'ютерних мереж;
7) аналіз отриманих результатів.

Об'єкт дослідження - процес класифікації стану комп'ютерних мереж.

Предмет дослідження - розробка та дослідження моделі класифікації стану комп'ютерних мереж.

Методи дослідження: полягають у використанні принципів та методів

системного аналізу, методів машинного навчання, а також застосування імітаційного та математичного моделювання, теорії математичної статистики, теорії множин, теорії ймовірностей, теорії графів, лінійну алгебру, методи математичної оптимізації, диференціального аналізу, теорії штучних нейронних мереж.

За результатами дослідження: розроблена модель класифікації стану комп’ютерних мереж, здатна ідентифікувати та класифікувати мережеві стани з високою точністю. Запропонована модель пройшла етапи навчання та тестування на реальних даних мережевого трафіку, що дало змогу оцінити її продуктивність, точність та адаптивність до змінних умов у мережі.

За результатами експериментів модель продемонструвала високу точність у виявленні аномалій, таких як перевантаження, спроби несанкціонованого доступу та збої у зв’язку. Зокрема, середня точність класифікації сягнула понад 80%, що підтверджує її ефективність у завданнях моніторингу мереж. Також вдалося досягти значного зниження кількості хибно-позитивних спрацювань, що є важливим фактором для ефективного управління мережевими інцидентами та забезпечення стабільності системи.

Ключовим аспектом у роботі моделі стало використання алгоритмів машинного навчання, зокрема методів глибокого навчання, які дозволили ефективно аналізувати великі обсяги даних про мережевий трафік. Крім того, було виявлено, що точність і швидкість класифікації суттєво залежить від вибору ознак, які використовуються для навчання моделі. Завдяки проведеному відбору ознак вдалося зменшити обсяг оброблюваних даних і підвищити швидкість роботи моделі без втрати її точності.

У результаті дослідження також було розроблено рекомендації щодо впровадження моделі в реальних мережах, зокрема методи налаштування і тестування системи на специфічних мережевих параметрах. Отримані результати підтверджують, що запропонована модель класифікації може стати ефективним інструментом для забезпечення надійності комп'ютерних мереж і значно полегшити роботу мережевих адміністраторів, надаючи їм

можливість оперативного виявлення та усунення мережевих аномалій і загроз.

Практична новизна роботи полягає в розробці та впровадженні моделі класифікації стану комп'ютерних мереж, яка дозволяє з високою точністю ідентифікувати мережеві аномалії та стан системи в режимі реального часу. Запропонований підхід базується на використанні сучасних методів машинного навчання та алгоритмів глибокого навчання, що забезпечують ефективну обробку великих обсягів мережевих даних і точне виявлення потенційних загроз.

На відміну від традиційних методів моніторингу, модель дозволяє не тільки виявляти аномалії, а й здійснювати прогнозування мережевих станів, що є важливим кроком до створення проактивних систем управління мережею. Завдяки цьому мережеві адміністратори отримують можливість запобігати інцидентам до їх виникнення, що значно підвищує стабільність і надійність мережі.

Таким чином, розроблена модель класифікації може використовуватися як універсальний інструмент для забезпечення безпеки та стабільності комп'ютерних мереж у різних сферах — від комерційних до державних організацій, що підвищує її цінність для практичного впровадження.

Апробація результатів дослідження. Основні теоретичні результати роботи були представлені на І Всеукраїнській науково-практичній студентській конференції «ІТ-простір сьогодення: тенденції, інновації та перспективи розвитку» з темою «Модель класифікації стану комп’ютерних мереж».

РОЗДІЛ 1
АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ ТЕХНОЛОГІЙ КОМП'ЮТЕРНИХ МЕРЕЖ

1.1 Аналіз сучасного стану розвитку комп'ютерних мереж

Сучасний розвиток інформаційних технологій супроводжується зростанням значущості телекомунікаційних систем різного призначення та комп'ютерних мереж. Це обумовлено потребою у швидшій передачі інформації, зокрема управлінської, для якої важливі оперативність і своєчасна доставка користувачам. Засоби електронного обміну документами, такі як електронна пошта та браузери, набувають все більшої ваги, оскільки завдяки їм значно підвищується ефективність роботи фахівців на різних рівнях управління сучасними підприємствами та організаціями.

Сучасні технології комп'ютерних мереж, зокрема локальні та глобальні мережі, відіграють важливу роль у виконанні цих завдань. Це зумовлено потребою в доступі до корпоративної інформації, яка зберігається в корпоративних базах даних, розташованих як у різних підрозділах підприємства, так і поза його межами. Відтак, сучасні технології обробки документів різного призначення повинні спиратися на засоби телекомунікаційного зв'язку та стандарти комп'ютерних мереж, які виступають транспортними системами для передачі даних.

З кожним роком посилюється тенденція до інтеграції комп'ютерних і телекомунікаційних мереж різних типів. Ставиться мета створити універсальну мультисервісну мережу, яка могла б надавати послуги як для комп'ютерних, так і для телекомунікаційних систем.

До телекомунікаційних мереж належать телефонні, радіо та телевізійні мережі. Їх об'єднує з комп'ютерними мережами те, що основним ресурсом, який надається користувачам, є інформація. Проте, ці мережі зазвичай передають інформацію в різних формах. Комп'ютерні мережі спочатку були

розроблені для передачі алфавітно-цифрової інформації, відомої як дані, тому їх часто називають мережами передачі даних. У той час телефонні та радіомережі створювалися для передачі лише голосу, тоді як телевізійні мережі забезпечують передачу як звуку, так і зображення [1-3].

У загальному випадку телекомунікаційна мережа складається з таких компонентів (рис. 1.1):

- мережа доступу – призначена для концентрації інформаційних потоків, що надходять через численні канали зв'язку від користувацького обладнання, у порівняно обмеженій кількості вузлів магістральної мережі;

- магістраль – об'єднує різні мережі доступу, забезпечуючи транзит трафіку між ними через високошвидкісні канали;

- інформаційні центри або центри керування сервісами – це внутрішні інформаційні ресурси мережі, на базі яких здійснюється обслуговування користувачів.

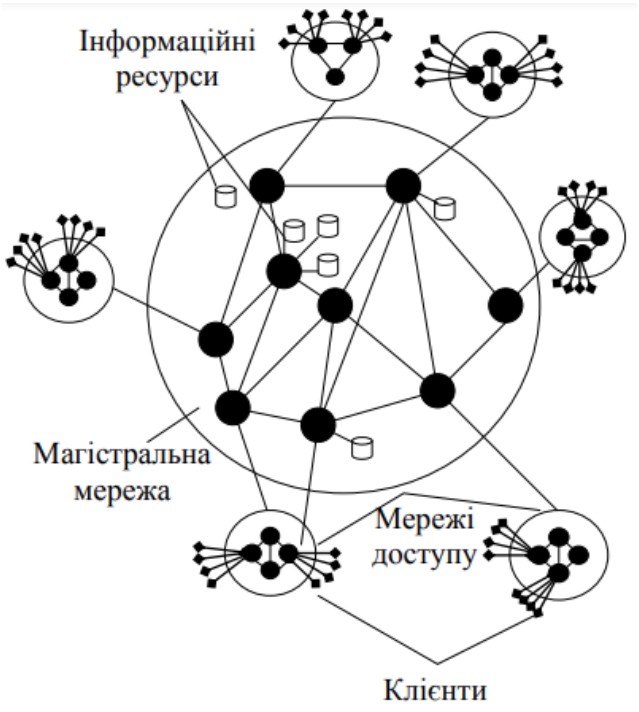

Рисунок 1.1 − Структура телекомунікаційної мережі [5]

Мережа доступу та магістральна мережа формуються на основі

комутаторів. Кожен комутатор обладнаний певною кількістю портів, які з'єднуються з портами інших комутаторів через канали зв'язку.

Термін «інформаційна мережа» означає розгляд телекомунікаційної мережі разом із об'єктами, що взаємодіють через неї. Таким чином, інформаційна мережа являє собою телекомунікаційну мережу, яка функціонує під «навантаженням» [4].

Загалом, інформаційною мережею будемо називати сукупність кінцевих систем, які розташовані на різних територіях, разом із телекомунікаційною мережею, що їх об'єднує. Ця мережа забезпечує доступ прикладних процесів з будь-якої з цих систем до всіх ресурсів інформаційної мережі та їх спільне використання.

Сьогодні існує безліч методів класифікації комп’ютерних мереж, які ґрунтуються на різних критеріях. Розглянемо один з варіантів такої класифікації.

Залежно від територіальної поширеності мережі поділяються на локальні, регіональні та глобальні. У класифікації мереж використовуються два основні терміни: LAN і WAN [4, 5].

Локальні мережі охоплюють невеликі території площею не більше 1000-2000 м². Як правило, ці мережі є комунікаційними системами, що належать одній організації. Завдяки коротким відстаням у локальних мережах є можливість використовувати досить дорогі високоякісні канали зв'язку, які дозволяють досягати високих швидкостей передачі даних (до 100 Мбіт/с і вище) за допомогою простих методів. Таким чином, послуги, що надаються локальними мережами, відрізняються великою різноманітністю і зазвичай реалізуються в режимі онлайн.

Глобальні WAN мережі охоплюють територію однієї країни або групи країн. Глобальні комунікаційні мережі покривають великі географічні області, що включають як локальні мережі, так і інші телекомунікаційні мережі та пристрої. Прикладом WAN є мережа з комутацією пакетів, яка дозволяє різним комп’ютерним мережам спілкуватися між собою.

Регіональні (міські) мережі розташовані на території міста або області і призначені для їх обслуговування. Локальні комп'ютерні мережі найкраще підходять для розподілу ресурсів на коротких відстанях, тоді як глобальні комп'ютерні мережі забезпечують зв'язок на великих відстанях, але з обмеженою швидкістю та невеликим набором послуг. Міські комп'ютерні мережі займають проміжну позицію. Вони використовують цифрові магістральні лінії зв'язку (часто оптоволоконні) зі швидкостями від 45 Мбіт/с і призначені для з'єднання локальних мереж у межах міста, а також для інтеграції їх з глобальними мережами.

Комп'ютерні мережі класифікуються за швидкістю передачі інформації на [5]:

- низькошвидкісні (до 10 Мбіт/с);
- середньошвидкісні (до 100 Мбіт/с);
- високошвидкісні (понад 100 Мбіт/с).

Для вимірювання швидкості передачі даних у мережі часто використовується термін «бод». Бод є одиницею виміру швидкості передачі сигналу, яка визначається як кількість дискретних переходів або подій за секунду.

Мережі класифікуються за типом середовища передачі на:

- провідні – коаксіальні, на скрученій парі, оптоволоконні;
- безпровідні – які передають інформацію через радіоканали або в інфрачервоному діапазоні.

Комп'ютерні мережі класифікуються за топологією на повнозв'язні та неповнозв'язні. Неповнозв'язні мережі, у свою чергу, можуть бути комірковими типів «загальна шина», «зірка», «кільце», а також змішаними (рис. 1.2). Топологія комп'ютерної мережі визначає метод, за допомогою якого комп'ютери з'єднуються один з одним (спосіб організації фізичних зв'язків). Іншими словами, топологія є конфігурацією графа, де вершинами є комп'ютери мережі (а іноді й інше обладнання, наприклад, концентратори), а ребра відображають фізичні з'єднання між ними [5].

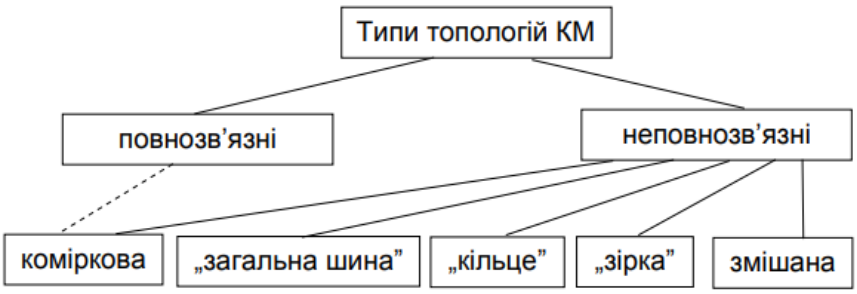

Рисунок 1.2 − Типи топологій комп'ютерних мереж [5]

Повнозв'язна топологія комп'ютерних мереж (рис. 1.3 а) характеризується тим, що кожен комп'ютер у мережі має прямі з'єднання з усіма іншими комп'ютерами. Це означає, що всі вузли можуть спілкуватися один з одним без проміжних пристроїв. ЇЇ переваги: швидкість передачі даних (висока швидкість обміну інформацією між вузлами, оскільки немає необхідності в передачі через інші комп'ютери) та надійність (якщо один з комп'ютерів виходить з ладу, це не вплине на зв'язок інших вузлів, оскільки з'єднання є незалежними). До недоліків можна віднести складність, оскільки повнозв'язна топологія вимагає великої кількості кабелів і комунікаційних портів, що ускладнює прокладку мережі та вартість (висока вартість встановлення через необхідність забезпечення зв'язків між усіма вузлами). Зазвичай повнозв'язна топологія використовується в обмежених середовищах, таких як малі офіси або лабораторії, де є невелика кількість комп'ютерів.

Коміркова топологія комп'ютерних мереж (рис. 1.3 б) є одним з варіантів організації мережевих з'єднань, що характеризується використанням центрального вузла, до якого підключаються всі інші пристрої. Цей центральний вузол, часто названий «коміркою» або «пунктом доступу», виконує функцію управління та маршрутизації даних між пристроями мережі.

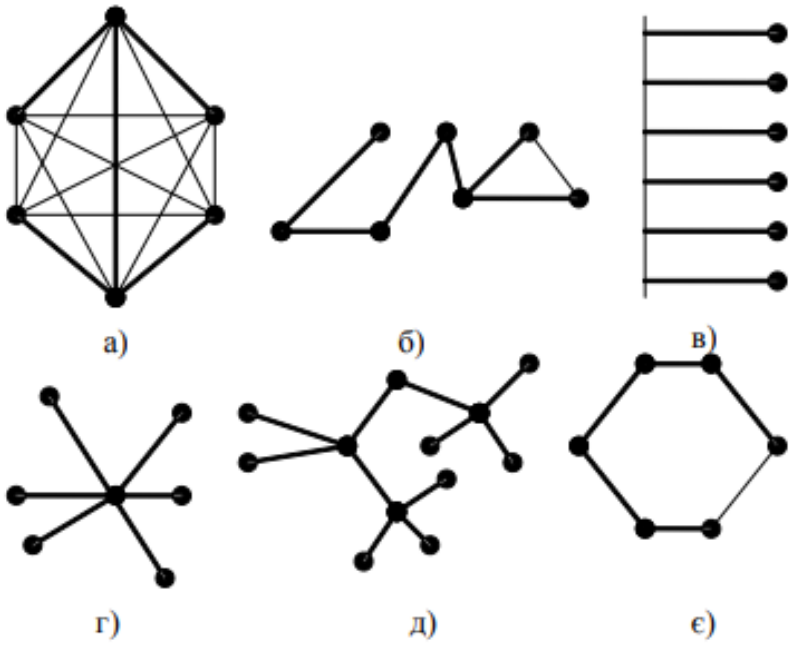

Рисунок 1.3 − Типові топології комп’ютерної мережі: а) повнозв’язна топологія; б) коміркова топологія; в) топологія «загальна шина»; г) топологія «зірка»; д) ієрархічна (розширена) «зірка»; є) кільцева топологія [5]

Основні характеристики коміркової топології:

- централізоване управління: усі дані проходять через центральний вузол, що спрощує управління мережею та моніторинг її стану;
- масштабованість: додаючи нові пристрої, можна легко розширити мережу, підключаючи їх до центрального вузла;
- простота в обслуговуванні: випадки поломки або збої в роботі можуть бути швидше виявлені і усунені, оскільки вся інформація проходить через один пункт;
- гнучкість: ця топологія дозволяє змінювати конфігурацію мережі без значних зусиль.

До переваг коміркової топології комп'ютерних мереж можна віднести:

- легке налаштування та управління: завдяки централізованій структурі, налаштування мережі стає простішим;
- зниження витрат на кабелі: можливість використання меншої кількості кабелів для підключення;
- контроль доступу: централізований вузол дозволяє ефективно

контролювати доступ до мережі.

До недоліків коміркової топології комп'ютерних мереж можна віднести:

- єдина точка відмови: якщо центральний вузол виходить з ладу, це може призвести до зупинки роботи всієї мережі;
- високе навантаження на вузол: велика кількість підключених пристроїв може призвести до перевантаження центрального вузла, що вплине на продуктивність мережі.

Коміркова топологія часто використовується в локальних мережах (LAN), де є потреба в ефективному управлінні та контролі.

Топологія «загальна шина» (рис. 1.3 в) - це тип мережевої топології, де всі комп’ютери або інші пристрої підключені до одного загального кабелю, який називається шиною. Кожен вузол мережі передає дані через цей спільний кабель, а всі інші вузли отримують ці дані.

Переваги використання топології «загальна шина» такі:

- економічність: вимагає мінімальної кількості кабелів порівняно з іншими топологіями, що знижує витрати на інфраструктуру;
- простота встановлення: кабель легко прокладається, особливо в невеликих мережах, що робить її зручною для розгортання;
- широкомовність: можливість одночасної передачі даних на всі станції, що дозволяє швидко розповсюджувати інформацію.

Недоліки використання топології «загальна шина» такі:

- низька надійність: у випадку пошкодження кабелю вся мережа перестає працювати, оскільки всі пристрої залежні від одного каналу;
- обмежена пропускна здатність: в один момент часу лише один пристрій може передавати дані, що знижує загальну продуктивність мережі;
- проблеми з масштабуванням: у міру збільшення кількості підключених пристроїв ефективність мережі знижується, оскільки всі користувачі ділять один і той самий канал.

Загальна шина часто використовувалася в старих мережах, але з

розвитком технологій її витіснили більш ефективні топології, такі як зірка і кільце.

Топологія «зірка» — це тип мережевої топології, де всі пристрої (комп'ютери, принтери тощо) підключені до центрального вузла або концентратора (хабу або комутатора). Цей центральний вузол керує всіма з'єднаннями та передачею даних між підключеними пристроями.

Переваги використання топології «зірка»:

- надійність: якщо один з вузлів виходить з ладу, це не впливає на роботу інших пристроїв у мережі, оскільки вони не залежать один від одного;
- простота в діагностиці: у разі неполадок легко ідентифікувати несправний вузол або з'єднання, оскільки кожен пристрій підключений окремо до центрального вузла;
- гнучкість: легко додавати нові пристрої до мережі без серйозних змін в її структурі.

Недоліки топології «зірка» такі:

- залежність від центрального вузла: якщо центральний хаб або комутатор вийде з ладу, вся мережа перестане працювати, оскільки всі з'єднання проходять через нього;
- вартість: потребує більше кабелів, оскільки кожен пристрій підключений окремо до центрального вузла, що збільшує витрати на прокладку мережі;
- завантаженість центрального вузла: у великих мережах можливе перевантаження центрального вузла через велику кількість запитів на передачу даних.

Топологія «зірка» є популярною для сучасних локальних мереж (LAN) завдяки своїй надійності та зручності в адмініструванні.

Часом, доцільно створювати мережу, використовуючи декілька концентраторів, з’єднаних між собою ієрархічно у вигляді зірки (рис. 1.3 д). Наразі ієрархічна (розширена) зірка є найпоширенішою топологією зв'язку як у локальних, так і у глобальних комп'ютерних мережах.

Мережі з кільцевою топологією (рис. 1.3 є) - це тип мережевої топології, де всі вузли (комп'ютери або інші пристрої) з'єднані один з одним у замкнене кільце. Дані передаються в одному напрямку по кільцю, або в обох напрямках, проходячи через кожен вузол мережі, доки не досягнуть кінцевого пункту.

Переваги кільцевої топології:

- рівномірний розподіл навантаження: усі вузли мають однакові права на передачу даних, що зменшує ймовірність перевантаження мережі;
- простота розширення: додавання нових вузлів не потребує значних змін в існуючій мережі;
- чітка передача даних: дані передаються в заздалегідь визначеному порядку, що знижує можливість колізій при одночасній передачі.

Недоліки кільцевої топології:

- залежність від кожного вузла: якщо один вузол або з'єднання вийде з ладу, це може порушити роботу всієї мережі;
- затримка передачі: дані повинні проходити через кожен вузол мережі, що може збільшити затримку, особливо в великих мережах;
- складність в налаштуванні та діагностиці: виявлення і усунення проблем може бути складним завданням через залежність кожного вузла від інших.

Кільцева топологія рідко використовується в сучасних мережах через свої обмеження, але вона була популярною в минулому, особливо для мереж, які вимагали точної та послідовної передачі даних, як, наприклад, в мережах Token Ring.

У той час як невеликі комп'ютерні мережі зазвичай мають типову топологію «зірка», «кільце» або «загальна шина», великі мережі часто характеризуються випадковими зв’язками між комп'ютерами. У таких мережах можна виділити окремі фрагменти (підмережі) з типовою топологією, що призводить до виникнення мереж зі змішаною топологією (рис. 1.4).

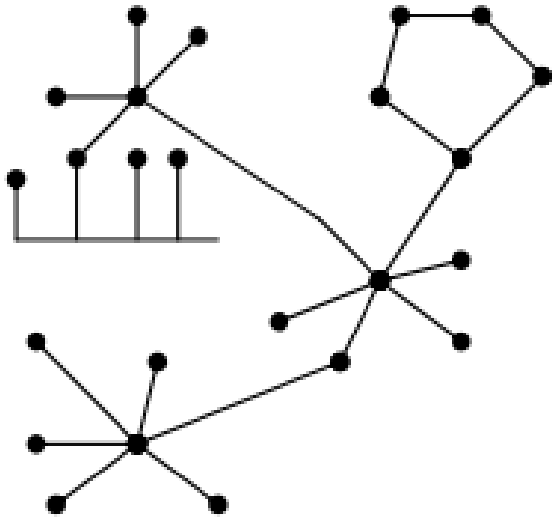

Рисунок 1.4 − Змішана топологія [5]

Згідно з організацією взаємодії між комп'ютерами, мережі можна розділити на одноранговi (Peer-to-Peer) та мережі з виділеним сервером (Dedicated Server Network). У комп'ютерах одноранговоï мережі всі пристроï мають однакові права. Кожен користувач може отримати доступ до даних, які зберігаються на будь-якому іншому комп'ютері в мережі.

В ієрархічній мережі під час налаштування системи заздалегідь обираються один або декілька комп'ютерів (серверів), які відповідають за управління обміном даних у мережі та розподіл ресурсів. Будь-який комп'ютер, який отримує доступ до сервісів сервера, називається клієнтом мережі або робочою станцією. Сам сервер може виступати клієнтом лише для сервера вищого рівня ієрархії. Через це ієрархічні мережі іноді називають мережами з виділеними серверами.

Існують дві основні технології використання сервера: технологія файл-сервера та архітектура клієнт-сервер. У моделі файл-сервера основна частина програм і даних зберігається на файловому сервері, який на запит користувача надає необхідну інформацію та програми. Обробка даних виконується на робочій станції.

У системах з архітектурою клієнт-сервер обмін даними відбувається

між клієнтським додатком і серверним додатком. Тут дані зберігаються і обробляються на потужному сервері, який також відповідає за контроль доступу до ресурсів і інформації. Робоча станція отримує лише результати своїх запитів. Цю технологію зазвичай використовують розробники програм, які займаються обробкою інформації.

1.2 Огляд моделі взаємодії відкритих систем OSI

Модель взаємодії відкритих систем (OSI) - це концептуальна рамка, що описує, як дані передаються в мережах комп'ютерів. Модель OSI була розроблена Міжнародною організацією зі стандартизації (ISO) і складається з семи рівнів, кожен з яких виконує певні функції. У моделі OSI (рис. 5) засоби взаємодії поділяються на сім рівнів: прикладний, представницький, сеансовий, транспортний, мережевий, канальний та фізичний [4, 5].

Коротко охарактеризуємо кожен рівень семирівневої моделі взаємодії відкритих систем OSI [1-3, 5].

1) Фізичний рівень (Physical Layer):

- відповідає за передачу необроблених бітів через фізичні середовища, такі як кабелі, радіохвилі тощо;
- включає апаратні елементи, такі як мережеві адаптери, роз'єми та кабелі.

2) Канальний рівень (Data Link Layer):

– забезпечує надійну передачу даних між сусідніми пристроями, формуючи кадри;
– включає протоколи для управління доступом до середовища передачі (наприклад, Ethernet).

3) Мережевий рівень (Network Layer):

– відповідає за маршрутизацію пакетів даних між різними мережами;
– використовує логічні адреси (наприклад, IP-адреси) для визначення шляху передачі;

– використовує протоколи, такі як TCP (надійний) і UDP (менш надійний).

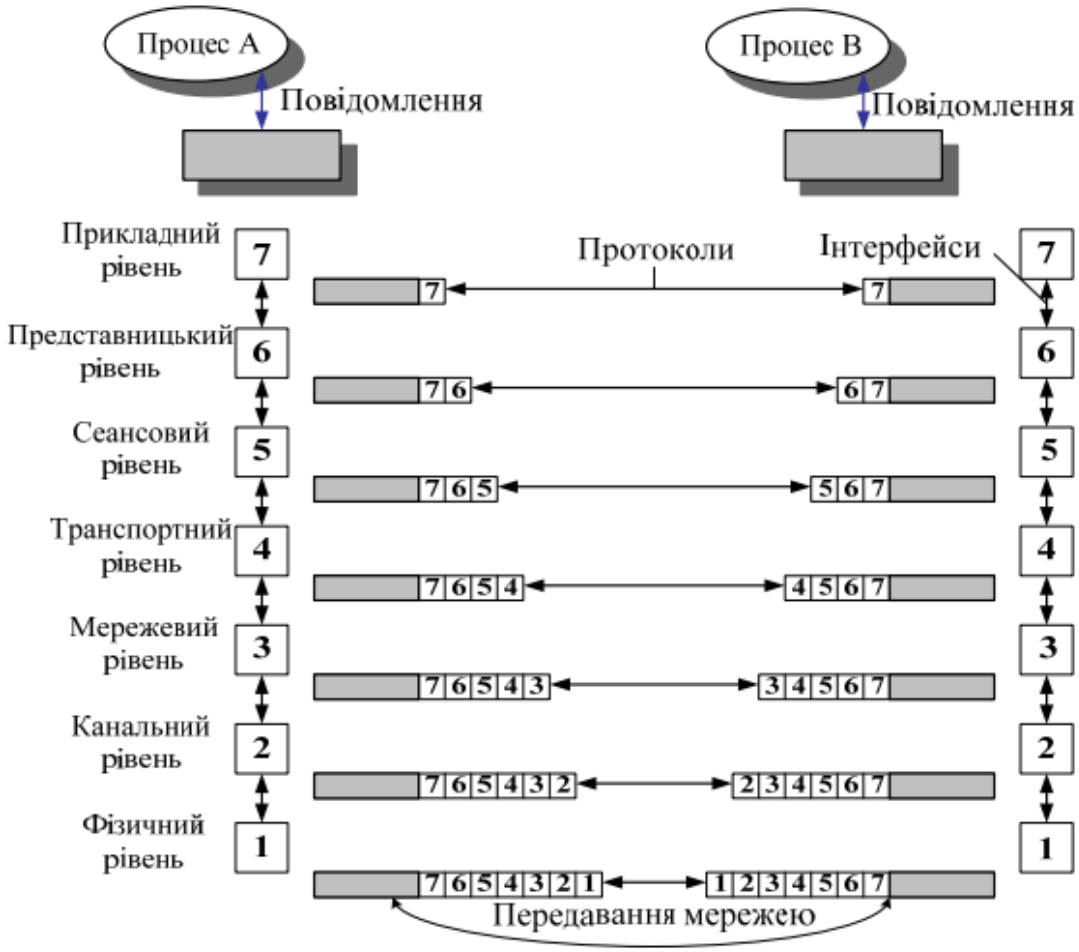

Рисунок 1.5 − Моделі взаємодії відкритих систем OSI [7]

4) Транспортний рівень (Transport Layer):

- забезпечує надійність передачі даних, контролюючи помилки та порядок пакетів;

- використовує протоколи, такі як TCP (надійний) і UDP (менш надійний).

5) Сеансовий рівень (Session Layer):

– відповідає за встановлення, підтримку та завершення сесій зв'язку між прикладними процесами;

– забезпечує механізми синхронізації та управління діалогами.

6) Представницький рівень (Presentation Layer):

– відповідає за форматування і кодування даних, а також за їхнє стиснення і шифрування;

– забезпечує сумісність між різними форматами даних.

7) Прикладний рівень (Application Layer):

– найближчий до користувача рівень, де реалізуються протоколи для взаємодії з прикладними програмами (наприклад, HTTP, FTP, SMTP);

– забезпечує інтерфейси для користувачів та програм.

Отже, додаток надсилає запит до прикладного рівня, наприклад, до файлової служби. На основі цього запиту програмне забезпечення прикладного рівня формує повідомлення у стандартному форматі.

Таке повідомлення складається із заголовка та поля даних. Заголовок містить службову інформацію, яку потрібно передати через мережу комп’ютеру-одержувачу, щоб вказати, яку задачу необхідно виконати. У нашому випадку заголовок повинен містити дані про місцезнаходження файлу та тип операції, яку слід виконати. Поле даних повідомлення може бути як пустим, так і містити деякі дані, наприклад, ті, що потрібно записати у віддалений файл. Однак для успішної доставки цієї інформації потрібно вирішити ще багато завдань, які покладені на нижчі рівні.

Після створення повідомлення рівень застосувань надсилає його вниз по стеку до рівня представлення. Протокол представлення, спираючись на інформацію з заголовка прикладного рівня, виконує необхідні дії та додає до повідомлення свій власний заголовок, який містить інструкції для вузла призначення. Це повідомлення передається далі до сеансового рівня, який також додає свій заголовок, і так продовжується далі. У деяких реалізаціях протоколів службова інформація може бути додана не лише на початку повідомлення, а й у кінці, у вигляді кінцевика. Врешті-решт, повідомлення досягає нижнього фізичного рівня, який передає його через лінії зв’язку до комп’ютера-одержувача. Таким чином, до цього моменту повідомлення накопичує заголовки всіх рівнів (рис. 1.6).

Коли повідомлення досягає комп'ютера призначення, його приймає фізичний рівень, після чого воно поступово піднімається до вищих рівнів. Кожен рівень аналізує та обробляє заголовок, який йому належить, виконує відповідні функції, а потім видаляє цей заголовок і передає повідомлення наступному, вищому рівню.

Спеціальні назви використовуються для позначення блоків даних на різних рівнях:

- сегмент (segment) — PDU транспортного рівня (TCP-сегмент);
- дейтаграма (datagram) — PDU транспортного рівня (UDP-дейтаграма);
- пакет (packet) — PDU мережевого рівня;
- кадр (frame) — PDU канального рівня.

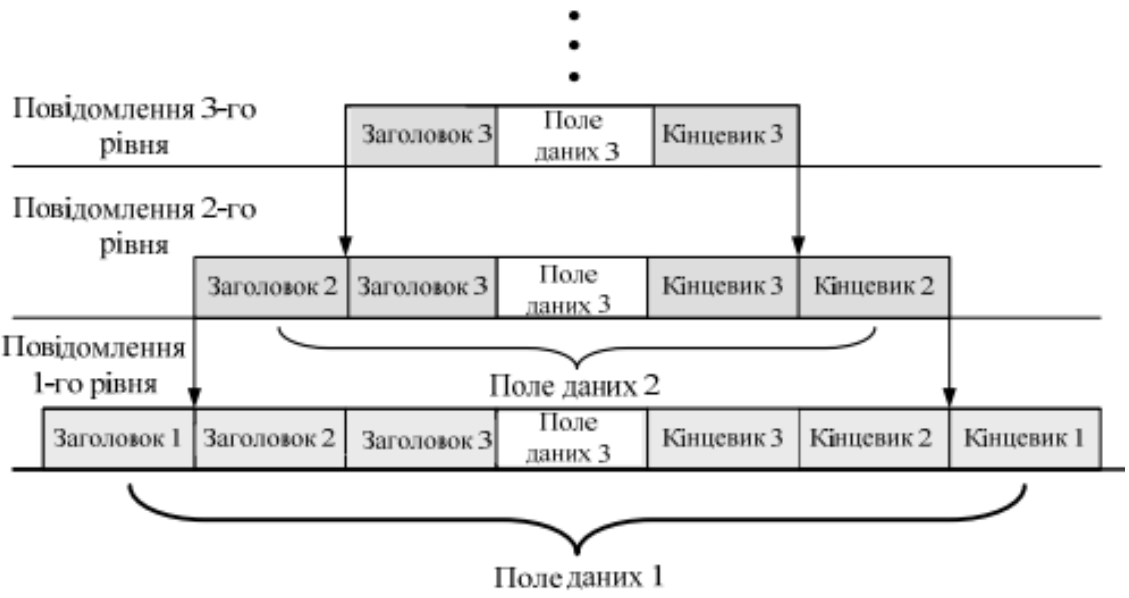

Рисунок 1.6 − Вкладеність повідомлень різних рівнів [7]

Модель OSI широко використовується для навчання та розуміння принципів мережевої комунікації, хоча на практиці багато сучасних протоколів, такі як TCP/IP, реалізують деякі функції з декількох рівнів одночасно.

1.3 Основні характеристики комп’ютерних мереж

Основною вимогою до комп’ютерних мереж є забезпечення користувачам доступу до ресурсів усіх спільно використовуваних комп’ютерів, що входять до складу мережі. Усі інші вимоги, такі як продуктивність, надійність, сумісність, керованість, безпека, розширюваність і масштабованість, залежать від якості виконання цієї основної функції [4, 5].

Продуктивність. Потенційно висока продуктивність є однією з ключових характеристик розподілених систем, до яких належать і комп'ютерні мережі. Вона досягається завдяки можливості розподілу завдань між кількома мережами. Існує кілька основних показників продуктивності мережі:

- час відгуку;
- пропускна здатність;
- затримка передачі та варіація затримки передачі.

Надійність комп'ютерної мережі є комплексним показником, до якого входять такі елементи:

- готовність;
- ймовірність доставки пакета без помилок;
- безпека;
- відмовостійкість.

Готовність (або коефіцієнт готовності) визначає тривалість часу, протягом якого система залишається функціональною. Поліпшити цю характеристику можна шляхом впровадження надлишковості: основні компоненти системи мають бути представлені в кількох копіях, щоб у разі відмови одного з них могли продовжувати працювати інші.

Щоб система демонструвала високу надійність, вона повинна мати високий рівень готовності та забезпечувати збереження даних.

Оскільки мережа функціонує на основі механізму передачі пакетів між кінцевими вузлами, однією з ключових характеристик надійності є

ймовірність безпомилкової доставки пакета до вузла призначення.

Безпека визначається як здатність системи захищати дані від несанкціонованого доступу.

Відмовостійкість мережі – це здатність мережі забезпечувати безперервність своєї роботи навіть за умови виникнення несправності або відмов її компонентів. Відмовостійка мережа може автоматично реагувати на збої, наприклад, за рахунок перенаправлення трафіку на альтернативні маршрути або використання резервних пристроїв. Завдяки цьому вона зменшує час простою та підвищує надійність і стабільність функціонування.

Відмовостійкість досягається через комбінацію апаратних і програмних рішень, таких як дублювання ключових елементів мережі (серверів, комутаторів, маршрутизаторів), використання протоколів маршрутизації, що підтримують резервування, а також застосування засобів моніторингу та автоматичного відновлення роботи. Основною метою відмовостійкості є забезпечення високої доступності мережевих ресурсів для користувачів і зменшення впливу інцидентів на бізнес-процеси та обслуговування клієнтів.

Керованість мережі визначає здатність централізовано контролювати стан основних компонентів мережі, виявляти і усувати проблеми, які можуть виникати під час її функціонування, проводити аналіз продуктивності та планувати подальший розвиток мережі.

Розширюваність означає здатність просто і швидко додавати нові елементи до мережі, такі як користувачі, комп’ютери, програми або служби, а також можливість збільшувати довжину сегментів мережі та замінювати існуюче обладнання на більш потужне. Важливо зазначити, що легкість розширення системи може бути обмежена певними рамками.

Масштабованість означає, що мережа може збільшувати кількість вузлів і довжину з'єднань в значних межах, не знижуючи при цьому продуктивність. Щоб забезпечити цю масштабованість, необхідно використовувати додаткове комунікаційне обладнання та структурно організовувати мережу певним чином.

Сумісність (або інтегрованість) означає, що мережа може функціонувати з різноманітним програмним і апаратним забезпеченням. Це означає, що в мережі можуть використовуватися різні операційні системи, які підтримують різні стеки комунікаційних протоколів, а також апаратні ресурси та програми від різних виробників. Мережа, що складається з елементів різних типів, називається неоднорідною (гетерогенною). Якщо така гетерогенна мережа функціонує без збоїв, її вважають інтегрованою.

Щоб підвищити ефективність роботи мережі підприємства, слід використовувати засоби її розширення у разі зростання кількості робочих станцій і користувачів.

Отже, як видно з аналізу літературних джерел, комп’ютерні мережі, будучи основою сучасної індустрії обробки інформації, пред’являють високі вимоги до ефективного використання засобів зв'язку та характеристик обслуговування мережевих абонентів. У зв'язку з цим однією з найважливіших проблем, яку доводиться вирішувати при практичному втіленні мережевих проектів та їх експлуатаційному супроводі, є проблема адмініструванні та організації ефективної роботи мережі у різних умовах функціонування. Тому актуальною є задача побудови комп’ютерної моделі класифікації стану комп’ютерних мереж, що є важливим інструментом для аналізу, моніторингу та управління мережевими ресурсами.

Порівняння типів мереж за ключовими характеристиками, які дозволяють визначити їх придатність для конкретних умов або застосувань наведені в таблиці 1.1.

Таблиця 1.1

Типи комп'ютерних мереж у розрізі їх основних характеристик

Характеристика	LAN (Локальна мережа)	WAN (Глобальна мережа)	MAN (Регіональна мережа)	PAN (Персональна мережа)
Охоплення	Кілька метрів до кількох кілометрів (офіс, будівля)	Від сотень до тисяч кілометрів (різні країни та континенти)	Декілька кілометрів (місто, регіон)	В межах кількох метрів (кімната, персональна зона)
Топологія	Зірка, шина, кільце, дерево, сітка	Зазвичай сітка з маршрутизацією	Зазвичай сітка, іноді зірка	Зірка, точка-точка
Пропускна здатність	Висока, зазвичай 1 Гбіт/с і більше	Низька до середньої, залежить від підключення	Середня, зазвичай сотні Мбіт/с	Низька, від кількох Кбіт/с до кількох Мбіт/с
Затримка	Низька	Висока	Середня	Дуже низька
Надійність	Висока, із можливістю швидкого відновлення	Від середньої до високої, залежить від операторів	Висока для міських умов	Низька
Безпека	Високий контроль доступу, внутрішній доступ	Потребує додаткових засобів безпеки, зокрема VPN	Середній, зазвичай через інтернет-провайдера	Зазвичай базова безпека, залежить від особистих налаштувань

Продовження таблиці 1.1

Характеристика	LAN (Локальна мережа)	WAN (Глобальна мережа)	MAN (Регіональна мережа)	PAN (Персональна мережа)
Відмовостійкість	Висока, з резервуванням	Середня, залежить від інфраструктури провайдерів	Висока для міських інфраструктур	Низька
Масштабованість	Обмежена, зазвичай в межах будівлі або кампусу	Висока, може охоплювати весь світ	Середня, обмежена містом або регіоном	Дуже обмежена, лише кілька пристроїв
Тип передавання даних	Дротова (Ethernet), бездротова (Wi-Fi)	Зазвичай через оптоволокно, супутниковий зв’язок	Дротова або бездротова, іноді з використанням радіохвиль	Зазвичай бездротова (Bluetooth, інфрачервоний зв’язок, Wi-Fi)
Протоколи	Ethernet, Wi-Fi	MPLS, ATM, Frame Relay, IPsec, іноді TCP/IP	Ethernet, Wi-Fi, іноді TCP/IP	Bluetooth, ZigBee, інші протоколи низької потужності
Управління мережею	Централізоване (мережевий адміністратор)	Децентралізоване, операторське	Централізоване або через інтернет-провайдер	Індивідуальне, просте

Продовження таблиці 1.1

Характеристика	LAN (Локальна мережа)	WAN (Глобальна мережа)	MAN (Регіональна мережа)	PAN (Персональна мережа)
Типи передавання	Одноадресний (unicast), груповий (multicast)	В основному одноадресний, іноді груповий	Залежить від типу підключення (unicast або multicast)	Одноадресний
Якість обслуговування (QoS)	Висока, контрольована якість	Може змінюватися, залежить від мережевих умов	Середня, залежить від оператора	Базова, рідко застосовується QoS
Мобільність	Обмежена фізичним місцезнаходженням	Може мати мобільний доступ через операторів	Можливий обмежений мобільний доступ	Висока мобільність, призначена для персональних пристроїв

Висновки до розділу 1

У першому розділі розглянуто теоретичні основи сучасних технологій комп’ютерних мереж. Сучасні технології комп’ютерних мереж є основою для розвитку інформаційних систем та забезпечення ефективної взаємодії між користувачами та ресурсами в різних середовищах. Аналіз стану розвитку комп’ютерних мереж демонструє зростаючу роль інноваційних підходів та рішень, таких як програмно-визначені мережі (SDN), віртуалізація мережевих функцій (NFV) та 5G технології, які надають можливість для більш гнучкого, безпечного та масштабованого управління мережею.

Огляд моделі OSI (Open Systems Interconnection) розкриває її значення для стандартизації процесів взаємодії між різними мережевими пристроями. OSI забезпечує структурований підхід, розділяючи процеси обміну даними на сім рівнів, що дозволяє ефективно розв'язувати задачі сумісності та інтероперабельності. Завдяки OSI розробники можуть створювати мережеві продукти та рішення, що відповідають міжнародним стандартам, забезпечуючи тим самим спрощену інтеграцію.

Основні характеристики комп'ютерних мереж, такі як продуктивність, масштабованість, надійність і безпека, залишаються критично важливими. Удосконалення технологій дозволяє покращувати ці показники, що забезпечує стабільну та безпечну роботу в складних умовах і підвищує якість обслуговування користувачів.

Таким чином, сучасні комп'ютерні мережі стають все більш динамічними та здатними до адаптації, а використання новітніх технологій та стандартів сприяє підвищенню їхньої ефективності, надійності та функціональності, відповідаючи вимогам сучасного цифрового суспільства. У зв'язку з цим однією з найважливіших проблем, яку доводиться вирішувати при практичному втіленні мережевих проектів та їх експлуатаційному супроводі, є проблема адмініструванні та організації ефективної роботи мережі у різних умовах функціонування.

РОЗДІЛ 2
ДОСЛІДЖЕННЯ МЕТОДІВ КЛАСИФІКАЦІЇ

Методи класифікації дозволяють розділити об’єкти відповідно до заздалегідь визначених класів. Найпростіше з технічного боку завдання класифікації — це бінарна класифікація, яка полягає в розподілі об’єктів між двома класами. Наприклад, якщо на вхід подаються транспортні засоби, ми знаємо, що це або автомобілі, або велосипеди. За певними алгоритмами машина розподіляє кожен з транспортних засобів до одного з визначених класів (автомобіль або велосипед) [6-9].

У багатокласовій класифікації кількість класів може досягати кількох тисяч, що ускладнює процес прийняття рішень. Існують також перетинні класи, коли об’єкт може належати одночасно до кількох класів, а також нечіткі класи, де належність до певного класу визначається ймовірністю [6].

Сьогодні алгоритми класифікації застосовуються для багатьох завдань, таких як визначення мови, спам-фільтрація, виявлення шахрайства (наприклад, у випадках, коли зловмисник користується вкраденими коштами для оплати послуг), а також для пошуку схожих документів тощо.

Щоб класифікація відбулася, необхідно мати дані з позначеними категоріями та характеристиками, які машина повинна навчитися розпізнавати. Алгоритм, виходячи з певних характеристик, визначає, до якого класу належить об'єкт [9, 10].

Алгоритми класифікації здатні розділяти об'єкти відповідно до попередньо визначених класів. В роботі буде розглянуто чотири основні та найпоширеніші методи [6-14]:

- дерево рішень (Decision Tree);
- метод k-найближчих сусідів (k-Nearest Neighbors);
- метод опорних векторів (Support Vector Machine);
- логістична регресія (Logistic Regression).

2.1 Дерева прийняття рішень

Дерево прийняття рішень (Decision Tree) — це один із методів класифікації та регресії в машинному навчанні, що базується на структурі дерева. У цьому методі дерево будується на основі набору правил, що розгалужуються від кореня до листових вузлів, і кожне з цих правил визначає умову або питання для класифікації чи прогнозування [15-18, 23].

Основні компоненти дерева:

1) вузол (Node) - представляє певну характеристику або властивість (фічу) об’єкта. На кожному вузлі виконується перевірка значення цієї характеристики;

2) гілка (Branch) - результат перевірки умови на вузлі. Кожна гілка відповідає за перехід на новий вузол на основі результату (правда/хиба, або інші можливі варіанти);

3) листовий вузол (Leaf Node) - це кінцевий вузол дерева, який містить остаточне рішення або клас. Це результат, який модель передбачає для об’єкта;

4) кореневий вузол (Root Node) — це початковий вузол дерева, з якого починається процес класифікації.

Процес побудови дерева складається з наступних операцій:

1) Вибір ознаки для розбиття: на кожному кроці вибирається найбільш інформативна ознака для поділу даних. Критеріями для цього можуть бути:

- інформаційна ентропія та приріст інформації (Information Gain) — показує, наскільки зменшиться невизначеність після розбиття за цією ознакою;
- коефіцієнт Джині (Gini Index) — показує рівень «чистоти» вузлів після поділу;
- та різноманітні інші показники, залежно від алгоритму.

2) Розбиття даних: на кожному вузлі дерево розбиває дані на підмножини на основі значення ознаки.

3) Рекурсія: процес розбиття продовжується рекурсивно на кожному з підмножин до тих пір, поки всі дані не будуть класифіковані або до досягнення заданих умов (наприклад, мінімальної кількості даних в підмножині або максимальної глибини дерева).

4) Обрізка дерева: щоб уникнути перенавчання (overfitting), дерево можна обрізати, тобто видаляти вузли, які не несуть важливої інформації для прийняття рішень. Це можна робити за допомогою методу «ранньої зупинки» або «після обрізки» (post-pruning).

Переваги алгоритму класифікації дерева прийняття рішень:

- легкість інтерпретації: рішення, прийняті моделлю, легко пояснити у вигляді логічних умов;
- гнучкість: можна використовувати для задач класифікації та регресії;
- не вимагає масштабування даних.

Недоліки алгоритму, полягають в наступному:

- схильність до перенавчання, особливо коли дерево стає занадто глибоким.
- чутливість до невеликих змін у даних (може змінити структуру дерева).

Древа рішень є основою для складніших методів, таких як Random Forest (випадкові ліси) і Gradient Boosting (градієнтний бустинг).

2.2 Метод k-найближчих сусідів

Метод k-найближчих сусідів (k-nearest neighbors, k-NN) — це простий, але ефективний метод класифікації та регресії в машинному навчанні. Він базується на тому, що об'єкти, які мають схожі характеристики, як правило, належать до одного класу [19-22, 24].

Основна ідея методу полягає в наступному: для класифікації нового зразка k-NN шукає найближчих сусідів у тренувальній вибірці та присвоює

йому клас на основі більшості класів серед цих сусідів. Для регресії передбачається середнє або середнє зважене значення сусідів.

Компоненти алгоритму:

1) Число сусідів (*k*): це число визначає, скільки найближчих сусідів враховується для прийняття рішення. Значення *k* має бути заздалегідь обрано.

- Якщо *k* мале, модель може бути чутливою до шуму і перенавчатися.
- Якщо *k* велике, модель може стати занадто простою і недооцінювати локальні властивості даних.

2) Метрика відстані: для пошуку найближчих сусідів необхідно вимірювати відстань між зразками. Найпоширеніші метрики, які використовуються в методі k-найближчих сусідів:

- Евклідова відстань (найчастіше використовується для безперервних ознак):

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}, \tag{2.1}$$

де x_i – *i*-та ознака об’єкта, що класифікується;

y_i – ознака класифікованих об’єктів.

- Манхеттенська відстань:

$$d(x, y) = \sum_{i=1}^{n} | x_i - y_i |. \tag{2.2}$$

- Косинусна подібність (особливо використовується для текстових даних):

$$d(x,y) = 1 - \frac{x \cdot y}{\|x\| \|y\|}. \quad (2.3)$$

3) Правило голосування (для класифікації): після того, як знайдено *k* найближчих сусідів, новий зразок відноситься до класу, який має більшість голосів серед цих сусідів. Існують також варіанти зваженого голосування, де ваги залежать від відстані (ближчі сусіди мають більшу вагу).

Процес класифікації складається з наступних дій:

1) вибір *k*: обирається кількість сусідів для моделі;

2) обчислення відстані: вимірюється відстань між новим зразком і всіма зразками тренувальної вибірки;

3) вибір *k* найближчих сусідів: вибираються *k* найближчих точок (зразків) до нового зразка;

4) голосування: новому зразку присвоюється той клас, якого найбільше серед сусідів.

Переваги використання методу k-найближчих сусідів для класифікації об’єктів:

- простота та легкість реалізації;
- не потребує явного тренування моделі, оскільки метод базується на використанні всього набору тренувальних даних;
- гнучкість: k-NN підходить як для класифікації, так і для регресії.

До недоліків методу k-найближчих сусідів можна віднести:

- низька швидкість та висока обчислювальна складність: метод k-NN потребує зберігання всього набору даних та обчислення відстаней до кожного зразка, що може бути ресурсомістким для великих даних;
- чутливість до вибору метрики відстані: результати сильно залежать від того, яку метрику використовувати і як масштабуються ознаки;
- вплив на шум: маленьке значення *k* може призвести до перенавчання через вплив шумових зразків.

Метод k-найближчих сусідів часто застосовується в задачах

розпізнавання образів, класифікації тексту, аналізу медичних даних, системах рекомендацій, тощо.

2.3 Метод опорних векторів

Метод опорних векторів (Support Vector Machine, SVM) — це один з потужних методів класифікації в машинному навчанні, що використовується для розв’язання як лінійних, так і нелінійних задач класифікації та регресії. Його основна ідея полягає в пошуку оптимальної гіперплощини, яка найкраще розділяє класи об'єктів у багатовимірному просторі ознак [6-9, 25].

Основна ідея методу SVM: метод опорних векторів будує гіперплощину (або гіперлінію в двовимірному просторі), яка максимізує відстань (маржин) між двома класами даних. Оптимальна гіперплощина — це та, яка має максимальний зазор (маржин) до найближчих зразків з кожного класу, які називаються опорними векторами.

Ключові компоненти:

1) Гіперплощина: у задачі класифікації SVM шукає лінію (в двовимірному просторі), площину (в тривимірному просторі) або гіперплощину (в просторі більшої розмірності), яка найкраще розділяє класи. Гіперплощина визначається як рівняння:

$$w \cdot x + b = 0, \tag{2.4}$$

де w - це вектор ваг,

x - точка в просторі ознак,

b - зміщення (bias).

2) Опорні вектори - це ті зразки навчального набору, які лежать найближче до гіперплощини та визначають її положення. Вони є критичними для побудови гіперплощини, оскільки відстань від цих зразків до

гіперплощини є маржином.

3) Маржин (відстань до гіперплощини) - це відстань між гіперплощиною та найближчими зразками кожного класу. SVM намагається максимізувати цю відстань, щоб покращити здатність до узагальнення на нових даних.

4) Кернел-функція (ядро): SVM може працювати з нелінійними даними за допомогою кернел-функцій. Ці функції перетворюють дані в простір вищої розмірності, де їх можна лінійно розділити. Найпоширеніші кернел-функції:

- лінійне ядро: використовується для лінійно роздільних даних;
- поліноміальне ядро: розширює простір ознак за допомогою поліноміальних співвідношень;
- радіальна базисна функція (RBF): найбільш поширена для нелінійних класифікацій, оскільки вона добре працює з більшістю видів даних;
- сигмоїдне ядро.

Формулювання задачі класифікації методом опорних векторів: SVM розв'язує оптимізаційну задачу для побудови гіперплощини. Якщо дані є лінійно роздільними, задача полягає в тому, щоб знайти таку гіперплощину, яка максимізує маржин між класами. Це виражається як задача мінімізації:

$$\min \frac{1}{2} \| w \|^2, \tag{2.5}$$

під умовою, що всі зразки правильно класифіковані:

$$y_i (w \cdot x_i + b) \geq 1, \quad \forall i, \tag{2.6}$$

де y_i - мітка класу для зразка i.

У реальних задачах класи можуть бути не повністю лінійно роздільними, і SVM дозволяє допускати помилки за допомогою параметра *C*, який контролює баланс між максимізацією маржина і кількістю помилок. Якщо *C* велике, алгоритм буде намагатися мінімізувати кількість помилок, жертвуючи маржином, і навпаки.

Переваги класифікації методом опорних векторів:

- ефективність: SVM є ефективним для задач з високою розмірністю;
- різноманітність кернел-функцій: може використовуватися як для лінійно, так і для нелінійно роздільних даних;
- мала схильність до перенавчання: через максимізацію маржину, SVM часто добре узагальнює дані.

До недоліків методу опорних векторів можна віднести:

- чутливість до вибору параметрів: ефективність SVM сильно залежить від правильного вибору параметрів, таких як коефіцієнт *C* і параметри кернел-функції;
- обчислювальна складність: для великих наборів даних обчислення відстаней до кожного зразка може бути ресурсномістким;
- інтерпретованість: на відміну від деяких інших методів (наприклад, дерев рішень), результат класифікації SVM важко інтерпретувати.

SVM широко використовується для розпізнавання образів, класифікації тексту, аналізу біологічних даних, обробки сигналів, виявлення спаму та інших задач, де дані можуть бути складними або мати високу розмірність.

2.3 Логістична регресія

Логістична регресія (Logistic Regression) — це статистичний метод класифікації, який використовується для моделювання ймовірності належності об'єкта до одного з двох класів. Хоча назва містить слово «регресія», цей метод є популярним для задач класифікації, оскільки його метою є передбачення ймовірності належності об'єкта до певного класу

(бінарна класифікація) [6, 14, 26].

Основна ідея логістичної регресії полягає в наступному: логістична регресія використовує лінійну функцію для прогнозування ймовірності події, але, щоб ймовірності були між 0 і 1, використовує логістичну функцію (сигмоїдну функцію). Таким чином, вона перетворює будь-яке значення виходу лінійної моделі у значення, що лежить в інтервалі [0, 1].

Залежність між вектором ознак $X=(x_1, x_2, \ldots, x_n)$ і ймовірністю події $P(y=1|X)$ виражається за допомогою логістичної функції (сигмоїдальної функції):

$$P(y=1|\ X)=\frac{1}{1+e^{-(w_0+w_1x_1+w_2x_2+\ldots+w_nx_n)}}, \tag{2.7}$$

де: w_0 - вільний член (intercept),

$w_1, w_2, \ldots, w_n$ - коефіцієнти (ваги) ознак,

$x_1, x_2, \ldots, x_n$ - значення ознак,

e - основа натурального логарифма.

Сигмоїдна функція має вигляд:

$$\sigma(z)=\frac{1}{1+e^{-z}}, \tag{2.8}$$

де z — це лінійна комбінація ознак.

Прийняття рішення: після того, як модель обчислює ймовірність належності зразка до класу, рішення приймається за пороговим значенням (звичайно, 0.5):

- якщо $P(y=1|X)\geq 0,5$, то прогнозується, що y=1.
- якщо $P(y=1|X)<0,5$, то прогнозується, що y=0.

Логістична регресія також може бути записана через логіт-функцію, яка представляє лінійну регресію на логарифмі співвідношення шансів (логіт):

$$\ln \frac{P(y=1|\ X)}{1-P(y=1|\ X)} = w_0 + w_1 x_1 + w_2 x_2 + ... + w_n x_n , \qquad (2.9)$$

Ця форма підкреслює, що логістична регресія лінійна за своїми коефіцієнтами в просторі логітів.

Коефіцієнти w_1, w_2, . . . , w_n навчаються за допомогою методу максимізації правдоподібності. Це метод, який знаходить такі параметри моделі, що максимізують ймовірність отримання спостережуваних даних.

Переваги використання методу логістичної регресії для класифікації об’єктів:

- простота інтерпретації: кожен коефіцієнт можна інтерпретувати як внесок відповідної ознаки у зміну ймовірності належності до класу;
- швидкість та ефективність: логістична регресія є швидкою і ефективною для тренування на невеликих або середніх наборах даних;
- добра узагальнюваність: завдяки регуляризації (наприклад, L1 або L2) логістична регресія може добре узагальнювати дані, що допомагає уникнути перенавчання.

До недоліків методу логістичної регресії можна віднести:

- лінійність: логістична регресія передбачає лінійну залежність між ознаками та логарифмом шансів. Вона може погано працювати на сильно нелінійних даних;
- чутливість до мультиколінеарності: якщо деякі ознаки сильно корелюють між собою, це може вплинути на точність моделі;
- потреба в балансуванні класів: логістична регресія може не працювати добре на даних із сильною дисбалансованістю класів.

Метод логістичної регресії застосовується в медичній діагностиці - для

визначення ймовірності наявності хвороби на основі ознак пацієнта; для кредитного скорингу (при оцінці ризику неповернення кредиту на основі фінансових показників клієнта), в маркетингу – для передбачення ймовірності того, що клієнт відреагує на маркетингову кампанію.

Логістична регресія є одним із найпопулярніших базових алгоритмів класифікації завдяки її простоті, ефективності та інтерпретованості результатів [26, 27].

Висновки до розділу 2

У другому розділі досліджено методи класифікації, що є важливим етапом в аналізі та обробці даних, оскільки класифікація дозволяє структурувати інформацію та розпізнавати закономірності для прийняття обґрунтованих рішень. Існує широкий спектр методів класифікації, таких як дерева рішень, методи опорних векторів (SVM), нейронні мережі, байєсівські класифікатори та методи ансамблю (наприклад, Random Forest і Gradient Boosting), кожен з яких має свої особливості, переваги та недоліки.

Дерева рішень забезпечують простоту інтерпретації та візуалізації результатів, що робить їх зручними для використання в завданнях з помірною кількістю характеристик. Методи опорних векторів дозволяють отримати точні результати для задач з високою вимірністю, але потребують ретельного налаштування параметрів. Нейронні мережі, зокрема глибокі, здатні обробляти великі обсяги складних даних, проте вимагають значних обчислювальних ресурсів та часу на навчання.

Кожен метод класифікації має своє оптимальне застосування в залежності від особливостей даних, обчислювальних ресурсів та вимог до точності. Комплексне використання методів, зокрема за допомогою ансамблевих підходів, дозволяє підвищити точність класифікації та знизити ризик помилок, пов’язаних із вибірковістю конкретного методу.

Отже, вибір підходящого методу класифікації залежить від специфіки

задачі, типу даних та обмежень системи, що підкреслює важливість ретельного аналізу та експериментальної перевірки під час розробки моделей класифікації.

Отже, в роботі пропонується побудувати модель класифікації стану комп’ютерної мережі на основі розглянутих вище алгоритмів, а саме: дерева прийняття рішень (Decision Tree), метод k-найближчих сусідів (k-Nearest Neighbors), метод опорних векторів (Support Vector Machine) та логістична регресія, що дасть змогу підвищити точність класифікації стану комп’ютерних мереж.

РОЗДІЛ 3
РОЗРОБКА КОМП'ЮТЕРНОЇ МОДЕЛІ КЛАСИФІКАЦІЇ СТАНУ КОМП'ЮТЕРНИХ МЕРЕЖ

3.1 Вибір засобів та технологій для створення комп'ютерної моделі

Для реалізації комп'ютерної моделі класифікації стану комп'ютерних мереж було визначено та проаналізовано кілька ключових критеріїв вибору програмного забезпечення, які забезпечують ефективність, зручність та надійність під час розробки, навчання та експлуатації моделі. Серед основних критеріїв можна виділити [28-30]:

1) Продуктивність: ПЗ має забезпечувати високу обчислювальну потужність та ефективність роботи з великими обсягами даних. Це включає швидкість обробки даних та час виконання алгоритмів, особливо якщо йдеться про великі набори даних чи складні обчислення.

2) Гнучкість та масштабованість: ПЗ має підтримувати можливість адаптації під різні архітектури та обсяги даних. Важливою є також підтримка розширення функціоналу для інтеграції з іншими інструментами та платформами.

3) Підтримка алгоритмів машинного навчання: ключовим фактором є наявність різноманітних алгоритмів класифікації, таких як дерева рішень, нейронні мережі, методи опорних векторів, ансамблеві методи тощо, що дозволяє реалізовувати різні підходи в класифікації.

4) Простота у використанні та наявність документації: інтуїтивно зрозумілий інтерфейс, наявність документації та навчальних матеріалів є важливими для швидкого освоєння ПЗ та зручної роботи, що допомагає скоротити час на навчання й оптимізацію моделі.

5) Підтримка обробки великих даних та інтеграція з базами даних. Для класифікації стану комп'ютерних мереж важливим є можливість інтеграції з базами даних та обробки великих обсягів інформації в режимі реального

часу, що забезпечує актуальність і точність результатів.

6) Вартість. При виборі програмного забезпечення враховується його вартість, зокрема можливість використання безкоштовних або ліцензованих версій, які б задовольняли технічні потреби проєкту, враховуючи наявний бюджет.

Зважаючи на ці критерії, вибір ПЗ для реалізації комп'ютерної моделі класифікації є комплексним завданням, що потребує балансу між продуктивністю, гнучкістю, функціональністю та вартістю.

3.1.1 Вибір мов програмування для створення комп’ютерної моделі

Розробка комп’ютерної моделі класифікації стану комп’ютерних мереж вимагає вибору мови програмування, яка відповідала б вимогам ефективності, продуктивності та підтримки сучасних алгоритмів машинного навчання. При виборі мови враховувались такі критерії, як доступність бібліотек для машинного навчання, зручність інтеграції з базами даних, можливість швидкої обробки великих обсягів даних та активна підтримка спільнотою. Нижче розглянуто найбільш популярні мови, які підходять для реалізації таких моделей, та їх основні переваги та недоліки [29, 30].

1) ***Python***. Python є однією з найбільш популярних мов програмування для задач машинного навчання та аналізу даних. Його переваги включають простоту синтаксису, що прискорює розробку, а також широкий спектр бібліотек, таких як Scikit-Learn, TensorFlow, Keras, PyTorch та Pandas, які забезпечують готові алгоритми класифікації, обробки даних та роботи з нейронними мережами. Python також підтримує інтеграцію з іншими мовами та платформами, що дозволяє розширити функціональність та зручність використання в складних проєктах. Водночас недоліком Python є дещо нижча продуктивність у порівнянні з компільованими мовами, такими як C++ або Java, особливо у випадках високонавантажених обчислень.

2) ***R***. Мова R також є популярною серед дослідників і розробників у

сфері машинного навчання, зокрема завдяки спеціалізації на статистичному аналізі та обробці даних. R має широкий спектр бібліотек для класифікації та візуалізації даних, таких як caret, randomForest, e1071, що спрощує аналіз даних. R добре підходить для створення моделей класифікації, але має певні обмеження у швидкодії та гнучкості у порівнянні з Python, а також складніший синтаксис, що може вимагати додаткових навичок від розробників.

3) ***Java***. Java забезпечує високу продуктивність і стабільність, що є важливим для застосувань у великих корпоративних мережах з високим навантаженням. Java має низку бібліотек для машинного навчання, таких як Weka, DL4J (DeepLearning4J), що дозволяють реалізовувати базові алгоритми класифікації та обробки даних. Крім того, Java підтримує платформну незалежність і добре інтегрується з іншими системами, що робить її доцільним вибором для розгортання моделей в масштабованих мережевих середовищах. Однак у порівнянні з Python, Java менш зручна для швидкої розробки та тестування моделей, і потребує більше часу для написання коду.

4) ***C++***. C++ забезпечує високу швидкість і ефективність роботи, що є критичним для обробки великих обсягів даних і виконання ресурсомістких алгоритмів класифікації в реальному часі. Ця мова дозволяє тонке налаштування пам’яті та обчислювальних ресурсів, що є перевагою для інтенсивних обчислень. Проте, C++ не має такої кількості спеціалізованих бібліотек для машинного навчання, як Python чи R, і потребує більше часу на розробку та налагодження, тому не завжди підходить для швидкого прототипування.

5) ***Julia***. Julia є відносно новою мовою, але її популярність зростає завдяки поєднанню високої продуктивності та простого синтаксису. Julia має бібліотеки, такі як Flux.jl та MLJ.jl, які підтримують машинне навчання та обробку даних, а також забезпечує високу продуктивність, що може конкурувати з C++ при значно простішому коді. Однак екосистема Julia ще не настільки розвинута, як у Python, що може обмежити доступ до деяких

спеціалізованих інструментів.

Отже, для реалізації комп'ютерної моделі класифікації стану комп'ютерних мереж найкращим вибором є Python завдяки простоті використання, багатству бібліотек для машинного навчання, широкій підтримці спільноти та можливості інтеграції з різними інструментами і платформами. Проте, в умовах підвищених вимог до продуктивності варто також розглянути C++ для вузькоспеціалізованих обчислень або Julia як перспективну альтернативу для продуктивної розробки.

3.1.2 Огляд бібліотек Python для машинного навчання

Python є однією з найпопулярніших мов програмування для реалізації моделей машинного навчання завдяки її простому синтаксису та наявності великої кількості спеціалізованих бібліотек. Ці бібліотеки надають готові алгоритми та інструменти для розробки, навчання, оцінки та розгортання моделей машинного навчання, що значно спрощує процес їх реалізації. Розглянемо основні бібліотеки Python, які широко використовуються для машинного навчання, а також їх основні функції та особливості.

1) ***NumPy.*** NumPy — це основна бібліотека для числових обчислень у Python, яка забезпечує підтримку багатовимірних масивів і матриць, а також надає велику кількість функцій для виконання математичних операцій над ними. NumPy є критично важливим інструментом у машинному навчанні, оскільки дозволяє швидко та ефективно обробляти великі обсяги даних. Бібліотека також слугує базою для багатьох інших бібліотек, таких як Scikit-Learn і TensorFlow, що робить її необхідною для роботи з даними та математичними операціями. Завдяки високій швидкості обробки даних та простоті інтеграції, NumPy використовується для створення масивів, операцій над матрицями та виконання лінійної алгебри, що є основою багатьох алгоритмів машинного навчання [6, 9, 27-30].

2) ***Matplotlib.*** Matplotlib є популярною бібліотекою для візуалізації

даних у Python. Вона дозволяє створювати графіки, гістограми, діаграми розсіювання та інші типи візуалізацій, що робить її важливим інструментом для аналізу та презентації результатів машинного навчання. Візуалізація є важливою частиною процесу машинного навчання, оскільки вона дозволяє зрозуміти розподіл даних, оцінити результати моделі, а також виявляти аномалії чи тенденції. Matplotlib також добре інтегрується з іншими бібліотеками, такими як NumPy та Pandas, що забезпечує зручність у використанні та гнучкість при роботі з різними типами даних.

3) ***Scikit-Learn.*** Scikit-Learn є однією з найпопулярніших бібліотек для класичних алгоритмів машинного навчання. Вона надає широкий набір інструментів для класифікації, регресії, кластеризації, вибору ознак, попередньої обробки даних і оцінки моделей. Scikit-Learn має простий інтерфейс та гарну документацію, що робить її зручною для швидкої побудови та тестування моделей. Бібліотека також дозволяє налаштовувати гіперпараметри моделей та надає методи для крос-валідації, що допомагає покращити якість моделі. Scikit-Learn ідеально підходить для класичних задач машинного навчання, але не включає засобів для роботи з нейронними мережами.

4) ***TensorFlow.*** TensorFlow, розроблений Google, є потужною бібліотекою для глибокого навчання. Вона дозволяє створювати та навчати складні нейронні мережі, обробляти великі обсяги даних та використовувати графічні процесори (GPU) для прискорення обчислень. TensorFlow включає в себе набір інструментів для створення, налаштування та оптимізації багаторівневих нейронних мереж, а також підтримує роботу з TensorBoard для візуалізації процесу навчання моделі. Хоча TensorFlow має багатий функціонал, вона може бути складною для початківців, проте є високо адаптивною для складних та масштабованих завдань машинного навчання.

5) ***Keras.*** Keras - це високорівневий API, який працює поверх TensorFlow, що забезпечує спрощений інтерфейс для розробки нейронних мереж. Keras має простий і зрозумілий синтаксис, що дозволяє швидко

створювати та навчати моделі, навіть без глибоких знань у програмуванні нейронних мереж. Бібліотека Keras підтримує основні типи нейронних мереж, такі як згорткові (CNN), рекурентні (RNN) та багатошарові перцептрони, що робить її чудовим вибором для швидкого прототипування моделей глибокого навчання.

6) ***PyTorch.*** PyTorch, розроблений компанією Facebook, є популярною бібліотекою для глибокого навчання, яка відзначається динамічним обчислювальним графом, що дозволяє змінювати структуру моделі в процесі навчання. PyTorch є зручною для дослідників і розробників, оскільки забезпечує гнучкість і простоту налагодження, а також дозволяє писати код, схожий на звичайний Python. PyTorch має численні додаткові бібліотеки, наприклад, torchvision для обробки зображень, що робить її популярною для завдань з обробки природної мови та комп'ютерного зору. Ця бібліотека також широко використовується в наукових дослідженнях.

7) ***XGBoost.*** XGBoost (eXtreme Gradient Boosting) - це бібліотека для створення моделей на основі ансамблевих методів, зокрема градієнтного бустингу. Вона є дуже ефективною для задач класифікації та регресії, особливо для роботи з табличними даними, та часто показує високу точність у порівнянні з іншими методами. XGBoost забезпечує високу швидкість роботи та підтримує GPU, що дозволяє обробляти великі обсяги даних. Бібліотека також добре інтегрується з іншими інструментами Python, такими як Scikit-Learn.

8) ***LightGBM.*** LightGBM, розроблений Microsoft, є ще однією бібліотекою для градієнтного бустингу, яка відзначається високою продуктивністю та ефективністю при роботі з великими даними. LightGBM використовує алгоритм, що працює на основі навчання на розподілених даних і листах дерева, що забезпечує високу швидкість роботи. Вона також добре підходить для задач класифікації, регресії та ранжування і має функцію автоматичної оптимізації гіперпараметрів.

9) ***CatBoost.*** CatBoost є бібліотекою для градієнтного бустингу, яка

добре справляється із задачами з табличними даними та автоматично обробляє категоріальні ознаки, що знижує необхідність попередньої обробки даних. CatBoost є стабільною і швидкою бібліотекою, яка демонструє високу точність у задачах класифікації і регресії. Її перевагою є також стійкість до переобучення завдяки використанню методів регуляризації.

Кожна з перелічених бібліотек Python має свої особливості та оптимально підходить для різних задач машинного навчання. Для числових обчислень і обробки масивів даних базовими бібліотеками є NumPy та Matplotlib, які забезпечують обчислення та візуалізацію. Для класичних алгоритмів підходить Scikit-Learn, тоді як для глибокого навчання можна обирати TensorFlow, Keras та PyTorch. Для задач класифікації та регресії з табличними даними ефективними є XGBoost, LightGBM та CatBoost.

3.1.3 Візуалізація результатів роботи алгоритмів класифікації

Для візуалізації роботи алгоритмів класифікації та їх результатів можна використати кілька видів діаграм, щоб краще зрозуміти, як кожна модель працює з даними. Ось основні діаграми, які можуть допомогти:

1) Матриця плутанини (Confusion Matrix):

- показує, як модель класифікує кожен клас: коректні та некоректні передбачення для кожної категорії;
- можна побудувати для кожної моделі, щоб бачити, де саме вона помиляється.

```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

for model_name, model in models.items():
    predictions = model.predict(X_test)
    cm = confusion_matrix(y_test, predictions)
    disp = ConfusionMatrixDisplay(confusion_matrix=cm)
    disp.plot()
    plt.title(f"Confusion Matrix for {model_name}")
    plt.show()
```

Рисунок 3.1 – Лістинг коду побудови матриці плутанини

2) Крива ROC та AUC (для моделей, що підтримують бінарну класифікацію):

- показує, як змінюється точність та чутливість при зміні порога класифікації;
- коефіцієнт AUC (Area Under Curve) дає уявлення про ефективність моделі: значення ближче до 1 вказує на кращу модель.

```
from sklearn.metrics import roc_curve, auc

for model_name, model in models.items():
    # Перевірка, чи підтримується predict_proba
    if hasattr(model, "predict_proba"):
        y_probs = model.predict_proba(X_test)[:, 1]
        fpr, tpr, _ = roc_curve(y_test, y_probs)
        roc_auc = auc(fpr, tpr)
        plt.plot(fpr, tpr, label=f"{model_name} (AUC = {roc_auc:.2f})")

plt.plot([0, 1], [0, 1], 'k--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc="lower right")
plt.show()
```

Рисунок 3.2 – Лістинг коду побудови кривих ROC та AUC

3) Precision-Recall крива, яка є корисною для задач із незбалансованими даними, оскільки показує співвідношення точності та повноти.

```
from sklearn.metrics import precision_recall_curve, average_precision_score

for model_name, model in models.items():
    if hasattr(model, "predict_proba"):
        y_probs = model.predict_proba(X_test)[:, 1]
        precision, recall, _ = precision_recall_curve(y_test, y_probs)
        ap_score = average_precision_score(y_test, y_probs)
        plt.plot(recall, precision, label=f"{model_name} (AP = {ap_score:.2f})")

plt.xlabel('Recall')
plt.ylabel('Precision')
plt.title('Precision-Recall Curve')
plt.legend(loc="upper right")
plt.show()
```

Рисунок 3.3 – Лістинг коду побудови Precision-Recall кривої

4) Дерева рішень (тільки для Decision Tree): візуалізація структури дерева рішень допомагає зрозуміти, які умови модель використовує для

прийняття рішень.

```
from sklearn.tree import plot_tree

decision_tree_model = DecisionTreeClassifier()
decision_tree_model.fit(X_train, y_train)
plt.figure(figsize=(15, 10))
plot_tree(decision_tree_model,
          filled=True, feature_names=X.columns,
          class_names=['Class 0', 'Class 1'])
plt.title("Decision Tree Visualization")
plt.show()
```

Рисунок 3.4 – Лістинг коду для візуалізація структури дерева рішень

5) Важливість ознак (Feature Importance) для ансамблевих методів (Random Forest) - вказує, які ознаки найбільше впливають на результат, що корисно для інтерпретації моделі.

```
importances = models['Random Forest'].feature_importances
_indices = np.argsort(importances)[::-1]
plt.bar(range(X.shape[1]), importances[indices])
plt.xticks(range(X.shape[1]), X.columns[indices], rotation=90)
plt.title("Feature Importances for Random Forest")
plt.show()
```

Рисунок 3.5 – Лістинг коду побудови діаграми важливості ознак

Ці діаграми допоможуть детально проаналізувати кожен алгоритм та зрозуміти його ефективність і особливості роботи з даними.

3.1.3 Вибір інтегрованого середовища розробки

Наступним кроком є вибір інтегрованого середовища розробки Інтегроване середовище розробки (IDE, Integrated Development Environment) є ключовим інструментом для ефективного написання, тестування та налагодження програмного коду. Вибір IDE має важливе значення при реалізації комп’ютерних моделей машинного навчання, оскільки від цього залежить зручність роботи з кодом, можливості інтеграції з бібліотеками та підтримка додаткових функцій. Далі розглянемо основні критерії вибору IDE та огляд найпопулярніших середовищ розробки для Python, які підходять для

проєктів у сфері машинного навчання.

Розглянемо критерії вибору IDE. При виборі IDE для проєктів машинного навчання важливо враховувати такі аспекти:

1) Підтримка Python та бібліотек для машинного навчання. IDE повинне забезпечувати зручну роботу з бібліотеками, такими як NumPy, Pandas, Scikit-Learn, TensorFlow, PyTorch та іншими.

2) Можливість інтеграції з інструментами візуалізації. Бібліотеки на зразок Matplotlib, Seaborn та інші вимагають зручних засобів для перегляду графіків і візуалізації даних.

3) Засоби для налагодження та відладки. Ідеальне IDE повинно мати інструменти для відстеження помилок і аналізу коду в реальному часі.

4) Підтримка роботи з даними та обробки великих обсягів інформації. Наявність підтримки для роботи з великими наборами даних, а також можливість запуску коду на різних пристроях, включно з GPU.

5) Зручність роботи з версіями коду. Можливість інтеграції з системами контролю версій, такими як Git, важлива для командних проєктів та трекінгу змін у коді.

Зробимо огляд найпопулярніших IDE для Python.

1) ***Jupyter Notebook***. Jupyter Notebook є одним з найпоширеніших середовищ для роботи з Python у сфері машинного навчання. Воно забезпечує зручність у написанні та виконанні коду за допомогою поділу на комірки, що дозволяє легко тестувати окремі блоки коду, а також швидко аналізувати дані. Jupyter Notebook ідеально підходить для обробки даних і візуалізації результатів, оскільки підтримує інтеграцію з бібліотеками Matplotlib, Seaborn та іншими. Завдяки своїй простоті й гнучкості, Jupyter широко використовується для дослідницьких проєктів та швидкого прототипування моделей машинного навчання.

2) ***PyCharm.*** PyCharm, розроблений JetBrains, є потужним і багатофункціональним IDE, що спеціалізується на роботі з Python. PyCharm пропонує велику кількість інструментів для розробки, зокрема налагоджувач,

інтеграцію з системами контролю версій, підтримку бібліотек для машинного навчання та можливість віддаленого запуску на сервері. PyCharm також підтримує інтеграцію з інструментами на зразок Jupyter Notebook, що дозволяє працювати з кодом у вигляді окремих комірок. PyCharm є чудовим вибором для комплексних проєктів, де потрібна робота з великими обсягами даних і складним кодом.

3) ***Spyder.*** Spyder (Scientific Python Development Environment) є спеціалізованим IDE для наукових обчислень і машинного навчання. Воно надає широкий функціонал для аналізу даних і візуалізації, зокрема інтеграцію з бібліотеками, такими як NumPy, Pandas і Matplotlib. Spyder має інтерфейс, схожий на MATLAB, що може бути зручним для користувачів, які працюють у сфері науки та обчислень. Це середовище особливо популярне серед дослідників та студентів, оскільки воно підтримує основні функції для аналізу й обробки даних, має зручний інтерфейс і простоту налаштування.

4) ***Visual Studio Code.*** Visual Studio Code (VS Code) є легким та адаптивним редактором коду від Microsoft, який підтримує роботу з багатьма мовами програмування, зокрема Python. Завдяки встановленню розширень, таких як Python, Jupyter та інші, VS Code можна налаштувати під потреби машинного навчання. IDE забезпечує зручне налагодження, можливості для запуску окремих комірок Jupyter Notebook, інтеграцію з Git та підтримку віддаленого підключення до серверів. Visual Studio Code є потужним і гнучким інструментом, який підходить як для простих, так і для складних проєктів, а його функціональність можна розширювати за допомогою численних додаткових модулів.

5) ***Google Colab.*** Google Colab - це онлайн середовище розробки, яке базується на Jupyter Notebook і пропонує безкоштовний доступ до обчислень на GPU та TPU, що робить його особливо привабливим для роботи з великими обсягами даних і глибокого навчання. Colab підтримує всі популярні бібліотеки для машинного навчання, надає інтеграцію з Google Drive для зберігання даних і проектів. Це середовище підходить для

дослідницьких завдань та розробки моделей глибокого навчання без необхідності налаштування спеціального обладнання.

Отже, враховуючи все вищенаведене у роботі для побудови комп'ютерної моделі було обрано середовище PyCharm, оскільки воно не лише забезпечує усі потреби реалізації проєктів машинного та візуалізації даних, а і забезпечує комплексний підхід до розробки з повним функціоналом для тестування й налагодження коду, що підходить для складних проєктів.

3.2 Модель класифікації стану комп'ютерних мереж

Розглянемо модель класифікації стану комп'ютерних мереж. Вона використовується для визначення поточного стану мережі, ідентифікації потенційних проблем та їхньої природи (наприклад, перевантаження, несправності, атаки) та прийняття рішень щодо оптимізації мережевої інфраструктури. Така модель може бути корисною для виявлення проблем у мережі на ранніх стадіях, що дозволяє вчасно вжити заходів для їх усунення.

Розглянемо основні етапи розробки моделі класифікації стану мережі:

1) Збір даних – дані для тренування моделі можуть включати параметри мережевого трафіку, логі файли, дані про пропускну здатність, затримки, частоту помилок тощо. Дані можуть збиратися з допомогою мережевих моніторингових систем або спеціалізованого обладнання.

2) Попередня обробка даних – обробка зібраних даних включає видалення шуму, фільтрацію та нормалізацію, а також створення нових ознак (features), які можуть бути корисними для класифікації.

3) Вибір моделі – можна використовувати різні алгоритми машинного навчання, такі як:

- Логістична регресія (Logistic Regression);
- Метод опорних векторів (SVM);
- K-найближчих сусідів (KNN);

- Random Forest;
- Naive Bayes;
- Дерева рішень (Decision Tree);
- Нейронні мережі (MLPClassifier).

4) Тренування та тестування моделі – на цьому етапі модель навчається на історичних даних з позначками (метками) та перевіряється на нових, не бачених раніше, щоб оцінити її точність.

5) Оцінка та оптимізація – проводиться аналіз ефективності моделі за допомогою метрик, таких як точність (accuracy), повнота (recall), F1-міра та інші. За потреби модель налаштовують, щоб досягти кращих результатів.

6) Впровадження та моніторинг – модель інтегрується у систему моніторингу мережі, щоб у режимі реального часу аналізувати стан і попереджати про можливі проблеми.

Для візуалізації отриманих результатів будуються графіки:

- матриці кореляцій;
- діаграми розсіювання (scatter plot);
- графіки точності для порівняння різних моделей.

Для побудови моделей та графіків будемо використовувати бібліотеки pandas, numpy, scikit-learn, matplotlib та seaborn.

Алгоритм роботи моделі класифікації стану комп’ютерних мереж складається з наступних дій:

- збір даних: імпорт даних із CSV-файлу.

В обраному наборі даних містяться наступні параметри мережевого трафіку:

- traffic – трафік;
- connections – підключення;
- latency – затримка;
- packet_loss - втрати пакетів;
- errors – помилки;
- failure_time - час відмови;

- cpu_usage - використання ЦП;
- state – стан мережі.

- попередня обробка: стандартизація даних для коректної роботи моделей.

- алгоритми класифікації: використовуються кілька моделей класифікації (логістична регресія, SVM, KNN, Random Forest, Naive Bayes, дерева рішень, нейронна мережа). Для кожної моделі оцінюється точність і виводиться звіт по класифікації.

- візуалізація отриманих результатів:

- матриця кореляцій для аналізу взаємозв'язків між ознаками (рис. 3.9);
- діаграми розсіювання для візуалізації класифікації різних алгоритмів (рис. 3.10 – 3.15);
- матриця плутанини для кожного алгоритму (рис. 3.16 - 3.22);
- діаграма точності для порівняння результатів моделей (рис. 3.23).

Цей підхід дозволить порівняти точність різних моделей і візуалізувати результати для кращого розуміння ефективності кожного з алгоритмів.

Результати оцінки точності класифікації для вибірки даних за всіма застосованими методами класифікації наведено у табл. 3.1.

Таблиця 3.1

Результати оцінки точності класифікації

Метод класифікації	Точність класифікації
Logistic Regression (логістична регресія)	0,7733
SVM (метод опорних векторів)	0,9400
KNN (k-найближчих сусідів)	0,8933
Random Forest (випадковий ліс)	0,9900
Naive Bayes (простий байєсівський класифікатор)	0,8467
Decision Tree (дерева прийняття рішень)	0,9933
Neural Network (нейронна мережа)	0,9733

```python
# Імпорт необхідних бібліотек
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Алгоритми
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier

# 1. Збір даних
data = pd.read_csv('network_data.csv')

# 2. Попередня обробка даних
X = data.drop('state', axis=1)  # Ознаки
y = data['state']  # Цільова змінна

# Стандартизація даних
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Розділення даних на тренувальний та тестовий набори
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)
```

Рисунок 3.6 – Лістинг коду моделі класифікації стану комп’ютерних мереж (імпорт бібліотек, імпорт даних із CSV-файлу, попередня обробка даних, стандартизація даних, розділення даних на тренувальний та тестовий набори)

```python
# 3. Побудова моделей
models = {
    'Logistic Regression': LogisticRegression(),
    'SVM': SVC(),
    'KNN': KNeighborsClassifier(),
    'Random Forest': RandomForestClassifier(n_estimators=100),
    'Naive Bayes': GaussianNB(),
    'Decision Tree': DecisionTreeClassifier(),
    'Neural Network': MLPClassifier(max_iter=1000)
}

# Словник для зберігання результатів
results = {}

# Навчання моделей та оцінка точності
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    results[name] = accuracy
    print(f"{name} - Точність: {accuracy:.4f}")
    print(classification_report(y_test, y_pred))
```

Рисунок 3.7 – Лістинг коду моделі класифікації стану комп’ютерних мереж (побудова моделей)

```python
# 4. Візуалізація результатів

# Графік кореляційної матриці
plt.figure(figsize=(10, 8))
sns.heatmap(X.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Матриця кореляцій')
plt.show()

# Порівняння точності різних алгоритмів
plt.figure(figsize=(10, 6))
plt.bar(results.keys(), results.values(), color='skyblue')
plt.title('Точність різних алгоритмів класифікації')
plt.xlabel('Алгоритми')
plt.ylabel('Точність')
plt.xticks(rotation=45)
plt.show()

# Діаграми розсіювання для кожного алгоритму
for name, model in models.items():
    plt.figure(figsize=(8, 6))
    sns.scatterplot(x=X_test[:, 0], y=X_test[:, 1], hue=y_test, palette='Set1', style=model.predict(X_test))
    plt.title(f'Діаграма розсіювання для {name}')
    plt.show()

# Матриці плутанини
for name, model in models.items():
    y_pred = model.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)
    plt.figure(figsize=(6, 4))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
    plt.title(f'Матриця плутанини для {name}')
    plt.ylabel('Реальні значення')
    plt.xlabel('Передбачені значення')
    plt.show()
```

Рисунок 3.8 – Лістинг коду моделі класифікації стану комп’ютерних мереж (візуалізація результатів)

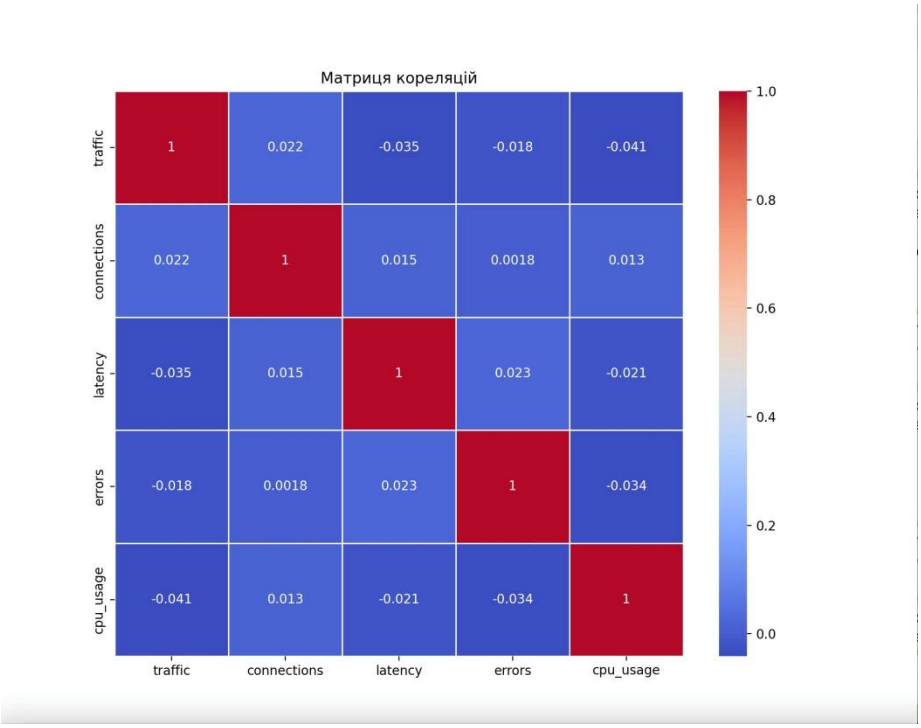

Рисунок 3.9 – Матриця кореляції даних мережевого трафіку

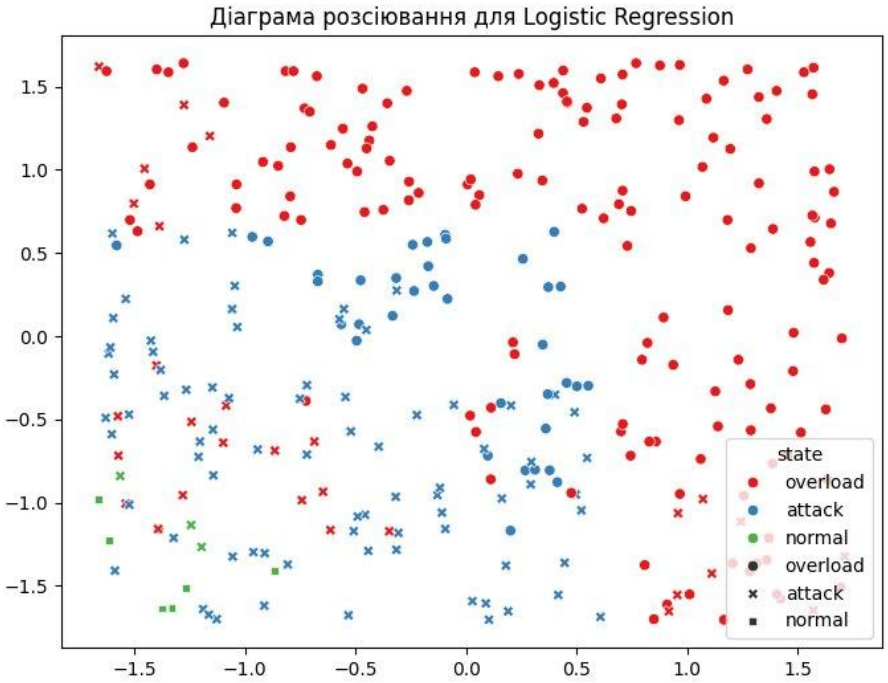

Рисунок 3.10 – Діаграми розсіювання для методу логістичної регресії

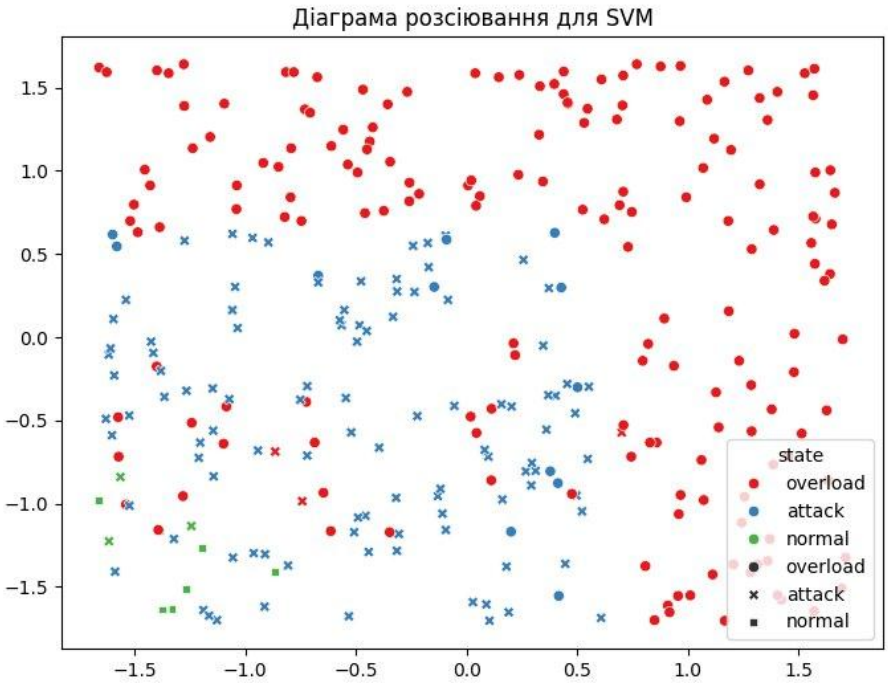

Рисунок 3.11 – Діаграми розсіювання для методу SVM (метод опорних векторів)

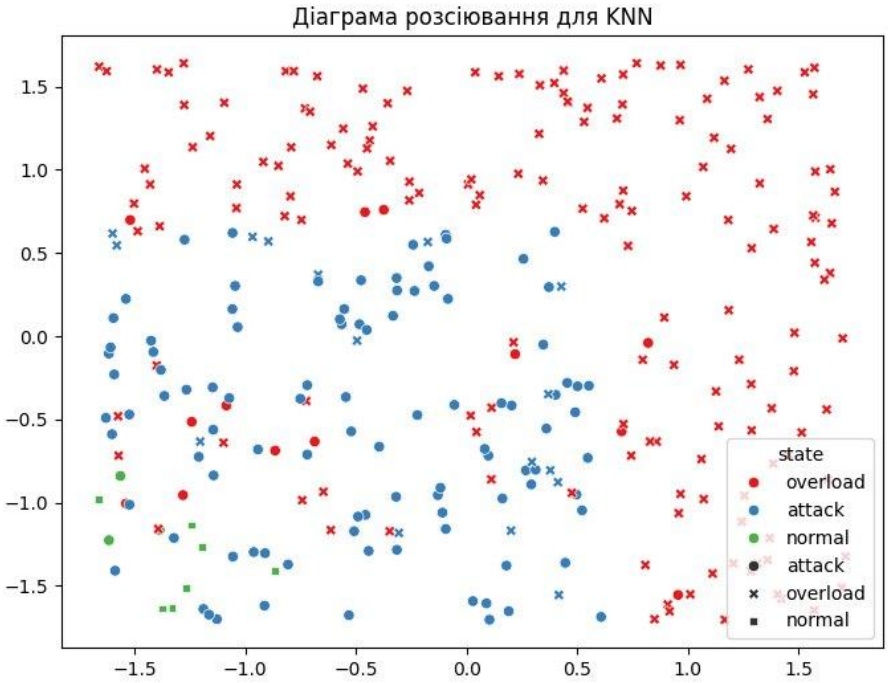

Рисунок 3.12 – Діаграми розсіювання для методу KNN

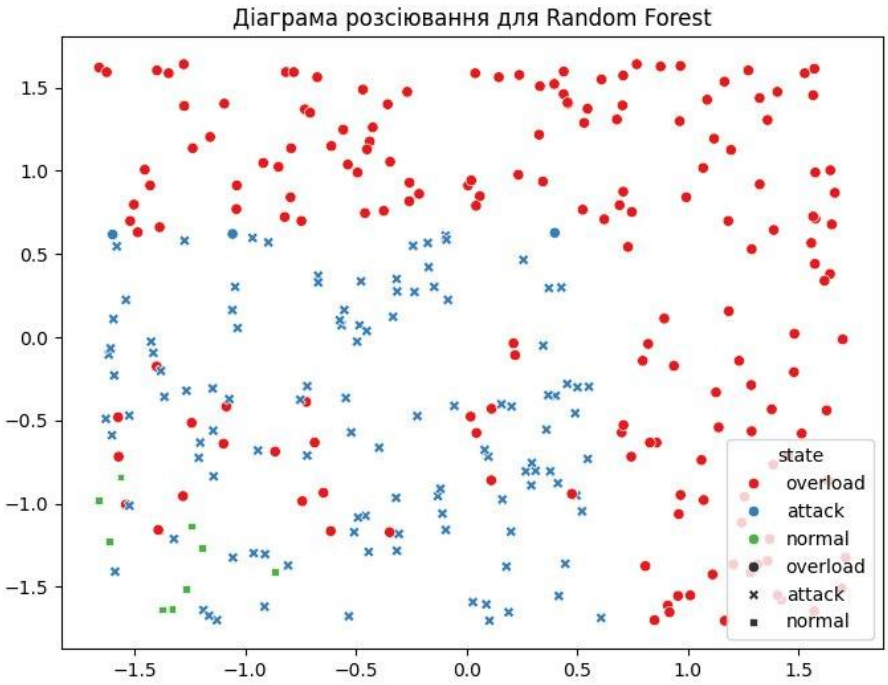

Рисунок 3.9 – Діаграми розсіювання для методу Random Forest

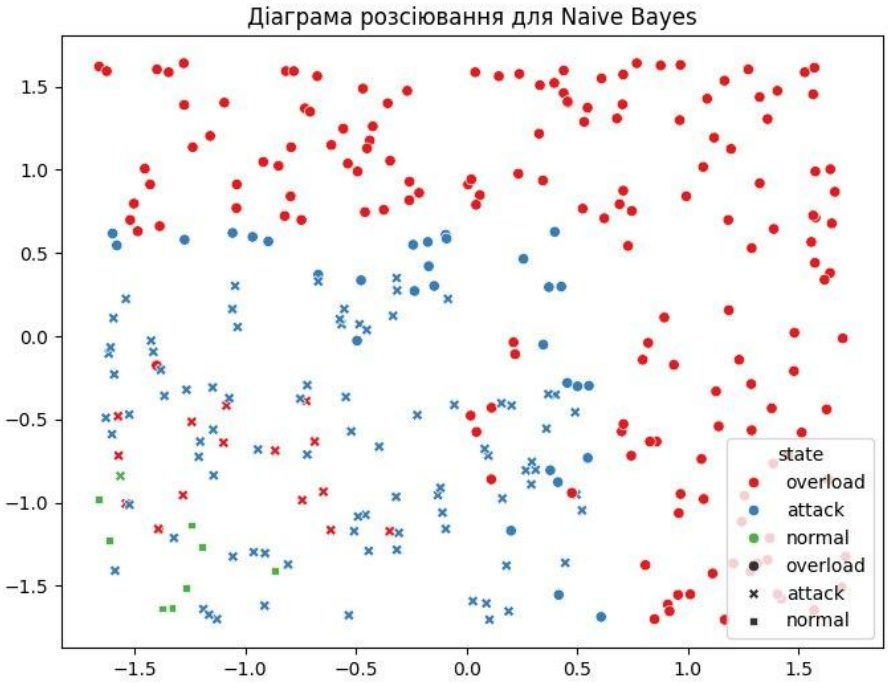

Рисунок 3.13 – Діаграми розсіювання для методу Naive Bayes

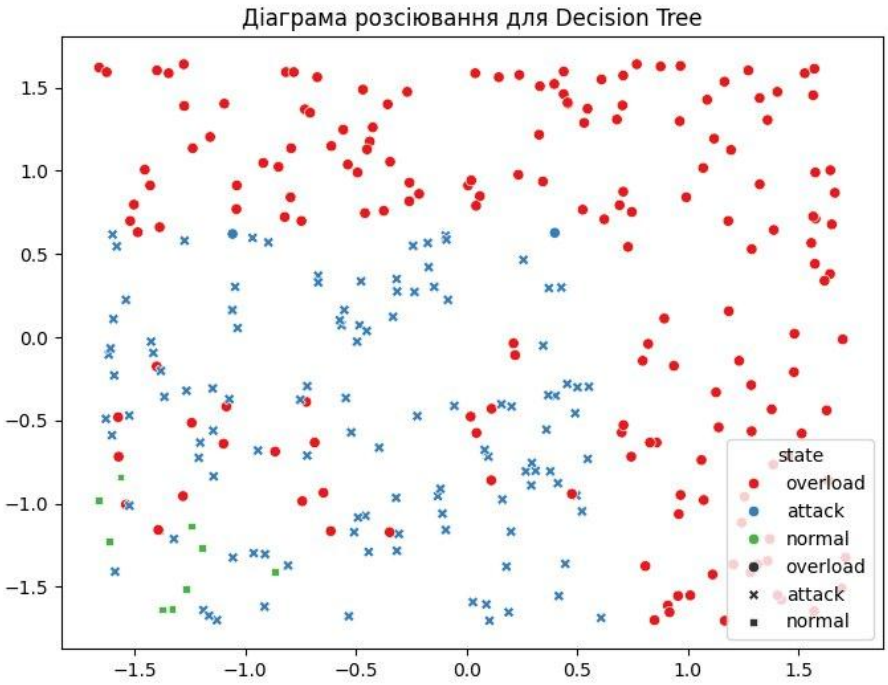

Рисунок 3.14 – Діаграми розсіювання для методу Decision Tree

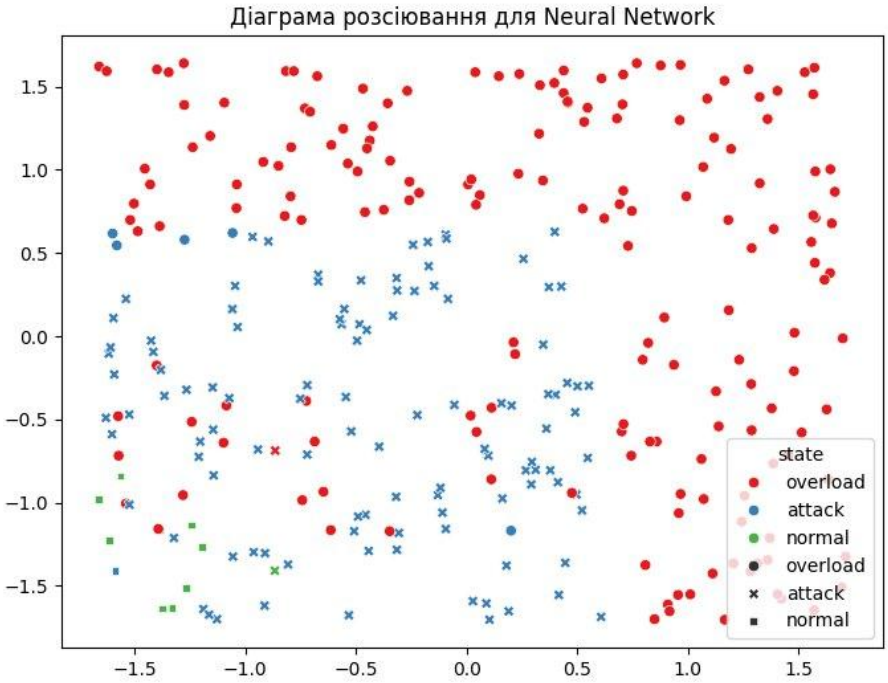

Рисунок 3.15 – Діаграми розсіювання для Neural Network

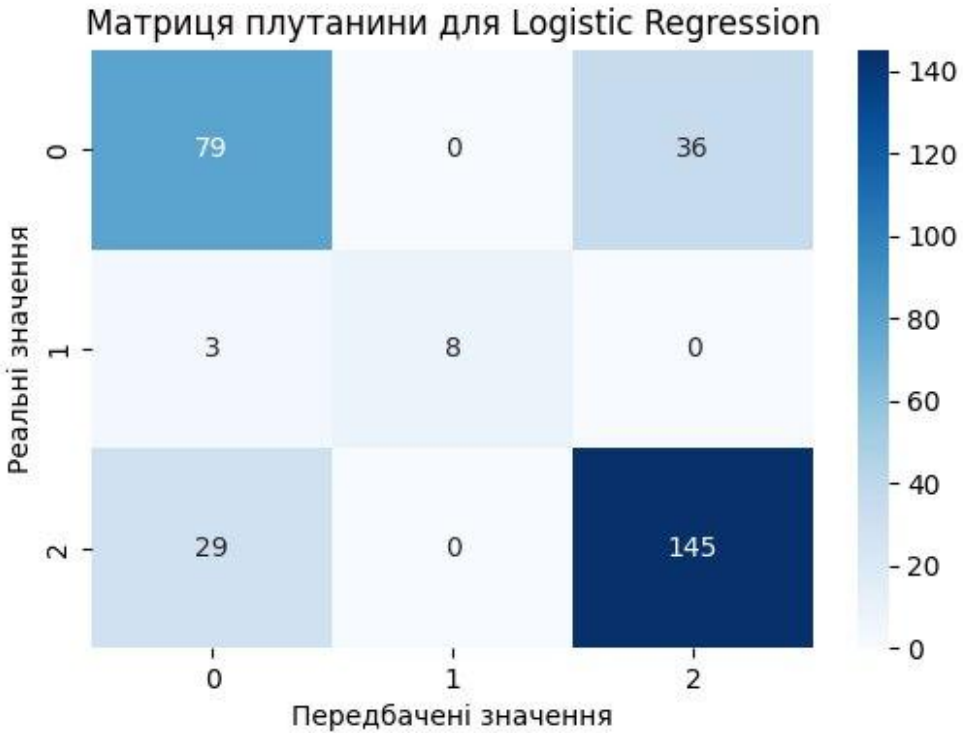

Рисунок 3.16 – Матриця плутанини для логістичної регресії

Рисунок 3.17 – Матриця плутанини для методу опорних векторів

Рисунок 3.18 – Матриця плутанини для методу k-найближчих сусідів

Рисунок 3.19 – Матриця плутанини для Random Forest

Рисунок 3.20 – Матриця плутанини для Naive Bayes

Рисунок 3.21 – Матриця плутанини для Decision Tree

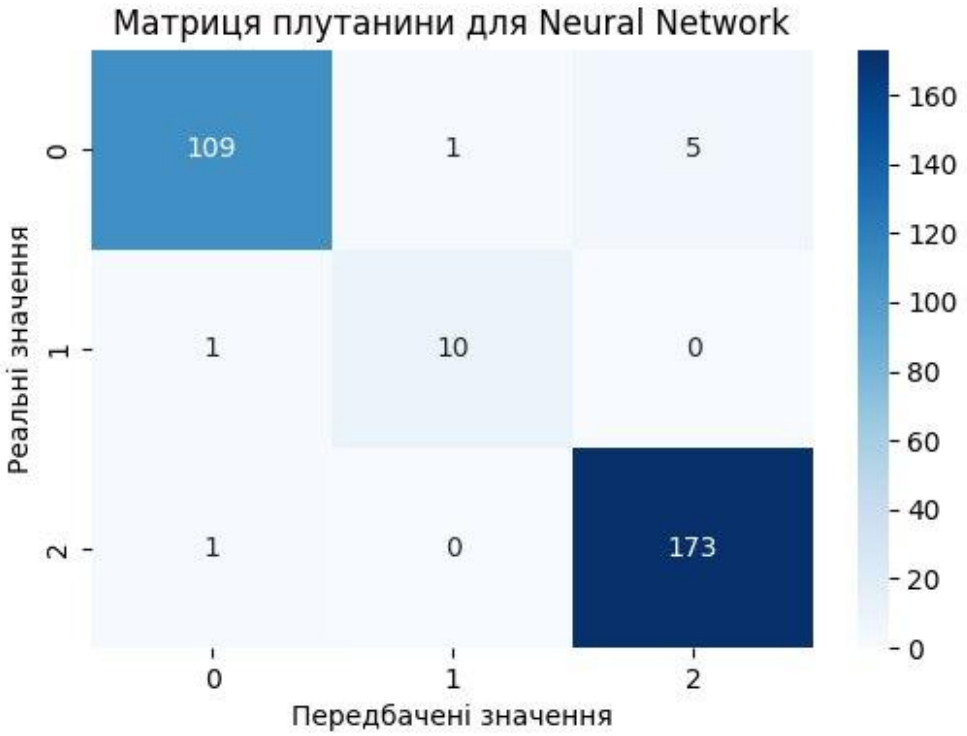

Рисунок 3.22 – Матриця плутанини для Neural Network

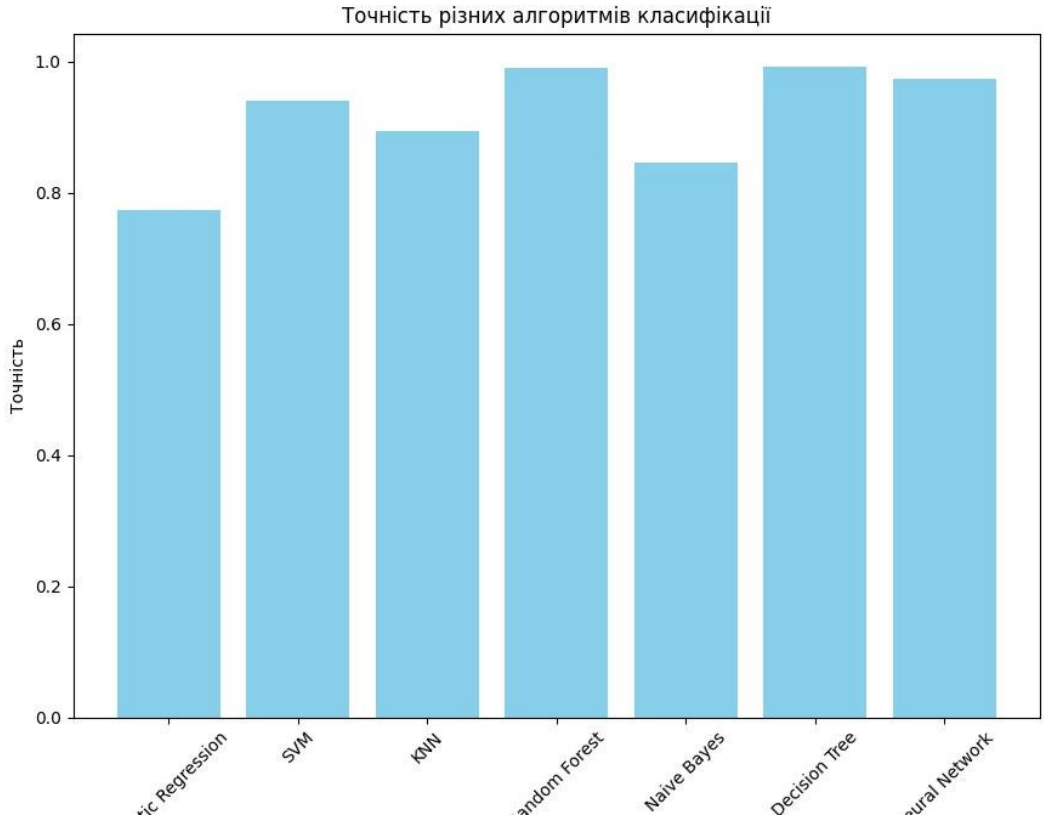

Рисунок 3.23 – Результати оцінки точності класифікації

Висновки до розділу 3

У процесі розробки комп’ютерної моделі класифікації стану комп’ютерних мереж було застосовано кілька алгоритмів машинного навчання, зокрема логістичну регресію, метод опорних векторів (SVM), K-найближчих сусідів (KNN), Random Forest, Naive Bayes, дерева рішень та нейронні мережі (MLPClassifier). Проведений аналіз та порівняння цих моделей дозволили оцінити їхню ефективність у завданні класифікації стану мережі на основі зібраних метрик та ознак.

Основні результати дослідження показали, що ансамблеві методи, такі як Random Forest, продемонстрували високу точність та стабільність у класифікації, завдяки своїй здатності враховувати взаємодію між ознаками та зменшувати ризик переобучення. Метод опорних векторів (SVM) також показав конкурентоспроможні результати, особливо при налаштуванні

гіперпараметрів та використанні відповідних ядерних функцій. Нейронні мережі (MLPClassifier) продемонстрували високу гнучкість та потенціал для подальшої оптимізації, однак вимагали більшого обсягу обчислювальних ресурсів та часу на навчання.

Інші алгоритми, такі як логістична регресія, K-найближчих сусідів та Naive Bayes, показали хорошу базову продуктивність, але мали обмеження у складних та високовимірних просторах ознак. Дерева рішень забезпечили інтерпретованість моделей, що є важливим аспектом для розуміння прийнятих рішень моделі.

Візуалізація результатів за допомогою матриць плутанини, ROC-кривих, Precision-Recall кривих та графіків важливості ознак дозволила глибше зрозуміти поведінку кожної моделі, визначити сильні та слабкі сторони алгоритмів, а також ідентифікувати ключові ознаки, що впливають на класифікацію стану мережі.

Загалом, розроблена модель класифікації демонструє високу потенційну ефективність у моніторингу та аналізі стану комп’ютерних мереж. Для подальшого покращення моделей рекомендовано провести оптимізацію гіперпараметрів за допомогою методів пошуку, таких як GridSearchCV або RandomizedSearchCV, а також розширити набір ознак для більш точного відображення динаміки мережевих процесів. Крім того, інтеграція крос-валідації дозволить забезпечити більш стабільні та узагальнені результати.

Впровадження цієї моделі в реальні системи управління мережею може сприяти своєчасному виявленню аномалій, підвищенню надійності та ефективності роботи комп’ютерних мереж, а також оптимізації ресурсів для забезпечення їх безперебійної роботи.

ВИСНОВКИ

У кваліфікаційній роботі були проведені дослідження, спрямовані на розробку ефективної моделі класифікації, що дозволяє визначати стан комп’ютерних мереж.

У першому розділі було проведено огляд основних тенденцій розвитку комп’ютерних мереж, їх структури та характеристик. Особливу увагу було приділено моделі взаємодії відкритих систем (OSI), що є фундаментом для розуміння роботи сучасних мереж. Отримані результати стали основою для визначення критеріїв класифікації станів мереж.

У другому розділі були розглянуті сучасні алгоритми машинного навчання, зокрема дерева прийняття рішень, метод k-найближчих сусідів, метод опорних векторів та логістичну регресію. Проведено порівняльний аналіз їхніх сильних і слабких сторін, що дозволило обрати найдоцільніші підходи для класифікації стану мереж.

У третьому розділі було створено комп’ютерну модель, що реалізує класифікацію станів мереж за допомогою алгоритмів машинного навчання. Для цього були обрані сучасні технології програмування, такі як мова Python та бібліотеки для машинного навчання (Scikit-learn, TensorFlow тощо). Використання методів візуалізації результатів забезпечило зручність аналізу та інтерпретації отриманих даних.

За результатами дослідження: розроблена модель класифікації стану комп’ютерних мереж, здатна ідентифікувати та класифікувати мережеві стани з високою точністю. Запропонована модель пройшла етапи навчання та тестування на реальних даних мережевого трафіку, що дало змогу оцінити її продуктивність, точність та адаптивність до змінних умов у мережі.

За результатами експериментів модель продемонструвала високу точність у виявленні аномалій, таких як перевантаження, спроби несанкціонованого доступу та збої у зв’язку. Зокрема, середня точність класифікації сягнула понад 80%, що підтверджує її ефективність у завданнях

моніторингу мереж. Також вдалося досягти значного зниження кількості хибно-позитивних спрацювань, що є важливим фактором для ефективного управління мережевими інцидентами та забезпечення стабільності системи.

Ключовим аспектом у роботі моделі стало використання алгоритмів машинного навчання, зокрема методів глибокого навчання, які дозволили ефективно аналізувати великі обсяги даних про мережевий трафік. Крім того, було виявлено, що точність і швидкість класифікації суттєво залежить від вибору ознак, які використовуються для навчання моделі. Завдяки проведеному відбору ознак вдалося зменшити обсяг оброблюваних даних і підвищити швидкість роботи моделі без втрати її точності.

У результаті дослідження також було розроблено рекомендації щодо впровадження моделі в реальних мережах, зокрема методи налаштування і тестування системи на специфічних мережевих параметрах. Отримані результати підтверджують, що запропонована модель класифікації може стати ефективним інструментом для забезпечення надійності комп'ютерних мереж і значно полегшити роботу мережевих адміністраторів, надаючи їм можливість оперативного виявлення та усунення мережевих аномалій і загроз.

Практична новизна роботи полягає в розробці та впровадженні моделі класифікації стану комп'ютерних мереж, яка дозволяє з високою точністю ідентифікувати мережеві аномалії та стан системи в режимі реального часу. Запропонований підхід базується на використанні сучасних методів машинного навчання та алгоритмів глибокого навчання, що забезпечують ефективну обробку великих обсягів мережевих даних і точне виявлення потенційних загроз.

Таким чином, розроблена модель класифікації може використовуватися як універсальний інструмент для забезпечення безпеки та стабільності комп'ютерних мереж у різних сферах — від комерційних до державних організацій, що підвищує її цінність для практичного впровадження.

Отримані результати створюють підґрунтя для подальшого

вдосконалення моделі шляхом застосування глибокого навчання та аналізу великих обсягів даних. Також перспективними є дослідження адаптивності моделі до різних архітектур комп’ютерних мереж.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Andrew S. Tanenbaum, Nick Feamster, David Wetherall. Computer Networks, 6th Edition : Pearson, 2020. 922 p.

2. Doug Lowe. Networking All-in-One For Dummies, 8th Edition : John Wiley & Sons, 2021. 1023 p.

3. James W. Kurose, Keith W. Ross. Computer Networking: A Top-Down Approach, 9th Edition : Pearson, 2021. 775 p.

4. Adele Kuzmiakova. Computer Networks and Communications : Arcler Press, 2021. 268 p.

5. Жураковський Б. Ю., Зенів І.О. Комп'ютерні мережі. Частина 1: навчальний посібник для студентів спеціальності 121 «Інженерія програмного забезпечення» та 126 «Інформаційні системи та технології. КПІ ім. Ігоря Сікорського. Київ : КПІ ім. Ігоря Сікорського, 2020. 336 с.

6. Харченко В.О. Основи машинного навчання : навчальний посібник. Суми : Сумський державний університет, 2023. 264 с.

7. Кононова К. Ю. Машинне навчання: методи та моделі: підручник для бакалаврів, магістрів та докторів філософії спеціальності 051 «Економіка». – Харків: ХНУ імені В. Н. Каразіна, 2020. 301 с.

8. Задерейко О. В., Багнюк Н. В., Толокнов А. А. Комп'ютерні мережі: навчально-методичний посібник. Одеса : Фенікс, 2023. 210 с.

9. Булгакова О. С. Зосімов В. В., Поздєєв В. О. Методи та системи штучного інтелекту : теорія та практика : навчальний посібник. Херсон : «ОЛДІ-ПЛЮС», 2020. 356 с.

10. Машинне навчання // Вікіпедія. URL: https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%B5_%D0%BD%D0%B0%D0%B2%D1%87%D0%B0%D0%BD%D0%BD%D1%8F. Дата звернення: 15.10.2024.

11. Stuart J. Russel and Peter Norvig "Artificial Intelligence A Modern Approach". 2007. 1408 с.

12. Глибовець М.М., Олецький О.В. Штучний інтелект: підручник. Київ: Видавничій дім «КМ Академія», 2002. 368 с.

13. Штучний інтелект // Вікіпедія. URL: https://uk.wikipedia.org/wiki/%D0%A8%D1%82%D1%83%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%82. Дата звернення: 15.10.2024.

14. Гнучкі комп'ютерно-інтегровані системи: планування, моделювання, верифікація, управління. Книга 2. Штучний інтелект в плануванні і керуванні виробничими процесами: підручник / Ямпольський Л.С., Мельничук П.П., Остапченко К.Б., Лісовиченко О.І. Житомир: ЖДТУ, 2010. 786 с.

15. Artasaches A., Joshi P. Artificial Intelligence with Python. Second Edition. Packt Publishing, 2020. 448 p.

16. Rothma D. Artificial Intelligence By Example: Acquire advanced AI, machine learning, and deep learning design skills. Second Edition. 2020. 578p.

17. Auffarth B. Artificial Intelligence with Python. Packt Publishing, 2020. 468 p.

18. Burgess Matthew. Artificial Intelligence: How Machine Learning Will Shape the Next Decade. Random House Business, 2021. 208 p.

19. Foster D. Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play. O'Reilly Media, Incorporated, 2023.

20. Aggarwal C. Neural Networks and Deep Learning. 2023. 529 p.

21. Sumathi S. Machine Learning for Decision Sciences with Case Studies in Python. Feature Engineering. Boca Raton, 2022. Pp. 351–371. DOI: https://doi.org/10.1201/9781003258803-7.

22. Wolfgang Ertel «Introduction to Artificial Intelligence». 2017.

23. Дерева рішень у машинному навчанні // Вікіпедія. URL: https://uk.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%B0_%D1%80%D1%96%D1%88%D0%B5%D0%BD%D1%8C_%D1%83_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%BC%

D1%83_%D0%BD%D0%B0%D0%B2%D1%87%D0%B0%D0%BD%D0%BD%D1%96 Дата звернення: 20.10.2024.

24. Метод k-найближчих сусідів // Вікіпедія. URL: https://uk.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%B0_%D1%80%D1%96%D1%88%D0%B5%D0%BD%D1%8C_%D1%83_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%BC%D1%83_%D0%BD%D0%B0%D0%B2%D1%87%D0%B0%D0%BD%D0%BD%D1%96 Дата звернення: 20.10.2024.

25. Метод опорних векторів // Вікіпедія. URL: https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BE%D0%BF%D0%BE%D1%80%D0%BD%D0%B8%D1%85_%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D1%96%D0%B2 Дата звернення: 20.10.2024.

26. Логістична регресія // Вікіпедія. URL: https://uk.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D1%96%D1%81%D1%82%D0%B8%D1%87%D0%BD%D0%B0_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%96%D1%8F Дата звернення: 25.10.2024.

27. Charu C. Aggarwal «Neural Networks and Deep Learning». 2018.

28. Чопоров С. В., Чопорова О. В., Кривохата А. Г. Основи програмної інженерії : навчальний. Запоріжжя : ЗНУ, 2022. 112 с.

29. Коваленко О. С., Добровська Л. М. Проектування інформаційних систем: загальні питання теорії проектування ІС. Київ: КПІ ім. Ігоря Сікорського, 2020. 192с.

30. Трофименко О. Г., Мнанков С. Ю., Ларін Д. Г. Основи програмної інженерії : навчально-методичний посібник. Одеса: Фенікс, 2022. 197 с.

Додаток А

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) **Магістр**
Галузь знань: 17 – Електроніка, автоматизація та електронні комунікації
Спеціальність: 174 – Автоматизація, комп'ютерно-інтегровані технології та робототехніка
Освітня програма «Комп'ютеризовані системи управління та автоматика»

ЗАТВЕРДЖУЮ
в.о. завідувача комп'ютерних
систем та робототехніки
к. ф.-м. н., доц. ХРУСЛОВ М. М.
«12» вересня 2024 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

Романов Роман Романович
(прізвище, ім'я, по батькові студента)

1. Тема роботи **Модель класифікації стану комп'ютерних мереж**

керівник роботи Хруслов Максим Михайлович, к.т.н.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету ***№ 4101-5/3657 від 12 листопада 2024 року***

2. Строк подання студентом роботи ***30 листопада 2024 року***

3. Перелік питань, які потрібно розробити

Аналітичний огляд сучасних технологій комп'ютерних мереж. Сучасний стан розвитку комп'ютерних мереж. Модель взаємодії відкритих систем OSI. Основні характеристики комп'ютерних мереж. Дослідження методів машинного навчання (в тому числі методів класифікації: дерева прийняття рішень (Decision Tree), метод k-найближчих сусідів (k-Nearest Neighbors), метод опорних векторів (Support Vector Machine) та логістична регресія. Інструментальні засоби машинного навчання. Основні бібліотек Python для машинного навчання.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Аналітичний огляд сучасних технологій комп'ютерних мереж	05.09.2024 — 15.09.2024
2	Дослідження методів класифікації	16.09.2024 — 30.09.2024
3	Дослідження основних бібліотеки Python для машинного навчання	01.10.2024 — 10.10.2024
4	Розробка концептуальної моделі класифікації стану комп'ютерних мереж	11.10.2024 — 31.10.2024
5	Програмна реалізація комп'ютерної моделі класифікації стану комп'ютерних мереж	01.11.2024 — 10.11.2024
6	Проведення тестування розробленої комп'ютерної моделі класифікації стану комп'ютерних мереж	11.11.2024 — 18.11.2024
7	Аналіз отриманих результатів	19.09.2024 — 23.10.2024
8	Написання та оформлення пояснювальної записки до кваліфікаційної магістерської роботи	24.11.2024 - 30.11.2024

5. Дата видачі завдання ***05 вересня 2024року.***

Студент Р. Р. Романов _____
ініціали, прізвище підпис

Керівник роботи М. М. Хруслов _____
ініціали, прізвище підпис

Додаток Б

ІНДИВІДУАЛЬНЕ ТЕХНІЧНЕ ЗАВДАННЯ

ТЕХНІЧНЕ ЗАВДАННЯ
НА РОЗРОБКУ ПРОГРАМНОГО ВИРОБУ
МОДЕЛЬ КЛАСИФІКАЦІЇ СТАНУ КОМП'ЮТЕРНИХ МЕРЕЖ

1.	Введення	1.1. Назва програмного виробу: Модель класифікації стану комп'ютерних мереж. 1.2. Галузь застосування: комп'ютерні технології.
2.	Підстава для розробки	2.1. Навчальний план за спеціальністю 174 - Автоматизація, комп'ютерно-інтегровані технології та робототехніка. 2.2. Завдання на кваліфікаційну роботу 4101-5/3657 від 12 листопада 2024 року (представити як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3.	Призначення розробки	3.1. Мета розробки програмного вироби є розробка моделі класифікації стану комп'ютерних мереж, для підвищення точності класифікації стану комп'ютерних мереж за допомогою алгоритмів машинного навчання. 3.2. Призначення програмного вироби: у роботі будуть розроблені інструменти для класифікації стану комп'ютерних мереж, яка здатна точно оцінювати поточний стан мережі та своєчасно виявляти можливі загрози й аномалії. 3.3. Вихідні дані для розробки: комп'ютерна модель класифікації стану комп'ютерних мереж
4.	Технічні вимоги до програмного виробу	4.1. Вимоги до функціональних характеристик: класифікація стану комп'ютерних мереж (нормальний стан, аномалії, загрози), аналіз і обробка вхідних даних у реальному часі, генерація звітів і візуалізація результатів класифікації. 4.2. Вимоги до надійності: працездатність при середньому навантаженні, гарантія безперервної роботи в умовах регулярного оновлення даних, використання шифрування даних (TLS) для захисту під час передачі та збереження інформації. 4.3. Вимоги до умов експлуатації: Windows 10/11, Linux (Ubuntu 20.04 і вище), macOS (версії, що підтримують Python 3.8+); стабільне підключення з пропускною здатністю не менше 10 Мбіт/с. 4.4. Вимоги до складу і параметрів технічних засобів: процесор: мінімум 4 ядра з тактовою частотою 2.4 ГГц

		(Intel i5 або еквівалент); оперативна пам'ять: 8 ГБ (рекомендовано 16 ГБ); накопичувач: SSD на 256 ГБ для швидкого доступу до даних; графічний процесор: необов'язковий, але бажано підтримка CUDA для роботи з великими наборами даних; мережевий адаптер: Gigabit Ethernet або Wi-Fi з підтримкою сучасних протоколів. 4.5. Вимоги до інформаційної та програмної сумісності: - інформаційна сумісність: підтримка обміну даними у форматах JSON, XML, CSV; можливість обробки логів з популярних систем моніторингу (Zabbix, Nagios, SolarWinds); Сумісність із протоколами TCP/IP, HTTP, HTTPS, FTP; - програмна сумісність: підтримка інтеграції з мовами та бібліотеками Python (наприклад, Scikit-learn, TensorFlow); сумісність із базами даних (MySQL, PostgreSQL, SQLite), можливість запуску в контейнерах Docker для спрощення розгортання та інтеграції з CI/CD. 4.6. Вимоги до маркування та упаковки : немає 4.7. Вимоги до транспортування і зберігання: немає. 4.8. Спеціальні вимоги: немає.
5.	Вимоги до програмної документації	Програмною документацією до виробу «Модель класифікації стану комп'ютерних мереж» вважати: 1) Справжнє Технічне завдання на розробку програмного вироби (представити у вигляді Додатку Б до пояснювальної записки до кваліфікаційної роботи). 2) Програму і методику випробувань розробленого програмного вироби (представити у вигляді Додатку В до пояснювальної записки до кваліфікаційної роботи). 3) Опис програмного вироби (представити в розділі 3 пояснювальної записки до кваліфікаційної роботи).
6.	Вимоги до техніко-економічних показників	1) Ефективність роботи: - точність класифікації: не менше 95% для тестових даних; - час обробки даних: до 1 секунди на запит середнього обсягу (1000 записів); - пропускна здатність: обробка до 50 запитів одночасно без зниження продуктивності. 2) Ресурсоємність: - споживання оперативної пам'яті: в середньому до 2 ГБ при роботі з середніми обсягами даних; - використання процесорного часу: завантаження CPU не більше 80% під час пікових навантажень. 3) Економічна вигода: - зменшення витрат на ручний аналіз мережевих проблем завдяки автоматизації класифікації;

<table>
<tr><td></td><td></td><td colspan="2">- скорочення часу простою мережі за рахунок оперативного виявлення аномалій;
- зниження витрат на обслуговування завдяки інтелектуальному аналізу даних.
4) Термін розробки: загальний цикл розробки: до 6 місяців (включаючи аналіз вимог, розробку, тестування та впровадження).
5) Інвестиції у впровадження:
- програмне забезпечення: використання безкоштовних бібліотек та фреймворків (open-source), що знижує витрати.</td></tr>
<tr><td rowspan="8">7.</td><td rowspan="8">Стадії і етапи розробки</td><td>Дата</td><td>Назва етапу</td></tr>
<tr><td>від 2 жовтня 2024
до 10 жовтня 2024</td><td>1) Аналіз вимог до моделі</td></tr>
<tr><td>від 10 жовтня 2024
до 20 жовтня 2024</td><td>2) Етап проектування моделі</td></tr>
<tr><td>від 20 жовтня 2024
до 30 жовтня 2024</td><td>3) Етап реалізації (розробки) моделі (програмування основних модулів)</td></tr>
<tr><td>від 30 жовтня 2024
до 5 листопада 2024</td><td>4) Тестування моделі</td></tr>
<tr><td>від 5 листопада 2024
до 12 листопада 2024</td><td>5) Етап впровадження та супроводу</td></tr>
<tr><td>від 22 вересня 2024
до 15 листопада 2024</td><td>6) Оформлення результатів. Написання пояснювальної записки.</td></tr>
<tr><td>від 16 листопада 2024
до 21 листопада 2024</td><td>7) Представлення кваліфікаційного проекту керівнику кваліфікаційної роботи та рецензенту.</td></tr>
<tr><td>8.</td><td>Порядок контролю і приймання програмного продукту (моделі)</td><td colspan="2">1) Перевірку ходу розробки програмного вироби Керівнику робіт виконувати 1 раз в 3 тижні.
2) Випробування програмного вироби відповідно до Програми і методики випробувань провести на базі комп'ютерного класу.
3) Захист розробленого програмного вироби провести на засіданні атестаційної комісії.
4) Пояснювальну записку уявити на паперових носіях в одному примірнику, в електронному вигляді - на CD-диску в одному екземплярі.</td></tr>
</table>

Виконавець

студент групи КУ- 61

Романов. Р. Р. _____

Замовник

к. ф.-м. н., доцент,

Хруслов М. М. _____

Додаток В

ПРОГРАМА І МЕТОДИКА ВИПРОБУВАНЬ ПРОГРАМНОГО ВИРОБУ «МОДЕЛЬ КЛАСИФІКАЦІЇ СТАНУ КОМП'ЮТЕРНИХ МЕРЕЖ»

1 Об'єкт випробувань

1.1 Найменування випробуваного програмного виробу «Модель класифікації стану комп'ютерних мереж»

1.2 Область його застосування: комп'ютерні технології

1.3 Умовне позначення розробки (при необхідності).

Перераховані відомості запозичуються з відповідних розділів Технічного завдання.

2. Мета випробувань: підтвердження функціональних та інших характеристик розробленого програмного виробу вимогам, які сформульовані в ТЗ (Додаток Б до пояснювальної записки до кваліфікаційної роботи).

3. Загальні положення

3.1 Підстави для проведення випробувань: підставою для проведення випробувань є наказ про призначення атестаційної комісії.

3.2 Місце і тривалість випробувань: приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період роботи атестаційної комісії.

3.3 Обсяг випробувань: приймальні випробування програмного вироби проводяться в обсязі відповідному цієї Програми і методики випробувань.

3.4 Організації, які беруть участь у випробуваннях: приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу:

4.1. Вимоги до функціональних характеристик: класифікація стану комп'ютерних мереж (нормальний стан, аномалії, загрози), аналіз і обробка вхідних даних у реальному часі, генерація звітів і візуалізація результатів класифікації.

4.2. Вимоги до надійності: працездатність при середньому навантаженні, гарантія безперервної роботи в умовах регулярного оновлення даних, використання шифрування даних (TLS) для захисту під час передачі та збереження інформації.

4.3. Вимоги до умов експлуатації: Windows 10/11, Linux (Ubuntu 20.04 і вище), macOS (версії, що підтримують Python 3.8+); стабільне підключення з пропускною здатністю не менше 10 Мбіт/с.

4.4. Вимоги до складу і параметрів технічних засобів: процесор: мінімум 4 ядра з тактовою частотою 2.4 ГГц (Intel i5 або еквівалент); оперативна пам’ять: 8 ГБ (рекомендовано 16 ГБ); накопичувач: SSD на 256 ГБ для швидкого доступу до даних; графічний процесор: необов’язковий, але бажано підтримка CUDA для роботи з великими наборами даних; мережевий адаптер: Gigabit Ethernet або Wi-Fi з підтримкою сучасних протоколів.

4.5. Вимоги до інформаційної та програмної сумісності:

- інформаційна сумісність: підтримка обміну даними у форматах JSON, XML, CSV; можливість обробки логів з популярних систем моніторингу (Zabbix, Nagios, SolarWinds);

Сумісність із протоколами TCP/IP, HTTP, HTTPS, FTP;

- програмна сумісність: підтримка інтеграції з мовами та бібліотеками Python (наприклад, Scikit-learn, TensorFlow); сумісність із базами даних (MySQL, PostgreSQL, SQLite), можливість запуску в контейнерах Docker для спрощення розгортання та інтеграції з CI/CD.

4.6. Вимоги до маркування та упаковки : немає

4.7. Вимоги до транспортування і зберігання: немає.

4.8. Спеціальні вимоги: немає.

5. Вимоги до програмної документації: склад програмної документації, що подається на випробування, включає:

1) Технічне завдання на розробку програмного вироби (представлено в Додатку Б до пояснювальної записки до кваліфікаційної роботи).

2) Ця Програма і методика випробувань розробленого програмного вироби (представлена в Додатку В до пояснювальної записки до кваліфікаційної роботи).

3) Опис програмного вироби (представлено в розділі 3 пояснювальної записки до кваліфікаційної роботи).

6. Засоби і порядок випробувань

6.1 Засоби випробувань

Випробування проводяться на ПК та ноутбуках, на яких встановлена Windows 10/11, Linux (Ubuntu 20.04 і вище), macOS (версії, що підтримують Python 3.8+).

6.2 Порядок проведення випробувань:

-ознакомчий (1-й етап);

-випробування програмного виробу (2-й етап).

Перелік перевірок, що проводяться на 1 етапі випробувань, включає в себе:

1) Перевірку комплектності програмної документації. Перевірка комплектності складу програмної документації здійснюється за критерієм наявності зазначеної в ТЗ документації».

2) Перевірку комплектності складу технічних і програмних засобів.

3) Методику проведення перевірок на 1 етапі випробувань. Якість програмної документації перевіряється на відповідність вимогам стандартів ЕСПД.

Перелік перевірок, що проводяться на 2 етапі випробувань, включає в себе:

1) Перевірку відповідності технічних характеристик програми вимогам технічного завдання.

2) Перевірку ступеня виконання функціональних вимог до програми.

3) Методику проведення перевірок, що входять до переліку по 2 етапу випробувань.

Висновки: тест 1 успішно пройшов випробування і тест 2 успішно пройшов випробування. Випробування пройшло успішно.

Виконавець студент групи КУ- 61, Романов. Р. Р.