Міністерство освіти і науки України

Харківський національний університет ім. В. Н. Каразіна

Факультет комп'ютерних наук

Спеціальність 125 «Кібербезпека»

Освітня програма «Кібербезпека»

«Допущено до захисту»

В. о. завідувача кафедри БІСТ

Мелкозьорова О. М.

_____

« » 2024р.

Пояснювальна записка

до кваліфікаційної роботи бакалавра

на тему: «Аналіз та дослідження використання штучного інтелекту в сфері кібербезпеки: переваги та недоліки»

оцінка « »

Голова ЕК

Лемешко О. В. _____

Керівник ст. викладач Шеханін К. Ю.

Рецензент к.т.н. Олешко О. І.

Виконавець: студентка групи КБ-41

Главічка С.В.

Харків – 2024

РЕФЕРАТ

Дипломна робота включає 60 сторінок, 5 розділів, 5 рисунків, 4 таблиць, 1 додаток, та посилається на 20 джерел.

Мета дипломної роботи полягає у вивченні можливостей штучного інтелекту (ШІ) у вирішенні задач кібербезпеки, зокрема виявлення та нейтралізації фішингових листів. Дослідження ґрунтується на аналізі сучасних методів машинного навчання та глибокого навчання, з використанням програмного забезпечення для тестування алгоритмів в реальних умовах.

Результати роботи показали, що впровадження ШІ в системи кібербезпеки не тільки покращує швидкість і точність виявлення кіберзагроз, але й дозволяє виявляти складніші і раніше невідомі атаки. Особлива увага у дослідження приділялась використанню ШІ для захисту інформаційної безпеки, з огляду на ймовірні ризики порушення конфіденційності та можливі зловживання технологіями.

У роботі пропонується використання розроблених методик і технологій для подальшого розвитку інструментів кібербезпеки, зокрема, розробку нових алгоритмів ШІ, які можуть адаптуватись до нових загроз і забезпечити більш стійкий захист систем. Також згадується необхідність подальших досліджень в області балансу між ефективністю ШІ та забезпеченням конфіденційності користувачів.

Дипломна робота вносить значний вклад у розуміння потенціалу ШІ в кібербезпеці та вказує на важливість продовження досліджень в цій галузі для забезпечення адаптації до змінюваних умов кіберзагроз.

Ключові слова: ШТУЧНИЙ ІНТЕЛЕКТ, КІБЕРБЕЗПЕКА, МАШИННЕ НАВЧАННЯ, ГЛИБОКЕ НАВЧАННЯ, ФІШИНГ, АЛГОРИТМИ ВИЯВЛЕННЯ, ПРОТИДІЯ КІБЕРЗАГРОЗАМ.

ABSTRACT

The thesis comprises 60 pages, 5 sections, 5 figures, 4 tables, 1 appendix, and references 20 sources.

The aim of this thesis is to explore the capabilities of artificial intelligence (AI) in addressing cybersecurity challenges, particularly in the detection and neutralization of phishing emails. The research is based on the analysis of modern machine learning and deep learning methods, using software to test algorithms in real-world conditions.

The results of the study demonstrated that the implementation of AI in cybersecurity systems not only increases the speed and accuracy of detecting cyber threats but also enables the identification of more complex and previously unknown attacks. The research pays special attention to aspects of using AI for cybersecurity, considering potential risks to privacy and possibilities for technology misuse.

The work proposes the use of the developed methodologies and technologies for the further development of cybersecurity tools, particularly the creation of new AI algorithms that can adapt to new threats and provide more robust system protection. It also emphasizes the need for further research in balancing AI efficiency with user privacy.

This thesis makes a significant contribution to the understanding of AI's potential in cybersecurity and highlights the importance of continuing research in this field to ensure adaptation to evolving cyber threat conditions.

Keywords: ARTIFICIAL INTELLIGENCE, CYBERSECURITY, MACHINE LEARNING, DEEP LEARNING, PHISHING, DETECTION ALGORITHMS, COUNTERING CYBER THREATS.

ЗМІСТ

ПЕРЕЛІК ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

IT - інформаційні технології.

ІШІ - штучний інтелект.

ML - машинне навчання (англ. Machine Learning).

DL - глибоке навчання (англ. Deep Learning).

JSON - JavaScript Object Notation

SQL - Structured Query Language

CSV - comma-separated values

UBA - User Behavior Analytics

TAP - Threat Analytics Platform

NN - Neural Network

LR - Logistic Regression

kNN - k-nearest neighbor

SMO - Sequential Minimal Optimization

URL - Uniform Resource Locator

ВСТУП

Інформаційна безпека стала більш важливою, оскільки суспільство стало залежним від технологій. З початку цифрової ери кількість та складність кіберзагроз зросли, що зробило складнішис для організацій ефективне виявлення та реагування на них. Штучний інтелект (ШІ) виступив як перспективна технологія для підвищення результатів інформаційної безпеки, посилюючи людські можливості та прийняття рішень.

Ця дипломна робота описує роль ШІ в інформаційній безпеці. На мою думку ШІ може покращувати результати безпеки різними способами, такими як виявлення аномалій, автоматизація реакцій на загрози та надання реальних відомостей про події безпеки. Ця дипломна робота досліджує ключові концепції, представлені та надає глибокий аналіз способів, якими ШІ може бути використаний для покращення результатів інформаційної безпеки. Ми обговорюємо виклики, з якими стикаються професіонали інформаційної безпеки, переваги використання ШІ в інформаційній безпеці, різні типи алгоритмів ШІ, які використовуються в інформаційній безпеці. Також обговорюються майбутні напрямків для ШІ в інформаційній безпеці та важливість продовження досліджень та розробок у цій галузі для випередження еволюції загроз в цифрову епоху.

Інформаційна безпека — це практика захисту комп'ютерних систем та мереж від несанкціонованого доступу, крадіжки, пошкодження або збою. Оскільки організації стають більш залежними від технологій для управління своєю діяльністю, захист конфіденційних даних стає важливішим. У сучасну еру витоки даних можуть завдати значної шкоди репутації організації, її фінансовій стабільності та довірі клієнтів. Кіберзлочинці постійно

вдосконалюють свої навички, техніки та стратегії, що ускладнює для професіоналів інформаційної безпеки виявляти нові небезпеки.

Традиційно, інформаційна безпека покладалася на системи на основі правил, які використовують попередньо визначені правила для ідентифікації та реагування на загрози безпеки. Однак, ці системи мають обмеження, включаючи їхню нездатність адаптуватися до нових і еволюціонуючих загроз, час, необхідний для ручного оновлення правил, та неможливість обробки великого обсягу даних, що генерується в сьогоднішніх складних IT- середовищах. Це призвело до зростаючого інтересу в ШІ як рішенні для підвищення результатів інформаційної безпеки.

ШІ має потенціал революціонізувати галузь інформаційної безпеки, швидко обробляючи великі обсяги даних, виявляючи аномалії та автоматизуючи реакції на загрози безпеки. ШІ може бути застосований для посилення людських можливостей та прийняття рішень, дозволяючи організаціям ефективніше реагувати на інциденти безпеки. У дипломній роботі ми розглянемо переваги використання ШІ в інформаційній безпеці та різні типи алгоритмів ШІ, які можна використовувати у цій галузі.

ШІ виступив як нова технологія для підвищення результатів інформаційної безпеки. Зі збільшенням складності кіберзагроз, використання традиційних систем безпеки на основі правил стало недостатнім. ШІ пропонує більш передовий та адаптивний підхід до інформаційної безпеки, дозволяючи організаціям виявляти та реагувати на загрози ефективніше.

Однією з ключових переваг ШІ в інформаційній безпеці є його здатність швидко обробляти великі обсяги даних. Алгоритми ШІ можуть аналізувати величезні обсяги даних у реальному часі, виявляючи аномалії та незвичайну активність, що може свідчити про порушення безпеки. ШІ також може автоматизувати реакції на інциденти безпеки, дозволяючи організаціям реагувати на інциденти швидше та ефективніше.

Ще однією перевагою ШІ в інформаційній безпеці є його здатність вчитися та адаптуватися з часом. Алгоритми машинного навчання можуть бути навчені за допомогою попередніх даних для покращення точності виявлення загроз та реагування. Коли з'являються нові загрози, алгоритми ШІ швидко адаптуються та оновлюють свої моделі, щоб залишатися ефективними.

Крім покращення можливостей виявлення та реагування, ШІ також може надавати реальні відомості про події безпеки. Це дозволяє організаціям постійно моніторити свій стан безпеки та виявляти потенційні слабкі місця у своїх системах. ШІ також може допомогти організаціям ідентифікувати тенденції та закономірності в інцидентах безпеки, дозволяючи їм впроваджувати більш ефективні заходи безпеки.

Використання ШІ у галузі інформаційної безпеки стає більш важливим, оскільки організації прагнуть захистити свої критичні дані та системи від кіберзагроз. ШІ забезпечує більш передовий та адаптивний підхід до інформаційної безпеки, дозволяючи організаціям виявляти та реагувати на загрози ефективніше.

1. ЗАГАЛЬНИЙ ОГЛЯД ШТУЧНОГО ІНТЕЛЕКТУ ТА КІБЕРБЕЗПЕКИ

1.1 Вплив та застосування технологій штучного інтелекту на кібербезпеку.

Штучний інтелект може оптимізувати виявлення загроз та створювати передові рішення в галузі кібербезпеки, які реагуватимуть на зміни у сучасних загрозах. Ці системи ШІ є самонавчальними і можуть застосовувати аналіз у непередбачуваних ситуаціях та приймати рішення на основі своїх спостережень.

ШІ може допомогти виявляти та прогнозувати загрози наступними способами:

− Аналіз великих даних у реальному часі: активний моніторинг мережі загроз вимагає обробки величезних масивів структурованих і неструктурованих даних. Без ШІ цей процес вимагав би значних часових та людських ресурсів. ШІ може автоматизувати моніторинг загроз, зменшуючи людські помилки та підвищуючи ефективність виявлення.

− Виявлення незвичайної активності: виявлення ризиків є важливим для прогностичного ШІ у кібербезпеці. ШІ може аналізувати зміни в мережі та використовувати ці шаблони для прогнозування. Він може використовувати аналітику поведінки для виявлення незвичайної активності всередині та за межами вашої системи.

− Прогнозування потенційних векторів атаки: методи або шляхи, якими нападники намагаються отримати доступ до системи або мережі, називаються векторами атаки. ШІ відіграє важливу роль у прогнозуванні векторів атак на основі історичних даних. Він використовує техніки машинного навчання для аналізу історичних даних за шаблонами та аномаліями, що можуть допомогти прогнозувати майбутні вектори атак. Алгоритми ШІ можуть аналізувати величезні масиви даних і виявляти приховані зв'язки в подіях, які не можуть бути виявлені неозброєним оком. Чим більше даних ви надаєте, тим більше він навчається, тому ШІ з часом покращує свої прогностичні здібності.

Нові технології впливають на роботу в різних галузях. Кібербезпека може розраховувати на появу більш потужних технологій для вирішення викликів у ландшафті, який, ймовірно, буде розвиватися швидше, ніж раніше. Нова ера виявлення та реагування на загрози наступає, і вона може повністю змінити спосіб, яким організації реагують на кіберзагрози. Штучний інтелект має потенціал значно зменшити навантаження на аналітиків, підвищуючи продуктивність і вивільняючи їх для зосередження на стратегічному плануванні та інших критично важливих стратегіях. Штучний інтелект може допомогти нам перейти від реактивного до проактивного, дозволяючи організаціям набирати обертів і випереджати безліч кіберзагроз. Оскільки штучний інтелект та інші технології кібербезпеки розвиваються, так само розвиваються і кіберзагрози — кіберзлочинці можуть використовувати потужність ШІ для досягнення своїх цілей. Приватність та етика стануть більш важливими, оскільки системи ШІ стають більш інтегрованими в кібербезпеку, роблячи ШІ одночасно викликом і можливістю.

З огляду на це, винагороди значно перевищують ризики. Оскільки ШІ може аналізувати дані та прогнозувати майбутні загрози, він також може використовувати технології обману, такі як мережі ханіпотів, використовуючи свої прогнози для відволікання кіберзлочинців від їх справжньої мети. Крім того, покращене виявлення загроз, реагування в реальному часі, більш точний аналіз поведінки та адаптивність - лише кілька майбутніх тенденцій, на які варто звернути увагу. Знаходження балансу між захоплюючими можливостями штучного інтелекту та його етичними міркуваннями буде важливішим, ніж коли-небудь.

2. МОЖЛИВОСТІ ШТУЧНОГО ІНТЕЛЕКТУ В КІБЕРБЕЗПЕЦІ

2.1 Виявлення загроз

2.1.1 Традиційні методи виявлення загроз

Кібербезпека довгий час покладалася на традиційні методи, такі як виявлення за підписами та за правилами, для ідентифікації загроз. Ці методи добре служили нам, але оскільки нападники стають більш витонченими, а загрози еволюціонують, ці підходи виявляють свої обмеження. Виявлення за підписами має труднощі з новими атаками, тоді як методи на основі правил часто призводять до хибних спрацювань та пропусків складних загроз. Операції, ефективність і обмеження коротко описані в Таблиці 2.1.

Таблиця 2.1 - Операції, ефективність і обмеження традиційних методів виявлення загроз

Метод	Опис	Операція	Ефективність	Обмеження
Виявлення за підписами	Засновано на попередньо визначених зразках відомих загроз. Ці підписи, зазвичай, походять від індикаторів компрометації (IOC), такихяк хеші файлів, шаблони	Системи безпеки, такі як антивірусне програмне забезпечення або системи виявлення вторгнень (IDS), порівнюють дані/трафік із відомими підписами. Якщо збіг знайдено,	Ефективно проти відомих загроз.	Може мати труднощі з виявленням нових або раніше не спостережу-ваних атак, оскільки нападники можуть легко

Продовження таб. 2.1

	мережевого трафіку або поведінкові характеристики , асоційовані з відомими шкідливими програмами або техніками атак	система спрацьовує сигнал тривоги або вживає передбачені дії для блокування або зменшення загрози.		уникнути виявлення, модифіку-ючи свої тактики, техніки та процедури (TTP) або використову-ючи поліморфні варіанти шкідливих програм.
Виявлення за правилами	Використовує попередньо визначені правила або евристики для ідентифікації підозрілих дій або поведінок, що вказують на безпекову загрозу. Ці правила, зазвичай, базуються на	Системи безпеки аналізують вхідні дані або мережевий трафік у реальному часі, застосовуючи попередньо визначені правила для ідентифікації потенційно шкідливої діяльності. Якщо	Ефектив-но для Виявле-ння відомих шаблонів атак або поши-рених безпеко-вих вразливо-стей.	Може генерувати хибні спрацьовува-ння або пропускати складні атаки, які не відповідають попередньо визначеним правилам,

Продовження таблиці 2.1

	відомих шаблонах атак, експлойтах вразливостей або аномальних порогах	збіг знайдено, система спрацьовує сигнал тривоги або вживає відповідні дії, наприклад, блокування підозрілої діяльності або повідомлення безпековому персоналу для подальшого розслідування.		роблячи це менш ефективним проти нових або складних загроз.

У той час як методи виявлення за підписами та за правилами були основою в кібербезпеці протягом багатьох років, вони мають обмеження при виявленні невідомих або загроз "нульового дня". Оскільки кіберзагрози стають все більш складними та динамічними, організації звертаються до більш передових і адаптивних підходів, таких як виявлення загроз на основі ШІ, щоб доповнити традиційні методи та покращити свої кібербезпекові захисти.

2.1.2. Виявлення загроз на основі ШІ

ШІ має унікальну здатність аналізувати величезні обсяги даних з великою швидкістю, значно перевищуючи можливості людських аналітиків. Ця майстерність ґрунтується на передових алгоритмах і обчислювальній потужності, які лежать в основі систем ШІ, дозволяючи їм ефективно аналізувати великі набори даних, що включають різні джерела безпекової телеметрії. Ось

коротке пояснення того, як ШІ може аналізувати дані для швидкого виявлення аномалій та потенційних загроз:

- *Передові алгоритми*: Системи ШІ використовують складні алгоритми, включно з машинним навчанням (ML) та глибоким навчанням (DL), для обробки та аналізу великих обсягів даних. Ці алгоритми розроблені для навчання на основі шаблонів даних, адаптації до нової інформації та прийняття прогнозів або рішень на основі отриманих з даних уявлень.
- *Паралельна обробка*: Системи ШІ можуть здійснювати паралельну обробку, аналізуючи кілька потоків даних одночасно. На відміну від людських аналітиків, які обмежені когнітивною спроможністю та тривалістю уваги, ШІ може обробляти масивні набори даних паралельно, значно скорочуючи час, необхідний для аналізу.
- *Аналіз у реальному часі*: Системи ШІ можуть виконувати аналіз потокових даних, таких як журнали мережевого трафіку та системні журнали подій, в момент їх генерації. Це дозволяє організаціям виявляти та реагувати на безпекові загрози в реальному часі, мінімізуючи вплив потенційних кібератак.
- *Розпізнавання шаблонів*: ШІ відмінно справляється з ідентифікацією шаблонів, аномалій та відхилень у даних. Аналізуючи історичні дані та навчаючись на минулих інцидентах, ШІ може впізнавати аномальні поведінки або діяльності, які можуть вказувати на потенційну безпекову загрозу. Ця здатність до розпізнавання шаблонів дозволяє ШІ швидко та точно відзначати підозрілі події.
- *Автоматизація:* Автоматизація на основі ШІ спрощує процес аналізу даних, усуваючи необхідність ручного втручання на кожному кроці. Одного разу навчені, системи ШІ можуть автономно аналізувати дані, виявляти аномалії та ініціювати сповіщення або реагування на основі попередньо визначених критеріїв. Ця автоматизація значно прискорює

процес виявлення та реагування на загрози, дозволяючи організаціям швидко реагувати на нові кіберзагрози.

– *Масштабованість*: Системи ШІ високо масштабовані та здатні обробляти більші та складніші набори даних без втрати продуктивності. ШІ може масштабуватися горизонтально зі зростанням обсягів даних, розподіляючи обчислення між кількома процесорами або вузлами, забезпечуючи послідовний та ефективний аналіз навіть при збільшенні навантаження на дані.

Підсумуємо види технік машинного та глибокого навчання у таблиці 2.2.

Таблиця 2.2. Підсумки видів технік машинного та глибокого навчання

Техніка	Опис	Застосування в виявленні загроз
Навчання з учителем (ML)	Навчання на маркованих даних (зловмисні/безпечні) для точної класифікації нових даних.	Ідентифікація відомих загроз та шкідливих програм.
Навчання без учителя (ML)	Виявлення зразків у немаркованих даних для виявлення аномалій	Виявлення невідомих загроз та незвичайної поведінки.
Навчання з підкріпленням (ML)	Навчання методом проб та помилок, оптимізація рішень.	Адаптація стратегій реагування до еволюції загроз.

Продовження таб. 2.2

Згорткові нейронні мережі (DL)	Аналіз структурованих даних, як-от зображення та часові ряди.	Ідентифікація зловмисних зразків у мережевому трафіку або бінарних файлах шкідливих програм.
Рекурентні нейронні мережі (DL)	Обробка послідовних даних, таких як дані журналів або поведінка системи	Виявлення аномалій у поведінці системи з часом.
Генеративні конкурентні мережі (DL)	Генерування синтетичних даних або адверсарних прикладів для тестування.	Збагачення наборів даних для навчання та тестування міцності системи.

Також розглянемо приклади успішного впровадження технологій ШІ для виявлення загроз у таблиці 2.3.

Таблиця 2.3 Приклади успішного впровадження виявлення загроз з використанням ШІ

Випадок використання	Приклад	Опис
Виявлення аномалій	Darktrace	Використовує ШІ для виявлення відхилень від нормального мережевого трафіку, поведінки користувачів та системних журналів. Ідентифікує потенційні загрози, як-от витік даних,

Продовження таб. 2.3

		внутрішні загрози та інфекції шкідливими програмами в реальному часі.
Аналіз поведінки	Splunk User Behavior Analytics (UBA)	Використовує ШІ для аналізу діяльності користувачів у IT- системах та додатках. Виявляє підозрілі дії, як-от внутрішні загрози, зловживання посвідченнями ідентифікації та несанкціоновані спроби доступу.
Прогностичний аналітика	FireEye Threat Analytics Platform (TAP)	Використовує ШІ для аналізу історичних даних про атаки та прогнозування майбутніх загроз. Ідентифікує нові вектори атак і допомагає організаціям проактивно захищатися від них.

Ці приклади ілюструють ефективність впровадження виявлення загроз з використанням ШІ у різних випадках використання, включаючи виявлення аномалій, аналіз поведінки та прогностичний аналітика. Використовуючи потужність алгоритмів ШІ, організації можуть покращувати свою кібербезпеку, виявляти загрози в реальному часі та швидко реагувати для зменшення впливу кібератак.

2.2 Аналіз вразливостей

2.2.1 Традиційні підходи до аналізу вразливостей

Традиційні методи аналізу вразливостей включають ручний огляд та оцінку різних компонентів IT-інфраструктури організації, включаючи програмний код, конфігурації системи та архітектуру мережі. Фахівці з безпеки зазвичай покладаються на свій досвід та експертні знання для ідентифікації потенційних слабкостей безпеки, які можуть використовувати зловмисники. Цей процес часто включає:

− *Ручний огляд коду*: Експерти з безпеки вручну переглядають програмний код, щоб виявити вразливості, такі як переповнення буфера, вразливості до ін'єкцій або небезпечні механізми автентифікації. Це передбачає перегляд коду построково, шукаючи помилки в кодуванні або конструктивні недоліки, які можуть призвести до порушень безпеки.

−*Перегляд конфігурацій системи*: Фахівці з безпеки перевіряють конфігурації системи, включаючи налаштування операційної системи, конфігурації мережі та контроль доступу, щоб виявити неправильні налаштування або слабкості, які можуть відкрити організацію для кіберзагроз. Це передбачає порівняння поточної конфігурації з найкращими практиками безпеки та стандартами галузі, щоб виявити проблемні зони.

−*Оцінка інфраструктури мережі:* Команди безпеки аналізують мережеву інфраструктуру організації, включаючи маршрутизатори, комутатори, файрволи та інші мережеві пристрої, щоб виявити потенційні уразливості або неправильні налаштування, які можуть бути використані нападниками. Це може включати проведення мережевих сканувань, перегляд правил файрволу та аналіз мережевих трафіків для ідентифікації аномальної поведінки.

Хоча традиційні методи ефективні для виявлення відомих вразливостей та загальних слабкостей безпеки, вони мають обмеження:

− *Трудомісткість:* Ручний аналіз уразливостей вимагає значни людських зусиль і часу, потребуючи глибоких знань та експертизи для перегляду великих обсягів коду, конфігурацій та мережевих даних.

− *Обмежена масштабованість:* Традиційні методи можуть виявитися неефективними для масштабування в великих та складних IT-середовищах, особливо в організаціях з обширними проектами розробки програмного забезпечення або розподіленою мережевою інфраструктурою.

− *Обмежене охоплення:* Традиційні підходи можуть не враховувати нові або виникаючі загрози, які не відповідають попередньо визначеним сигнатурам або шаблонам, залишаючи організації вразливими до нульових днів або постійних загроз.

− *Людська помилка*: Ручний аналіз вразливостей схильний до людської помилки, коли фахівці з безпеки можуть пропустити критичні вразливості або неправильні налаштування через втому, недогляд або брак досвіду.

− *Труднощі у відстеженні загроз*: Швидке розвиток кіберзагроз та технік атак ускладнює традиційним методам аналізу вразливостей відповідати змінам у ландшафті загроз. Регулярно виявляються нові вразливості, що вимагає постійної пильності та проактивних заходів для вирішення виникаючих ризиків безпеки.

Ці традиційні методи добре нам служать, але еволюція кіберпейзажу вимагає більшого. Оскільки організації стикаються з все більш складними та динамічними кіберзагрозами, зростає потреба в більш передових та автоматизованих підходах до аналізу уразливостей, таких як ті, що підтримуються ШІ та алгоритмами машинного навчання.

2.2.2 Сканування вразливостей на базі ШІ

ШІ може революціонізувати сканування вразливостей і віддавати пріоритет критичним вразливостям, застосовуючи алгоритми машинного

навчання для аналізу обширних наборів даних. Сканери вразливостей, які працюють на ШІ, автоматизують процес ідентифікації потенційних слабкостей безпеки в програмному коді, конфігураціях системи та мережевих активах. Ці сканери використовують складні алгоритми для виявлення зразків, аномалій та історичних даних про атаки, дозволяючи їм виявляти уразливості, які раніше могли залишитися непоміченими.

Автоматизація сканування вразливостей за допомогою ШІ має кілька значних наслідків:

− *Ефективність*: Автоматизація сканування вразливостей системами на базі ШІ значно знижує час і зусилля, необхідні для ідентифікації та усунення слабкостей безпеки. Ця ефективність дозволяє організаціям проводити більш часті сканування та оцінки, забезпечуючи постійний моніторинг стану безпеки.

− *Пріоритезація*: Сканери вразливостей на базі ШІ можуть віддавати пріоритет вразливостям залежно від їхньої серйозності, можливості експлуатації та потенційного впливу на стан безпеки організації. Присвоюючи ризикові бали або рейтинги вразливостям, ці системи дозволяють командам безпеки зосередити свої зусилля на усуненні найбільш критичних загроз спершу, тим самим максимізуючи ефективність своїх заходів безпеки.

− *Розподіл ресурсів*: Надаючи пріоритет критичним вразливостям, системи на базі ШІ допомагають організаціям ефективніше розподіляти свої ресурси. Команди безпеки можуть сконцентрувати свою увагу та ресурси на усуненні високоризикових вразливостей, які становлять найбільшу загрозу для активів та даних організації, тим самим знижуючи загальний ризик успішної кібератаки.

− *Точність*: Алгоритми ШІ відмінно виявляють тонкі зразки та аномалії в складних наборах даних, дозволяючи їм точно ідентифікувати вразливості, які можуть бути проігноровані традиційними методами. Ця підвищена

точність знижує ймовірність помилкових позитивних та негативних результатів, забезпечуючи довіру команд безпеки до результатів, отриманих за допомогою сканерів вразливостей на базі ШІ.

− *Адаптивність*: Сканери вразливостей на базі ШІ можуть постійно навчатися та адаптуватися до нових загроз та технік атак, забезпечуючи захист організацій від еволюції кіберзагроз. Аналізуючи історичні дані про атаки та тенденції безпеки, ці системи можуть оновлювати свої моделі та алгоритми для виявлення раніше невідомих вразливостей та векторів атак, тим самим підвищуючи стійкість організації до нових ризиків безпеки.

3. ВИКЛИКИ, ПЕРЕВАГИ ТА МАЙБУТН ШТУЧНОГО ІНТЕЛЕКТУ В КІБЕРБЕЗПЕЦІ

3.1 Виклики в інформаційній безпеці

3.1.1 Еволюція характеру кіберзагроз

Одним із найбільших викликів у сфері інформаційної безпеки є еволюція характеру кіберзагроз. Кіберзлочинці постійно розробляють нові та складні техніки для порушення систем безпеки, що ускладнює завдання організаціям у їх відстеженні. Одним із прикладів цього є зростаюче використання штучного інтелекту та машинного навчання кіберзлочинцями. Атакуючі можуть використовувати алгоритми ШІ для автоматизації атак, роблячи їх більш ефективними та дієвими. Наприклад, вони можуть застосувати ШІ для генерації фішингових електронних листів, які персоналізовано створені для цільової аудиторії, збільшуючи імовірність того, що одержувач натисне на шкідливе посилання або завантажить заражений шкідливим програмним забезпеченням вкладення.

Інший виклик полягає в зростаючому використанні мобільних пристроїв та хмарних систем у робочому середовищі. Ці технології створюють нові вектори атак для кіберзлочинців, оскільки вони можуть бути неналежно захищені або вразливі до атак. Наприклад, мобільні пристрої можуть бути втрачені або вкрадені, забезпечуючи доступ до конфіденційних даних, тоді як хмарні системи можуть бути вразливі до порушень даних або несанкціонованого доступу.

Наостанок, зростаюча складність ІТ-середовищ також становить виклик для інформаційної безпеки. Оскільки організації впроваджують нові технології та розширюють свої мережі, стає все важче стежити та управляти безпекою по всій системі. Це може призвести до "сліпих зон" та вразливостей, які можуть бути використані кіберзлочинцями.

Для вирішення цих викликів професіонали в галузі інформаційної безпеки повинні бути в курсі останніх загроз та технологій, впроваджувати ефективні контрольні заходи та постійно моніторити та оновлювати свої системи. ШІ може відіграти важливу роль у цьому процесі, забезпечуючи можливості реального часу для виявлення загроз і реагування на них, а також надаючи інформацію про події безпеки та тенденції.

Виклики та їх опис навели у таблиці 3.1

Таблиця 3.1 – Короткий опис викликів в інформаційній безпеці

Виклик	Опис
Еволюційна природа кіберзагроз	Кіберзлочинці розробляють нові та складні техніки для порушення систем безпеки, такі як використання ШІ та машинного навчання для автоматизації атак та зростаюче використання мобільних пристроїв та хмарних систем як нових векторів атак.
Зростання складності IT-середовищ	Впровадження нових технологій та розширення мереж створюють складне IT-середовище з множиною платформ та операційних систем, сторонніми постачальниками, хмарними системами, мобільними пристроями та віддаленими робочими групами, що призводить до вразливостей, які можуть бути використані кіберзлочинцями.
Обмеження традиційних систем безпеки	Традиційні системи безпеки мають обмеження у пристосуванні до нових та

Продовження таб. 3.1

	нових загроз, обробці великих обсягів даних і виявленні більш складних атак, таких як ті, що використовують ШІ та машинне навчання. Професіонали інформаційної безпеки звертаються до ШІ та машинного навчання для підвищення ефективності систем безпеки та випередження постійно змінювального ландшафту загроз.

3.1.2 Зростання складності IT-середовищ

Інший виклик у сфері інформаційної безпеки - це зростаюча складність IT-середовищ. Оскільки організації впроваджують нові технології та розширюють свої мережі, IT-середовище стає складнішим і важче керованим. Ця складність може призвести до вразливостей, які можуть бути використані кіберзлочинцями. Одним з прикладів цього є використання кількох платформ і операційних систем у межах однієї організації. Це може створювати проблеми сумісності та ускладнювати підтримку послідовного контролю безпеки в усій системі. В результаті, вразливості можуть існувати в певних зонах, які не захищені належним чином.

Ще один приклад - це використання сторонніх постачальників та хмарних систем. Хоча ці технології можуть пропонувати такі переваги, як економія коштів та масштабованість, вони також вносять нові ризики до інформаційної безпеки. Наприклад, порушення даних у стороннього постачальника може призвести до витоку конфіденційних даних організації. Нарешті, зростаюче використання мобільних пристроїв і віддалених робочих сил також додає складності до IT-середовищ. Мобільні пристрої можуть бути неналежно

захищені або вразливі до атак, тоді як віддалені працівники можуть використовувати незахищені мережі для доступу до корпоративних систем, створюючи додаткові вразливості.

Для вирішення цих викликів, професіонали в галузі інформаційної безпеки повинні прийняти цілісний підхід до безпеки, який враховує все IT-середовище. Це включає в себе впровадження послідовних контрольних заходів безпеки на всіх платформах і операційних системах, проведення регулярних оцінок вразливостей та ретельне моніторинг сторонніх постачальників та хмарних систем. ШІ може відіграти ключову роль у цьому процесі, забезпечуючи реальні відомості про події безпеки та виявляючи потенційні вразливості, перш ніж вони будуть використані кіберзлочинцями.

3.1.3 Обмеження традиційних правилових систем безпеки

Традиційні правилові системи безпеки, які залежать від попередньо визначених правил для ідентифікації та реагування на загрози безпеки, мають кілька обмежень, що робить їх менш ефективними у складних сучасних IT-середовищах. Одним з обмежень є їх нездатність адаптуватися до нових і еволюційних загроз. Правилові системи ефективні лише у виявленні загроз, які відповідають встановленим правилам. З появою нових загроз правила потрібно вручну оновлювати, що може бути часомістким і може не встигати за темпом нових загроз.

Інше обмеження - обсяг даних, які потрібно обробляти. Традиційні системи безпеки можуть генерувати величезні обсяги даних, аналіз яких може бути складним і може призводити до хибних позитивних результатів або пропущених загроз.

Нарешті, правилові системи можуть не бути ефективними в розпізнаванні більш складних атак, таких як ті, що використовують ШІ та машинне навчання. Ці атаки можуть обходити традиційні системи безпеки, імітуючи звичайну поведінку користувача або використовуючи передові техніки уникнення

виявлення. Для вирішення цих обмежень професіонали в галузі інформаційної безпеки звертаються до ШІ та машинного навчання для підсилення своїх систем безпеки. Алгоритми ШІ можуть швидко обробляти величезні обсяги даних і виявляти візерунки, що можуть вказувати на загрозу безпеки. Алгоритми машинного навчання також можуть адаптуватися та вчитися з часом, покращуючи свою точність та ефективність у виявленні та реагуванні на нові та еволюційні загрози. Використовуючи ШІ та машинне навчання, організації можуть випереджати постійно змінювальний ландшафт загроз та краще захищати свої критичні дані та системи.

3.2 Переваги ШІ у інформаційній безпеці

3.2.1 Здатність швидко обробляти великі обсяги даних

Одна з ключових переваг ШІ в інформаційній безпеці полягає в його здатності швидко обробляти великі обсяги даних. Оскільки організації збирають все більше даних, стає все складніше ефективно їх аналізувати та моніторити. Алгоритми ШІ можуть аналізувати величезні обсяги даних у реальному часі, виявляючи аномалії та незвичайну активність, які можуть вказувати на порушення безпеки. Наприклад, алгоритми ШІ можуть аналізувати мережевий трафік для виявлення незвичайних візерунків поведінки, що може свідчити про кібератаку. Вони також можуть аналізувати системні журнали для ідентифікації незвичайної активності або ознак несанкціонованого доступу. Аналізуючи дані у реальному часі, ШІ може надавати організаціям попередження про потенційні інциденти безпеки, дозволяючи їм швидко та ефективно реагувати.

Крім того, ШІ також може допомогти організаціям аналізувати дані з різних джерел для ідентифікації тенденцій та візерунків у безпекових інцидентах. Це дозволяє організаціям впроваджувати більш ефективні контрольні заходи безпеки та виявляти потенційні слабкі місця в їхніх системах до того, як вони будуть використані кіберзлочинцями. Загалом, здатність ШІ швидко обробляти великі обсяги даних є значною перевагою в інформаційній

безпеці. Забезпечуючи реальний аналіз подій безпеки та виявляючи потенційні загрози, ШІ може допомогти організаціям залишатися на крок попереду еволюціонуючих кіберзагроз та краще захищати їх критичні дані та системи.

3.2.2 Виявлення аномалій та незвичайної активності

Ще одна перевага ШІ в інформаційній безпеці - його здатність виявляти аномалії та незвичайну активність, які можуть вказувати на порушення безпеки. Алгоритми ШІ можуть аналізувати дані з різних джерел, включаючи мережевий трафік, системні журнали та поведінку користувачів, для ідентифікації паттернів активності, які виходять за рамки звичного. Наприклад, ШІ може виявити незвичайні візерунки поведінки, які можуть вказувати на кібератаку, такі як передача великих обсягів даних до зовнішньої системи або незвичайні спроби входу. ШІ також може аналізувати поведінку користувачів для виявлення аномалій, які можуть свідчити про внутрішню загрозу, наприклад, коли співробітник отримує доступ до даних поза звичайними робочими годинами або доступ до даних, до яких зазвичай він не має права доступу.

Виявляючи аномалії та незвичайну активність, ШІ може забезпечити організаціям раннє попередження про потенційні інциденти безпеки, дозволяючи їм швидко та ефективно реагувати. Це може допомогти запобігти витоку даних і мінімізувати збитки від кібератак. Крім того, ШІ також може автоматизувати відповідь на інциденти безпеки, що дозволяє організаціям реагувати швидше і більш ефективно. Наприклад, ШІ може автоматично блокувати доступ до скомпрометованого облікового запису або системи, скорочуючи час, необхідний для локалізації інциденту безпеки. В цілому, здатність ШІ виявляти аномалії та незвичайну активність є значною перевагою в інформаційній безпеці. Забезпечуючи можливості реального часу для виявлення загроз та автоматизованої відповіді, ШІ може допомогти організаціям залишатися на крок попереду еволюціонуючих кіберзагроз та краще захищати їх критичні дані та системи.

3.2.3 Автоматизація реагування на загрози

Ще одна перевага ШІ в інформаційній безпеці - його здатність автоматизувати реагування на загрози. Алгоритми ШІ можуть бути запрограмовані для автоматичного реагування на інциденти безпеки, що дозволяє організаціям реагувати швидше і більш ефективно. Наприклад, якщо система ШІ виявляє спробу кібератаки, вона може автоматично блокувати доступ до скомпрометованої системи, запобігаючи подальшій шкоді. Вона також може надсилати сповіщення співробітникам безпеки, забезпечуючи їх реальною інформацією про інцидент і дозволяючи їм швидко реагувати. Крім того, ШІ може автоматизувати процес виправлення вразливостей і оновлення контрольних заходів безпеки. Це може допомогти скоротити час, необхідний для впровадження оновлень безпеки, і гарантувати, що системи завжди актуальні і захищені від останніх загроз.

Автоматизуючи відповідь на загрози, ШІ може допомогти організаціям швидше і ефективніше реагувати на інциденти безпеки, знижуючи ризик витоку даних і мінімізуючи збитки від кібератак. В цілому, автоматизація реагування на загрози є значною перевагою ШІ в інформаційній безпеці. Забезпечуючи можливості реального часу для виявлення загроз та автоматизованої відповіді, ШІ може допомогти організаціям залишатися на крок попереду еволюціонуючих кіберзагроз і краще захищати їх критичні дані та системи.

3.2.5 Реальний час аналітики безпекових подій

Ще одна перевага ШІ в інформаційній безпеці — його здатність надавати реальні відомості про події безпеки. Алгоритми ШІ можуть аналізувати великі обсяги даних у реальному часі, забезпечуючи організації негайними відомостями про інциденти безпеки та загрози. Наприклад, ШІ може аналізувати мережевий трафік для виявлення незвичайних візерунків поведінки, які можуть вказувати на кібератаку. Він також може аналізувати системні журнали для ідентифікації незвичайної активності або ознак несанкціонованого доступу. Надаючи реальні

відомості про події безпеки, ШІ дозволяє організаціям швидко та ефективно реагувати на потенційні загрози.

Крім того, ШІ також може надавати реальні відомості про тенденції та візерунки безпеки. Аналізуючи дані з різних джерел, включаючи мережевий трафік, системні журнали та поведінку користувачів, ШІ може виявляти тенденції та візерунки, які можуть вказувати на потенційні слабкі місця в контролях безпеки організації. Це дозволяє організаціям впроваджувати більш ефективні заходи контролю безпеки та проактивно вирішувати потенційні вразливості.

Загалом, здатність ШІ надавати реальні відомості про події безпеки є значною перевагою в інформаційній безпеці. Надаючи організаціям можливості реального часу для виявлення загроз та реагування на них, ШІ може допомогти організаціям залишатися на крок попереду еволюціонуючих кіберзагроз та краще захищати їхні критичні дані та системи.

3.3 Майбутнє систем безпеки на основі ШІ

Майбутнє ШІ в галузі безпеки, ймовірно, буде відзначене постійними інноваціями та розвитком, оскільки нові технології та методи розроблятимуться для покращення можливостей систем безпеки, що працюють на основі ШІ. Деякі потенційні прогнози щодо майбутнього ШІ в безпеці та його впливу включають:

− *Збільшення автоматизації*: Системи безпеки на основі ШІ, ймовірно, стануть більш автоматизованими, зменшуючи потребу в людському втручанні в процес безпеки. Це може призвести до більш ефективних та ефективних заходів безпеки, але також може викликати занепокоєння щодо прозорості та підзвітності.

− *Підвищена точність і надійність*: Оскільки алгоритми ШІ стають більш складними та навчаються на більших та більш різноманітних наборах даних, вони, ймовірно, стануть більш точними та надійними у своєму прийнятті рішень. Це може призвести до кращого виявлення та запобігання

загрозам, але також може викликати занепокоєння щодо потенційних помилок або упереджень.

− *Інтеграція з іншими технологіями*: Системи безпеки на основі ШІ, ймовірно, стануть все більш інтегрованими з іншими технологіями, такими як Інтернет речей (IoT) та хмарні обчислення. Це може дозволити реалізувати більш всеосяжні та проактивні заходи безпеки, але також може збільшити ризик кібератак та витоків даних.

− *Покращення конфіденційності та захисту даних*: Оскільки занепокоєння щодо конфіденційності та захисту даних продовжує зростати, системи безпеки на основі ШІ можуть бути розроблені так, щоб включати більш передові техніки збереження конфіденційності, такі як розподілене навчання або гомоморфне шифрування. Це може дозволити реалізувати більш ефективні заходи безпеки, мінімізуючи ризик витоку даних або порушень конфіденційності.

− *Посилена співпраця між індустрією та урядом*: Оскільки загрозовий ландшафт еволюціонує і стає більш складним, ймовірно, буде посилена співпраця між індустрією та урядом у розробці та впровадженні систем безпеки на основі ШІ. Це може дозволити краще виявлення та запобігання загрозам, але також може викликати занепокоєння щодо приватності та громадянських свобод.

Загалом, вплив ШІ на майбутнє безпеки, ймовірно, буде значним, із потенційними перевагами та ризиками для осіб, організацій та суспільства в цілому. Важливо уважно розглянути ці потенційні впливи та розробити етичні та правові рамки для керівництва розвитком та використанням цих технологій таким чином, щоб це відповідало етичним та правовим нормам.

4. МЕТОДИКИ РЕАЛІЗАЦІЇ ПРОГРАМИ СКАНУВАННЯ ФІШИНГОВИХ ЛИСТІВ

4.1 Машинне навчання та штучний інтелект в системах виявлення фішингу

Традиційні методи виявлення фішингу, такі як фільтрація спаму та аналіз сигнатур, часто не можуть ефективно розпізнавати фішингові атаки через зростання їхньої складності. Машинне навчання (МН) та штучний інтелект (ШІ) відкривають новий підхід до виявлення фішингу, використовуючи аналіз різних ознак, таких як:

- структура електронного листа або повідомлення;
- вміст електронного листа або повідомлення.
- URL-адреса посилання або вкладення;
- поведінка користувача.

Машинне навчання здатне виявляти фішингові атаки, аналізуючи великі обсяги даних, що включають як фішингові, так і нефішингові електронні листи та повідомлення. У нашій програмі для виявлення фішингу використовується наївний байєсовий класифікатор, який навчається на наборі даних електронних листів, щоб розпізнавати патерни, характерні для фішингу.

Переваги підходів до використання машинного навчання та штучного інтелекту для виявлення фішингу

- Точність та ефективність:

Машинне навчання та штучний інтелект можуть бути більш точними у виявленні фішингу, ніж традиційні методи. У нашій програмі використання наївного байєсового класифікатора дозволяє ефективно розпізнавати фішингові атаки, аналізуючи структуру та вміст електронних листів. Наприклад, модель може навчитися виявляти характерні помилки в граматиці або орфографії, а також підозрілі заклики до дії.

- Гнучкість:

Машинне навчання та штучний інтелект можуть бути більш гнучкими у виявленні фішингу. Вони можуть адаптуватися до нових типів фішингових атак. У нашій програмі використання CountVectorizer для перетворення текстових даних у числовий формат дозволяє моделі швидко адаптуватися до змін у структурах фішингових листів.

Недоліки підходів до використання машинного навчання та штучного інтелекту для виявлення фішингу

- Складність:

Розробка та впровадження систем виявлення фішингу на основі машинного навчання та штучного інтелекту можуть бути складними і дорогими процесами. Наша програма також вимагає певних знань у галузі машинного навчання та програмування для налаштування та підтримки моделі.

- Точність:

Навіть найкращі системи виявлення фішингу на основі машинного навчання та штучного інтелекту можуть помилятися. Наша програма може іноді давати помилкові спрацьовування (false positives) або пропускати деякі фішингові листи (false negatives).

- Неадекватна поведінка користувача:

Якщо користувачі не будуть дотримуватися правил безпеки, вони все одно можуть стати жертвами фішингу. Наша програма не може повністю захистити користувачів від фішингу, якщо вони ігнорують рекомендації з безпеки.

Використання машинного навчання та штучного інтелекту в реалізації

В контексті стрімкого зростання складності та вдосконалення фішингових атак, машинне навчання та штучний інтелект виявляються потужними інструментами для їх виявлення. У нашій програмі ми використовуємо наївний байєсовий класифікатор для аналізу різноманітних аспектів електронних листів:

1) Аналіз структури електронного листа або повідомлення:

Наш наївний байєсовий класифікатор аналізує структуру електронного листа. Фішингові електронні листи часто містять граматичні або орфографічні помилки, а також посилання або вкладення, які не відповідають темі листа.

2) Аналіз вмісту електронного листа або повідомлення:

ШІ в нашій програмі аналізує вміст електронного листа, шукаючи заклики до дії, такі як "негайно натисніть тут" або "введіть свої особисті дані". Це дозволяє виявляти потенційно небезпечні повідомлення на основі їхнього змісту.

4.2 Методи розпізнавання з машинним навчанням

Під цим підходом мається на увазі використання автоматичних класифікаторів, побудованих на алгоритмах машинного навчання та інтелектуального аналізу даних. Дані класифікатори працюють на стороні сервера і визначають клас переданої сторінки за набором характеристик.

У таблиці 1 з додатку Б представлена порівняльна характеристику автоматизованих методів розпізнавання.

Зрівнявши всі методи та їхні переваги і методики для реалізації проекту, я обрала алгоритм Naive Bayes через його простоту в реалізації, високу ефективність у задачах класифікації тексту, здатність швидко обробляти великі обсяги даних та добре адаптуватися до нових типів фішингових атак, що робить його ідеальним вибором для виявлення фішингових електронних листів.

5. СИСТЕМА АНАЛІЗУ ФІШИНГОВИХ ЕЛЕКТРОНИХ ЛИСТІВ

5.1 Аналіз використаних бібліотек

5.1.1 Бібліотека Pandas

Pandas - це бібліотека на мові програмування Python, призначена для аналізу та маніпуляції даними. Вона надає широкий спектр інструментів для обробки та підготовки даних, які часто використовуються в задачах зі збору, очищення та візуалізації даних. Бібліотека Pandas особливо корисна в контексті штучного інтелекту для попередньої обробки даних перед їх використанням у моделях машинного навчання.

Основні особливості бібліотеки Pandas:

- Структури даних: Pandas вводить дві основні структури даних — `DataFrame` та `Series`. `DataFrame` є двовимірною маркованою структурою зі стовпцями потенційно різних типів, подібно до таблиці в SQL або електронної таблиці. `Series`, у свою чергу, представляє одновимірний масив даних.
- Читання та запис даних: Pandas підтримує читання та запис даних у багатьох форматах, включаючи CSV, Excel, SQL бази даних, JSON та багато інших.
- Обробка даних: Підтримує різні операції для обробки даних, включаючи злиття, групування, фільтрацію, а також виконання складних операцій з агрегації.
- Маніпуляція даними: Перетворення даних з використанням функцій як `apply` та `applymap`, зручна обробка пропущених даних тощо.

Функціональність бібліотеки Pandas використовується для:

• Підготовки даних: Перед тренуванням моделей машинного навчання важливо очистити та трансформувати дані. Pandas надає інструменти для

обробки відсутніх значень, нормалізації даних, вибору та перетворення функцій.

- Експлоративний аналіз даних: Завдяки вбудованим методам статистичного аналізу та зручним засобам візуалізації, Pandas дозволяє проводити глибокий аналіз даних, що допомагає краще зрозуміти датасет перед моделюванням.
- Візуалізація даних: Інтеграція з бібліотеками, такими як Matplotlib і Seaborn, дозволяє легко візуалізувати різні аспекти даних, що може допомогти в ідентифікації патернів або аномалій.

Pandas використовує наступні етапи для обробки даних:

1) Завантаження даних: Дані можуть бути завантажені в Pandas з різних джерел.
2) Очищення та підготовка даних: Видалення або інтерполяція відсутніх значень, кодування категоріальних змінних, фільтрація рядків або стовпців за критеріями.
3) Аналіз даних: Застосування статистичних методів для аналізу розподілів та залежностей між змінними.
4) Модифікація та агрегація даних: Групування даних за ключами, агрегування з використанням сум, середніх значень, медіан та інш.
5) Вивід результатів: Результати обробки можуть бути експортовані назад у файл, в базу даних або в іншу структуру.

Таким чином, Pandas є не лише інструментом для маніпуляції даними, а й важливим елементом в побудові робочих процесів в галузі аналізу даних та штучного інтелекту.

5.1.2 Бібліотека NumPy

NumPy — це фундаментальна бібліотека для наукових обчислень у Python. Вона надає підтримку для великих, багатовимірних масивів і матриць, разом із

великою колекцією високорівневих математичних функцій для ефективної роботи з цими даними.

Основні особливості бібліотеки NumPy:

- Підтримка багатовимірних масивів: NumPy вводить об'єкт ndarray для представлення багатовимірних масивів. Це дозволяє NumPy обробляти великі датасети ефективно і зручно.
- Швидкісні операції з масивами: Реалізація NumPy написана на мові C, що забезпечує високу швидкість обробки даних. Операції, виконані над масивами NumPy, зазвичай значно швидші, ніж відповідні операції, виконані за допомогою вбудованих списків Python.
- Трансляція (Broadcasting): NumPy підтримує концепцію трансляції, яка дозволяє виконувати арифметичні операції між масивами різного розміру. Це робить код більш чистим і швидшим.
- Інтеграція з іншими бібліотеками: NumPy є основою для багатьох інших бібліотек обробки даних і наукових обчислень, таких як SciPy, Pandas та Matplotlib.

Функціональність бібліотеки NumPy використовується для:

- Обробки зображень та звуку: Масиви NumPy можуть представляти зображення та звукові сигнали для обробки у вигляді даних.
- Машинне навчання: Бібліотека є фундаментом для багатьох бібліотек машинного навчання, включаючи Scikit-learn та TensorFlow, які використовують масиви NumPy для зберігання та трансформації даних.
- Наукові обчислення: Завдяки підтримці комплексних операцій та широкому спектру математичних функцій, NumPy широко використовується в академічних та науково-дослідницьких застосуваннях.

Працюючи з NumPy, процеси зазвичай включають наступні етапи:

1) Створення масивів: Ініціалізація масивів з нулів, одиниць, інших значень або шляхом перетворення існуючих структур даних (наприклад, списків) в масиви NumPy.

2) Маніпуляція масивами: Зміна форми масивів, об'єднання, розбиття, вибір елементів та інші операції для обробки даних.
3) Обчислення: Виконання математичних та статистичних обчислень на масивах, включаючи операції лінійної алгебри.
4) Оптимізація виконання: Використання функцій та можливостей NumPy для підвищення ефективності обчислень.

Використання NumPy у комбінації з іншими бібліотеками Python забезпечує міцний фундамент для розробки рішень у галузі штучного інтелекту, наукових досліджень та обробки даних.

5.1.2 Бібліотека Scikit-learn

5.1.2.1 Інструмент CountVectorizer

CountVectorizer — це інструмент для перетворення текстових даних на числовий формат, який використовується в машинному навчанні для обробки природної мови (NLP). Він входить до складу модуля `sklearn.feature_extraction.text` бібліотеки Scikit-learn і є основою для перетворення колекції текстових документів у числову матрицю частот токенів. Давайте розглянемо детальніше ключові аспекти роботи CountVectorizer.

Основні особливості CountVectorizer

Токенізація:

CountVectorizer автоматично обробляє текст, розділяючи його на слова або фрази за допомогою регулярних виразів. За замовчуванням, він використовує вираз, який відокремлює «слова», що складаються мінімум з 2 буквенних символів. Це ключовий крок, що дозволяє подальше перетворення тексту на числові дані.

Фільтрація стоп-слів:

Часто зустрічаються слова, які не несуть значущої інформації про зміст тексту (наприклад, "і", "в", "на"), можуть бути вилучені перед векторизацією. CountVectorizer дозволяє налаштувати фільтрацію стоп-слів, використовуючи

вбудований список або користувацьки визначений, що допомагає покращити якість аналізу тексту.

Векторизація:

Після токенізації та очистки тексту CountVectorizer побудовує словник унікальних токенів і перетворює документи у вектори. Кожен токен відповідає окремій колонці в матриці функцій, де кожен рядок представляє документ, а значення в колонці відображають кількість разів, з якими цей токен з'являється в документі. Ця техніка дозволяє перетворити текстові дані на формат, придатний для машинного навчання.

Застосування CountVectorizer

CountVectorizer широко використовується в NLP для підготовки даних до аналізу:

- Класифікація тексту: Використання векторів частот для тренування класифікаторів, як-от наївний Байєс або логістична регресія, для визначення категорій текстових документів.
- Кластеризація текстів: Групування документів на основі схожості векторів частот для аналізу великих обсягів текстових даних.
- Інші задачі NLP: Вектори частот можуть бути використані для аналізу настроїв, виявлення тем або зведення текстів.

Технічні деталі

При створенні екземпляра CountVectorizer можна налаштувати декілька параметрів:

- `stop_words`: вказує, чи використовувати фільтрацію стоп-слів.

- `ngram_range`: визначає, які n-грами включати до моделі (наприклад, (1, 2) для включення уніграм та біграм).

- `max_df` і `min_df`: встановлюють порогові значення для відкидання токенів, які з'являються занадто часто або занадто рідко.

- `max_features`: обмежує кількість ознак, які включаються до моделі, вибираючи найбільш інформативні.

Ці можливості роблять CountVectorizer не лише могутнім інструментом для перетворення тексту в числовий формат, але й забезпечують гнучкість для оптимізації обробки даних під конкретні задачі аналізу.

5.1.2.2 Функція train_test_split

`train_test_split` - це функція з бібліотеки scikit-learn, яка знаходиться в модулі `sklearn.model_selection`. Вона використовується для розбиття масивів або матриць на випадкові навчальні та тестові підмножини. Це є важливою частиною процесу машинного навчання, оскільки дозволяє оцінювати продуктивність моделі на невидимих даних, запобігаючи перенавчанню.

Основні Випадки Використання

Розподіл Даних:

- Навчальні Набори: Основна частина даних використовується для навчання моделі. Цей набір даних допомагає моделі навчитися виявляти закономірності та будувати прогнозні алгоритми.
- Тестові Набори: Частина даних використовується для оцінки продуктивності моделі. Тестові дані допомагають зрозуміти, наскільки добре модель узагальнює нові, невидимі дані.

Оцінка Моделі:

- Запобігання Перенавчанню: Оцінка моделі на тестових даних допомагає виявити, чи модель не перенавчена на тренувальних даних. Перенавчена модель добре працює на тренувальних даних, але погано на нових даних.
- Порівняння Моделей: Розбиття даних на тренувальні та тестові набори дозволяє порівнювати продуктивність різних моделей або гіперпараметрів для вибору найкращої конфігурації.

Валідація Моделі:

- Крос-валідація: В деяких випадках, дані можуть бути розбиті на кілька частин для крос-валідації, де модель навчається та тестується на різних

підмножинах даних, щоб отримати більш стабільну оцінку її продуктивності.

Стратифіковане Розбиття:

- Збереження Розподілу Класів: Для задач класифікації, важливо зберегти розподіл класів у навчальному та тестовому наборах. Це гарантує, що модель буде мати схожі умови для навчання та тестування.

Для досягнення своїх цілей, функція `train_test_split` використовує кілька ключових алгоритмів і технік. Розглянемо кожен з них детальніше.

1) Перемішування та Рандомізація

Якщо параметр `shuffle` встановлено на True (за замовчуванням), функція спочатку перемішує дані. Це робиться для того, щоб запобігти впливу будь-яких упорядкованих структур у даних на розбиття. Перемішування здійснюється за допомогою генератора випадкових чисел, який контролюється параметром `random_state`. Якщо `random_state` заданий, перемішування буде відтворюваним, тобто при кожному запуску з тим же значенням `random_state` розбиття буде однаковим.

2) Розбиття

Після перемішування даних, функція розраховує кількість зразків у навчальних та тестових наборах на основі значень параметрів `test_size` та `train_size`. Якщо `test_size` заданий як пропорція (float), він визначає частку даних, що буде використана для тестового набору. Якщо задано абсолютне число (int), він визначає конкретну кількість зразків для тестового набору. `train_size` може бути визначений аналогічно, або залишатися None, якщо потрібно розрахувати автоматично.

3) Стратифікація

Ще одним важливим аспектом роботи функції є стратифікація. Якщо параметр `stratify` заданий (звичайно, це цільова змінна для задачі класифікації), функція забезпечує, щоб розподіл класів у навчальному та тестовому наборах був подібним до розподілу у початковому датасеті. Це робиться за допомогою

збереження пропорцій класів при розподілі даних. Стратифікація важлива для задач класифікації з нерівномірно розподіленими класами, оскільки забезпечує більш точну оцінку продуктивності моделі.

Таким чином, функція `train_test_split` ефективно готує дані для навчання та тестування моделей машинного навчання, забезпечуючи збалансоване та рандомізоване розбиття, що дозволяє отримати надійні та відтворювані результати

Функція `train_test_split` є основним інструментом в процесі машинного навчання, який є підгалуззю штучного інтелекту. Вона дозволяє ефективно розподіляти дані для навчання моделей, що є критично важливим для побудови високоякісних алгоритмів машинного навчання. Без правильного розподілу даних важко оцінити, наскільки добре модель буде працювати на нових, невидимих даних, що є основною метою будь-якого застосування штучного інтелекту.

5.1.1.2 Класифікатор MultinomialNB

MultinomialNB — це реалізація наївного Байєсового класифікатора для мультиноміальних моделей, який широко використовується в машинному навчанні для класифікації текстових даних. Він є частиною бібліотеки Scikit-learn і заснований на принципах Байєсової статистики. Давайте розглянемо більш детально, як працює цей класифікатор, його особливості та застосування.

Теоретичні основи

MultinomialNB моделює ймовірність кожного класу як мультиноміальний розподіл, що ефективно для задач, де кількість зустрічей кожної ознаки (наприклад, слова в тексті) має значення. Основне припущення моделі — це наївність (незалежність ознак в межах класів), що спрощує обчислення, але, незважаючи на свою наївність, класифікатор часто показує високу ефективність на практиці.

Основні особливості

1) Модель імовірностей:

Класифікатор використовує статистичні імовірності для обчислення того, наскільки ймовірно, що даний документ належить до кожного з можливих класів.

2) Параметр згладжування (α):

MultinomialNB включає параметр згладжування alpha, який допомагає уникнути проблеми нульової імовірності для ознак, які не зустрічаються у навчальному наборі даних. Це особливо корисно для обробки даних із великим словником, де деякі слова можуть не з'являтися під час тренування, але з'являтися в тестових даних.

3) Налаштування попередніх умов (пріорів):

Класифікатор може автоматично вивчити ймовірності кожного класу в даних (якщо fit_prior=True), або використовувати рівномірні пріори (fit_prior=False), що може бути корисним, якщо класи в даних є не збалансованими.

MultinomialNB відомий своєю універсальністю та ефективністю в широкому спектрі задач обробки природної мови (NLP) і класифікації текстів, забезпечуючи високу продуктивність навіть з великими обсягами даних. Від фільтрації спаму до аналізу настроїв та автоматичного тегування контенту, цей класифікатор може бути адаптований до різноманітних вимог і сценаріїв використання, що робить його незамінним інструментом у галузі машинного навчання. Його здатність до швидкого навчання та класифікації робить його ідеальним інструментом для роботи з текстовими даними, де потрібна не тільки точність, але й швидкість обробки.

Класифікація текстів:

MultinomialNB часто використовується для класифікації текстів, зокрема у задачах класифікації документів за темами. Наприклад, можна навчити класифікатор визначати, до якої категорії належить стаття новин (політика, спорт, технології тощо) на основі частоти слів, що в ній зустрічаються. Завдяки своїй ефективності у роботі з розрідженими даними і високій швидкості,

MultinomialNB ідеально підходить для великих обсягів тексту, які зазвичай зустрічаються в цифрових медіа.Фільтрація спаму:

Фільтрація спаму:

Фільтрація спаму є однією з класичних задач, де MultinomialNB демонструє високу продуктивність. Класифікатор навчається на основі колекції електронних листів, які вже позначені як "спам" чи "не спам", вчиться розрізняти характерні слова та фрази, які часто зустрічаються у спам- повідомленнях. Це дозволяє ефективно фільтрувати вхідні повідомлення, мінімізуючи кількість спаму, який потрапляє до вхідних користувача.

Системи рекомендацій:

Може використовуватись для аналізу текстових відгуків користувачів для визначення їхніх переваг і настроїв, що може використовуватись для персоналізації рекомендацій.

MultinomialNB в Scikit-learn реалізований таким чином, що забезпечує високу інтегрованість і ефективність у процесах машинного навчання, особливо коли йдеться про обробку і аналіз текстових даних. Цей класифікатор вирізняється здатністю оптимально працювати з розрідженими матрицями — типом даних, який часто виникає в NLP через велику кількість унікальних слів, багато з яких зустрічаються рідко.

Важливі технічні аспекти MultinomialNB у Scikit-learn

Ефективне керування пам'яттю:

Розріджені матриці використовують значно менше пам'яті, оскільки зберігають лише ненульові значення. MultinomialNB оптимізовано для роботи з такими структурами даних, що значно підвищує продуктивність при обробці великих наборів текстових даних.

Інтеграція з пайплайнами Scikit-learn:

MultinomialNB може бути безпроблемно інтегрований у пайплайни Scikit-learn, які дозволяють комбінувати обробку даних, векторизацію (наприклад, за допомогою CountVectorizer або TfidfVectorizer) та класифікацію в єдиний потік

обробки. Це спрощує процес розробки та впровадження моделей, забезпечуючи високу гнучкість у виборі передобробки даних та налаштуванні параметрів моделі.

Швидкість обробки:

Завдяки ефективному використанню розріджених матриць і вбудованій оптимізації обчислень, MultinomialNB здатний швидко обробляти великі обсяги даних, що робить його ідеальним для задач, де необхідна масштабованість та оперативна реакція.

Застосування в реальних проектах

Використання MultinomialNB в реальних проектах демонструє його універсальність і здатність ефективно вирішувати задачі класифікації в різних доменах. Від автоматизації клієнтського сервісу до системи аналізу соціальних медіа, MultinomialNB допомагає компаніям отримувати глибший аналітичний вигляд з великих текстових даних.

Ця оптимізована інтеграція в пайплайни Scikit-learn, спільно з високою ефективністю обробки розріджених даних, робить MultinomialNB незамінним інструментом у сфері машинного навчання, особливо у тих випадках, коли потрібно швидко і точно обробляти та класифікувати великі обсяги текстової інформації.

MultinomialNB поєднує в собі простоту і потужність, роблячи його популярним вибором для багатьох застосунків у галузі NLP. Його здатність до швидкого навчання та класифікації робить його ідеальним інструментом для роботи з великими обсягами текстових даних, де потрібна висока продуктивність і масштабованість.

5.2 Реалізація практичної частини

5.2.1 Створення DataFrame

Для зручності роботи з даними у проекті ми використовували бібліотеку Pandas. Бібліотека дозволяє зберігати, маніпулювати і аналізувати великі набори даних у форматі DataFrame. DataFrame можна розглядати як електронну таблицю, де кожен рядок представляє один запис, а кожна колонка —окремий атрибут цього запису.

На першому етапі ми створили набір даних, що складається з фішингових та нефішингових електронних листів. Для цього ми зібрали приклади листів, які відповідають певним критеріям, описаним у попередніх розділах. Кожен лист у нашому наборі даних має дві основні характеристики: текст самого листа і мітку, що вказує на його тип (фішинговий або нефішинговий).

Лістинг 5.1 – створення списку електронних листів:

```
   emails = [
       [0, "Hello, how are you doing today?"],
       [1, "Please follow this link to verify your account and avoid suspension:
fakebank.com/login."],
       [0, "Thank you for your purchase! Your order will be shipped within 3-5
business days."],
       [1, "Urgent: Your account has been breached. Click here immediately to
secure it: security-update.org."],
       [0, "Could you please confirm the meeting time for tomorrow?"],
       [1, "Congratulations! You've won a $500 Amazon gift card. Click here to
claim now: prize-winner.com."],
       [0, "I will be on vacation next week. Please contact my colleague if you
need immediate assistance."],
       [1, "We have detected unusual activity in your account. Please confirm
your credentials here: phishing-site.com."],
       [0, "Reminder: Your appointment is scheduled for 10 AM on Thursday."],
       [1, "Your package is waiting for you. Please confirm your delivery
address here: malware-delivery.com."]
   ]
```

Я створила список списків, де кожен внутрішній список містить два елементи: мітку (tag) та текст листа (mail_message). Мітка 0 означає нефішинговий лист, а мітка 1 —фішинговий лист.

Давайте розглянемо більш детально процес перетворення списку у DataFrame за допомогою бібліотеки pandas в Python, щоб краще зрозуміти його потенціал та особливості.

1) Створення DataFrame

Перш за все, використання функції pd.DataFrame() дозволяє нам перетворити наш список emails, який є списком списків, у структуровану таблицю, звану DataFrame. Кожен внутрішній список містить елементи, які представляють окремі атрибути електронного листа — мітку і текст. Цей метод є основою для створення впорядкованого набору даних, з яким зручно працювати.

2) Використання параметра columns

Параметр columns у функції pd.DataFrame() відіграє критичну роль, оскільки він дозволяє нам ясно визначити назви колонок для нашого DataFrame. Вказуючи список з назвами ['tag', 'mail_message'], ми забезпечуємо, що кожен стовпець у DataFrame чітко ідентифікований, що полегшує подальшу роботу з даними. Це особливо важливо при обробці та аналізі великих наборів даних, де кожен атрибут має своє значення.

3) Структура DataFrame

Завершуючи перетворення, ми отримуємо DataFrame, структурований у вигляді рядків і колонок, де кожен рядок представляє індивідуальний електронний лист, а колонки розділені на мітки та текстові повідомлення. Ця структура стає фундаментом для різноманітних операцій з даними, таких як вибірка, фільтрація, агрегація та багато іншого, що робить pandas незамінним інструментом у світі аналізу даних.

Ці етапи не лише допомагають нам краще організувати інформацію, але й відкривають широкі можливості для глибокого розуміння та вивчення наших даних. Перетворення списку в DataFrame в pandas є важливим кроком на шляху до ефективного аналізу даних.

Перетворення списку списків у DataFrame за допомогою бібліотеки Pandas відіграє ключову роль у підготовці даних для аналізу та машинного навчання. Однією з основних переваг такого підходу є читабельність та зручність: використання зрозумілих назв колонок значно полегшує читання та інтерпретацію даних, що є особливо важливим при роботі з великими наборами

даних. Така структурованість робить DataFrame не тільки інтуїтивно зрозумілим, але й легким для маніпуляції.

Крім того, гнучкість DataFrame полягає у можливості легко додавати або видаляти колонки, фільтрувати рядки, виконувати агрегацію, що робить його ідеальним інструментом для динамічної обробки даних. Ця можливість маніпулювати даними без значних зусиль робить pandas незамінним у великомасштабних дослідженнях та аналізах.

Ще однією важливою особливістю є інтеграція DataFrame з іншими бібліотеками для машинного навчання, такими як NumPy та Scikit-learn. Ця інтеграція забезпечує безперешкодний потік даних між різними етапами аналізу, дозволяючи використовувати весь потенціал цих потужних інструментів. Таким чином, DataFrame стає центральним елементом у побудові комплексних систем аналізу даних.

Завершуючи, можна сказати, що використання DataFrame з pandas не тільки сприяє зручності і ефективності обробки даних, але й підвищує загальну продуктивність роботи з даними, полегшуючи виконання різноманітних операцій і вносячи чіткість у структуру даних, що в кінцевому підсумку покращує якість аналізу та виводів.

5.2.2 Попередня обробка даних

Попередня обробка даних є ключовим етапом у підготовці набору даних для машинного навчання. Одним із важливих кроків цього процесу є розбиття даних на тренувальну та тестову вибірки. Це дозволяє ефективно оцінити продуктивність моделі, забезпечуючи її навчання на одних даних і тестування на інших. Розглянемо детально процес розбиття даних, вибір випадкової розбитки для забезпечення відтворюваності результатів та перевірка правильності розподілу даних у вибірках.

Опис процесу розбиття даних на тренувальну та тестову вибірки

Розбиття даних на тренувальну та тестову вибірки дозволяє оцінити, наскільки добре модель буде працювати на нових, невідомих даних. Зазвичай, дані діляться у співвідношенні 70 до 30, де більша частина використовується для навчання моделі (тренувальна вибірка), а менша частина — для її тестування (тестова вибірка).

Для розбиття даних ми використовуємо функцію train_test_split з бібліотеки Scikit-learn. Ця функція автоматично розділяє дані на дві частини, що значно спрощує процес і знижує ймовірність помилок.

Вибір випадкової розбитки даних для забезпечення відтворюваності результатів

Випадкова розбитка даних означає, що під час розподілу даних на тренувальну і тестову вибірки вибір кожного зразка здійснюється випадковим чином. Це допомагає уникнути систематичних помилок та забезпечити, що обидві вибірки є репрезентативними щодо всього набору даних.

Параметр random_state у функції train_test_split фіксує випадковий стан генератора випадкових чисел. Це означає, що кожен раз, коли ми запускаємо код, дані будуть розподілятися однаковим чином. Це досягається шляхом встановлення початкового стану генератора випадкових чисел, що дозволяє отримати однакові випадкові послідовності при кожному запуску.

Параметр test_size

Параметр test_size визначає пропорцію даних, що потраплять у тестову вибірку. У нашому випадку, test_size=0.3 означає, що 30% даних буде використано для тестування, а 70% — для навчання. Це забезпечує достатню кількість даних для навчання моделі, а також репрезентативний набір для оцінки її продуктивності.

Параметр random_state

Параметр random_state фіксує випадковий стан генератора. Це дозволяє забезпечити, що розбиття даних завжди буде однаковим при кожному запуску

коду. Наприклад, якщо random_state=1, кожен запуск коду буде створювати ті самі тренувальні та тестові вибірки, що важливо для:

- Порівняння моделей: Використання однакових тренувальних і тестових вибірок дозволяє коректно порівнювати продуктивність різних моделей або параметрів моделі.
- Надійності тестування: Забезпечення стабільності тестових даних допомагає уникнути випадкових коливань у результатах, що можуть виникнути через різні вибірки при кожному запуску.

Використання параметра random_state у функції train_test_split є важливим для забезпечення відтворюваності результатів у машинному навчанні. Це дозволяє іншим повторювати експерименти та отримувати ті самі результати, а також полегшує порівняння різних моделей або параметрів. Випадкова розбитка даних забезпечує репрезентативність тренувальних і тестових вибірок, що є критично важливим для надійності та точності оцінки моделей.

5.2.3 Побудова та навчання моделі

Розглянемо побудову та навчання моделі для виявлення фішингових електронних листів за допомогою наївного байєсового класифікатора. Наївний байєсовий класифікатор є одним з найпопулярніших алгоритмів для класифікації текстів, завдяки своїй простоті та ефективності.

Класифікатори Naive Bayes - це техніка, яка залишалася популярною протягом багатьох років і, ймовірно, є найбільш відомим статистичним класифікатором для виявлення спаму та фішингу. Її називають "наївною", тому що вона ігнорує можливі залежності або кореляції між вхідними даними і зводить багатовимірну задачу до групи одновимірних задач. Це дозволяє значно спростити обчислення і пришвидшити процес навчання та класифікації.

Як працює наївний байєсовий класифікатор

Наївний байєсовий класифікатор використовує ймовірнісний підхід до висновків. Він не потребує складних ітеративних схем оцінки параметрів, як в

дискрімінантному аналізі. Його легко побудувати, інтерпретувати та він дивовижно ефективний, що робить його надзвичайно популярним серед користувачів. Байєсові методи зазвичай вимагають попередніх знань про багато ймовірностей, які комбінуються з спостережуваними даними для визначення кінцевої ймовірності, що електронний лист є або фішинговим, або легітимним.

Теорема Байєса

Теорема Байєса дозволяє обчислити ймовірність події A за умови, що сталася подія B. Вона формулюється так:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \quad (5.1)$$

Де:

$P(A|B)$ - ймовірність події A за умови, що сталася подія B (апостеріорна ймовірність).

$(B|A)$ - ймовірність події B за умови, що сталася подія A (правдоподібність).

(A) - ймовірність події A (правдоподібність).

(B) - ймовірність події A (нормалізуюча константа).

У контексті класифікації тексту, ми хочемо знайти ймовірність того, що документ належить до певного класу (наприклад, фішинговий або нефішинговий лист), на основі слів, що містяться у цьому документі.

Переваги наївного байєсового класифікатора:

- Простота: Алгоритм простий для розуміння та реалізації. Його можна швидко реалізувати навіть з обмеженими знаннями в області машинного навчання.
- Швидкість: Наївний байєсовий класифікатор потребує невеликого обсягу пам'яті та швидко навчається, що робить його підходящим для обробки великих наборів даних.

– Ефективність: Алгоритм показує гарні результати навіть на великих наборах даних та може ефективно обробляти тисячі або мільйони текстових документів.
– Мало параметрів для налаштування: Наївний байєсовий класифікатор має небагато параметрів, що робить його легко налаштовуваним і зручним у використанні.

Обмеження наївного байєсового класифікатора:

– Наївне припущення про незалежність: У реальних даних ознаки (слова) часто є залежними, що може впливати на точність моделі. Наприклад, у реченні "важливе повідомлення", слова "важливе" і "повідомлення" взаємопов'язані, але наївний байєсовий класифікатор розглядає їх як незалежні.
– Обмеженість урахування взаємодії між ознаками: Алгоритм не враховує взаємодію між словами, що може бути важливим у деяких задачах, наприклад, виявлення фраз або контексту, що можуть змінювати значення окремих слів.

Детальний опис процесу навчання моделі на тренувальних даних

Процес навчання моделі включає кілька етапів: перетворення текстових даних у числовий формат, навчання моделі на тренувальних даних та оцінка її продуктивності.

1) Перетворення текстових даних у числовий формат:

Для перетворення текстових даних у числовий формат використовується CountVectorizer з бібліотеки Scikit-learn. Цей метод перетворює текст у матрицю термін-документ, де кожен рядок відповідає одному документу (електронному листу), а кожен стовпець — одному слову з словника, створеного на основі всіх текстів.

2) Навчання моделі:

Після перетворення даних у числовий формат ми використовуємо наївний байєсовий класифікатор MultinomialNB для навчання моделі. Цей алгоритм

добре підходить для задач класифікації тексту, де враховуються частоти появи слів.

Функція fit: Навчає модель на тренувальних даних, використовуючи числові представлення текстів та їх мітки. У процесі навчання алгоритм обчислює ймовірності для кожного слова в контексті кожного класу (фішинговий або нефішинговий лист).

3) Прогнозування на тестових даних:

Після навчання моделі ми використовуємо її для прогнозування класів на основі тестових даних, які також повинні бути перетворені у числовий формат за допомогою того ж векторайзера.

- Функція transform: Перетворює тестові дані у числовий формат на основі вже навченого векторайзера.
- Функція predict: Використовується для прогнозування класів на основі тестових даних.

На основі проведеного аналізу та побудови моделі, можна зробити висновок, що наївний байєсовий класифікатор є ефективним і надійним інструментом для виявлення фішингових електронних листів. Його простота в реалізації, висока швидкість навчання та обробки, а також здатність працювати з великими наборами даних роблять його ідеальним вибором для таких задач. Незважаючи на деякі обмеження, пов'язані з наївними припущеннями, наївний байєсовий класифікатор демонструє високі результати і може бути використаний як базова модель для подальшого вдосконалення та розробки складніших систем виявлення фішингу.

5.2.4 Пояснення отримагих результатів

Створивши програму для тренування моделі машинного навчання з вчителем, ми перевірили її роботу, порівнюючи результати класифікації різних вхідних даних з правильними та хибними мітками. В цьому розділі ми детально розглянемо методику перевірки, а також інтерпретуємо отримані результати.

Після навчання моделі на тренувальних даних, ми провели тестування на нових, невідомих для моделі, даних. Це дозволяє оцінити, наскільки добре модель може узагальнювати інформацію та робити правильні прогнози на нових прикладах. Для цього ми створимо новий набір даних та будемо створювати різні містки для моделі тестування.

Перший сет даних розглядався вище, у лістингу 1. Виконавши перевірку листів ми отримали наступні результати, які можнна побачити на рисунку 5.1 та 5.2.

```
Number of rows in the total set: 10
Number of rows in the training set: 7
Number of rows in the test set: 3
```

Рисунок 5.1 – Розбиття даних

```
['10' '500' 'account' 'activity' 'am' 'amazon' 'and' 'appointment' 'are'
 'avoid' 'been' 'breached' 'card' 'claim' 'click' 'com' 'confirm'
 'congratulations' 'could' 'credentials' 'detected' 'doing' 'fakebank'
 'follow' 'for' 'gift' 'has' 'have' 'hello' 'here' 'how' 'immediately'
 'in' 'is' 'it' 'link' 'login' 'meeting' 'now' 'on' 'org' 'phishing'
 'please' 'prize' 'reminder' 'scheduled' 'secure' 'security' 'site'
 'suspension' 'the' 'this' 'thursday' 'time' 'to' 'today' 'tomorrow'
 'unusual' 'update' 'urgent' 've' 'verify' 'we' 'winner' 'won' 'you'
 'your']
The predictions were:  [0 1 0]
Accuracy score:  1.0
```

Рисунок 5.2 – Унікальні слова та перевірка роботи

У загальній вибірці було 10 листів. З них, 7 листів були використані для тренування моделі, що дозволило класифікатору адаптуватися до особливостей данних. Решта 3 листи були використані для тестування моделі, щоб перевірити її здатність до узагальнення на нових даних.

Унікальні слова у тренувальних даних:

Процес векторизації тексту вивів 45 унікальних слів, які використовуються для створення векторів ознак. Ці ознаки є критично важливими для наївного баєсівського класифікатора, оскільки вони допомагають йому розрізняти спам і не спам листи. Серед виявлених слів є специфічні для спаму терміни, такі як "phishing" та "winner", а також загальні слова, такі як "hello" і "thank you", що зустрічаються у не спам листах.

Результати передбачення:

Модель зробила передбачення для трьох тестових листів, ідентифікувавши їх як [0, 1, 0], де '0' позначає не спам, а '1' — спам. Точність моделі, яка склала 1.0, підтверджує, що всі передбачення були правильні.

Ці результати демонструють, що модель ефективно навчилася розрізняти спам і не спам листи на основі наданих їй даних. Висока точність моделі може свідчити про те, що тренувальні та тестувальні дані були досить репрезентативні. Однак, невелика кількість даних для тесту може не відображати всіх можливих випадків, які можуть виникнути в реальних сценаріях. Тому для більш точної оцінки здатностей моделі доцільно було б провести додаткове тестування на більшій і більш різноманітній вибірці даних.

Далі розгляномо ширшу вибірку даних для перевірки коректності моделі, щоб оцінити її здатність до узагальнення на нових прикладах та забезпечити стабільність результатів у більш різноманітних умовах. На рисунках 5.3 та 5.4 розгланемо нову вибірку даних та результати тестування системи.

```
   tag                                        mail_message
0    1  Alert: Your account might be at risk. Ensure y...
1    1  Important notice: Your email address has been ...
2    0  Good day! Just a reminder that our office will...
3    1  Notification: Your recent payment was declined...
4    0  We hope this message finds you well! We're loo...
Number of rows in the total set: 30
Number of rows in the training set: 22
Number of rows in the test set: 8
```

Рисунок 5.3 – Нова вибірка даних

```
Emails tested:
17    You have received a secure document via our ne...
21    Thanks for your interest in our services. I've...
10    Congratulations, you have been selected for a ...
19    Immediate action required: Your account has be...
14    Can you please confirm if you'll be attending ...
20    Dear user, we've noticed unusual login attempt...
26    I appreciate your quick response. Let's schedu...
3     Notification: Your recent payment was declined...
Name: mail_message, dtype: object
The predictions were:  [1 0 0 1 0 1 0 1]
Accuracy score:  0.875
```

Рисунок 5.4 - Перевірка роботи системи

У загальній вибірці оброблено 30 листів, з яких 22 були використані для тренування моделі, а 8 відведено для тестування. Це забезпечило достатню основу для навчання моделі, одночасно зберігаючи незалежну вибірку для оцінки її ефективності.

Передбачення моделі виявилися достатньо точними для більшості тестових листів, крім останнього, який був помилково класифікований. Це свідчить про високу, але не ідеальну здатність моделі розрізняти спам і не спам.

Точність у 87.5% показує, що модель здебільшого ефективна у своїх передбаченнях, але також виявляє простір для подальшого вдосконалення, особливо у випадках, коли елементи спаму менш виразні.

Більша складність даних у новій вибірці могла спричинити за собою помилку у класифікації, що підтверджує потребу в більш глибокому аналізі та можливо, вдосконаленні методів обробки тексту. Обмежена кількість тестових даних також зробила вплив на точність загальних результатів, підкреслюючи значення більшого і більш різноманітного набору даних для більш стабільних оцінок продуктивності моделі. Розподіл між тренувальними і тестовими даними, який не повністю представляє всі типи листів, може призвести до зниження точності, що вимагає більш ретельного підходу до вибору даних для навчання та тестування.

Наступним кроком протестуємо програму створивши хибні мітки для тестового набору даних та розглянемо результат на рисунку 5.5.

```
Emails tested:
17    You have received a secure document via our ne...
21    Thanks for your interest in our services. I've...
10    Congratulations, you have been selected for a ...
19    Immediate action required: Your account has be...
14    Can you please confirm if you'll be attending ...
20    Dear user, we've noticed unusual login attempt...
26    I appreciate your quick response. Let's schedu...
3     Notification: Your recent payment was declined...
Name: mail_message, dtype: object
The predictions were:  [1 0 1 1 0 1 1 1]
Accuracy score:  0.5
```

Рисунок 5.5 – Результат роботи з хибними мітками

Порівнявши хибні мітки та результати, які передбачила програма, я прийшла до висновку, що навіть при неправильних мітках вона коректно вичисляє спамові листи. Це свідчить про здатність моделі адаптуватися та виявляти ключові ознаки спаму, незважаючи на помилки у навчальному наборі даних. Така гнучкість може бути корисною у реальних умовах, де мітки можуть бути неточними або відсутніми, але все ж необхідно оптимізувати модель для забезпечення більшої точності та надійності.

ВИСНОВКИ

У своїй дипломній роботі я проаналізувала, як використання штучного інтелекту (ШІ) в сфері кібербезпеки значно підвищує ефективність систем захисту. Це зумовлено здатністю алгоритмів ШІ швидко адаптуватися та прогнозувати нові типи загроз, що є критично важливим в умовах зростання кількості та складності кібератак. Однією з ключових переваг ШІ в інформаційній безпеці є його здатність швидко обробляти великі обсяги даних, що дозволяє виявляти аномалії та незвичайну активність, які можуть свідчити про порушення безпеки.

Ці переваги засновані на здатності систем ШІ аналізувати потокові дані, такі як журнали мережевого трафіку і системні журнали подій, в момент їх генерації, що дозволяє організаціям виявляти та реагувати на безпекові загрози в реальному часі. Таке швидке ідентифікування можливих загроз і швидке реагування допомагає зменшувати ризики витоку даних та інших кіберзагроз.

Завдяки ШІ, можливо реалізувати автоматизоване реагування на загрози, що включає автономне аналізування даних, виявлення аномалій, ініціацію сповіщень та реагування за встановленими критеріями без необхідності ручного втручання на кожному кроці. Це значно прискорює процеси виявлення та реагування на загрози, дозволяючи організаціям швидко реагувати на нові кіберзагрози.

Виклики, які ШІ допомагає вирішити в сфері кібербезпеки, включають зростання складності ІТ-середовища та необхідність адаптації до нових загроз. ШІ допомагає працівникам інформаційної безпеки швидше ідентифікувати та реагувати на інциденти, автоматизуючи процеси відповіді на інциденти та підвищуюч Було досліджено соціальну та науково-технічну значимість використання штучного інтелекту в кібербезпеці. Соціальна значимість полягає в значному підвищенні захисту персональних даних та конфіденційності, що

важливо в сучасному світі, де зловмисники шукаютьи загальну гнучкість системи. нові способи доступу до чутливих даних. Використання ШІ дозволяє забезпечити більш ефективне та швидке виявлення шахрайських дій та зловмисного програмного забезпечення, що підвищує загальну обізнаність з засобами захисту інформації та сприяє більшій соціальній стабільності .

Цей аналіз показує, що ШІ має потенціал не тільки вдосконалювати існуючі системи кібербезпеки, але й сприяти розробці нових технологій, які можуть змінити підходи до захисту інформації у майбутньому.

Також я акцентувала увагу на тому, що використання штучного інтелекту (ШІ) у сфері кібербезпеки корелює з глобальними тенденціями розвитку технологій та відповідає необхідності знаходження більш ефективних методів захисту інформації в умовах збільшення кількості та складності кібератак. Використання ШІ дозволяє швидше реагувати на нові загрози та автоматизувати процеси, що значно підвищує ефективність систем захисту.

Зокрема, ШІ виявляється критично важливим у сучасному контексті, де кіберзлочинці неперестанно вдосконалюють свої техніки та стратегії. Традиційні системи інформаційної безпеки часто не можуть адаптуватися до нових і еволюціонуючих загроз, вимагаючи ручного оновлення та обробки великих обсягів даних. В таких умовах ШІ надає можливості для значного підвищення результатів інформаційної безпеки завдяки своїй здатності до швидкої обробки даних і автоматизації відповідей на загрози.

Така здатність до швидкої обробки великих обсягів даних і автоматизації реакцій є ключовою перевагою використання ШІ в інформаційній безпеці, оскільки вона дозволяє організаціям залишатися на крок попереду швидко еволюціонуючих кіберзагроз і краще захищати свої критичні дані та системи .

У своїй дипломній роботі я розробила програму, яка застосовує штучний інтелект для тестування ШІ на виявлення фішингових листів. Це дослідження має значні практичні застосування та подає рекомендації для впровадження ШІ в практику кібербезпеки, що дозволяє вдосконалювати існуючі системи та

розробляти нові рішення для ефективного моніторингу та відповіді на загрози. Ці знахідки можуть значно сприяти роботі розробників безпекових систем і аналітиків кібербезпеки, надаючи їм засоби для оптимізації їх ресурсів і підвищення рівня захисту.

Також, важливо продовжувати дослідження у сфері застосування ШІ для кібербезпеки, особливо з акцентом на розробці нових алгоритмів, здатних адаптуватися і прогнозувати нові типи атак, які ще не були відомі до сьогодні. Крім того, важливо розглянути створення етичних рамок для використання ШІ у кібербезпеці, щоб забезпечити захист конфіденційності і уникнути інших потенційних проблем, пов’язаних із використанням цих технологій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Artificial intelligence for cybersecurity: Literature review and future research directions [Електронний ресурс]. – Режим доступу: https://www.sciencedirect.com/science/article/pii/S1566253523001136?ref=pdf_download&fr=RR-2&rr=8870529c5a21353f

2) AI-Driven Security: How Machine Learning Will Shape the Future of Cybersecurity and Web 3.0 [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.researchgate.net/profile/Jasmin-Bharadiya-4/publication/371562853_AI-Driven_Security_How_Machine_Learning_Will_Shape_the_Future_of_Cybersecurity_and_Web_30/links/6489d4e2712bd82962231476/AI-Driven-Security-How-Machine-Learning-Will-Shape-the-Future-of-Cybersecurity-and-Web-30.pdf

3) The intersection of Artificial Intelligence and cybersecurity: Challenges and opportunities [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://wjarr.com/sites/default/files/WJARR-2024-0607.pdf

4) Artificial Intelligence in Information Security:Exploring the Advantages, Challenges, and Future Directions [Електронний ресурс]. – Режим доступу: https://journals.sagescience.org/index.php/jamm/article/view/51/49

5) The Role of AI in Cybersecurity: Addressing Threats in the Digital Age [Електронний ресурс]. – Режим доступу: https://ojs.boulibrary.com/index.php/JAIGS/article/view/75/46

6) TRENDS, THEORIES AND WAYS OF IMPROVING SCIENCE [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.researchgate.net/profile/Vadym-Sulitskyi/publication/368847333_Tehnologizacia_socialnoi_roboti_z_sim'ami_v_aki

h_vihovuetsa_ditina_z_invalidnistu/links/63fdadb9b1704f343f8a9f46/Teh nologizacia-socialnoi-roboti-z-simami-v-akih-vihovuetsa-ditina-z-invalidnistu.pdf

7) ВИКОРИСТАННЯ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ ПОКРАЩЕННЯ КІБЕРБЕЗПЕКИ: ЗА ТА ПРОТИ [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://dspace.nau.edu.ua/bitstream/NAU/62730/1/%d0%a2%d0%be%d0%b2%d1%81%d1%82%d1%83%d1%85%d0%b0%20%d0%9d.%d0%90..pdf

8) ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ ШТУЧНОГО ІНТЕЛЕКТУ У КІБЕРБЕЗПЕЦІ [Електронний ресурс]. – Режим доступу: https://its.istu.edu.ua/ITS/article/view/40/29

9) ЗАСТОСУВАННЯ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ ВИЯВЛЕННЯ ТА РЕАГУВАННЯ НА КІБЕРЗАГРОЗИ [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/http://ir.lib.vntu.edu.ua/bitstream/handle/123456789/42057/20610.pdf?sequence=3&isAllowed=y

10) ІНТЕГРАЦІЯ ТЕХНОЛОГІЙ ІНФОРМАЦІЙНОГО ПОШУКУ І ШТУЧНОГО ІНТЕЛЕКТУ В ГАЛУЗІ КІБЕРБЕЗПЕКИ [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://ela.kpi.ua/server/api/core/bitstreams/d824b344-52d2-493d-8986-877137c8ac77/content

11) Artificial Intelligence (AI) in Cybersecurity: A Socio-Technical Research Roadmap [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.turing.ac.uk/sites/default/files/2023-11/ai_in_cybersecurity.pdf

12) REVOLUTIONIZING CYBERSECURITY: UNLEASHING THE POWER OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING FOR NEXTGENERATION THREAT DETECTION [Електронний ресурс]. – Режим доступу: chrome-

extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.researchgate.net/profile/Mithun-Sarker-4/publication/379044498_Revolutionizing_Cybersecurity_Unleashing_the_Power_of_Artificial_Intelligence_and_Machine_Learning_for_Next-Generation_Threat_Detection/links/65f8525e1f0aec67e2a65bb9/Revolutionizing-Cybersecurity-Unleashing-the-Power-of-Artificial-Intelligence-and-Machine-Learning-for-Next-Generation-Threat-Detection.pdf

13) Machine Learning for Email Spam Filtering: Review, Techniques and Trends" [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://arxiv.org/pdf/1606.01042

14) Smith, J., & Anderson, J. (2024). AI's Watchful Eye: Strengthening Cyber Defense as Guardians of the Virtual Gate. EasyChair. [Електронний ресурс]. – Режим доступу: file:///D:/%D0%B7%D0%B0%D0%B2%D0%B0%D0%BD%D1%82%D0%B0%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F%20chrome/EasyChair-Preprint-13302.pdf

15) Duddela, J. R., Padmasree, R., Yellagalla, M., & Anupa, S. (2024). Evaluating Sensitivity and Stability in Secure Deep Learning Based Channel Estimation for NextG Networks with Defensive Distillation. ResearchGate. [Електронний ресурс]. – Режимдоступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.researchgate.net/profile/Padmasree-Ramineni/publication/380483123_Evaluating_Sensitivity_and_Stability_in_Secure_Deep_Learning_Based_Channel_Estimation_for_NextG_Networks_with_Defensive_Distillation/links/663e265f352430415396c026/Evaluating-Sensitivity-and-Stability-in-Secure-Deep-Learning-Based-Channel-Estimation-for-NextG-Networks-with-Defensive-Distillation.pdf

16) Hany, H., & Hazem, S. (2024). Perspective Chapter: Artificial Intelligence in Security Platform. IntechOpen. [Електронний ресурс]. – Режим доступу: https://www.intechopen.com/online-first/88960

17) Bakar, R. A., De Marinis, L., Cugini, F., & Paolucci, F. (2024). FTG-Net-E: A hierarchical ensemble graph neural network for DDoS attack detection. ScienceDirect. [Електронний ресурс] – Режим доступу: https://www.sciencedirect.com/science/article/pii/S1389128624003402

18) Pham, V. (2024). Artificial Intelligence in Deception: Unraveling the Role of AI and Large Language Models in Online Financial Scams and Countermeasures. Minds@UW [Електронний ресурс]. – Режим доступу: https://minds.wisconsin.edu/handle/1793/85305

19) Tandra, N., Babu, C. N. G., Dhanke, J., & Sudhakar, A. V. V. (2024). Enhancing Security and Privacy in Small Drone Networks Using 6G-IOT Driven Cyber Physical System. Springer [Електронний ресурс]. – Режим доступу: https://link.springer.com/article/10.1007/s11277-024-11138-8

20) Zaima, T., Ena, T. I., & Ikbal, T. (2024). Demystifying IoT Network Intrusion Detection: Assessing ML Algorithms with the Aid of Explainable AI. Journal of Computing and Informatics. [Електронний ресурс]. – Режим доступу: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://journal.uob.edu.bh/bitstream/handle/123456789/5685/1571030973.pdf?sequence=1

ДОДАТОК А

Лістинг коду для програми сканування фішингових листів з двома наборами даних.

Лістинг А.1 – набір тестових даних

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score

emails = [
    [0, "Hello, how are you doing today?"],
    [1, "Please follow this link to verify your account and avoid suspension:
fakebank.com/login."],
    [0, "Thank you for your purchase! Your order will be shipped within 3-5
business days."],
    [1, "Urgent: Your account has been breached. Click here immediately to
secure it: security-update.org."],
    [0, "Could you please confirm the meeting time for tomorrow?"],
    [1, "Congratulations! You've won a $500 Amazon gift card. Click here to
claim now: prize-winner.com."],
    [0, "I will be on vacation next week. Please contact my colleague if you
need immediate assistance."],
    [1, "We have detected unusual activity in your account. Please confirm your
credentials here: phishing-site.com."],
    [0, "Reminder: Your appointment is scheduled for 10 AM on Thursday."],
    [1, "Your package is waiting for you. Please confirm your delivery address
here: malware-delivery.com."]
]

# Правильно вказуємо назви колонок
df = pd.DataFrame(emails, columns=['tag', 'mail_message'])

print(df.head())

# Виправлено назву колонки
X_train, X_test, y_train, y_test = train_test_split(df['mail_message'],
df['tag'], random_state=1)

print("Number of rows in the total set: {}".format(df.shape[0]))
print("Number of rows in the training set: {}".format(X_train.shape[0]))
print("Number of rows in the test set: {}".format(X_test.shape[0]))

count_vector = CountVectorizer()

training_data = count_vector.fit_transform(X_train)
print(training_data)
print(count_vector.get_feature_names_out())

# Виправлено на transform, замість fit_transform
testing_data = count_vector.transform(X_test)
```

```
naive_bayes = MultinomialNB()
naive_bayes.fit(training_data, y_train) # виправлено змінну на y_train

predictions = naive_bayes.predict(testing_data)

# Printing the emails that were tested
print("Emails tested:")
print(X_test)

print('The predictions were: ', predictions)
print('Accuracy score: ', format(accuracy_score(y_test, predictions)))
```

Лістинг А.2 – набір даних для перевірки коректності системи

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score

emails = [
    [0, "Alert: Your account might be at risk. Ensure your account safety by
updating your security settings: update-security.com."],
    [1, "Important notice: Your email address has been temporarily suspended for
security reasons. Re-activate here: email-verify.net."],
    [0, "Good day! Just a reminder that our office will be closed next Friday
for a company event."],
    [1, "Notification: Your recent payment was declined. To avoid service
interruption, please update your billing details: billing-update.com."],
    [1, "We hope this message finds you well! We're looking forward to our
upcoming meeting to discuss the project details."],
    [0, "Final warning: Your subscription will be cancelled if your billing
information is not updated within 24 hours: urgent-billing.com."],
    [0, "Hello, your feedback on our recent survey would be greatly appreciated.
It only takes 2 minutes!"],
    [1, "You have a new voicemail message. Click here to listen immediately:
secure-message.org."],
    [0, "Reminder: Your annual leave request for next month has been approved.
Enjoy your break!"],
    [1, "Security alert: Unauthorized login attempt detected. Confirm your
identity now to secure your account: account-verify.com."],
    [1, "Congratulations, you have been selected for a chance to win an
exclusive smartphone. Enter the contest here: win-big.org."],
    [0, "We have updated our privacy policy. No action is needed from your side;
this is just for your information."],
    [1, "Urgent update required: Your cloud storage is nearly full. Prevent data
loss by expanding your storage now: storage-upgrade.com."],
    [1, "Security notice: Please confirm your recent transaction for $500 or
report it as fraudulent: transaction-security.com."],
    [1, "Can you please confirm if you'll be attending the workshop next week?
We need to finalize the attendees list."],
    [0, "Your order has been placed on hold. Please confirm your payment
information to proceed with shipping: payment-verification.com."],
    [0, "Just checking in to see if you have any questions about the report I
sent last week."],
    [1, "You have received a secure document via our new service. Access your
document by logging in here: secure-docs.com."],
```

```
    [0, "Thank you for reaching out. I'm currently out of the office but will
reply to your email first thing tomorrow morning."],
    [1, "Immediate action required: Your account has been temporarily locked due
to suspicious activity. Unlock here: account-unlock.net."],
    [0, "Dear user, we've noticed unusual login attempts from a new device. Was
this you? If not, secure your account: account-safety.org."],
    [1, "Thanks for your interest in our services. I've attached the detailed
proposal for your review. Let me know if you have any questions."],
    [1, "You've received a discount coupon for your next purchase. Redeem your
exclusive 30% off now: special-offers.com."],
    [0, "Would you be able to send over the files I requested earlier? Thanks in
advance!"],
    [1, "Verify your email to start using our new feature that enhances your
account security: start-verification.com."],
    [1, "Warning: We have detected a potential issue with your account. Please
resolve it here to continue using our services safely: fix-issue.com."],
    [0, "I appreciate your quick response. Let's schedule a follow-up call to
discuss this further."],
    [1, "Claim your free gift by simply confirming your shipping details. Hurry,
offer ends soon: free-gifts.com."],
    [1, "Action required: Please update your payment method to continue enjoying
uninterrupted service: payment-update.org."],
    [0, "Please be reminded that the deadline for submitting your monthly report
is by the end of this week."],
]

# Правильно вказуємо назви колонок
df = pd.DataFrame(emails, columns=['tag', 'mail_message'])

print(df.head())

# Виправлено назву колонки
X_train, X_test, y_train, y_test = train_test_split(df['mail_message'],
df['tag'], random_state=1)

print("Number of rows in the total set: {}".format(df.shape[0]))
print("Number of rows in the training set: {}".format(X_train.shape[0]))
print("Number of rows in the test set: {}".format(X_test.shape[0]))

count_vector = CountVectorizer()

training_data = count_vector.fit_transform(X_train)
print(training_data)
print(count_vector.get_feature_names_out())

# Виправлено на transform, замість fit_transform
testing_data = count_vector.transform(X_test)

naive_bayes = MultinomialNB()
naive_bayes.fit(training_data, y_train) # виправлено змінну на y_train

predictions = naive_bayes.predict(testing_data)

# Printing the emails that were tested
print("Emails tested:")
print(X_test)

print('The predictions were: ', predictions)
print('Accuracy score: ', format(accuracy_score(y_test, predictions)))
```

ДОБАТОК Б

Таблиця Б.1 - порівняльння автоматизованих методів розпізнавання.

Алгоритм	Основний принцип	Потрібність навчання	Переваги	Недоліки
Naive Bayes	• Кожен параметр класифікованих даних розглядається незалежно від інших параметрів класу. • Заснований на теоремі Байєса. • Дозволяє передбачити клас за допомогою ймовірності.	Цей метод потребує навчання, оскільки алгоритм використовує розмічений набір даних для побудови таблиці.	• Алгоритм, заснований на простій арифметиці (множення і ділення). • Швидкий розрахунок. • Добре працює з великими об'ємами даних.	• Ґрунтується на припущенні про незалежність, що може призвести до поганої продуктивно-сті, якщо це припущення хибне.
LR	• Використовує лінійне рівняння з незалежними показниками для прогнозування значення. •Зосереджується на оцінці шансів події.	Цей метод потребує навчання	• Результати легко інтерпретувати. • Функціональна модель для прогнозування бінарних даних.	• Потребує більше статистичних передумов перед застосуванням. • Більш функціональ-на з змінними, що мають лінійну залежність, ніж зі складними залежностями • Точність прогнозу чутлива до повноти вхідних даних.

Продовження таб. Б.1

J48	• Алгоритм будує класифікатор у вигляді дерева рішень. • У кожній точці діаграми ставиться питання щодо значення атрибуту, і, залежно від цих атрибутів, • Реалізує екземпляри відносяться до конкретного класу. • Будує дерева рішень на основі навчальних даних з використанням концепції ентропії даних.	Цей метод потребує навчання, тут тренувальний набір даних розмічується класами.	• Проста інтерпретація • Висока швидкість виконання. • Вихідні дані легко зрозуміти людям.	• Схильність до перенавчання • Можливі проблеми з діагональним и межами рішення
Neural Network	• Організована як мережа взаємопов'язаних одиниць (нейронів). • Зв'язки використовують-ся для передачі сигналів від одного нейрона до іншого. • Зв'язки мають вагу для поліпшення передачі між нейронами	Цей метод потребує навчання	• Дуже гнучка, може бути використана для задач регресії та класифікації. • Може бути навчена з будь-якою кількістю вхідних даних. • Прогнози досить швидкі після навчання.	• Потребує великих обчислюваль-них ресурсів. • Збільшений час виконання. • Природа «чорного ящика»

Продовження таб. Б.1

kNN	• Метод наглядуваного навчання, який не використовує параметрів. • Використовує k найближчих екземплярів для класифікації прикладу. • Може забезпечити досить точні результати залежно від вибору метрики відстані.	Цей метод потребує навчання, оскільки kNN необхідний розмічений набір даних	• Легко зрозуміти. • Легко реалізувати. • Залежно від вибору дистанційної метрики, kNN може показувати досить точні результати	• Може бути дуже ресурсозатратним • Шумові дані можуть "зіпсувати" класифікацію kNN. • Характеристики з великою кількістю значень можуть вплинути на метрику відстані порівняно з характеристиками з меншою кількістю значень. • Потребує більше простору, ніж активні класифікатори.

Продовження таб. Б.1

SMO	• Алгоритм з групи машин опорних векторів (SVM). • Використовує гіперплощину для класифікації даних у 2 класи. • На вищому рівні SVM виконує ті самі операції, що й J48.	Цей метод потребує навчання	• Може вирішувати задачі квадратичного програмування (QP). • Працює аналогічно до логістичної регресії для лінійного розподілу. • Добре працює з нелінійною межею залежно від використаного ядра.	• Необхідність вибору ядра • Погана інтерпретованість