**Міністерство освіти і науки України**
**Національний технічний університет**
**«Дніпровська політехніка»**

---

**Інститут електроенергетики**
**Факультет інформаційних технологій**
**Кафедра безпеки інформації та телекомунікацій**

# ПОЯСНЮВАЛЬНА ЗАПИСКА
## кваліфікаційної роботи ступеня бакалавра

студента *Хричова Олекасандра Вадимовича*

академічної групи *125м-22-2*

спеціальності *125 Кібербезпека*

спеціалізації

за освітньо-професійною програмою *Кібербезпека*

на тему *Виявлення каналів витоку інформації соціальної інженерії за допомогою машинного навчання*

| **Керівники** | **Прізвище, ініціали** | **Оцінка за шкалою** | | **Підпис** |
|---|---|---|---|---|
| | | **рейтинго-вою** | **інституційною** | |
| кваліфікаційної роботи | к. ф-м.н., професор Магро В.І. | | | |
| розділів: | | | | |
| спеціальний | ст. викладач Святошенко В.О. | 95 | відмінно | |
| економічний | к.е.н., доц. Пілова Д.П. | 90 | відмінно | |

| | | | | |
|---|---|---|---|---|
| Рецензент | | | | |

| | | | | |
|---|---|---|---|---|
| Нормоконтролер | ст. викладач Мєшков В.І. | 90 | відмінно | |

**Дніпро**
**2023**

**ЗАТВЕРДЖЕНО:**
завідувач кафедри
безпеки інформації та телекомунікацій
_____ д.т.н., проф. Корнієнко В.І.

«_____»_____20___року

**ЗАВДАННЯ**
**на кваліфікаційну роботу ступеня бакалавра**

**студенту** *Хричову О. В.* **Академічної групи** **125м-22-2**
(прізвище та ініціали) (шифр)

**спеціальності** *125 Кібербезпека*

**за освітньо-професійною програмою** *Кібербезпека*

**на тему** ***Виявлення каналів витоку інформації соціальної інженерії за допомогою машинного навчання***

Затверджену наказом ректора НТУ «Дніпровська політехніка» від 09.10.23 № 1227-с

| Розділ | Зміст | Термін виконання |
|---|---|---|
| *Розділ № 1* | *Огляд літератури за темою, постановка задачі.* | жовтень |
| *Розділ № 2* | *Проведення аналізу навчання різних методів машинного навчання, їх застосування у виявленні витоків інформації .* | листопад |
| *Розділ № 3* | *Розрахунок витрат пов'язаними з впровадженням методів захисту* | грудень |

Завдання видано_____ Володимир СВЯТОШЕНКО
(підпис керівника) (ім'я, прізвище)

Дата видачі завдання: _____

Дата подання до екзаменаційної комісії: _____

Прийнято до виконання _____ Олександр ХРИЧОВ
(підпис студента) (ім'я, прізвище)

# РЕФЕРАТ

Пояснювальна записка: 93 ст., 18 рис., 3 табл., 4 додатків, 14 джерел

Предмет дослідження: канали витоку інформації у соціальних мережах які є цілью атак соціальної інженерії.

Методи дослідження: спостереження, обстеження, аналіз,порівняння, опис.

Мета роботи: дослідження та оцінка ефективності різних методів машинного навчання, в контексті аналізу соціальних мереж, для виявлення потенційних каналів витоку інформації та загроз соціальної інженерії.

У першому розділі кваліфікаційної роботи наведено опис, що саме підлягає визначенню соціальної інженерії. Які є її види атак та як вони реалізовуються, що таке машинне навчання, які саме методи навчання використовуються для виявлення. Розглянуто і описано, що таке машинне навчання, його види і їх можливості.

У другому розділі розглянуто соціальну інженерію у контексті соціальних мереж. Проведено аналіз і порівняння різних моделей машинного навчання для виявлення найкращої моделі, яку можна застосувати для виявлення відкритих каналів витоку у соціальних мережах.

У третьому розділі розраховано витрати на впровадження алгоритму виявлення відкритих каналів витоку у соціальних мережах. Прорахована економічна доцільність створення та використання такого алгоритму.

Практична цінність роботи: можливість застосування проведеного дослідження для створення системи виявлення витоків інформації та запобігання атакам соціальної інженерії, інтегрування алгоритму у вже існуючі ситеми безпеки.

ІНФОРМАЦІЙНА БЕЗПЕКА, СОЦІАЛЬНА ІНЖЕНЕРІЯ, ЗАХИСТ ІФОРМАЦІЇ, МАШИННЕ НАВЧАННЯ, МЕТОДИ РЕАЛІЗАЦІЇ АТАК, МОДЕЛІ МАШИННОГО НАВЧАННЯ .

# ABSTRACT

Explanatory Note: 93 pages, 18 figures, 3 tables, 4 appendices, 14 references.

Subject of Study: Channels of information leakage in social networks targeted by social engineering attacks.

Research Methods: Observation, survey, analysis, comparison, description. Objective of the Study: To investigate and evaluate the effectiveness of various

machine learning methods in the context of social network analysis for identifying potential information leakage channels and social engineering threats.

The first section of the thesis provides a description of what constitutes social engineering, the types of attacks it includes, and how they are implemented. It also covers what machine learning is, and the specific learning methods used for detection. It explores and describes machine learning, its types, and their potential.

The second section examines social engineering in the context of social networks. An analysis and comparison of different machine learning models are conducted to identify the best model applicable for detecting open channels of leakage in social networks.

The third section calculates the costs of implementing an algorithm to detect open channels of information leakage in social networks. The economic feasibility of creating and using such an algorithm is assessed.

Practical Value of the Work: The ability to apply the conducted research for creating a system to detect information leaks and prevent social engineering attacks, and integrating the algorithm into existing security systems.

INFORMATION SECURITY, SOCIAL ENGINEERING, INFORMATION PROTECTION, MACHINE LEARNING, METHODS OF ATTACK IMPLEMENTATION, MACHINE LEARNING MODELS.

# СПИСОК УМОВНИХ СКОРОЧЕНЬ

CNN – Конволюційні нейронні мережі (Convolutional Neural Network);

GDPR– Загальний регламент захисту даних (General Data Protection Regulation)

PCA – метод головних компонент (Principal Component Analysis);

RNN – Рекурентні нейронні мережі (Recurrent Neural Network);

ROC – Receiver Operating Characteristic analysis (Аналіз робочих характеристик приймача);

SMS – Сервіс Коротких Повідомлень (Short Message Service);

SVM – Машини опорних векторів (Support Vector Machines);

ПЗ – програмне забезпечення;

ПК – персональний комп'ютер.

ЗМІСТ с.

# ВСТУП

Сучасний світ неможливо уявити без обміну інформацією через глобальну інформаційну мережу. Величезний потік даних створює не тільки можливості для розвитку бізнесу та технологій, але й ризики для інформаційної безпеки. Однією з найбільш значущих загроз є соціальна інженерія – комплекс технік, спрямованих на маніпуляцію людьми для отримання несанкціонованого доступу до конфіденційної інформації. Витік інформації через соціальну інженерію може призвести до серйозних фінансових та репутаційних втрат для організацій.

Зростання складності інформаційних систем та збільшення кількості їх користувачів робить традиційні методи захисту недостатньо ефективними. В цьому контексті, машинне навчання виокремлюється як потужний інструмент, здатний аналізувати великі обсяги даних, виявляти складні закономірності та передбачати потенційні атаки до їх здійснення. Ця дипломна робота присвячена аналізу можливостей машинного навчання у виявленні та протидії каналам витоку інформації, створеним через соціальну інженерію.

Основною метою дослідження є визначення ефективності застосування різних алгоритмів машинного навчання для ідентифікації потенційних каналів витоку інформації, зокрема, в контексті соціальної інженерії. Дослідження спрямоване на аналіз різноманітних методів і моделей, які використовуються в галузі кібербезпеки для прогнозування та запобігання інформаційним загрозам.

# РОЗДІЛ 1 СТАН ПИТАННЯ. ПОСТАНОВКА ЗАДАЧІ

## 1.1 Соціальна інженерія у контексті інформаційної безпеки

Соціальна інженерія, яка поєднує в собі різноманітні методи маніпуляції людьми для отримання доступу до конфіденційної інформації, стає все більш розповсюдженою у сфері кібербезпеки. Ці методи використовують слабкі місця не в програмному забезпеченні чи апаратній частині, а в людській психології, експлуатуючи довіру та недостатню обізнаність користувачів.

Серед різних підходів, які використовуються в соціальній інженерії, найбільш поширеними є фішинг, вішинг та претекстинг. Кожен із цих методів має свою специфіку, але об'єднує їх спільна мета – обманом змусити жертву виконати дії, які ведуть до витоку конфіденційної інформації.

З розвитком інтернету та соціальних мереж соціальна інженерія набуває нових форм. Наприклад, атакуючи через соціальні мережі, зловмисники можуть використовувати зібрану інформацію про особу для створення переконливих фальшивих повідомлень або запитів.

Оскільки соціальна інженерія безпосередньо взаємодіє з людьми, це ставить перед організаціями складне завдання захисту своїх систем та даних. Не достатньо мати сильну технічну захисну стіну; необхідно також освічувати співробітників про ризики та методи протидії таким атакам.

Підсумовуючи, соціальна інженерія представляє собою серйозну загрозу у світі інформаційної безпеки. Вона вимагає не тільки технічних рішень, але й ретельного розуміння психології та поведінки людей.

Соціальна інженерія, будучи однією з найбільш хитромудрих форм кіберзлочинності, створює унікальний комплекс загроз для організацій усіх масштабів. Її наслідки варіюються від прямих фінансових втрат до більш суб'єктивних, але не менш руйнівних, як втрата репутації та довіри.

На сам перед від атак за допомогою соціальної інженерії страждають фінанси. Компанії часто зазнають прямого фінансового збитку в результаті шахрайських операцій, таких як незаконні перекази коштів або крадіжка фінансової інформації.

Наприклад, шахраї можуть використовувати інженерні методи для отримання доступу до банківських даних чи кредитних карток, що призводить до несанкціонованих транзакцій.

Крім фінансових збитків, соціальна інженерія становить серйозну загрозу конфіденційності даних. Шахраї можуть отримати доступ до важливих корпоративних документів, особистих даних клієнтів, а також логінів та паролів співробітників. Це не тільки порушує конфіденційність, але й ставить під загрозу цілісність і доступність даних, які є ключовими принципами інформаційної безпеки. Також компанії, що зазнали витоку даних через соціальну інженерію, можуть зіткнутися з юридичними наслідками, включно зі штрафами за порушення законодавства про захист даних. У багатьох країнах закон вимагає від компаній забезпечувати належний захист персональних даних, і в разі їх витоку організації можуть зіткнутися з великими штрафами, як, наприклад, у випадку з GDPR у Європейському Союзі.

Всі наслідки призводять до мабуть найбільш неприємного та найбільш тривалого наслідка атаки за допомогою соціальної інженерії є втрата довіри з боку клієнтів та партнерів. Коли інформація про витік стає відомою, це може негативно вплинути на репутацію компанії та знизити довіру до неї з боку громадськості. Особливо це стосується сфер, де довіра та конфіденційність інформації є ключовими, наприклад у фінансових послугах, охороні здоров'я та освіті.

Після атаки соціальної інженерії компанії змушені витрачати значні ресурси на розслідування та відновлення своїх систем. Це включає в себе аналіз інциденту, ідентифікацію вразливостей, які були використані, та вживання заходів для запобігання майбутнім атакам. Крім того, організації часто змушені наймати зовнішніх консультантів та експертів з кібербезпеки для відновлення втрачених даних та вдосконалення систем безпеки.

Тож соціальна інженерія становить серйозну загрозу у сучасному цифровому світі. Наслідки таких атак можуть бути довготривалими та різноманітними, від фінансових втрат до значної шкоди репутації та організаційної культури. Запобігання таким загрозам вимагає комплексного підходу, що включає як технічні рішення, так і освітні програми для підвищення обізнаності співробітників.

## 1.2 Загрози та наслідки

Соціальна інженерія, будучи однією з найбільш хитромудрих форм кіберзлочинності, створює унікальний комплекс загроз для організацій усіх масштабів. Її наслідки варіюються від прямих фінансових втрат до більш суб'єктивних, але не менш руйнівних, як втрата репутації та довіри.

На сам перед від атак за допомогою соціальної інженерії страждають фінанси. Компанії часто зазнають прямого фінансового збитку в результаті шахрайських операцій, таких як незаконні перекази коштів або крадіжка фінансової інформації. Наприклад, шахраї можуть використовувати інженерні методи для отримання доступу до банківських даних чи кредитних карток, що призводить до несанкціонованих транзакцій.

Крім фінансових збитків, соціальна інженерія становить серйозну загрозу конфіденційності даних. Шахраї можуть отримати доступ до важливих корпоративних документів, особистих даних клієнтів, а також логінів та паролів співробітників. Це не тільки порушує конфіденційність, але й ставить під загрозу цілісність і доступність даних, які є ключовими принципами інформаційної безпеки. Також компанії, що зазнали витоку даних через соціальну інженерію, можуть зіткнутися з юридичними наслідками, включно зі штрафами за порушення законодавства про захист даних. У багатьох країнах закон вимагає від компаній забезпечувати належний захист персональних даних, і в разі їх витоку організації можуть зіткнутися з великими штрафами, як, наприклад, у випадку з GDPR у Європейському Союзі.

Всі наслідки призводять до мабуть найбільш неприємного та найбільш тривалого наслідка атаки за допомогою соціальної інженерії є втрата довіри з боку клієнтів та партнерів. Коли інформація про витік стає відомою, це може негативно вплинути на репутацію компанії та знизити довіру до неї з боку громадськості. Особливо це стосується сфер, де довіра та конфіденційність інформації є ключовими, наприклад у фінансових послугах, охороні здоров'я та освіті.

Після атаки соціальної інженерії компанії змушені витрачати значні ресурси на розслідування та відновлення своїх систем. Це включає в себе аналіз інциденту, ідентифікацію вразливостей, які були використані, та вживання заходів для

запобігання майбутнім атакам. Крім того, організації часто змушені наймати зовнішніх консультантів та експертів з кібербезпеки для відновлення втрачених даних та вдосконалення систем безпеки.

Тож соціальна інженерія становить серйозну загрозу у сучасному цифровому світі. Наслідки таких атак можуть бути довготривалими та різноманітними, від фінансових втрат до значної шкоди репутації та організаційної культури. Запобігання таким загрозам вимагає комплексного підходу, що включає як технічні рішення, так і освітні програми для підвищення обізнаності співробітників.

### 1.2.1 Основні методи соціальної інженерії

Соціальна інженерія – складне явище, яке постійно змінюється, але все ж існують загальноприйняті методи атак, які мають свої особливості та специфіку. У розділі ми детально розглянемо ці основні методи соціальної інженерії. Важливо розуміти, що кожен з цих методів має свій унікальний підхід до маніпулювання та обману жертви, від використання лукавих фішингових електронних листів до більш прямих форм претекстингу чи бейтингу.

Ознайомлення з цими методами допоможе нам краще розуміти, як зловмисники використовують соціальні та психологічні трюки для досягнення своїх злочинних цілей, а також навчить, як цим загрозам можна запобігти або протистояти. Знання та розуміння цих методів є ключовим елементом у будуванні ефективної стратегії кібербезпеки.

Phishing (Фішинг) – це атака з використанням соціальної інженерії, де зловмисник маскується під надійний суб'єкт, щоб виманити конфіденційну інформацію від жертви. Такі атаки часто включають відправку обманних електронних листів, які імітують легітимні запити від відомих організацій або індивідів, з метою виманювання даних облікового запису, номера кредитної картки чи іншої конфіденційної інформації. Ціль фішингу – отримання цінних даних для використання у шахрайських цілях або продажу.

Концепція фішингу вперше була описана в 1987 році, а систематичні атаки почалися в мережі AOL у 1995 році [1]. Зловмисники часто видають себе за банки чи інші фінансові установи для змушення жертв заповнювати фальшиві форми та

отримання даних облікових записів. У минулому для виманювання даних часто використовувалися неправильно написані або оманливі доменні імена, але сучасні фішингові сторінки стали значно складнішими та візуально схожими на легітимні аналоги.

Vishing (Вішинг) – це форма соціальної інженерії, де шахраї телефонують жертвам, імітуючи іншу особу, щоб викрасти особисту інформацію або отримати гроші. Ця тактика ґрунтується на використанні соціальної інженерії та психології жертви, зловмисники можуть використовувати погрози або обіцянки вигоди, щоб змусити жертву виконати певні дії.

Шахраї також можуть використовувати "підміну" ідентифікатора абонента, щоб змусити жертву вірити, що дзвінок надходить з надійного джерела, наприклад, з легітимного бізнесу або урядового офісу. Іноді вони використовують особисту інформацію, отриману з інших джерел, щоб здатися більш переконливими.[2]

Існує багато видів вішингу, але їхня мета завжди одна – обдурити жертву для отримання інформації або грошей. Від 1 кварталу 2021 до 1 кварталу 2022 року кількість вішинг–атак зросла на 550%.[2]

Smishing (SMS–фішинг) – це тип фішингової атаки, який використовує текстові повідомлення (SMS) як засіб обману одержувачів. Термін "смішинг" був вперше введений в 2006 році. Незважаючи на свою ранню появу, смішингові атаки спочатку були відносно маловідомими, але останнім часом набули популярності через еволюцію тактики кіберзлочинності та зростаючу залежність від мобільних пристроїв.

Смішинг, по суті, є природним розвитком традиційного явища фішингу, яке історично було спрямоване на жертв через електронну пошту. Під час смішингових атак зловмисники надсилають фальшиві текстові повідомлення, що містять посилання, які виглядають легітимними, але ведуть на шахрайські сайти. Метою цих сайтів, як правило, є викрадення особистих облікових даних, розповсюдження мобільного шкідливого програмного забезпечення або шахрайство. Оманливі повідомлення часто створюють відчуття терміновості або використовують неправдиву інформацію, змушуючи одержувача діяти негайно.[3]

Особливо підступним смішинг робить використання людських вразливостей, таких як схильність занадто легко довіряти, діяти швидко та бути корисним. Цими аспектами людської природи маніпулюють, щоб обманом змусити жертв поставити під загрозу свою безпеку.

У світі, де переважає мобільний зв'язок, у поєднанні з тенденцією до віддаленої роботи, проактивний захист від атак зловмисників стає все більш важливим. Малий розмір екрану мобільних пристроїв та обмежені можливості перевірки посилань і вкладень перед тим, як натискати на них, наражають користувачів на більші ризики смішингу, ніж будь–коли раніше.

Pretexting (Претекстинг) – це метод соціальної інженерії, який включає створення переконливої, але вигаданої ситуації або історії з метою обману жертви для отримання конфіденційної інформації. Цей метод часто використовується для обходу стандартних захисних механізмів і викрадення даних або доступу до захищених систем.

У претекстингу зловмисник створює вигадану історію, щоб виглядати переконливим. Наприклад, він може вдавати з себе колегу по роботі, представника служби підтримки або офіційну особу. Ці сценарії часто включають деталі, які роблять історію реалістичною та викликають довіру у жертви.

Претекстинг може виконуватися через телефонні дзвінки, електронну пошту, соціальні мережі або навіть особисті зустрічі. Зловмисники часто ретельно досліджують свою жертву, щоб їхня історія була більш переконливою.

Наприклад, шахрай може зателефонувати співробітнику компанії, представившись IT–адміністратором, і попросити пароль для "оновлення системи". Або ж він може вдавати з себе банківського працівника, який потребує підтвердження банківських даних для "перевірки безпеки".

Загалом, претекстинг є витонченою формою соціальної інженерії, яка вимагає уважності та обережності для її розпізнавання та протидії.

Baiting (Бейтинг) є одним з методів соціальної інженерії, де зловмисники використовують привабливі пропозиції або обіцянки вигоди для виманювання конфіденційної інформації від жертв. Цей метод часто застосовується шляхом

надання безкоштовного програмного забезпечення або фізичних предметів, які насправді містять шкідливе ПЗ.

Метою бейтингу є залучення жертви за допомогою обіцянки чогось привабливого, такого як подарунки або ексклюзивний доступ до послуг або програм. Після того, як жертва взаємодіє з приманкою, наприклад, вставляючи заражений USB–накопичувач або завантажуючи програму, зловмисник отримує можливість встановити шкідливе програмне забезпечення або викрасти дані.

Концепція бейтингу виникла разом з розвитком інтернету, оскільки зловмисники шукали нові способи для проведення своїх атак. Спочатку використання фізичних носіїв, таких як дискети або USB–накопичувачі з шкідливим ПЗ, було поширеним методом бейтингу. З часом цей метод еволюціонував, і зловмисники почали пропонувати цифрові товари або послуги, такі як безкоштовне програмне забезпечення, що також може містити шкідливі компоненти.

Бейтинг вимагає від жертви активної взаємодії з приманкою, що відрізняє його від інших методів соціальної інженерії, таких як фішинг, де зловмисники частіше активно намагаються ввести в оману жертву. У випадку бейтингу, часто самі жертви роблять перший крок до взаємодії, спокусившись обіцянкою вигоди.

1.2.2 Статистика атак за допомогою соціальної інженерії

В контексті розгляду соціальної інженерії як серйозної загрози для інформаційної безпеки, важливо звернути увагу на статистичні дані, які ілюструють масштаб та вплив цього явища. Статистика не тільки підкреслює частоту і успішність таких атак, але й допомагає зрозуміти, які галузі та організації найбільше ризикують.

Враховуючи складність та різноманітність методів соціальної інженерії, статистичні дані дозволяють визначити ключові тренди та вектори атак, виявляючи найбільш уразливі місця в інформаційних системах. Така інформація є критично важливою для розробки ефективних стратегій захисту та навчання персоналу.

Далі наведено докладну статистику, яка відображає реальну картину атак соціальної інженерії, їх розвиток протягом останніх років, а також висвітлює

основні напрямки, по яких зловмисники намагаються реалізувати свої злочинні наміри.

Статистика Атак Соціальної Інженерії:

Згідно зі звітом IBM "Вартість витоку даних у 2023 році":

- Порушення даних, ініційовані методами соціальної інженерії, в середньому коштували понад $4,5 млн;
- Найпоширенішим вектором у звіті за 2022 рік були викрадені облікові дані, але фішинг зайняв перше місце з невеликим відривом від викрадених облікових даних;
- Внаслідок викрадення або компрометації облікових даних цього року в середньому знадобилося майже 11 місяців (328 днів), щоб виявити та локалізувати витоки даних, і близько 10 місяців (308 днів), щоб усунути витоки, ініційовані зловмисними інсайдерами.[4]

На основі останнього дослідження компанії Statista:

- У світі було виявлено 1 270 883 унікальних фішингових сайтів;
- Фішингові атаки найчастіше спрямовані на фінансові установи;
- За оцінками, 18% атак спрямовані на веб–програмне забезпечення та веб–пошту;
- Смішингові (SMS–фішинг) атаки націлені на 76% світових компаній;
- Лише у жовтні 2022 року фішингові атаки торкнулися 599 брендів.

На соціальну інженерію припадає 98% усіх кібератак.[5]

Останній звіт ФБР про інтернет–злочини показує, що на фішинг, вішинг, смішинг і фармінг припадає найбільша кількість жертв – 323 972.[6]

Згідно зі звітом Verizon про розслідування витоків даних за 2022 рік:

- 82% порушень були спричинені людськими помилками;
- Незважаючи на те, що лише 2,9% співробітників можуть натиснути на фішингові електронні листи, і ця статистика залишається відносно стабільною з плином часу, зловмисники все одно продовжують користуватися цим.[7]

Згідно зі звітом Spear Phishing від Barracuda:

- Щорічно керівники компаній отримують 57 цілеспрямованих фішингових

атак;

- Щороку IT–персонал отримує в середньому 40 цільових фішингових атак;
- Компрометації ділової електронної пошти (BEC) становлять одну з десяти атак соціальної інженерії;
- Типова організація щороку зазнає понад 700 атак соціальної інженерії.
- У другому кварталі 2023 року Microsoft очолила список брендів, які найчастіше використовують для фішингових атак. (MSSP Alert)[8]

Згідно з Індексом цифрової довіри та безпеки Sift за 3 квартал 2023 року:

- У 2023 році кількість атак на акаунти зросла на 354% порівняно з попереднім роком.;
- 73% споживачів вважають, що бренд несе відповідальність за атаки на акаунти та захист облікових даних, і тому вони відмовляються від нього;
- Серед жертв крадіжки акаунтів лише 43% отримали повідомлення від компанії;
- 24% жертв шахрайства із захопленням акаунтів змінили свої контактні дані після інциденту (наприклад, адресу електронної пошти або номер телефону).[9]

У першій половині 2023 року зловмисники вимагали рекордні $176 млн, що зробило цей рік другим найдорожчим в історії зловмисників.

У 2020 році компаніям електронної комерції знадобилося в середньому 250 годин на відновлення після захоплення акаунтів.

Захоплення акаунтів у 75% випадків пов'язане з атаками "підміни облікових даних".[10]

Тож зважаючи на світову статистику кількість атак соціальної інженерії продовжує зростати з кожним роком. Це пояснюється збільшенням цифрової активності людей та організацій, а також постійним удосконаленням тактик шахраїв. Зростання складності та винахідливості атак підкреслює важливість постійного оновлення методів захисту та обізнаності співробітників щодо потенційних загроз.

1.3 Захист від соціальної інженерії

Для захисту від атак соціальної інженерії необхідно поєднувати освітні програми для персоналу з технічними засобами безпеки. Тренінги повинні включати

інформацію про різні методи соціальної інженерії, як їх розпізнавати та реагувати на них. З технічного боку, важливо використовувати фільтри антифішингу, багаторівневу аутентифікацію, шифрування даних тощо.

Забезпечення захисту від атак соціальної інженерії вимагає багатогранного підходу, який охоплює як освітні, так і технічні аспекти. Ефективна стратегія захисту поєднує зусилля у сфері навчання персоналу з впровадженням передових технологічних рішень. Далі будуть наведені приклади рішень за допомогою яких можливо протистояти атакам .

Освітні програми та тренінги для персоналу:

- Тренінги можуть включати симуляції атак, інтерактивні навчальні сесії, відеоматеріали та тестування знань співробітників. Це допомагає їм краще розуміти, як виглядають атаки у реальному житті, та як на них реагувати;

Технічні засоби безпеки:

- Фільтри антифішингу: сучасні рішення для антифішингу використовують алгоритми машинного навчання для аналізу електронних листів та ідентифікації шахрайських спроб;
- Багаторівнева аутентифікація: використання додаткових методів перевірки особи, наприклад, через одноразові паролі або біометричні дані, значно знижує ризик несанкціонованого доступу;
- Шифрування даних: захист даних за допомогою шифрування запобігає їх витоку у випадку компрометації системи;
- Регулярне оновлення ПЗ: важливо постійно оновлювати всі системи та програмне забезпечення, щоб запобігти використанню відомих вразливостей.

Програмні рішення:

- Інструменти для автоматизованого відповіді на інциденти: системи, які автоматично реагують на інциденти безпеки, здатні швидко ізолювати загрози та мінімізувати їх вплив;
- Інструменти для розширеного аналізу поведінки користувачів: Використання поведінкових аналітичних інструментів для виявлення незвичайних або підозрілих дій користувачів, що можуть вказувати на втручання ззовні.

Створення культури безпеки в організації:

- Розвиток культури безпеки серед персоналу є фундаментальним аспектом захисту від соціальної інженерії. Це включає створення середовища, де безпека є пріоритетом для кожного співробітника, незалежно від їх ролі чи позиції;
- Залучення вищого керівництва до процесу навчання та просвітництва з питань кібербезпеки сприяє формуванню відповідального ставлення на всіх рівнях організації.

Але ці рішення не завжди працюють, так як люди не передбачувані, деякі співробітники можуть нехтувати правилами, ігнорувати тренінги, чи банально помилятись. Саме через свою непердбачуваність людина це найслабша ланка в інформаційній безпеці. Тому для певності у безпеці і точності захисту додатково краще використовувати машинне навчання.

1.3.1 Роль машинного навчання у протидії соціальній інженерії

Машинне навчання може використовуватися для аналізу великих обсягів даних на предмет виявлення шахрайської поведінки та підозрілої активності. Наприклад, алгоритми можуть аналізувати шаблони електронних листів, щоб ідентифікувати потенційні фішингові спроби, або використовувати методи обробки природної мови для оцінки змісту комунікацій на предмет маніпулятивних технік. Це включає в себе використання різних моделей, таких як нейронні мережі, випадкові ліси, градієнтний бустинг та інші, для виявлення і класифікації потенційних загроз.

1.3.2 Переваги використання машинного навчання

Застосування машинного навчання в галузі кібербезпеки відкриває нові горизонти в боротьбі з кіберзлочинністю, особливо в контексті соціальної інженерії. Його унікальні переваги дозволяють ефективно виявляти та протидіяти складним кіберзагрозам, які традиційні методи безпеки можуть не виявити.

Наведемо перелік головних переваг машинного навчання:

1. Виявлення складних шаблонів:

Машинне навчання здатне аналізувати великі масиви даних та виявляти складні шаблони, які можуть вказувати на атаки соціальної інженерії. Це включає в

себе не тільки текстовий аналіз, але й виявлення підозрілої поведінки користувачів, незвичайних транзакцій або неправильного використання системних ресурсів.

2. Адаптивність та гнучкість:

Системи машинного навчання можуть навчатися на основі нових даних, що робить їх надзвичайно адаптивними до змінних методів кібератак. Це особливо важливо у випадках соціальної інженерії, де тактики шахраїв постійно еволюціонують.

3. Прогностична аналітика:

Використання алгоритмів машинного навчання дозволяє не тільки реагувати на поточні загрози, але й прогнозувати майбутні атаки на основі аналізу тенденцій та шаблонів.

4. Виявлення непомітних аномалій:

Машинне навчання ефективно у виявленні тонких аномалій у поведінці користувачів або в структурі даних, які можуть свідчити про спробу шахрайства. Це включає в себе аналіз звичних шаблонів доступу до даних, використання системних ресурсів та інших нечітких показників.

5. Підтримка рішень щодо безпеки:

З використанням машинного навчання кібербезпека переходить від реактивних до превентивних заходів. Системи можуть автоматично блокувати підозрілі дії або навіть рекомендувати конкретні кроки для підвищення безпеки.

6. Оптимізація ресурсів:

Використання машинного навчання також дозволяє оптимізувати використання ресурсів, зосереджуючи увагу на тих областях, де ризик кібератак найвищий, тим самим підвищуючи ефективність кіберзахисту.

Машинне навчання надає значні переваги у сфері кібербезпеки, забезпечуючи компанії потужним інструментом для ідентифікації та протидії складним атакам соціальної інженерії. Впровадження цих технологій може значно підвищити здатність організацій захищатися від сучасних кіберзагроз.

### 1.3.3 Моделі машинного навчання для протидії соціальній інженерії

Щоб зрозуміти, як машинне навчання може допомогти в захисті від соціальної інженерії, треба розглянути всі його існуючі можливості та моделі. Машинне навчання відкриває широкі перспективи для ідентифікації та нейтралізації таких атак, використовуючи сучасні алгоритми та методи аналізу даних. Далі ми розглянемо декілька ключових моделей машинного навчання, які здатні ефективно виявляти та протистояти загрозам соціальної інженерії.

Нейронні мережі в машинному навчанні імітують роботу людського мозку, аналізуючи і вирішуючи складні завдання, такі як розпізнавання зображень чи обробка природної мови. Їхня структура організована у вигляді шарів, що містять вузли або "нейрони", подібно до нейронних з'єднань у мозку. Кожен шар – від вхідного до вихідного – має свою роль у обробці інформації. Ваги та зсуви в цих мережах допомагають моделі адаптуватися і вчитися, використовуючи функції активації, які визначають активність нейронів.

Процес навчання цих мереж включає коригування ваг на основі помилок у прогнозах, з використанням методу зворотного поширення помилки. Ця здатність до адаптації робить нейронні мережі ідеальними для виявлення шахрайських атак або змін у поведінці, що можуть вказувати на соціальну інженерію. Це може включати аналіз тексту в електронних листах, повідомленнях у соцмережах чи виявлення шахрайських спроб через аналіз зображень чи відео.

Однак, важливо враховувати певні обмеження нейронних мереж, як–от необхідність великих обсягів даних для ефективного навчання та складнощі у розумінні того, як саме мережа приходить до певних висновків. Крім того, ці мережі можуть бути вразливими до певних видів атак, спрямованих на введення їх в оману. Незважаючи на ці виклики, нейронні мережі залишаються важливим інструментом у боротьбі з кіберзагрозами, зокрема, соціальною інженерією, завдяки їхній гнучкості та глибокому аналітичному розумінню.

Глибоке навчання представляє собою розширену та більш складну форму нейронних мереж. Воно використовує глибокі нейронні мережі, що включають багато прихованих шарів, дозволяючи моделям виявляти складні і абстрактні зразки

в даних. Цей підхід стає особливо ефективним, коли йдеться про роботу з великими обсягами неструктурованих або складних даних, таких як зображення, звуки чи текст.

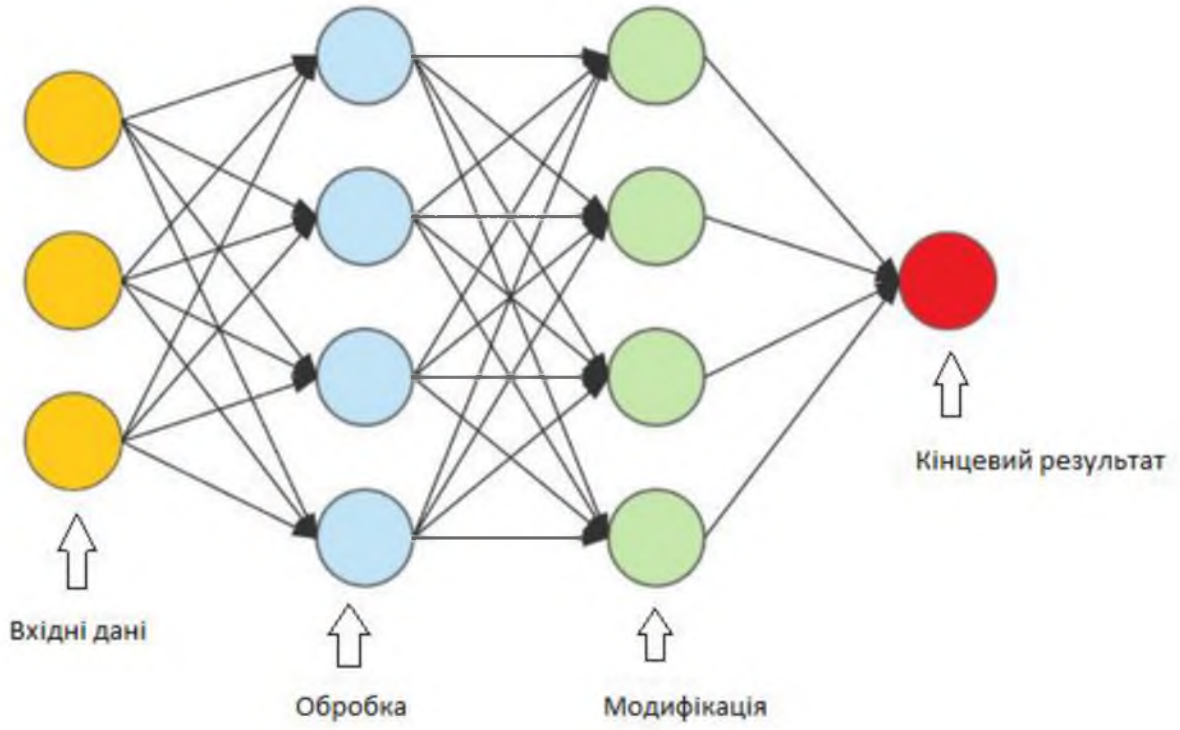



Рисунок 1.1 – Модель нейронної мережі

У контексті кібербезпеки, глибоке навчання використовується для виявлення шахрайської діяльності, включаючи засоби соціальної інженерії. Його здатність до обробки природної мови дозволяє аналізувати комунікації на предмет фішингу чи інших видів обману.

Конволюційні нейронні мережі (Convolutional Neural Network, CNN) є спеціалізованим типом нейронних мереж, оптимізованих для аналізу візуальних даних. Вони автоматично та ефективно виявляють важливі особливості на зображеннях, від базових форм до складніших об'єктів і шаблонів. Це робить CNN особливо цінними для аналізу зображень та відео в контексті кібербезпеки, наприклад, для виявлення підроблених ідентифікаційних документів або шахрайських дій.

Загалом, глибоке навчання та CNN грають ключову роль у сучасній кібербезпеці, надаючи потужні інструменти для виявлення складних та витончених форм кіберзагроз, включно з атаками за допомогою соціальної інженерії. Вони допомагають виявляти та протидіяти загрозам, які традиційні методи безпеки можуть

пропустити, завдяки своїй здатності аналізувати великі обсяги даних і виявляти складні поведінкові шаблони.

Випадкові ліси (Random Forests) є потужною технікою машинного навчання, яка поєднує декілька дерев рішень для створення більш стійкої та точної моделі. У контексті кібербезпеки, вони знайшли своє застосування завдяки своїй здатності до класифікації та регресії, що дозволяє аналізувати великі набори даних і виявляти потенційні загрози.

Однією з ключових особливостей випадкових лісів є їхня здатність до агрегування висновків від множини дерев рішень, що зменшує ризик перенавчання і покращує загальну точність моделі. Кожне дерево в лісі навчається на випадковій підвибірці даних, що робить процес навчання розподіленим і більш стійким до змін у даних.

У випадкових лісах кожне дерево рішень генерує свій прогноз, і найпоширеніший прогноз серед усіх дерев стає кінцевим висновком моделі. Ця техніка, відома як "bagging" або bootstrap aggregating, допомагає уникнути проблем, пов'язаних із зміщеністю або варіативністю, які часто спостерігаються у одиночних деревах рішень.

У кібербезпеці випадкові ліси використовуються для аналізу та класифікації різноманітних типів даних, від мережевого трафіку до поведінкових шаблонів користувачів. Вони здатні виявляти неправильні або підозрілі дії, які можуть вказувати на кібератаки або спроби соціальної інженерії.

Завдяки їхній здатності до ефективної роботи з великими даними та високої точності у класифікації, випадкові ліси стали однією з ключових технологій в сучасних системах кібербезпеки, дозволяючи компаніям краще зрозуміти та реагувати на постійно еволюціонуючі кіберзагрози.

Градієнтний бустинг є методом машинного навчання, що використовується для побудови прогностичних моделей, особливо в задачах класифікації та регресії. Цей метод входить до категорії ансамблевих методів, де кілька слабких моделей (часто дерев рішень) комбінуються для створення більш потужної та точної прогностичної моделі.

Основна ідея градієнтного бустингу полягає у послідовному додаванні моделей до ансамблю, де кожна наступна модель коригує помилки попередньої. Це відбувається шляхом застосування градієнтного спуску до функції втрат для мінімізації помилок. Завдяки цьому, градієнтний бустинг дозволяє покращити прогностичну здатність комплексу моделей, адаптуючись до складних даних і різноманітних закономірностей.

Градієнтний бустинг часто використовується в сфері кібербезпеки, зокрема для аналізу мережевого трафіку, виявлення шахрайства або прогнозування кібератак. Він ефективно виявляє складні шаблони в даних, які можуть бути ознаками зловмисної діяльності або спробами соціальної інженерії.

З точки зору реалізації, градієнтний бустинг вимагає ретельного підбору параметрів, таких як глибина дерева, швидкість навчання та кількість ітерацій, щоб уникнути перенавчання та досягти оптимальної продуктивності.

Машини опорних векторів (Support Vector Machines, SVM) є популярним методом у галузі машинного навчання, який використовується для класифікації та регресійних задач. Цей метод заснований на ідеї знаходження гіперплощини у високовимірному просторі, яка найкращим чином розділяє класи даних.

Основна ідея SVM полягає у максимізації відстані (маржі) між гіперплощиною та найближчими до неї точками даних з кожного класу, відомими як опорні вектори. Це дозволяє SVM ефективно розділити дані, навіть коли вони є лінійно нероздільними у вихідному просторі, за допомогою використання так званих ядерних трюків (kernel tricks), які перетворюють дані у простір більшої розмірності.

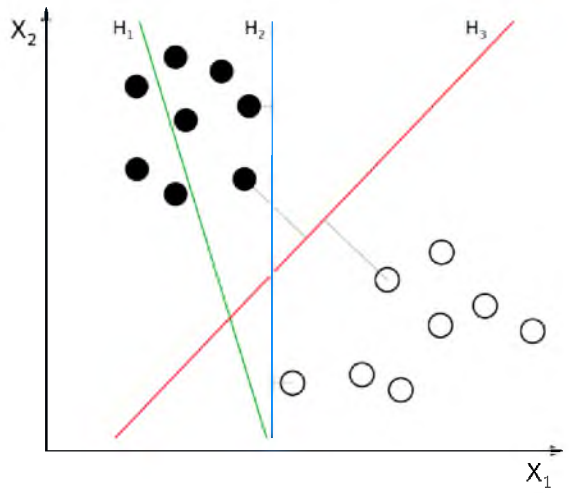



Рисунок 1.2 – Розділення данних у SVM

SVM широко використовуються у різних сферах, включаючи розпізнавання образів, біоінформатику та текстовий аналіз. У контексті кібербезпеки, SVM може застосовуватися для виявлення шахрайських дій, класифікації мережевого трафіку та ідентифікації малвару.

Однак, важливо зауважити, що SVM можуть вимагати значних обчислювальних ресурсів, особливо при роботі з великими наборами даних. Також вони менш ефективні у задачах з великою кількістю шуму в даних.

Логістична регресія – це статистичний метод, який використовується в машинному навчанні для розв'язання задач бінарної класифікації. Вона є розширенням звичайної лінійної регресії, але відрізняється тим, що передбачає ймовірність приналежності спостереження до одного з двох класів, а не безперервне значення.

Замість того, щоб прямо моделювати відповідь з використанням лінійної функції, як у випадку з лінійною регресією, логістична регресія використовує логістичну функцію для обмеження виведення моделі у діапазоні між 0 та 1. Це робить її ідеальною для задач, де треба вирішити, належить спостереження до одного класу чи до іншого (наприклад, чи електронний лист є спамом).

У контексті кібербезпеки, логістична регресія може застосовуватися для виявлення фішингових веб–сайтів, аналізу шкідливого програмного забезпечення та інших форм кіберзлочинності, де задача полягає у класифікації елементів на «шкідливі» або «безпечні».

Модель логістичної регресії включає оцінку вагових коефіцієнтів для різних ознак. Ці коефіцієнти визначають вплив кожної ознаки на ймовірність належності до певного класу. Процес навчання моделі полягає у мінімізації помилок передбачення, часто за допомогою методів, таких як градієнтний спуск.

Однак, важливо враховувати, що логістична регресія найкраще працює з даними, що мають лінійні відносини, і може бути менш ефективною у випадках, коли відносини між ознаками та класами є складнішими чи нелінійними.

Рішення дерев (Decision Trees) є одним із фундаментальних алгоритмів у машинному навчанні. Вони працюють за принципом серії рішень, які можна порівняти з розгалуженням дерева, де кожен вузол представляє рішення, а кожна гілка – можливий результат цього рішення. Ці дерева використовуються для класифікації або регресії, тобто вони можуть передбачати дискретні класи або неперервні значення відповідно.

Ключовим елементом рішення дерев є їхня структура. Вони складаються з кореневого вузла, з якого починається дерево, внутрішніх вузлів, які представляють рішення на основі певних ознак, і листя, які є кінцевими вузлами, представляючи передбачення або висновки. Процес передбачення здійснюється шляхом проходження від кореневого вузла до листа, приймаючи рішення на кожному етапі.

У контексті кібербезпеки, рішення дерев можуть використовуватися для ідентифікації шкідливої активності або класифікації типів загроз на основі різних характеристик або поведінкових показників. Вони особливо корисні, коли необхідно чітко розуміти логіку прийняття рішень, оскільки структура дерева є прозорою та інтуїтивно зрозумілою.

Однак, рішення дерев мають свої обмеження. Вони схильні до перенавчання, особливо у випадках, коли дерева стають занадто складними. Це може призвести до того, що модель добре працює на тренувальних даних, але показує погані результати на нових, невідомих даних. Ще однією проблемою є стійкість: невеликі зміни в даних можуть призвести до значної зміни в структурі дерева.

Кластеризація – це процес групування або сегментації набору даних на підмножини, відомі як кластери, таким чином, що об'єкти в одному кластері мають

схожі характеристики або поведінку, а об'єкти з різних кластерів – відрізняються. Головна ідея полягає в тому, що об'єкти всередині одного кластера повинні бути більш схожими один на одного, ніж на об'єкти в інших кластерах.

Кластеризація широко застосовується в різних областях, включаючи кібербезпеку, для розпізнавання і аналізу шаблонів або аномалій у даних. Наприклад, вона може бути використана для ідентифікації нетипових поведінкових патернів, які можуть вказувати на кібератаки або внутрішні загрози.

У кібербезпеці кластеризація може допомогти у виявленні шкідливого програмного забезпечення, аналізі мережевого трафіку для виявлення нестандартної активності або виявленні вразливостей шляхом групування схожих інцидентів безпеки. Це дозволяє фахівцям з кібербезпеки швидше реагувати на потенційні загрози, аналізуючи характеристики кластерів, замість того, щоб аналізувати кожен випадок окремо.

Однак, важливо зазначити, що кластеризація не завжди є точною, і вибір методу кластеризації та його параметрів може сильно вплинути на результати. Наприклад, різні алгоритми кластеризації, такі як k–середньої, ієрархічна кластеризація або DBSCAN (Density–Based Spatial Clustering of Applications with Noise), мають свої переваги та обмеження і можуть бути підходящими для різних типів даних та завдань.

Використання кластеризації в машинному навчанні для кібербезпеки вимагає глибокого розуміння даних та обережного вибору методів, щоб забезпечити точне та ефективне виявлення загроз.

У сфері кібербезпеки та протидії соціальній інженерії важливо вибирати відповідні моделі машинного навчання, оскільки кожна з них має свої унікальні переваги. Нейронні мережі, наприклад, ефективні для розпізнавання складних шаблонів у великих даних, тоді як випадкові ліси чи градієнтний бустинг забезпечують сильну класифікацію та регресію.

Вибір моделі залежить від конкретних вимог та особливостей задачі. Наприклад, для ідентифікації шахрайських атак може бути потрібна модель з високою

точністю, в той час як для швидкого сканування великих наборів даних краще підійде простіша та швидша модель.

Ефективність захисту від кіберзагроз збільшується за рахунок інтеграції різних моделей, що дозволяє створювати гнучкі та адаптивні системи безпеки. Такий підхід допомагає краще зрозуміти та протистояти різним формам кібератак, забезпечуючи комплексний захист у сфері кібербезпеки.

1.3.4 Чому машинне навчання є кращим засобом протидії соціальній інженерії

Використання машинного навчання дозволяє системам безпеки швидко адаптуватися до нових та розвиваючихся загроз. Традиційні методи безпеки часто базуються на відомих підписах шкідливого ПЗ або відомих техніках атак, тоді як методи машинного навчання можуть виявляти атаки на основі поведінкових шаблонів та аномалій, навіть якщо конкретний метод атаки раніше не зустрічався.

1.4 Застосування машинного навчання у реальних сценаріях

Машинне навчання стає ключовим інструментом у боротьбі з соціальною інженерією, оскільки воно забезпечує гнучкість та адаптивність, яких не можуть надати традиційні методи безпеки. Сучасні кіберзагрози постійно еволюціонують, роблячи традиційні системи, які покладаються на статичні правила та підписи шкідливих програм, менш ефективними.

Машинне навчання дозволяє ідентифікувати та реагувати на нові та невідомі атаки шляхом аналізу поведінкових шаблонів та аномалій. Це означає, що системи безпеки, засновані на машинному навчанні, можуть виявляти потенційні загрози, навіть якщо вони не відповідають жодному відомому шаблону атаки.

Такі системи можуть аналізувати великі обсяги даних, виявляючи складні зразки, які можуть вказувати на шахрайські дії, такі як фішингові листи, атаки на основі претекстингу, або інші види маніпуляцій. Вони також можуть використовуватися для моніторингу мережевої активності, виявляючи незвичайні зміни в поведінці, які можуть вказувати на внутрішні загрози або компрометацію акаунтів.

Крім того, машинне навчання надає можливість швидкої адаптації до нових загроз. Алгоритми можуть навчатися на нових даних та вдосконалюватися з часом,

що робить системи безпеки більш гнучкими та ефективними у виявленні та протидії соціальній інженерії.

Використання машинного навчання у кібербезпеці забезпечує не тільки високий рівень виявлення загроз, але й сприяє швидкій адаптації до змін у тактиці кіберзлочинців, забезпечуючи комплексний та актуальний захист.

Висновок:

Застосування машинного навчання у сфері кібербезпеки є критично важливим у боротьбі з соціальною інженерією. Воно не тільки підвищує ефективність ідентифікації загроз, але й дозволяє компаніям підтримувати крок зі швидкозмінним ландшафтом кіберзагроз. Однак, слід пам'ятати, що жодна технологія не може забезпечити повний захист. Комплексний підхід, який включає освіту персоналу, суворі політики безпеки та передові технології, є ключовим для ефективного захисту від загроз соціальної інженерії.

# РОЗДІЛ 2 СПЕЦІАЛЬНА ЧАСТИНА

## 2.1 Соціальна інженерія у соціальних мережах

Соціальні мережі відіграють ключову роль у атаках за допомогою соціальної інженерії, оскільки вони є справжнім сховищем інформації про людей. Люди часто навіть не усвідомлюють, наскільки цінною може бути інформація, яку вони діляться: від особистих інтересів до професійних зв'язків. Зловмисники використовують цю відкритість, створюючи фальшиві профілі для встановлення довіри з потенційними жертвами або навіть імітуючи реальних людей чи організації.

Через публікації та особисті повідомлення в соціальних мережах зловмисники можуть ефективно маніпулювати емоціями людей, створюючи ілюзію згоди або довіри, щоб змусити їх виконувати певні дії. Це може включати натискання на шкідливі посилання або передачу конфіденційної інформації. Іноді ці атаки стають високо персоналізованими, використовуючи зібрану інформацію для створення дуже переконливих фішингових повідомлень.

Соціальні мережі також дозволяють зловмисникам швидко поширювати шкідливі посилання або інформацію серед великої аудиторії, збільшуючи ефективність масових атак. Крім того, вони можуть використовуватися для підвищення власного авторитету або надійності через велику кількість "друзів" або підписників.

Таким чином, захист від таких атак вимагає не тільки технічних засобів, а й підвищеної обережності, критичного мислення щодо інформації, що зустрічається в соцмережах, та освіти користувачів про потенційні загрози.

Відкриті канали витоку і як їми скористуються

## 2.2 Аналіз потенційних джерел витоку у соціальних мережах

Відкриті канали витоку інформації у соціальних мережах представляють собою різноманітні способи, за допомогою яких особиста, конфіденційна або важлива інформація може стати доступною для сторонніх, включаючи зловмисників. Ці

канали мають велику цінність для зловмисників, оскільки вони надають інформацію, яку можна використовувати для різних мет:

1. Особиста інформація користувачів: це включає імена, дати народження, адреси, інформацію про роботу, сімейний стан та інші особисті деталі. Така інформація може використовуватися для ідентифікації або верифікації особи в інших контекстах.

2. Деталі професійного життя: інформація про роботодавців, професійні навички та кар'єрні досягнення може допомогти зловмисникам у створенні переконливих фішингових атак, спрямованих на конкретні компанії або індустрії.

3. Мережа контактів: знання про друзів, колег і зв'язки може використовуватися для створення мережевих атак, де зловмисники видають себе за одного з контактів жертви.

4. Інформація про інтереси та поведінку: Знання про хобі, інтереси та поведінкові звички користувачів може допомогти у виборі стратегій маніпулювання та в створенні цільового вмісту.

5. Візуальна інформація: фотографії та відео, які користувачі розміщують у своїх профілях, можуть викривати важливу інформацію про їхній спосіб життя, місцезнаходження, звички, а також надати матеріал для підробки документів або створення фальшивих профілів.

6. Геолокаційні дані: інформація про місцезнаходження, яка часто включається в пости та фотографії, може використовуватися для визначення звичок переміщення особи, її місцеперебування в певний час та планування фізичних або цифрових атак.

Зловмисники можуть комбінувати цю інформацію для створення детальних профілів потенційних жертв, підготовки персоналізованих атак, таких як соціальна інженерія або фішинг, а також для широкомасштабних кампаній, спрямованих на велику кількість користувачів. Тому важливо обережно ставитися до того, яку інформацію ми ділимося в соціальних мережах, та використовувати налаштування приватності для контролю над тим, хто може бачити цю інформацію.

### 2.2.1 Приклад витоків у твіттер

Щоб точніше зрозуміти які саме данні можуть представляти відкриті канали витоку, використаємо соціальну мережу твітер для наглядного прикладу. Твітер, як і багато інших соціальних мереж, містить кілька відкритих каналів, через які може відбуватися витік інформації. Ось кілька прикладів, як це може відбуватися на платформі Твітер:

1. Твіти: очевидно, що основним каналом витоку інформації є самі твіти. Користувачі часто діляться особистою інформацією, такою як місцеперебування, події з особистого життя, думки та враження, які можуть надати зловмисникам цінну інформацію.

2. Біографія користувача: у біографії користувачі часто вказують професійну інформацію, місцеперебування та інші особисті деталі. Ця інформація може бути використана для ідентифікації чи створення цільових атак.

3. Зображення та відео: фотографії та відео, які користувачі розміщують у своїх твітах, можуть виявити багато інформації про їхній спосіб життя, сім'ю, роботу, захоплення, а також можуть містити геолокаційні метадані.

4. Лісти підписки та підписники: аналізуючи, на кого підписаний користувач і хто підписаний на користувача, можна визначити соціальну мережу особи, її інтереси та потенційні професійні зв'язки.

5. Відкриті діалоги: Через відповіді на твіти, ретвіти та публічні бесіди можна визначити міжособистісні відносини, думки та реакції користувачів, що може бути використано для маніпулювання або мішені атак.

6. Геотеги у твітах: Якщо користувачі використовують геотеги у своїх твітах, це може вказувати на їхнє точне місцеперебування або звичайні місця відвідування, що є цінною інформацією для фізичного стеження або цифрових атак. Усвідомлення цих потенційних каналів витоку інформації та обережне використання соціальних мереж, зокрема Твітера, може значно знизити ризик стати жертвою атаки через соціальну інженерію.

## 2.3 Машинне навчання для перевірки соціальних мереж

Протистояння відкритим каналам витоку інформації та атакам соціальної інженерії у соціальних мережах за допомогою машинного навчання включає розробку та використання алгоритмів, які можуть ідентифікувати та запобігати потенційним загрозам. Ось декілька способів, як машинне навчання може бути використане в цьому контексті:

1. Виявлення аномальної поведінки: Моделі машинного навчання можуть аналізувати поведінку користувачів у соціальних мережах, виявляючи незвичайні зміни або аномалії, які можуть вказувати на компрометацію акаунта або маніпулятивну діяльність.

2. Виявлення фішингових та шкідливих посилань: Алгоритми машинного навчання можуть бути навчені виявляти фішингові атаки та шкідливі посилання, аналізуючи характеристики контенту та метадані посилань, що діляться у соціальних мережах.

3. Аналіз контенту: Машинне навчання може допомогти у виявленні та фільтрації маніпулятивного або шкідливого контенту, включаючи дезінформацію, хибну інформацію та контент, спрямований на соціальну інженерію.

4. Ідентифікація фальшивих акаунтів та ботів: Алгоритми машинного навчання можуть допомогти в ідентифікації фальшивих акаунтів та ботів, які часто використовуються для поширення шкідливого контенту або проведення атак соціальної інженерії.

5. Прогнозування та запобігання загрозам: З використанням великих даних та аналітичних інструментів, системи машинного навчання можуть прогнозувати потенційні загрози на основі вивчення тенденцій та шаблонів у соціальних мережах.

6. Навчання користувачів: Машинне навчання може використовуватися для створення персоналізованих попереджень та навчальних програм для користувачів, підвищуючи їхню обізнаність та вміння розпізнавати атаки соціальної інженерії.

7. Автоматизація відповідей: Машинне навчання може допомогти автоматизувати процеси реагування на виявлені загрози, швидко блокуючи або обмежуючи доступ до шкідливого контенту.

Застосування машинного навчання вимагає постійного оновлення та адаптації моделей, оскільки тактики зловмисників постійно еволюціонують. Це також вимагає балансу між ефективністю виявлення загроз і збереженням конфіденційності та приватності користувачів.

2.3.1 Задача машинного начання

Коли співробітники діляться інформацією в інтернеті, особливо у соціальних мережах, вони можуть, часто навіть не усвідомлюючи, викривати деталі, що стають корисними для зловмисників. Розглянемо приклад: публікації про робочі проекти або деталі відряджень містять інформацію, яка може бути використана для розробки переконливих фішингових атак або інших форм маніпуляції.

Саме тут алгоритми машинного навчання відіграють ключову роль. Вони здатні швидко обробляти великі обсяги даних, виявляючи можливі ризики. Наприклад, ці алгоритми можуть моніторити зміни в поведінці співробітників на соціальних платформах, що може свідчити про їх вразливість до зовнішніх впливів або маніпуляцій. Також важливим є відстеження комунікації співробітників із підозрілими акаунтами або групами, що може бути ознакою потенційної загрози.

Ці системи машинного навчання допомагають створити захисний бар'єр, аналізуючи публічну інформацію та виявляючи попередні ознаки атак. Вони функціонують як розумний охоронець, який постійно аналізує поведінку співробітників у цифровому просторі і попереджає про можливі ризики, що дозволяє компаніям швидко реагувати та вживати заходів до того, як відбудеться реальна атака.

Використання машинного навчання для моніторингу соціальних мереж стає невід'ємною частиною стратегії кібербезпеки, дозволяючи компаніям зміцнювати свою оборону від складних і постійно еволюціонуючих кіберзагроз.

Тож зупинимось на аналізі контенту співробітників задля передбачення і виявлення атак соціальної інженерії.

## 2.4 Реалізація навчання моделей

Для реалізації системи, що використовує машинне навчання для аналізу соціальних мереж співробітників, ключовими є два аспекти: збір даних та обрання моделей машинного навчання.

Збір даних – це фундамент усієї системи. Нам потрібно зібрати достатньо інформації з соціальних мереж, щоб мати комплексне розуміння того, як співробітники взаємодіють у цифровому просторі. Це означає аналізувати їхні публікації, коментарі, мережу контактів, а також реакції на різний контент. Важливо забезпечити, що збір даних відбувається у відповідності з правилами конфіденційності та законодавством.

Моделі машинного навчання – це серце системи. Вони повинні бути обрані так, щоб ефективно обробляти зібрані дані та виявляти потенційні ризики. Ці моделі можуть аналізуюти поведінкові шаблони, визначають аномалії та оцінювати потенційні загрози. Важливим аспектом є здатність моделей адаптуватися до нових тенденцій та еволюціонувати разом зі змінами у способах ведення атак соціальної інженерії.

Тому для обрання моделей які найкраще підійшли для задачі аналізу контенту з соціальних мереж порівняємо роботу декількох популярних моделей на даних взятих з соціальної мережі Twitter. Основною задачею моделей буде аналіз тексту з метою виявлення окремих користувачів і їх повідомлень.

## 2.5 Створення даних для навчання моделей

Датасет – це зібрання даних, яке використовується для обробки, аналізу та, зокрема, у сфері машинного навчання, для тренування моделей. У контексті машинного навчання, датасет складається з набору записів, які можуть включати різноманітні типи даних, такі як числа, текст, зображення або навіть більш складні дані, як–от аудіо чи відео.

Ключовою характеристикою датасету є його структурованість. Це означає, що дані організовані у певному порядку або форматі, який дозволяє системам машинного навчання ефективно їх обробляти та аналізувати. Наприклад, датасет може бути у формі таблиці, де рядки представляють окремі записи (наприклад, окремі

спостереження або випадки), а стовпці – різні атрибути або характеристики цих записів.

У контексті аналізу соціальних мереж, датасет може включати такі дані, як текстові пости, коментарі, метадані (дата та час публікації, інформація про автора посту тощо), відомості про взаємодії (наприклад, лайки, репости), а також може включати мережеві дані, які представляють зв'язки між користувачами.

Для тренування моделей машинного навчання, важливо, щоб датасет був не тільки багатий на дані, але й якісний, тобто відображав реальні сценарії, з якими модель буде працювати. Також важливим є представлення всієї різноманітності випадків, щоб система могла навчитися розпізнавати широкий спектр ситуацій.

2.5.1 Збір данних

Щоб створити датасет для дослідження, використовувався метод, відомий як скрапінг. Скрапінг – це процес автоматичного збору даних з інтернет–ресурсів. Цей процес включав використання спеціалізованого програмного забезпечення, яке "переглядає" Twitter, автоматично збираючи твіти за певними критеріямиб, а саме додатком TVINT .

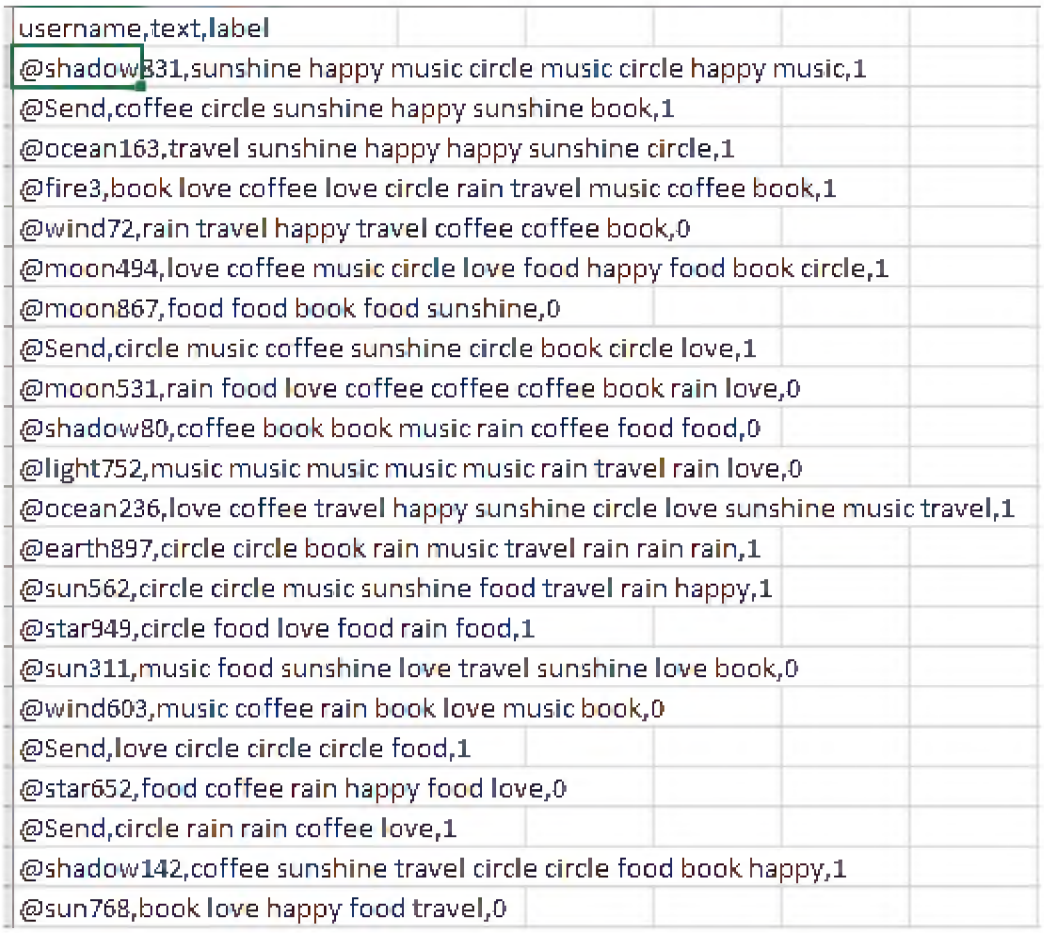
Були зібрані 6000 твітів від різних користувачів. Ця невелика кількість твітів допомогла забезпечити репрезентативність та різноманітність даних для моєї вибірки. Було ідентифікації 400 твітів від конкретних користувачів, які включали певне слово в своїх повідомленнях. Цей підхід дозволив мені виокремити та проаналізувати специфічні шаблони використання мови, тематику твітів та контекст, у якому використовувалося це слово.

Важливо зазначити, що процес скрапінгу вимагає дотримання правил і політик користування даними, встановлених ресурсами, з яких ведеться збір. У випадку з Twitter, це означає враховувати їхні умови використання та політику конфіденційності. Також твітером були встановлені обмеження на кількість читаємих твітів на добу, тому це також повпиливало на розмір вибірки в датасеті.

Данні в датасеті поділені на стовпці: username, text, label. У стовпці username зазначені імена користувачів, у text тексти твітів, а у стовпеці label вказані ідентифікатори для пошуку і розпізнавання. Всі твіти де зустрічається у username ім’я

@Send та слово circle, у стовпці label встановлено ідентифікатор 1. Де нічого з зазначенного не зустрічається вказуємо ідентифікатор 0.

Таким чином наш датасет виглядає так:



| username,text,label |
|---|
| @shadow831,sunshine happy music circle music circle happy music,1 |
| @Send,coffee circle sunshine happy sunshine book,1 |
| @ocean163,travel sunshine happy happy sunshine circle,1 |
| @fire3,book love coffee love circle rain travel music coffee book,1 |
| @wind72,rain travel happy travel coffee coffee book,0 |
| @moon494,love coffee music circle love food happy food book circle,1 |
| @moon867,food food book food sunshine,0 |
| @Send,circle music coffee sunshine circle book circle love,1 |
| @moon531,rain food love coffee coffee coffee book rain love,0 |
| @shadow80,coffee book book music rain coffee food food,0 |
| @light752,music music music music music rain travel rain love,0 |
| @ocean236,love coffee travel happy sunshine circle love sunshine music travel,1 |
| @earth897,circle circle book rain music travel rain rain rain,1 |
| @sun562,circle circle music sunshine food travel rain happy,1 |
| @star949,circle food love food rain food,1 |
| @sun311,music food sunshine love travel sunshine love book,0 |
| @wind603,music coffee rain book love music book,0 |
| @Send,love circle circle circle food,1 |
| @star652,food coffee rain happy food love,0 |
| @Send,circle rain rain coffee love,1 |
| @shadow142,coffee sunshine travel circle circle food book happy,1 |
| @sun768,book love happy food travel,0 |

Рисунок 2.1–Фрагмент датасету

## 2.6 Методи і метрики для аналізу моделей

У контексті машинного навчання існує цілий спектр методів та підходів, кожен з яких має свої унікальні особливості та застосування. Ці методи використовуються для створення моделей, які можуть ефективно обробляти та аналізувати дані. Також важливою частиною процесу машинного навчання є використання метрик, які допомагають оцінити та покращити ефективність цих моделей. У цілому, ці методи та метрики відіграють ключову роль у здатності моделей розуміти та інтерпретувати великі обсяги даних, особливо з соціальних мереж.

Методи машинного навчання відіграють ключову роль у виявленні шаблонів та витягуванні значущих інсайтів із сирих даних. Вони варіюються від простих до складних та залежать від конкретної задачі, з якою ми маємо справу. Методи можуть

бути навчанням з учителем, де модель тренується на попередньо позначених даних, або навчанням без учителя, яке зосереджується на виявленні шаблонів у даних без попереднього маркування.

Розглянемо більш детально два основних типи машинного навчання: навчання з учителем (supervised learning) та навчання без учителя (unsupervised learning).

Навчання з учителем передбачає, що у нас є датасет, який вже містить "відповіді" або мітки. Це означає, що кожен елемент даних у датасеті вже позначений або класифікований у певну категорію. Модель машинного навчання "навчається" на цих даних, намагаючись вивчити зв'язки та закономірності, які дозволяють їй робити прогнози або класифікації для нових, непозначених даних. Наприклад, якщо ми тренуємо модель розпізнавати емоційний зміст текстів (позитивний, негативний, нейтральний), то для навчання ми використовуємо вже позначені текстові дані. Після тренування модель може прогнозувати емоційний зміст нових текстів.

Навчання без учителя відбувається без попередньо позначених даних. Тут мета полягає в тому, щоб модель самостійно виявила структуру або шаблони в даних. Одним із найпоширеніших застосувань навчання без учителя є кластеризація, де модель групує дані на основі їхніх властивостей або характеристик. Наприклад, модель може аналізувати великий набір твітів і групувати їх на основі спільних тем чи ключових слів. Інший важливий приклад навчання без учителя – це методи зниження розмірності, які допомагають у виявленні найважливіших атрибутів у великих датасетах.

Обидва ці типи навчання мають своє місце в аналізі даних і використовуються в залежності від конкретних цілей дослідження та наявності та типу даних. Вибір між навчанням з учителем і без учителя часто залежить від того, наскільки добре ми розуміємо домен, з яким працюємо, і від того, чи є у нас доступ до великої кількості позначених даних.

Також варто звернути увагу на дві ключові концепції в області обчислень та машинного навчання: моделях з плаваючою та сталою точкою. Ці поняття мають

важливе значення при обробці та аналізі числових даних, що є фундаментальною частиною багатьох алгоритмів машинного навчання. Розуміння цих моделей допомагає у виборі правильних підходів до обчислень та оптимізації ресурсів, особливо в контексті великих даних та обмежених обчислювальних можливостей.

Моделі з Плаваючою Точкою:

- Плаваюча точка – це спосіб представлення реальних чисел у комп'ютерах, де число розбивається на мантису (або значущу частину) та експоненту.
- Моделі з плаваючою точкою дозволяють представляти дуже малі та дуже великі числа з високою точністю, що є важливим у наукових обчисленнях та машинному навчанні.
- Однак, такі моделі можуть бути досить ресурсомісткими з точки зору обчислювальної потужності та пам'яті, що може бути проблемою при роботі з великими обсягами даних або на пристроях з обмеженими ресурсами.

Моделі зі Сталою Точкою:

- Стала точка – це інший спосіб представлення чисел, де кількість цифр перед та після десяткового знаку є фіксованою.
- Цей метод є менш гнучким, ніж плаваюча точка, але займає менше пам'яті та може бути ефективнішим з точки зору обчислювальної потужності.
- Моделі зі сталою точкою часто використовуються в вбудованих системах та мобільних пристроях, де обмежені ресурси пам'яті та обчислювальна потужність.

Вибір між плаваючою та сталою точкою в машинному навчанні залежить від специфіки задачі та обмежень обладнання. Плаваюча точка зазвичай використовується для тренування моделей через її високу точність та гнучкість, тоді як стала точка може бути більш ефективною для впровадження моделей у виробництво, особливо в системах з обмеженими ресурсами.

### 2.6.1 Метрики

Для виявлення моделей, які найкраще підходили для подальшого використання в системах і алгоритмах аналізу соціальних мереж співробітників, проведемо порівняння різних моделей машинного навчання.

Основним компонентом оцінки та порівняння ефективності моделей машинного навчання є метрики. Метрики – це кількісні інструменти, які використовуються для оцінки різних аспектів ефективності моделі, включаючи її точність, надійність, швидкість та споживання ресурсів. Вони дозволяють нам систематично порівнювати різні моделі та обирати найбільш оптимальний варіант для конкретних дослідницьких завдань.

У дослідженні використовуємо наступні метрики:

1. Accuracy (Точність)

Формула:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{2.1}$$

Де Number of Correct Predictions (Кількість правильних прогнозів) – кількість випадків, коли модель правильно передбачила клас або вихідний результат.

А Total Number of Predictions (Загальна кількість прогнозів) – загальна кількість випадків, які були оцінені моделлю. Це включає як правильні, так і неправильні прогнози.

Accuracy (Точність) вимірює відсоток випадків, коли модель правильно класифікувала або прогнозувала дані.

2. F1 Score

Формула:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.2}$$

Де Precision (Точність) – відсоток випадків, коли модель правильно ідентифікувала позитивний клас серед усіх реальних позитивних випадків.

A Recall (Повнота) – відсоток випадків, коли модель правильно ідентифікувала позитивний клас серед усіх реальних позитивних випадків.

F1 Score допомагає збалансувати точність та повноту, особливо важливо у випадках, коли розподіл класів є нерівномірним.

3. Overfitting

Не має конкретної формули, але визначається порівнянням продуктивності моделі на тренувальних та тестових наборах даних.

Важливо для забезпечення здатності моделі ефективно працювати з невідомими даними.

4. Training Time (Час навчання)

Вимірюється як час, який потрібен моделі для навчання на заданому наборі даних. Це важливо для оцінки ресурсоемності моделі.

5. Correct Predictions (Кількість правильних прогнозів)

Формула:

Кількість правильних прогнозів=сума випадків, коли передбачений клас співпадає з фактичним класом.

Показує абсолютну кількість випадків, коли передбачення моделі були точними.

6. Model Size (Розмір моделі)

Розмір моделі визначається об'ємом пам'яті, який вона займає. Це важливо для розуміння ефективності використання ресурсів, особливо у системах з обмеженими ресурсами.

Для моделей які використовують навчання без вчителя додатково використовуватимуться трохи інші метрики. Ці метрики є аналогічними тим які були зазначені вище, але для кластеризації.

Метрики :

1. Model Inertia (Інерція моделі):

Інерція моделі в контексті кластеризації визначається як загальна сума квадратів відстаней між кожною точкою даних у кластері та центроїдом (центром) цього кластера.

Формула:

$$Inertia = \sum(d(x,c)2) \quad (2.3)$$

Де: $d(x,c)$ визначає відстань між точкою $x$ та центроїдом $c$.

Низьке значення інерції вказує на те, що точки даних згруповані близько до центроїдів, тоді як високе значення вказує на ширше розсіяння точок від центроїдів.

2. Індекс Данна (Dunn's Index)

Індекс Данна вимірює співвідношення між найменшою відстанню між кластерами (міжкластерна відстань) та найбільшою відстанню всередині кластеру (внутрішньокластерна відстань).

Формула:

$$DI = \frac{\min_{1 \le i < j \le k}(\delta(C_i,C_j))}{\max_{1 \le l \le k}(\Delta(C_l))} \quad (2.4)$$

Де:

- $DI$ – Індекс Данна;
- $k$ – кількість кластерів;
- $Ci,Cj$ – різні кластери;
- $\delta(Ci,Cj)$ – відстань між кластерами $Ci$ та $Cj$, яка зазвичай визначається як відстань між найближчими точками цих кластерів;
- $\Delta(Cl)$ – розмір кластера $Cl$, визначений як максимальна відстань між точками всередині кластера $Cl$.

3. Силуетний коефіцієнт (Silhouette Coefficient)

Формуда:

$$s(i) = \frac{b(i)-a(i)}{\max\{a(i),b(i)\}} \tag{2.5}$$

Де:

- $s(i)$ – силуетний коефіцієнт для об'єкта $i$;
- $a(i)$ – середня відстань від об'єкта $i$ до інших об'єктів у тому ж кластері;
- $b(i)$ – найменша середня відстань від об'єкта $i$ до об'єктів у інших кластерах, які не включають кластер $i$.

Середнє значення силуетних коефіцієнтів усіх об'єктів використовується як оцінка якості кластеризації. Значення цієї метрики коливається від –1 до +1, де вищі значення вказують на більш "щільну" та добре відокремлену структуру кластерів.

2.6.2 Візуальна оцінка результатів

У процесі оцінки ефективності моделей машинного навчання, які навчаються з використанням вчителя , часто використовуємо інструментами візуалізації, щоб краще зрозуміти їхню роботу. Два з таких популярних інструменти – це Матриця Помилок та ROC–крива. Матриця Помилок – це таблиця, яка показує нам, наскільки точно модель розрізняє класи: вона вказує, скільки разів модель правильно чи неправильно класифікувала випадки. ROC–крива – це графік, який демонструє, наскільки добре модель може відрізнити два класи, базуючись на її прогнозах.

Що стосується моделей, які навчаються без вчителя, ми використовуємо інші інструменти. Одним з них є Теплова карта кореляцій, яка показує, як різні характеристики даних пов'язані між собою. Ще одним корисним інструментом є візуалізація кластерів за допомогою PCA, методу, який допомагає зменшити складність даних для їх легшої візуалізації. Далі оремо розповімо про кожен інструмент візуалізації окремо.

Матриця Помилок (Confusion Matrix)

Матриця помилок – це таблиця, яка використовується для оцінки ефективності моделі класифікації. Вона дозволяє візуально порівняти фактичні мітки класів з прогнозованими моделлю. Основні елементи матриці помилок:

- True Positives (TP): Кількість випадків, коли модель правильно передбачила позитивний клас.
- True Negatives (TN): Кількість випадків, коли модель правильно передбачила негативний клас.
- False Positives (FP): Кількість випадків, коли модель помилково передбачила позитивний клас.
- False Negatives (FN): Кількість випадків, коли модель помилково передбачила негативний клас.

ROC–Крива (Receiver Operating Characteristic Curve)

ROC–крива – це графік, який ілюструє діагностичну здатність бінарного класифікатора при варіюванні порога класифікації. Ось основні компоненти ROC–кривої:

- Ось X (False Positive Rate, FPR): Відсоток негативних випадків, які помилково були класифіковані як позитивні. Формула:

$$FPR = \frac{FP}{FP+TN} \tag{2.6}$$

де:

- FP (False Positives) – кількість випадків, які модель невірно класифікувала як позитивні, тоді як вони насправді негативні;
- TN (True Negatives) – кількість випадків, які модель правильно класифікувала як негативні.
- Ось Y (True Positive Rate, TPR): Відсоток позитивних випадків, які були правильно класифіковані. Також відомий як чутливість або recall. Формула:

$$TPR = \frac{TP}{TP+FN} \tag{2.7}$$

де:

- TP (True Positives) – кількість випадків, які модель правильно класифікувала як позитивні.
- FN (False Negatives) – кількість випадків, які модель невірно класифікувала як негативні, тоді як вони насправді позитивні.

Ці дві метрики разом використовуються на ROC–кривій для оцінки загальної ефективності бінарних класифікаторів, де ідеальний класифікатор матиме TPR = 1 (або 100%) і FPR = 0 (або 0%).

PCA Visualization of Clusters (Візуалізація Кластерів за допомогою PCA)

Метод головних компонент (PCA) використовується для зменшення розмірності даних, перетворюючи багатовимірний набір даних на двовимірний або тривимірний, зберігаючи при цьому якомога більше варіативності. Візуалізація кластерів з використанням PCA дозволяє нам бачити, як різні об'єкти групуються у просторі зі зменшеною розмірністю.

Формула PCA включає кілька кроків:

1. Стандартизація даних: Кожна ознака даних центрується навколо 0 та масштабується.
2. Обчислення коваріаційної матриці: Визначається матриця, яка відображає коваріацію між кожною парою ознак.
3. Знаходження власних значень та власних векторів: Власні вектори вказують напрямки найбільшої варіативності, а власні значення визначають їхній "вагу" або важливість.
4. Вибір компонентів і формування нових ознак: Найважливіші власні вектори (головні компоненти) вибираються для створення нового, зменшеного простору ознак.

Heatmap of Feature Correlations (Теплова карта кореляцій ознак)

Теплова карта кореляцій використовується для візуалізації взаємозв'язків між різними ознаками в датасеті. Кореляційна матриця визначається як:

$$\mathrm{Corr}(X,Y) = \frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y} \tag{2.8}$$

де $\mathrm{Cov}(X,Y)$ – це коваріація між двома ознаками, а $\sigma X$ і $\sigma Y$ – стандартні відхилення ознак X та Y.

Формула для обчислення висоти стовпця гістограми:

Де:

$$\text{Height of Bin} = \frac{\text{Number of Observations in Bin}}{\text{Total Number of Observations} \times \text{Width of Bin}} \tag{2.9}$$

- Height of Bin (Висота стовпця) – це висота кожного стовпця в гістограмі, яка відображає частоту спостережень.
- Number of Observations in Bin (Кількість спостережень в біні) – кількість спостережень (або даних), які падають у певний інтервал значень (бін).
- Total Number of Observations (Загальна кількість спостережень) – загальна кількість спостережень у датасеті.
- Width of Bin (Ширина біну) – розмір інтервалу значень для кожного біна в гістограмі.

Гістограма дозволяє візуально оцінити розподіл даних за певною ознакою, показуючи, як часто певні значення зустрічаються в датасеті. Це корисно для ідентифікації розподілу даних, виявлення асиметрії, викидів та інших характеристик розподілу.

2.7 Програмна реалізація

Для основною задачею є класифікація твітів з датасету tweets_dataset.csv відповідно до їхніх позначок у стовпці Label. У нашому датасеті, значення 1 в стовпці Label позначає твіти, які відповідають певним критеріям пошуку, таким як наявність конкретного слова або приналежність до певного користувача. Відповідно, значення 0 в цьому стовпці позначає твіти, які не відповідають цим критеріям.

Перш ніж почати тренувати моделі, ми підготуємо дані, перетворивши текст твітів на числові вектори за допомогою TfidfVectorizer. Це дозволить моделям машинного навчання розпізнавати важливі ознаки в тексті. Одночасно, мітки класу (1

або 0) зі стовпця Label будуть використовуватися як цільова змінна (y) у процесі тренування.

Ми поділимо наш датасет на тренувальну та тестову вибірки. Тренувальна вибірка використовуватиметься для навчання моделей, а тестова – для оцінки їхньої здатності узагальнювати навчення на нових даних. Це дозволить переконатися, що моделі ефективно розрізняють твіти, які відповідають критеріям (1), від тих, що не відповідають (0).

Використовуючи обрані алгоритми машинного навчання, такі як MLPClassifier, RandomForestClassifier, SVC, і GradientBoostingClassifier, CNN, RNN ми тренуємо моделі на основі тренувальної вибірки. Моделі навчатимуться розрізняти між двома класами твітів на основі їх текстового вмісту та позначок у Label. Ефективність кожної моделі буде оцінена на тестовій вибірці з використанням метрик, таких як точність (accuracy) та F1–оцінка.

2.7.2 Опис коду

Перш за все, нам потрібно підключити необхідні бібліотеки для різних завдань, пов'язаних з обробкою даних, машинним навчанням та вимірюванням ефективності нашої моделі:

1. Pandas (import pandas as pd):

Підключаємо Pandas для роботи з датасетами, їх обробки та аналізу.

2. Scikit–learn:

Завантажуємо різні компоненти з Scikit–learn, що включає алгоритми машинного навчання, методи обробки тексту та інструменти для вимірювання якості моделі.

3. Time (import time):

Використовуємо бібліотеку time для відстеження часу, необхідного для тренування моделей.

4. Pickle (import pickle):

Імпортуємо pickle для серіалізації моделей та аналізу їх розміру.

5. Sys (import sys):

Sys використовується для отримання системної інформації, такої як розмір серіалізованого об'єкта.

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier,
GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, f1_score, classification_report
import time
import pickle
import sys
```

Тепер переходимо до завантаження даних та їх підготовки для подальшого аналізу:

- Завантажуємо датасет за допомогою Pandas.
- Обробляємо відсутні значення та видаляємо непотрібні рядки.

```
# Завантаження даних
data = pd.read_csv('/content/tweets_dataset.csv')
data['text'].fillna('', inplace=True)  # Заміна NaN значень на пусті рядки
data.dropna(subset=['label'], inplace=True)  # Видалення рядків з NaN у
'label'
```

Для перетворення текстових даних на числові вектори, які можна аналізувати алгоритмами машинного навчання, ми використовуємо TF–IDF векторизацію:

- Застосовуємо TF–IDF для перетворення тексту в числові вектори.

```
# Попередня обробка
tfidf = TfidfVectorizer(max_features=1000)
X = tfidf.fit_transform(data['text']).toarray()
y = data['label']
```

Щоб оцінити ефективність моделей, нам потрібно розділити дані на тренувальну та тестову вибірки:

- Використовуємо функцію train_test_split для розділення даних.

```
# Розбиття на тренувальну та тестову вибірку
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Ми визначаємо функцію, яка допоможе нам оцінити різні аспекти ефективності кожної моделі:

- Функція evaluate_model вимірює точність, F1–бал, час тренування та інші метрики для кожної моделі.

```
# Функція для обчислення розміру моделі
def model_size(model):
    return sys.getsizeof(pickle.dumps(model))
# Функція для оцінки моделі
def evaluate_model(model, X_train, y_train, X_test, y_test):
    start_time = time.time()
    model.fit(X_train, y_train)
    end_time = time.time()

    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    correct_predictions = sum(y_pred == y_test)

    training_time = end_time - start_time
    size = model_size(model)

    return accuracy, training_time, correct_predictions, size
```

Тепер ми можемо створити кілька моделей машинного навчання та оцінити їх:

- Ініціалізуємо та оцінюємо різні моделі, такі як нейронна мережа, випадковий ліс, SVM, градієнтний бустинг.

```
# Моделі та їхні назви
models = [MLPClassifier(), RandomForestClassifier(), SVC(),
GradientBoostingClassifier()]
model_names = ['Neural Network', 'Random Forest', 'SVM', 'Gradient Boosting']
results = {}

# Оцінка моделей
for model, name in zip(models, model_names):
    accuracy, training_time, correct_predictions, size = evaluate_model(model,
X_train, y_train, X_test, y_test)
    results[name] = {
        'Accuracy': accuracy,
        'Training Time': training_time,
        'Correct Predictions': correct_predictions,
        'Model Size': size
    }
```

Нарешті, ми виводимо результати, щоб порівняти ефективність різних моделей:

– Виводимо результати для кожної моделі, включаючи точність, F1–бал та інші важливі метрики.

```
# Виведення результатів
for name, metrics in results.items():
    print(f"Model: {name}")
    for metric, value in metrics.items():
        print(f"{metric}: {value}")
    print("-" * 30)
```

```
/usr/local/lib/python3.10/dist-packa
  warnings.warn(
Model: Neural Network
Accuracy: 0.924468085106383
F1 Score: 0.8922726791574045
Overfitting: 0.002925531914893642
Training Time: 11.528108358383179
Correct Predictions: 869
Model Size: 65954
------------------------------
Model: Random Forest
Accuracy: 0.925531914893617
F1 Score: 0.9029969296450592
Overfitting: 0.07260638297872346
Training Time: 0.5847892761230469
Correct Predictions: 870
Model Size: 3945042
------------------------------
Model: SVM
Accuracy: 0.9287234042553192
F1 Score: 0.8944021311802468
Overfitting: -0.003457446808510678
Training Time: 0.21960115432739258
Correct Predictions: 873
Model Size: 162189
------------------------------
Model: Gradient Boosting
Accuracy: 0.925531914893617
F1 Score: 0.8999568885393149
Overfitting: 0.013563829787234138
Training Time: 1.00404953956604
Correct Predictions: 870
Model Size: 117242
------------------------------
```

Рисунок 2.2 – Результати навчання

Врвховцючи отримані результати навчання на різних моделях, що найбільш вірних передбачень та найменший розмір моделі і найкращу середню точність і оптимальні показники інших метрик мають Нейронні мережі.

Тому для подальшого аналізу пропоную використовувати різні види моделей нейронної мережі.

2.7.1 Начання нейронних мереж

Існує безліч видів нейронних мереж, кожна з яких має свої особливості та оптимальні сценарії використання. Для аналізу текстових даних, які ми розглядаємо, найбільш підходящими є MLPClassifier, CNN та RNN (Recurrent Neural Network), оскільки кожна з них має унікальні переваги для обробки мови.

MLPClassifier ефективний для задач, де текст може бути представлений через фіксовані особливості, наприклад, за допомогою векторизації TF–IDF. Це робить MLP хорошим вибором для задач класифікації, де текст не вимагає аналізу послідовностей або контексту.

CNN, хоч і зазвичай використовується в обробці зображень, також виявляється ефективним у виявленні патернів та локальних особливостей у тексті. Це робить його вдалим вибором для задач, де необхідно виявити ключові слова або фрази, наприклад, у сентимент–аналізі або класифікації тексту.

RNN та його варіації, як–то LSTM або GRU, є оптимальним вибором для обробки послідовностей, оскільки вони здатні зберігати інформацію про попередні елементи в послідовності. Це особливо корисно в задачах, де контекст та послідовність слів мають велике значення, наприклад, при машинному перекладі або генерації тексту.

У наведеному нижче прикладі коду використовується модель нейронної мережі – MLPClassifier з вчителем, з бібліотеки scikit–learn. Ця модель представляє багатошаровий персептрон (Multi–Layer Perceptron), який є базовим типом нейронної мережі для класифікації. нейронок з графіками

Спершу імпортуємо необхідні бібліотеки. Pandas використовується для роботи з даними, Scikit–learn – для машинного навчання, Matplotlib і Seaborn – для візуалізації, time для вимірювання часу, pickle для зберігання моделей, sys для системних операцій та numpy для математичних обчислень.

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, roc_curve,
classification_report
import matplotlib.pyplot as plt
import seaborn as sns
import time
import pickle
import sys
import numpy as np
```

В цьому блоку завантажуємо датасет за допомогою Pandas, обробляємо відсутні значення в текстових полях, застосовуємо TF–IDF векторизацію для перетворення тексту у числові вектори, а потім ділимо наші дані на тренувальні та тестові набори.

```
# Завантаження та попередня обробка даних
data = pd.read_csv('/content/tweets_dataset.csv')
data['text'].fillna('', inplace=True)
tfidf = TfidfVectorizer(max_features=1000)
X = tfidf.fit_transform(data['text']).toarray()
y = data['label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Тут ініціалізуємо та тренуємо нашу модель нейронної мережі (MLPClassifier). Також вимірюємо час, який займає процес навчання, та використовуємо натреновану модель для прогнозування на тестових даних.

```
# Навчання моделі MLP
mlp = MLPClassifier(verbose=True)
start_time = time.time()
mlp.fit(X_train, y_train)
end_time = time.time()
y_pred = mlp.predict(X_test)
```

В цьому блоку розраховуємо точність моделі, створюємо матрицю помилок, розраховуємо показники для ROC–кривої та генеруємо звіт класифікації. Також перевіряємо модель на перенавчання, порівнюючи точність на тренувальних та тестових даних.

```
# Розрахунок правильних передбачень
correct_predictions = np.sum(y_pred == y_test)

# Оцінка моделі
accuracy = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
fpr, tpr, _ = roc_curve(y_test, mlp.predict_proba(X_test)[:,1])
```

```
report = classification_report(y_test, y_pred, output_dict=True)
f1_score = report['weighted avg']['f1-score']

# Перевірка на Overfitting
train_accuracy = mlp.score(X_train, y_train)
test_accuracy = accuracy
overfitting = "Yes" if train_accuracy > test_accuracy else "No"
```

Побудова теплової карти для матриці помилок та ROC–кривої для оцінки ефективності моделі.

```
# Візуалізації
plt.figure(figsize=(10, 5))
sns.heatmap(cm, annot=True, fmt="d")
plt.title('Confusion Matrix')
plt.show()

plt.figure(figsize=(10, 5))
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```

Виводимо ключові показники ефективності моделі, включаючи точність, F1–бал, інформацію про перенавчання, час тренування, кількість правильних передбачень та розмір моделі.

```
# Виведення результатів
print(f"Accuracy: {accuracy}")
print(f"F1 Score: {f1_score}")
print(f"Overfitting: {overfitting}")
print(f"Training Time: {end_time - start_time} seconds")
print(f"Correct Predictions: {correct_predictions}")
print(f"Model Size: {sys.getsizeof(pickle.dumps(mlp))} bytes")
```

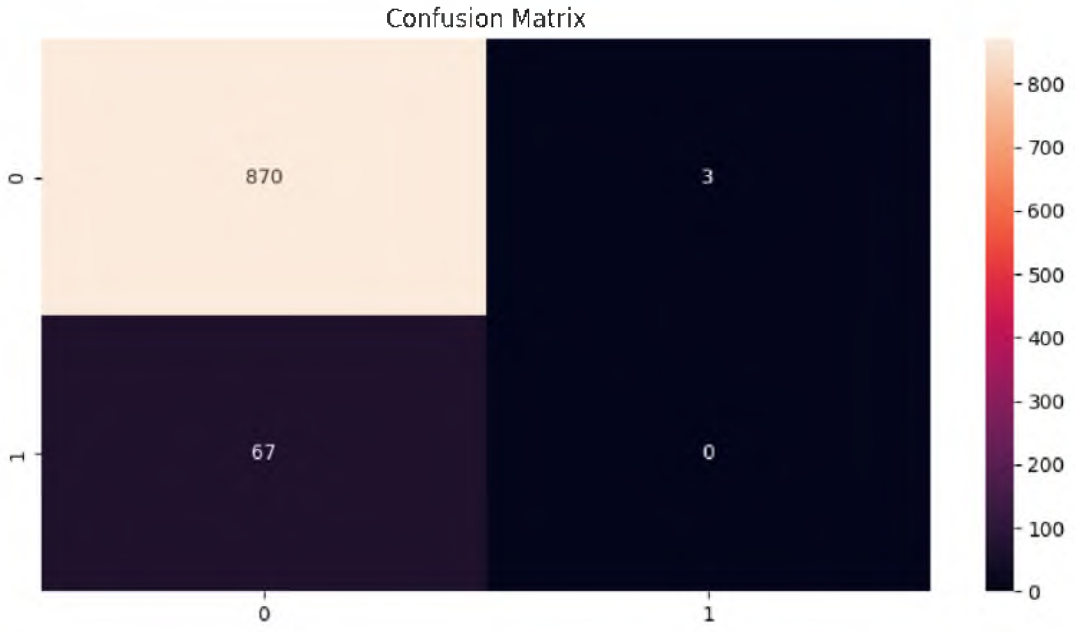



Рисунок 2.3 – Confusion Matrix

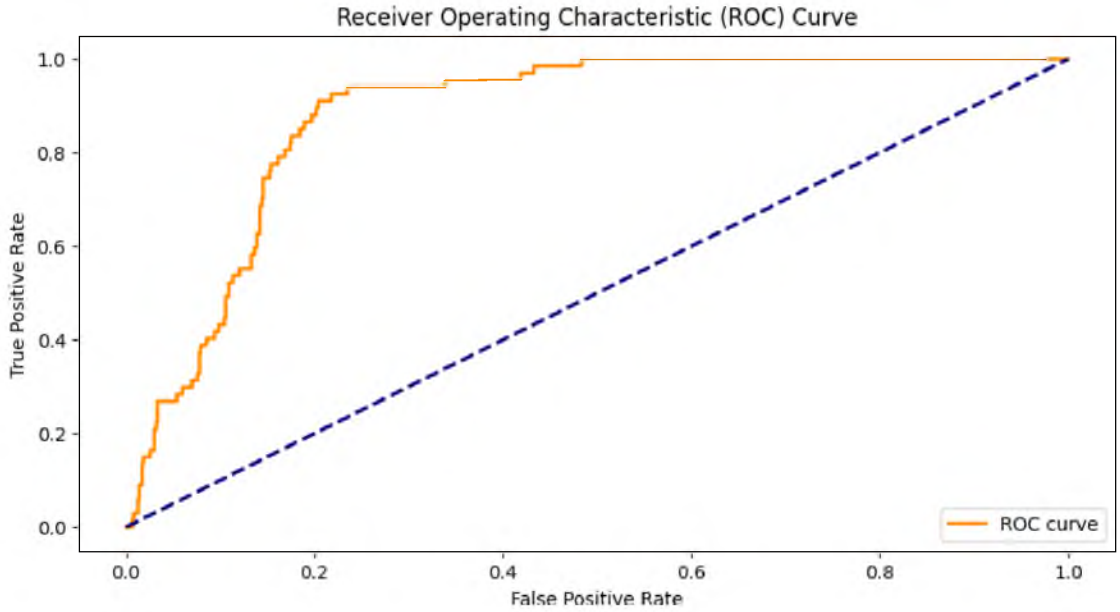



Рисунок 2.4 – ROC крива

```
Accuracy: 0.925531914893617
F1 Score: 0.8928059245327379
Overfitting: Yes
Training Time: 4.831637859344482 seconds
Correct Predictions: 870
Model Size: 65953 bytes
```

Рисунок 2.5 – Результати навчання

Моделі CNN та RNN будуть розглянуті разом через те, що мають спільні бібліотеки. Розглянемо особливості кожної з моделей

CNN (Convolutional Neural Networks) – це тип нейронної мережі, який спеціалізується на обробці даних з явною просторовою структурою, наприклад, зображень. Основа CNN полягає в застосуванні згорткових шарів, де через ряд фільтрів аналізується кожен фрагмент вхідного зображення для виявлення основних візуальних ознак, таких як краї, текстури чи візерунки. Згорткові шари, поєднані з активаційними функціями та пулінговими шарами, допомагають CNN ефективно визначати складніші ознаки та зменшувати обчислювальну складність. Після послідовності згорткових та пулінгових шарів, CNN використовують повнозв'язні шари для класифікації або інших задач на основі виявлених ознак.

RNN (Recurrent Neural Networks) – це клас нейронних мереж, призначений для обробки послідовностей, таких як мовні дані або часові ряди. Унікальна особливість RNN полягає в їх здатності "пам'ятати" попередні вхідні дані завдяки внутрішнім зворотнім зв'язкам, що дозволяє їм враховувати інформацію з попередніх кроків при обробці поточних даних. Це робить їх ідеальними для завдань, де контекст або послідовність елементів має велике значення, наприклад, у машинному перекладі або розпізнаванні мовлення. Проте RNN часто стикаються з проблемами зникання або вибуху градієнтів, які можуть бути частково подолані за допомогою таких варіацій, як LSTM або GRU.

Код для обох моделей так як вони використовують однакові бібліотеки.

На початку імпортуємо усі потрібні бібліотеки для обробки даних, побудови та тренування нейронних мереж, візуалізації результатів, вимірювання часу та інших важливих завдань.

```
import pandas as pd
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, Conv1D,
GlobalMaxPooling1D
from tensorflow.keras.metrics import AUC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, roc_curve,
roc_auc_score, f1_score
import matplotlib.pyplot as plt
import seaborn as sns
import time
```

```
import numpy as np
import pickle
import sys
```

Перетворюємо текстові дані в числові послідовності за допомогою токенізації, а потім вирівнюємо їх довжину.

```
# Завантаження та попередня обробка даних
data = pd.read_csv('/content/tweets_dataset.csv')
data['text'].fillna('', inplace=True)

# Токенізація та векторизація тексту
tokenizer = Tokenizer(num_words=5000)
tokenizer.fit_on_texts(data['text'])
sequences = tokenizer.texts_to_sequences(data['text'])
maxlen = 100
X = pad_sequences(sequences, maxlen=maxlen)
y = data['label']
```

Розділяємо дані на тренувальну та тестову вибірки для подальшого аналізу.

```
# Розбиття на тренувальну та тестову вибірки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Функції підрахунку розміру моделі а також коли модель правильно передбачила мітку.

```
# Функція для обчислення розміру моделі
def model_size(model):
    return sys.getsizeof(pickle.dumps(model))

# Функція для розрахунку правильних прогнозів
def correct_predictions(y_true, y_pred_binary):
    return np.sum(y_true == y_pred_binary)
```

Цей блок коду демонструє процес створення, компіляції, тренування та використання CNN для класифікації текстів твітів.

```
# CNN Модель
cnn_model = Sequential([
    Embedding(5000, 32, input_length=maxlen),
    Conv1D(64, 5, activation='relu'),
    GlobalMaxPooling1D(),
    Dense(1, activation='sigmoid')
])
cnn_model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy', AUC(name='auc')])
start_time = time.time()
cnn_history = cnn_model.fit(X_train, y_train, epochs=5,
validation_data=(X_test, y_test), verbose=1)
end_time = time.time()
y_pred_cnn = cnn_model.predict(X_test).ravel()
```

```
y_pred_cnn_binary = np.where(y_pred_cnn > 0.5, 1, 0)
```

Цей аналогічно попередньомублоку демонструє процес створення, компіляції, тренування та використання для класифікації текстів твітів але RNN.

```
# RNN Модель
rnn_model = Sequential([
    Embedding(5000, 32, input_length=maxlen),
    LSTM(64),
    Dense(1, activation='sigmoid')
])
rnn_model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy', AUC(name='auc')])
start_time_rnn = time.time()
rnn_history = rnn_model.fit(X_train, y_train, epochs=5,
validation_data=(X_test, y_test), verbose=1)
end_time_rnn = time.time()
y_pred_rnn = rnn_model.predict(X_test).ravel()
y_pred_rnn_binary = np.where(y_pred_rnn > 0.5, 1, 0)
```

Оцінюємо кожну модель, виводимо основні метрики, такі як точність і F1–бал, візуалізуємо матрицю помилок та ROC–криву. Перевіряємо моделі на перенавчання.

```
# Виведення результатів та додаткових метрик
for model, y_pred_binary, history, name in [(cnn_model, y_pred_cnn_binary,
cnn_history, "CNN"),
                                            (rnn_model, y_pred_rnn_binary,
rnn_history, "RNN")]:
    accuracy = accuracy_score(y_test, y_pred_binary)
    f1 = f1_score(y_test, y_pred_binary)
    cm = confusion_matrix(y_test, y_pred_binary)
    auc_score = roc_auc_score(y_test, model.predict(X_test).ravel())
    correct_preds = correct_predictions(y_test, y_pred_binary)
    model_size_bytes = model_size(model)

    # Перевірка на Overfitting
    train_acc = history.history['accuracy'][-1]
    val_acc = history.history['val_accuracy'][-1]
    overfitting = "Yes" if val_acc < train_acc else "No"

    # Візуалізація
    sns.heatmap(cm, annot=True, fmt="d")
    plt.title(f'Confusion Matrix for {name}')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()

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

```
    plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title(f'ROC Curve for {name}')
    plt.legend(loc="lower right")
    plt.show()

    # Результати
    print(f"{name} Accuracy: {accuracy}")
    print(f"{name} F1 Score: {f1}")
    print(f"{name} Overfitting: {overfitting}")
    print(f"{name} Training Time: {end_time - start_time if name == 'CNN' else
end_time_rnn - start_time_rnn} seconds")
    print(f"{name} Correct Predictions: {correct_preds}")
    print(f"{name} Model Size: {model_size_bytes} bytes")
```

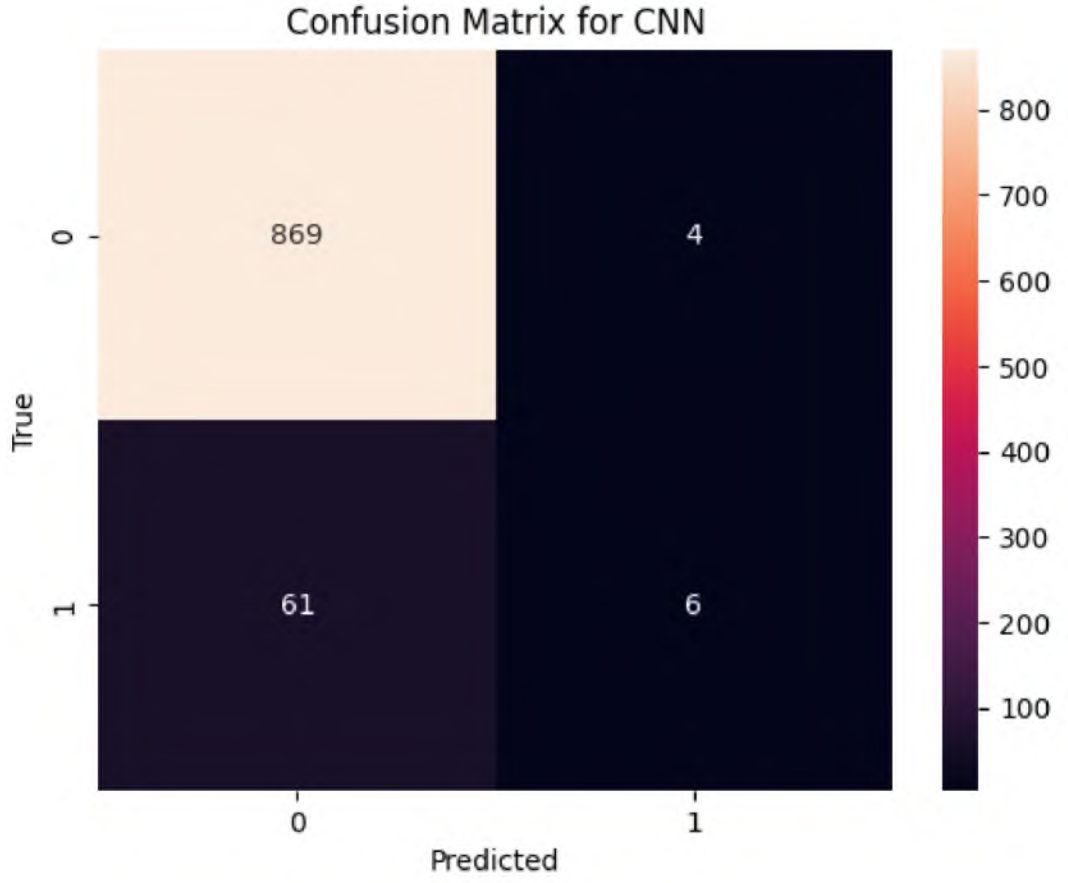



Рисунок 2.6 – Confusion Matrix для CNN

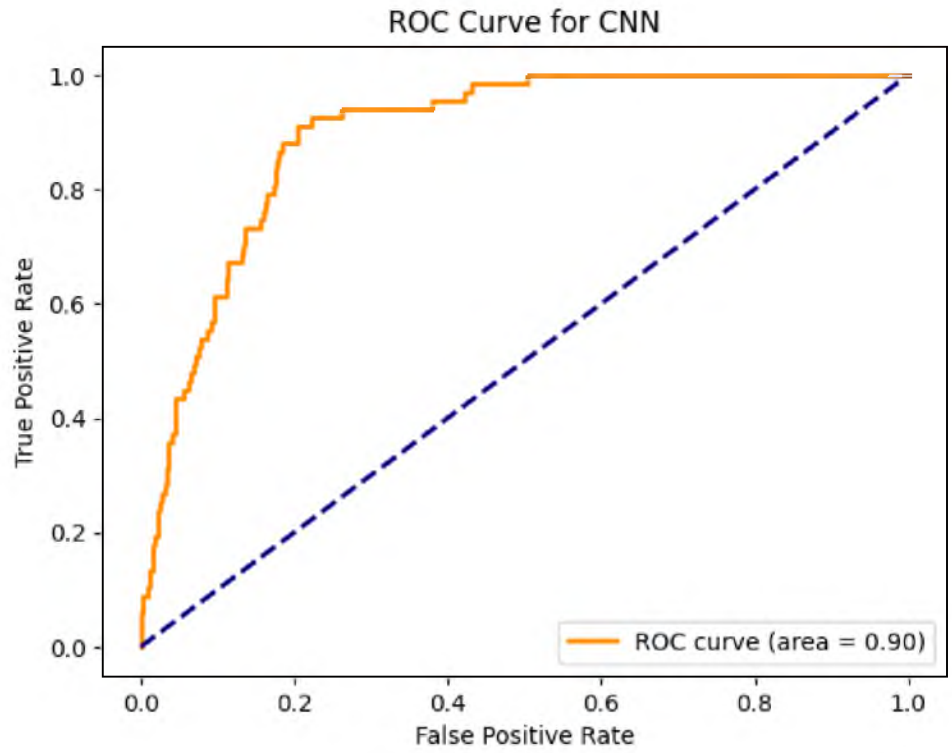



Рисунок 2.7 – ROC крива для CNN

```
CNN Accuracy: 0.9308510638297872
CNN F1 Score: 0.1558441558441558
CNN Overfitting: No
CNN Training Time: 11.236605644226074 seconds
CNN Correct Predictions: 875
CNN Model Size: 2078423 bytes
```

Рисунок 2.8 – Результати навчання

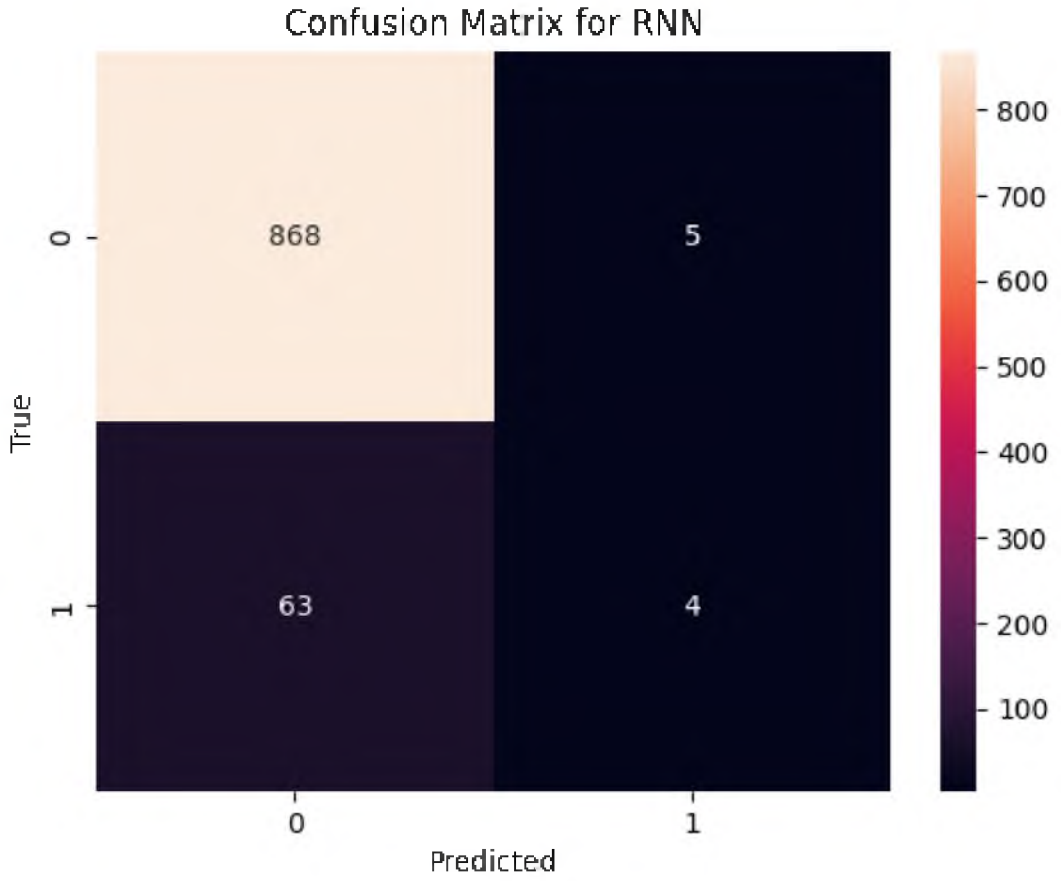



Рисунок 2.9 – Confusion Matrix для RNN

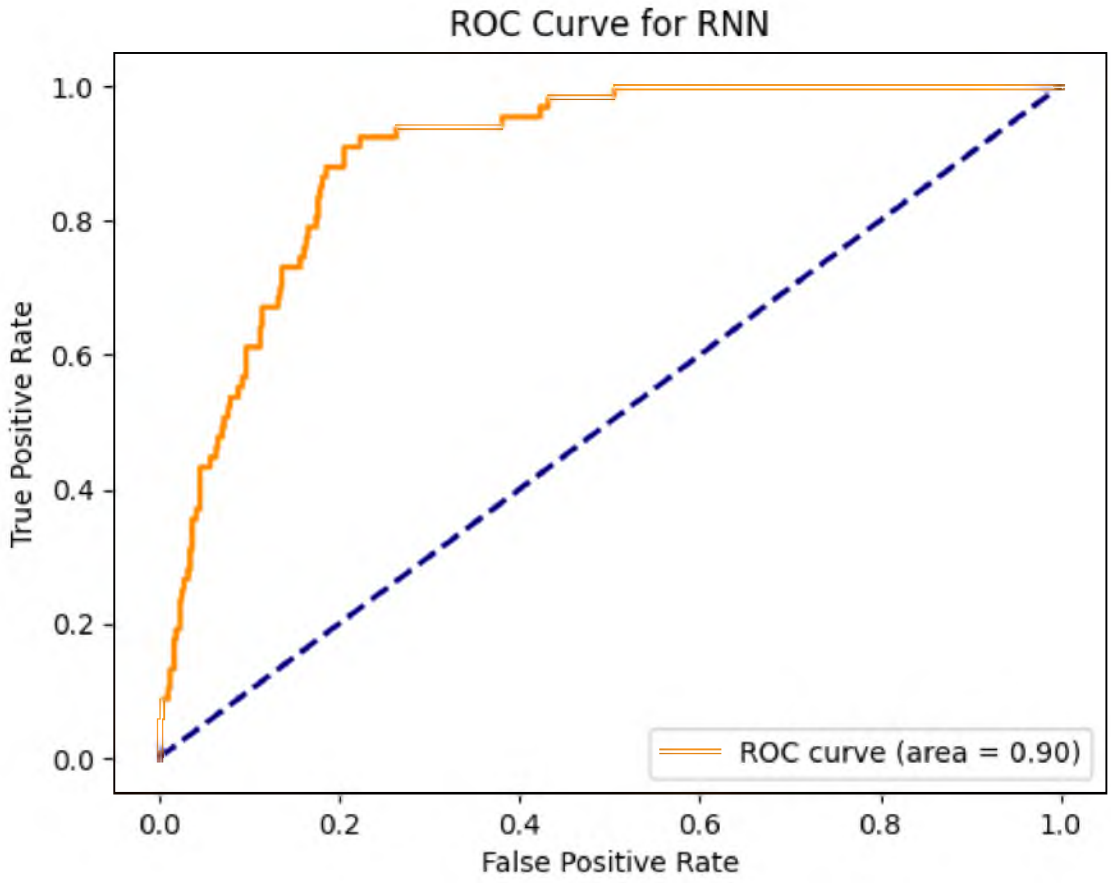



Рисунок 2.10 – ROC крива для RNN

```
RNN Accuracy: 0.9276595744680851
RNN F1 Score: 0.10526315789473685
RNN Overfitting: No
RNN Training Time: 42.37916922569275 seconds
RNN Correct Predictions: 872
RNN Model Size: 2253749 bytes
```

Рисунок 2.11 – Виводимі метрики

Розглянемо навчання нейронної мережі без вчителя з плавоючою комою для пошуку повідомлень користувача @Send у датасеті *tweets_dataset.csv*. Для цього використовуватиметься алгоритм KMeans. KMeans є популярним методом в машинному навчанні, який дозволяє ефективно групувати дані на основі їх характеристик без необхідності мануальної мітки чи надання інструкцій.

Процес використання KMeans у нашому починається з перетворення текстових даних твітів на числові вектори за допомогою TfidfVectorizer. Після цього KMeans аналізує ці вектори і розділяє їх на кластери. Це дозволяє нам

ідентифікувати групи твітів, що мають подібний зміст або тематику, та швидко знаходити повідомлення від конкретного користувача, як у нашому випадку з @Send.

Код починається з інсталяції бібліотеки validclust, яка потрібна для розрахунку Dunn's Index – метрики, що оцінює якість кластеризації.

```
# Інсталяція додаткової бібліотеки
!pip install validclust
```

Далі нам треба імпортувати всі бібліотеки, які будемо використовувати. Pandas допоможе нам працювати з даними, sklearn та validclust використаємо для кластеризації та оцінювання її якості, а matplotlib та seaborn допоможуть з візуалізацією результатів.

```
import pandas as pd

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score, pairwise_distances
from validclust import dunn
import matplotlib.pyplot as plt
import seaborn as sns
import sys
import pickle
```

Завантажуємо датасет із твітами, а потім заповнюємо відсутні значення в текстах твітів, щоб уникнути проблем при подальшій обробці. “fillna” використовується для заміни відсутніх значень у тексті, що гарантує, що векторизація тексту не зіткнеться з проблемами через відсутні дані.

```
# Завантаження даних
data = pd.read_csv('/content/tweets_dataset.csv')

# Заміна NaN значень на пусті рядки в стовпці 'text'
data['text'].fillna('', inplace=True)
```

Використовуємо TF–IDF для перетворення текстів на числові вектори, що дозволить алгоритму кластеризації працювати з текстовими даними.

```
# Попередня обробка
tfidf = TfidfVectorizer(max_features=1000)
X = tfidf.fit_transform(data['text']).toarray()
```

Тепер, коли у нас є векторизовані дані, можемо застосувати алгоритм KMeans для розділення даних на кластери.

```
# Визначаємо число кластерів
```

```
n_clusters = 5
kmeans = KMeans(n_clusters=n_clusters, random_state=42)
kmeans.fit(X)

# Додаємо інформацію про кластер до датасету
data['cluster'] = kmeans.labels_

# Візуалізація кластерів за допомогою PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
plt.figure(figsize=(10, 8))
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=data['cluster'],
palette="viridis")
plt.title("PCA Visualization of Clusters")
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.show()
```

Щоб зрозуміти, наскільки ефективною була наша кластеризація, обчислимо Silhouette Coefficient та Dunn's Index.

```
# Обчислення Silhouette Coefficient
silhouette_avg = silhouette_score(X, data['cluster'])
print(f"Silhouette Coefficient: {silhouette_avg:.2f}")

# Обчислення матриці дистанцій
dist_matrix = pairwise_distances(X)

# Обчислення Dunn's Index
dunn_index = dunn(dist_matrix, data['cluster'])
print(f"Dunn's Index: {dunn_index:.2f}")
```

Подивимося, скільки об'єктів у кожному кластері, а також оцінимо загальну інерцію моделі. Також розрахуємо розмір моделі.

```
# Виведення інформації про кластери та інерцію
print(f"Total clusters: {n_clusters}")
for i in range(n_clusters):
    print(f"Cluster {i} size: {sum(data['cluster'] == i)}")
print(f"Model Inertia: {kmeans.inertia_}")

# Розрахунок та виведення розміру моделі
model_size = sys.getsizeof(pickle.dumps(kmeans))
print(f"Model Size: {model_size} bytes")
```

Тепер давайте подивимося на кореляції між ознаками та побудуємо гістограми для окремих числових ознак.

```
# Теплова карта кореляцій
plt.figure(figsize=(10, 8))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm')
plt.title("Heatmap of Feature Correlations")
```

```
plt.show()

# Гістограми для окремих ознак
for column in data.select_dtypes(include=['int64', 'float64']).columns:
    plt.figure(figsize=(10, 6))
    sns.histplot(data[column], kde=True)
    plt.title(f"Histogram for {column}")
    plt.xlabel(column)
    plt.ylabel("Frequency")
    plt.show()
```

Спробуємо знайти, до якого кластера належать твіти конкретного користувача.

```
# Шукаємо кластер, до якого належать твіти користувача "@Send"
user_tweets = data[data['username'] == '@Send']
if not user_tweets.empty:
    cluster_label = user_tweets.iloc[0]['cluster']
    # Виведення твітів від "@Send" у цьому кластері
    tweets_from_user = data[(data['cluster'] == cluster_label) &
(data['username'] == '@Send')]
    print(f"Tweets from '@Send' in the same
cluster:\n{tweets_from_user['text']}")
else:
    print("No tweets found for '@Send'")
```

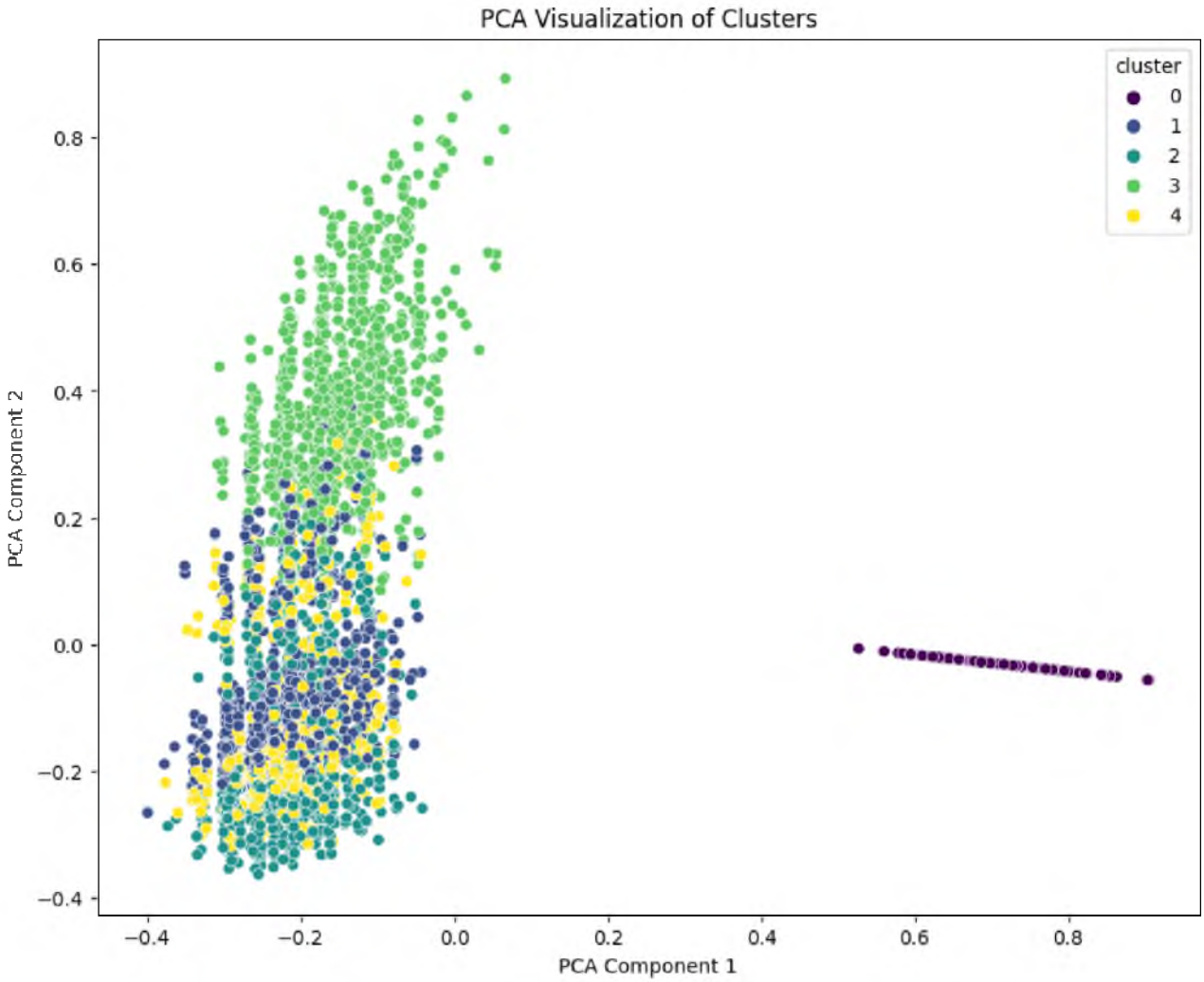



Рисунок 2.12 – Візуалізація Кластерів за допомогою PCA

```
Silhouette Coefficient: 0.13
Dunn's Index: 0.14
Total clusters: 5
Cluster 0 size: 1000
Cluster 1 size: 1093
Cluster 2 size: 869
Cluster 3 size: 908
Cluster 4 size: 830
Model Inertia: 2315.979829523314
Model Size: 20322 bytes
```

Рисунок 2.13 – Результати

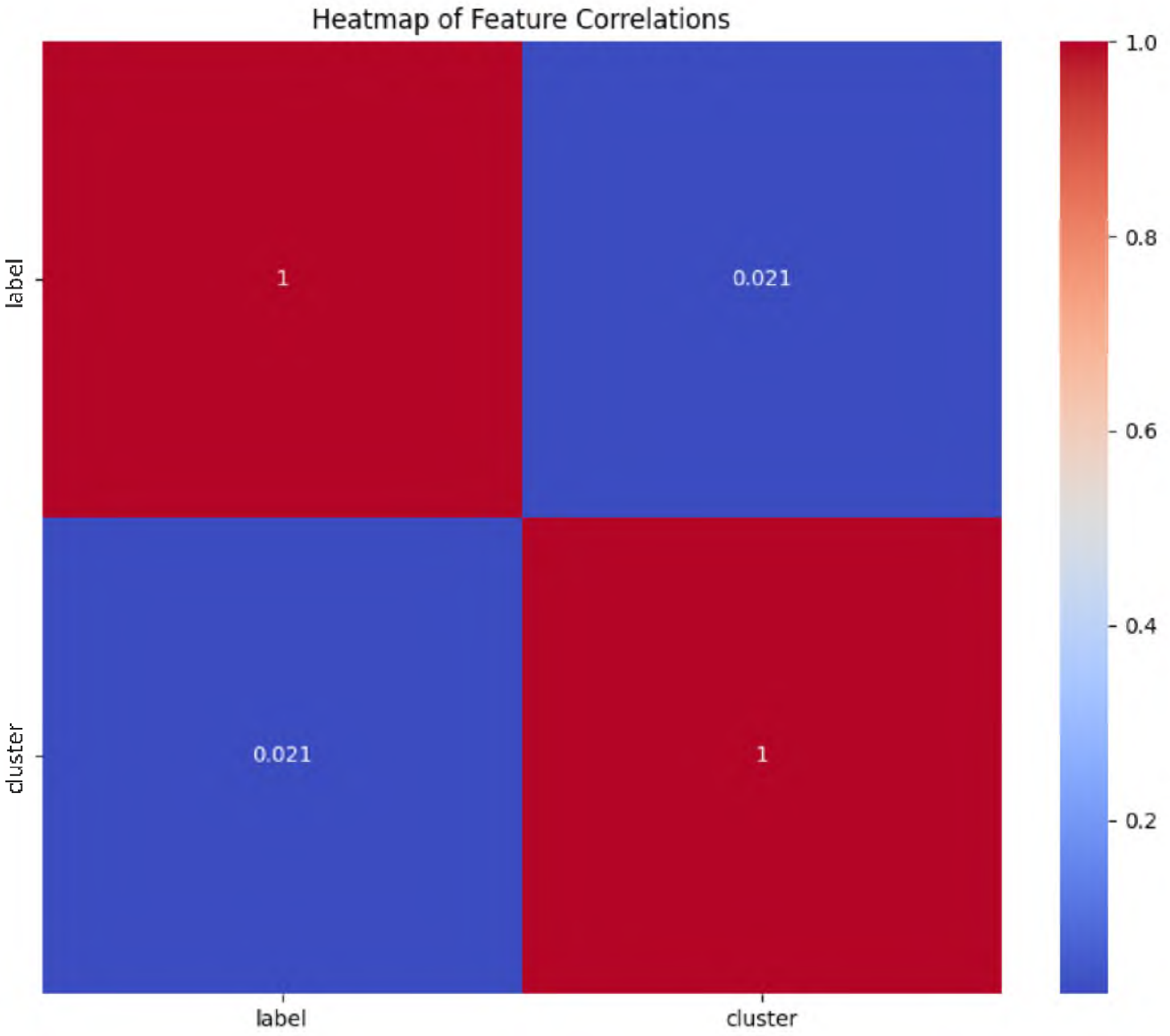



Рисунок 2.14 – Теплова карта кореляційних зв'язків ознак

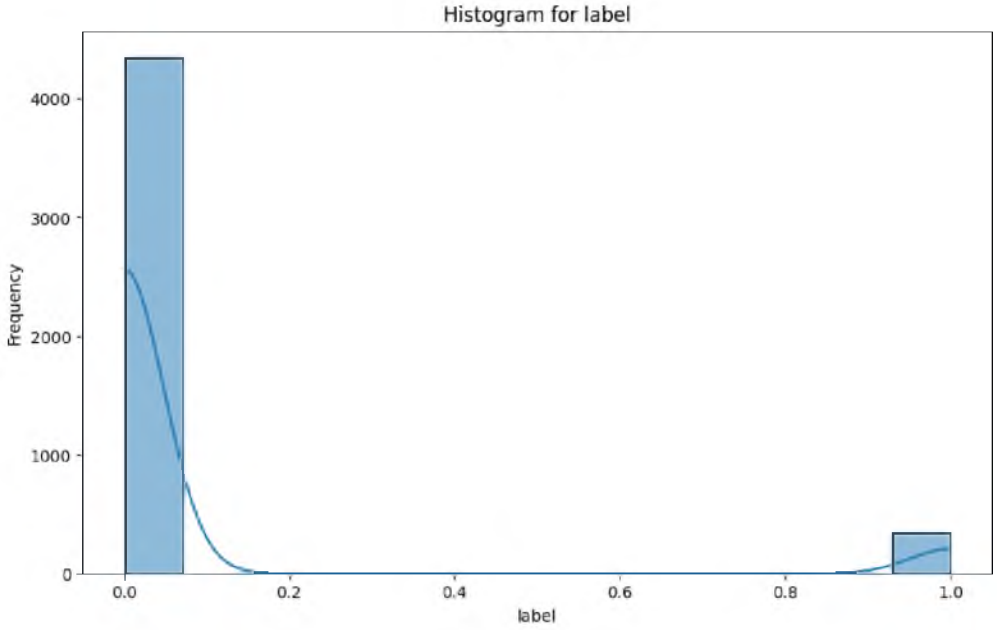



Рисунок 2.15 – Гістограма ознак

```
Tweets from '@Send' in the same cluster:
1              coffee circle sunshine happy sunshine book
26                        book circle music travel happy
83      love sunshine happy happy book circle food sun...
88                book food coffee coffee rain happy food
106      happy circle coffee sunshine happy circle circle
125                           book book book food sunshine
150                  sunshine sunshine happy circle food
168                rain sunshine coffee happy travel book
181             circle book travel happy circle book rain
187     book rain food happy circle sunshine circle fo...
189         happy rain book love music coffee happy happy
195     rain book happy sunshine book happy circle foo...
197     music circle happy sunshine coffee circle book...
216        food rain love food book coffee happy sunshine
228     love love book happy happy love coffee happy c...
236                      music sunshine travel book happy
248              sunshine circle coffee happy happy rain
267             coffee book circle food love music happy
270                circle book sunshine music food happy
303       music book book love sunshine happy circle book
329                happy happy happy circle circle happy
348     happy rain music music sunshine circle happy love
376     coffee circle food happy book love love coffee...
390      food book happy rain travel sunshine circle love
391     circle circle sunshine happy food sunshine rai...
397     coffee happy sunshine food coffee sunshine boo...
411                        food love happy book book food
429         book rain coffee food music book circle happy
430        coffee travel happy rain circle happy sunshine
446     love happy food travel happy happy food happy ...
452              happy travel sunshine sunshine food book
482      book happy love coffee food book circle sunshine
485                      happy book coffee circle coffee
2885    coffee happy rain book circle happy travel circle
3372               circle happy love circle happy circle
Name: text, dtype: object
```

Рисунок 2.16 – Результати кластеризації

## 2.8 Поріняння

Після отримання всіхї данних можно побудувати таблицю для проріняння та оцінити всі переваги кожної моделі.

Таблиці 2.1 – Метрики

| | Neural Network | MLPClassifier | CNN | RNN | KMeans |
|---|---|---|---|---|---|
| Accuracy | 0.9287 | 0.9255 | 0.9308 | 0.9276 | – |
| F1 Score | 0.8922 | 0.8928 | 0.1558 | 0.1052 | – |
| Overfitting | 0.0029 | 0.0 | -0.01 | -0.01 | – |
| Training Time | 11.5281 | 4.8316 | 11.2366 | 0.1052 | – |
| Correct Predictions | 869 | 870 | 875 | 872 | – |
| Model Size | 65954 bytes | 65953 bytes | 2078423 bytes | 2253749 bytes | 20322 bytes |
| Silhouette Coefficient | – | – | – | – | 0.13 |
| Dunn's Index | – | – | – | – | 0.14 |

Давайте розглянемо переваги та недоліки різних моделей нейронної мережі – Neural Network, MLPClassifier, CNN, RNN – та алгоритму KMeans, враховуючи їх особливості роботи та отримані результати навчання з таблиці (2.1).

Neural Network (Загальна Нейронна Мережа)

Переваги:

- Гнучкість: Може бути налаштована для вирішення різноманітних задач;
- Універсальність: Здатна виявляти складні шаблони та відносини в даних;
- Висока точність і F1 Score, швидке тренування, малий розмір моделі.

Недоліки:

- Перенавчання: присутність невеликого перенавчання. Схильна до навчання "за пам'яттю", особливо при великій кількості параметрів;
- Вимагає багато даних: для досягнення високої точності потрібні великі обсяги даних.

MLPClassifier (Multi–Layer Perceptron Classifier)

Переваги:

- Простота використання: Легше налаштовувати та тренувати порівняно з складнішими моделями. Добре підходить для табличних даних.
- Найвища точність і F1 Score серед усіх моделей, найшвидше тренування.

Недоліки:

- Обмежена здатність обробляти послідовні та просторові дані.
- Схильність до перенавчання при великих архітектурах.

CNN (Convolutional Neural Network)

Переваги:

- Ефективність в обробці зображень та просторових даних завдяки згортковим шарам. Часткова здатність до узагальнення завдяки спільній вагі фільтрів.
- Найвища точність, найбільша кількість правильних передбачень, відсутність перенавчання.

Недоліки:

- Висока обчислювальна складність та великі обсяги необхідних даних. Менш ефективні для обробки текстових даних порівняно з RNN.
- Низький F1 Score, що може свідчити про незбалансованість класів; довший час тренування та великий розмір моделі.

RNN (Recurrent Neural Network)

Переваги:

- Ефективність в обробці послідовних даних (текст, часові ряди). Здатність до зберігання інформації про попередній контекст у послідовностях.
- Висока точність, відсутність перенавчання.

Недоліки:

- Проблема зникання та вибуху градієнтів, яка ускладнює тренування.
- Вимагають багато часу та ресурсів для тренування.

KMeans (Алгоритм Кластеризації)

Переваги:

– Простота інтерпретації та реалізації. Ефективність у виявленні груп або кластерів у даних.

Недоліки:

– Вимагає заздалегідь визначеної кількості кластерів. Чутливий до масштабування ознак та вибору початкових точок.

У виборі між цими моделями важливо враховувати специфіку задачі, тип та обсяг даних, які у вас є, а також обчислювальні ресурси, доступні для тренування та використання моделі.

## 2.9. Висновок

В спеціальній частині було розглянуто різні методи машинного навчання, їх особливості, роботу і оцінено можливості їх роботи. Навчання моделей проводилось у середовищі Colaboratory. Для навчання було спеціально створено окрему вибірку даних та очищено її для подальшої роботи. Було обрано чотири різних методи машинного навчання, при порівнянні яких було виявлено, що для аналізу і пошуку певних твітів найкраще підходить нейронна мережа, тому в подальшому, для аналізу, було обрано різні види нейронних мереж.

При навчанні різних нейронних мереж було виявлено, що зокрема MLPClassifier продемонструвала високу ефективність. А саме оптимальне співвідношення швидкості тренування і класифікаційної точності, що є життєво важливим для обробки даних в реальному часі. Також ця модель забезпечує високий рівень точності та F1 Score, що гарантує ефективне визначення відповідних твітів, мінімізуючи помилкові позитивні та негативні результати. MLPClassifier показує хороші результати у різноманітних контекстах і з різними типами текстових даних, що робить його універсальним інструментом для аналізу даних. Також важливо, що завдяки порівняно невеликому розміру моделі, MLPClassifier вимагає менше обчислювальних ресурсів, що є важливим для аналізу в реальному часі. Ці фактори, у поєднанні з додатковими аспектами, такими як швидкість збору та обробки даних, масштабування системи та забезпечення стабільної інфраструктури, роблять MLPClassifier оптимальним вибором для використання в аналізі

соціальних мереж співробітників з метою виявлення каналів витоку інформації соціальної інженерії.

Тому цю модель і алгоритм навчання можно застосувати для системи виявлення каналів витоку інформації, яка би аналізувала контент співробітників у соціальних мережах. Це дозволить вчастно виявити і зупинити витік данних, про новий проєкт компанії, чи предачу важливих данних третім особам. Також при додаковому налаштуванні обрану модель можно буде застосувати безпосередньо для передбачення фішингових атак, за допомогою аналізу шаблонів фішингових повідомлень.

Але варто зазначити, що для нашої задачі, аналізу твітів користувачів чи пошуку конкретних слів, у випадку коли, необов'язкова точність і при невизначеності конретного об'єкта пошуку варто застосовувати алгоритм кластеризації KMeans. KMeans ефективний у виявленні природних груп або патернів у даних. Якщо вам потрібно ідентифікувати твіти, що містять певну тему чи ключові слова, KMeans може допомогти сгрупувати подібні твіти разом. KMeans є алгоритмом навчання без вчителя, що означає, що він не вимагає попередньо маркованих даних (лейблів чи ідентифікаторів) для тренування. Це знімає необхідність витрачати час на ручну обробку даних або використання додаткових ресурсів для отримання міток. KMeans не вимагає від даних певної репрезентації та може працювати з різними типами фіч, що дозволяє уникнути складного відбору та інженерії ознак, який часто потрібен для навчання з вчителем. Сукупність цих особливостей робить KMeans привабливим варіантом, коли швидкість є критично важливою.

Також ці моделі можливо використовувати не лише для аналізу тексту, а і для зображень. Що дозволить покрити весь можливий контент у соціальних мережах і дозволить передбачити більшість ситуацій.

Загалом використання машинного навчання, для виявлення і аналізу каналів витоку інформації, яка може стати цілью атаки соціальної інженерії, дозволить значно скоротити можливий вплив зловмисників на потенційні цілі, та дасть змогу передбачити можливі атаки.

Але варто зазначити що вибір найкращої моделі для конкретної задачі повинен базуватися на розумінні специфічних потреб та обмежень проекту, а також на

властивостях даних, з якими ви працюєте. Кожна з цих моделей має допомогти в розв'язанні широкого спектру задач, але потребує ретельного вибору та налаштування для досягнення найкращих результатів.

# РОЗДІЛ 3. ЕКОНОМІЧНА ЧАСТИНА.

## 3.1Розрахунок витрат

Розробка алгоритму на основі машинного навчання (нейронної мережі) для сканування та аналізу соціальних мереж співробітників має важливе значення для забезпечення інформаційної безпеки організації. Такий підхід дозволяє своєчасно виявляти відкриті канали потенційного витоку інформації через методи соціальної інженерії, що знижує ризики для конфіденційності даних та економічної безпеки підприємства. Однак, поряд з технічними аспектами реалізації такого алгоритму, необхідно враховувати й економічну доцільність впровадження системи. Вартість розробки та експлуатації системи повинна бути виважена з урахуванням потенційних ризиків та збитків, які можуть виникнути від інформаційних витоків. У цьому контексті проводиться оцінка витрат для організації, що включає в себе, наприклад, мережеву інфраструктуру з одним сервером, підключеним до інтернету, та п'ятьма комп'ютерами, з яких три також мають доступ до інтернету.

Нормування праці в процесі створення системи захисту від атак соціальної інжнерії істотно ускладнено через творчий характер праці спеціалістів з інформаційної безпеки. Проте трудомісткість розробки даної системи може бути розрахована на основі трудомісткості робіт, які виконуються.

### 3.1.1Трудомісткість

Трудомісткість створення ПЗ визначається тривалістю кожної робочої операції, починаючи з складання технічного завдання і закінчуючи оформленням документації (за умови роботи одного програміста): Трудомісткість створення ПЗ визначається тривалістю кожної робочої операції, починаючи з складання технічного завдання і закінчуючи оформленням документації (за умови роботи одного програміста):

$$t = t_{тз} + t_{в} + t_{а} + t_{пр} + t_{опр} + t_{д}, \text{годин}, \qquad (3.1)$$

де $t_{тз}$ – тривалість складання технічного завдання на розробку ПЗ,

де $t_{тз}$ = 8 годин

$t_в$ – тривалість вивчення ТЗ, літературних джерел за темою тощо;

$t_а$ – тривалість розробки блок–схеми алгоритма;

$t_{пр}$ – тривалість програмування за готовою блок–схемою;

$t_{опр}$ – тривалість опрацювання програми на ПК;

$t_б$ – тривалість підготовки технічної документації на ПЗ.

Складові трудомісткості визначаються на підставі умовної кількості операторів у програмному продукті Q (з урахуванням можливих учточнень у процесі роботи над алгоритмом і програмою)

Умовна кількість операторів у програмі:

$$Q = q * c(1 + p), \text{штук} \quad (3.2)$$

Де q – очікувана кількість операторів;

c – коефіцієнт складності програми. Коефіцієнт визначає відносну складність програми щодо типового завдання. Діапазон його зміни: 1.25…2

p – коефіцієнт корекції програми в процесі її опрацювання. Коефіцієнт визначає збільшення обсягу робіт за рахунок внесення змін в алгоритм або програму внаслідок уточнення технічного завдання. Діапазон его зміни 0,05…0,1

Очікувана кількість операторів (q): Код містить багато операторів, включаючи присвоєння, умовні конструкції, цикли, виклики функцій тощо. Усього їх приблизно 250.

Коефіцієнт складності програми (c): Код є досить складним, оскільки він включає обробку даних, машинне навчання та різні умовні конструкції. Візьмемо середнє значення: 1.5.

Коефіцієнт корекції програми (p): Припустимо, що зміни будуть помірними, та оберемо середнє значення 0.075.

За формолою (3.2) Розчитаємо кількість операторів у програмі:

$$Q=250*1.5*1.075=403$$

Тривалість вивчення технічного завдання складається з опрацювання довідкової літератури з урахуванням уточнення ТЗ і кваліфікації програміста можливо оцінити за формолою:

$$\text{tв} = \frac{Q * B}{(75 \dots 85) * k}, \text{годин} \qquad (3.3)$$

де В – коефіцієнт збільшення тривалості етапу внаслідок недостатнього опису завдання, В = 1,2…1,5;

k – коефіцієнт, що враховує кваліфікацію програміста і визначається стажем роботи за фахом:

- До 2 років – 0,8;
- Від 2 до 3 років 1,0;
- від 3 до 5 років 1,1…1,2;
- від 5 до 7 років - 1,3…1,4;
- понад 7 років - 1,5…1,6.

Внаслідок того що опис завдання буде дороблятися візьмемо середнє значення В = 1.3

Програміст, який зможе розробляти даний програмний код має мати кваліфакацію від 2 до 3 років, тому k = 1,0.

Розрахуємо тривалість вивчення ТЗ, літературних джерел за темою використовуючи формулу (3.3)

$t_в$= 6,5 годин

Тривалість розробки блок–схеми алгоритму:

$$ta = \frac{Q}{(20 \dots 25) * k}, \text{годин} \qquad (3.4)$$

Розрахуємо тривалість розробки блок–схеми алгоритму за формулою (3.4):

$t_а$=16 годин

Тривалість складання програми за готовою блок–схемою:

$$t\text{пр} = \frac{Q}{(20 \dots 25) * k}, \text{годин} \qquad (3.5)$$

Розрахуємо тривалість складання програми за готовою блок–схемою за формолою (3.5):

$t_{пр}$=16 годин

Тривалість опрацювання програми на ПК:

$$t\text{опр} = \frac{1{,}5Q}{(4 \dots 5) * k}, \text{годин} \qquad (3.6)$$

Розрахуємо тривалість опрацювання програми на ПК за формолою (3.6):

$t_{опр}$ = 121 годин

Тривалість підготовки технічної документації на ПЗ:

$$t\text{д} = \frac{Q}{(15 \dots 20) * k} + \frac{Q}{(15 \dots 20)} * 0{,}75, \text{годин} \qquad (3.7)$$

Розрахуємо тривалість підготовки технічної документації на ПЗ (3.7):

$t_д$ = 40 годин.

Підрахуємо трудомісткість використовуючи формулу (3.1):

де $t_{тз}$ = 8 годин, $t_в$ = 6,5 годин, $t_а$ = 16 годин, $t_{пр}$ = 16 годин, $t_{опр}$ = 121 годин, $t_д$ = 40 годин.

t = 8 + 6,5 + 16 +16 +121 + 50 = 217 годин

### 3.1.2 Розрахунок витрат на створення алгоритму захисту від атак соціальної інженерії.

Витрати на розробку алгоритму захисту від атак соціальної інженерії $К_{пз}$ складаються з витрат на заробітну плату спеціаліста з інформаційної безпеки $З_{зп}$ і вартості витрат машинного часу, що необхідний для розробки системи захисту $З_{мч}$

$$Кпз = Ззп + Змч \quad (3.8)$$

Заробітна плата виконавця враховує основну і додаткову заробітну плату, а також відрахування на соціальне потреби (пенсійне страхування, страхування на випадок безробіття, соціальне страхування тощо) и визначається за формулою:

$$Ззп = t * Зіб, грн$$

де *t* – загальна тривалість розробки аогоритму захисту, годин;

*Зіб* – средньо–годинна заробітна плата спеціаліста з інформаційної безпеки з нарахуваннями, грн/годину.

Заробітна плата в місяць спеціаліста інформаційної безпеки становить 25000, відповідно щоб отримати час за годину потрібно цю суму розділити на 160 годин. 4 годин середньомісячний час роботи з алгоритммом.

$$З_{ЗП} = 4 * 156 = 624 \text{ місяць} \quad (3.9)$$

Вартість машинного часу для налагодження програми на ПК визначається за формулою:

$$Змч = topр * Смч + tд, грн \quad (3.10)$$

де *topp* – трудомісткість налагодження програми на ПК, годин;

tд– трудомісткість підготовки документації на ПК, год.;

$C_{мч}$ – вартість 1 години машинного часу ПК, грн./година.

Вартість 1 години машинного часу ПК визначається за формулою:

$$\text{Смч} = \text{Р} * \text{t нал} * \text{Се} + \frac{\text{Фзал} * \text{На}}{Fp} + \frac{\text{Клпз} * \text{Напз}}{Fp}, \text{грн} \quad (3.11)$$

де $P$ – встановлена потужність ПК, кВт;

$t_{нал}$ – річний час використання ПК на рік у годинах

$C_e$ – тариф на електричну енергію, грн/кВт година;

$Ф_{зал}$ – залишкова вартість ПК на поточний рік, грн.;

$H_a$ – річна норма амортизації на ПК, частки одиниці;

$H_{апз}$ – річна норма амортизації на ліцензійне програмне забезпечення, частки одиниці;

$K_{лпз}$ – вартість ліцензійного програмного забезпечення, грн.;

$F_p$ – річний фонд робочого часу (за 40–годинного робочого тижня $F_p$ = 2080).

$$С_{мч} = 0,2 * 177 *6 + ((40000*0.2)/177) + ((7000*0.2)/177) = 265 \text{ грн}$$

$$З_{мч} = 121*2499 + 40*265 = 42665 \text{ грн}$$

Маючи всі необхідні числа можемо розрахуати витрати на розробку алгоритму за формулою (3.8):

$$К_{пз} = 42665+624=43289 \text{ грн}$$

Залишкова вартість ПК визначається виходячи з фактичного терміну його експлуатації як різниця між первісною вартістю та зносом за час використання. Визначена таким чином вартість розробки політики безпеки $Крп$ є частиною одноразових капітальних витрат разом з витратами на придбання і налагодження апаратури системи інформаційної безпеки.

Таким чином, капітальні (фіксовані) витрати на проектування та впровадження проектного варіанта системи інформаційної безпеки складають:

$$К = Крп + Кзпз + Кпз + Каз + Кнавч + Кн \quad (3.12)$$

де $К_{рп}$ – вартість розробки проекту інформаційної безпеки та залучення для цього зовнішніх консультантів, тис. гри. Зовнішні консультанти не наймалися ($К_{рп}$=1800)

$К_{зпз}$ – вартість закупівель ліцензійного основного й додаткового програмного забезпечення (ПЗ), тис. грн;

Таблиця 3.1 – Перелік придбаного Пз

| № | Назва | Кількість | Вартість за 1 шт |
|---|---|---|---|
| 1 | Microsoft Windows 11 | 1 | 6000 грн |
| 2 | Microsotf visual studio | 1 | 1000 грн |
| Всього: $К_{зпз}$ = 7000 грн | | | |

$К_{рп}$ – вартість розробки системи виявлення каналів витоку інформації яка би аналізувала контент співробітників у соціальних мережах, тис. гри; ($К_{пз}$ =72773)

$К_{аз}$ – вартість апаратного забезпечення та допоміжних матереалів. Приблизна ціна 1000грн

$К_{навч}$ – витрати на навчання технічних фахівців і обслуговуючого персоналу, тис. гри; ($К_{навч}$ =1250грн)

$К_{н}$ – витрати на встановлення обладнання та налагодження системи інформаційної безпеки, тис. гри. ($К_{н}$ = 160)

За формулою (3.12) розрахуємо капітальні витрати:

$$К = 1800+7000+43289+1000+1250 = 56339 \text{ грн}$$

3.1.3 Розрахунок поточних експлуатаційних витрат

Експлуатаційні витрати – це поточні витрати на експлуатацію та обслуговування об'єкта проектування за визначений період (наприклад, рік), що виражені у грошовій формі.

$$С = Св + Ск + Сак, тис. грн \quad (3.13)$$

Де $С_в$ – Витрати на Upgrade–відновлення й модернізацію системи інформаційної безпеки.

$С_к$ – витрати на керування системою в цілому;

$С_{ак}$ – «активність користувача», витрати викликані активністю користувачів системи інформаційної безпеки.

$С_в$ – орієнтовно визначимо виходячи із вагової частки статей витрат (11%) з сукупної вартості інформаційної системи

$$С_в = 43289 * 0.11 = 4761 \text{ грн.}$$

Витрати на керування системою інформаційної безпеки ($С_к$) складають:

$$Ск = Сн + Са + Сз + Сєв + Сел + со + Стос, грн \quad (3.14)$$

Витрати на навчання адміністративного персоналу и кінцевих користувачів визначаються за даними організації з проведення тренінгів персоналу, курсів підвищення кваліфікації тощо (Сн=3000 грн).

Річний фонд амортизаційних відрахувань ($С_а$) визначається у відсотках від суми капітальних інвестицій за видами основних фондів і нематеріальних активів (ПЗ); Вартість купівлі ліцензійного ПЗ – 7000 грн., мінімальний срок дії користування – 2 роки

$$С_а = 7000/2 * 100 = 3500 \text{ грн/рік}$$

Річний фонд заробітної плати інженерно–технічного персоналу, що обслуговує систему інформаційної безпеки ($С_З$), складає:

$$Сз = Зосн + Здод, грн \quad (3.15)$$

Де $З_{осн}$ $З_{дод}$ – основна і додаткова заробітна плата відповідно, грн на рік.

Основна заробітна плата визначається, виходячи з місячного посадового окладу, а додаткова заробітна плата – в розмірі 8–10% від основної заробітної плати.

$$З_{дод} = 7488 * 9\% = 673 \text{ грн. на рік}$$

Основна Заробітна плата в місяць спеціаліста інформаційної безпеки становить ($З_{осн}$ = 30180 грн/місяць):

$$С_З = 7488 + 673 = 8161 \text{ грн}$$

До річного фонду заробітної плати додається єдиний внесок на загальнообов'язкове державне соціальне страхування – консолідований страховий внесок, збір якого здійснюється відповідно до класів професійного ризику виробництва, до яких віднесено платників єдиного внеску, з урахуванням видів їх економічної діяльності.

Розмір єдиного внеску на загальнообов'язкове державне соціальне страхування визначається на підставі встановленого чинним законодавством відсотка від суми основної та додаткової заробітної плати (за узгодженням з консультантом економічної частини дипломного проекту):

$$С_{есв} = 7488 * 0.22 = 1647 \text{ грн}$$

*Вартість електроенергії*, що споживається апаратурою системою інформаційної безпеки протягом року ($С_{ел}$), визначається за формулою:

$$Сел = Р * FP * Це, грн \quad (3.16)$$

де Р – встановлена потужність апаратури інформаційної безпеки, кВт;

$$Р = 200 Вт = 0,2 кВт$$

$F_р$ – річний фонд робочого часу системи інформаційної безпеки (приблизно 177 год на рік, так як застосовується лише за необхідностю);

Це – тариф на електроенергію, грн/кВт–годин. (Це = 6).

$$С_{ел} = 0.2 *6 * 177 = 212 грн$$

*Витрати на залучення* сторонніх *організацій* для виконання деяких видів обслуговування, навчання та сертифікацію обслуговуючого персоналу визначаються за даними організації. ($С_о = 0$)

*Витрати на технічне й організаційне адміністрування та сервіс* системи інформаційної безпеки ($С_{тос}$) визначаються за даними організації або у відсотках від вартості капітальних витрат (1–3%).

$$К = 56339 грн$$
$$С_{тос} = 56339 * 2 \% = 1126 грн$$

Витрати, викликані активністю користувачів системи інформаційної безпеки ($С_{ак}$) можна орієнтовно визначити, користуючись даними табл. 1 про вагові частки статей витрат у сукупній вартості системи інформаційної безпеки.

$$С_{ак} = 56339 * 38\% = 21408 грн$$

Розрахуємо витрати на керування системою інформаційної безпеки за формулою (3.14):

$$С_к = 3000 + 3500 + 8161 + 1647 + 212 + 0 + 1126 = 14646 \text{ грн}$$

У кожному конкретному випадку можуть бути враховані й інші види поточних витрат, що визначаються специфікою експлуатації проектованої системи інформаційної безпеки.

3.2 Оцінка Величини збитку

Загалом можна виділити таки види збитків, які даний алгоритм виявлення витоків інформації може допомогти запобігти:

1. Втрата комерційної інформації. Витік комерційної інформації призведе до розриву контракту з виплатою штрафних санкцій. Сума збитків від розриву може становити близько 10 мільйонів гривень.

1. Розголос інформації підвищеної секретності (Державна таємниця). Втрата чи розголошення інформації підвищеної секретності тягне за собою кримінальну відповідальність, відповідних осіб.

1. Репутаційні втрати. Репутаційні втрати призведуть до труднощів у роботі з майбутніми клієнтами, а це може призвести до втрати 50 млн. грн на рік.

Загальний ефект від впровадження системи інформаційної безпеки визначається з урахуванням ризиків порушення інформаційної безпеки і становить:

$$E = B * R - C \qquad (3.17)$$

де $B$ – загальний збиток від атаки (витоку), тис. гри;

$R$ – очікувана імовірність атаки (витоку інформації), частки одиниці;

$C$ – щорічні витрати на експлуатацію системи інформаційної безпеки, гри.

Для розрахунку приймаємо один випадок атаки на рік, що призведе до втрат:

$$1000000+5000000 = 6 \text{ млн.грн}$$

$$E = 6000000 * 1 - 14646 = 5985354$$

## 3.3 Визначення та аналіз показників економічної ефективності

Показник сукупної вартості володіння (ТСО) використовується, якщо величину відверненого збитку від атаки на вузол або сегмент корпоративної мережі важко або неможливо визначити у вартісній формі.

У цьому випадку необхідно порівняти сукупну вартість володіння, розраховану для двох варіантів проектного рішення щодо створення або удосконалення системи інформаційної безпеки, і вибрати варіант із найменшою з них.

Коефіцієнт повернення інвестицій ROSI показує, скільки гривень додаткового прибутку приносить одна гривня капітальних інвестицій на впровадження системи інформаційної безпеки.

Щодо до інформаційної безпеки говорять не про прибуток, а про запобігання можливих втрат від атаки на сегмент або вузол корпоративної мережі, а отже:

$$\text{ROSI} = \frac{E}{K}, \text{частка одиниці} \qquad (3.18)$$

де Е – загальний ефект від впровадження системи інформаційної безпеки (розділ 3.2 методичних вказівок, формула 3.8), 5985354грн;

К – капітальні інвестиції за варіантами, що забезпечили цей ефект, тис. гри.

Якщо порівнюється два варіанти системи інформаційної безпеки, то обирається варіант з більшим значенням ROSI.

$$\text{Rosi} = 5985354/56339 = 106 \text{ частка одиниці}$$

Для остаточної оцінки варіантів і вибору найбільш ефективного з них необхідно порівняти розрахункове значення ROSI з бажаним значенням показника ефективності Ен.

Проект системи інформаційної безпеки визнається доцільним за умови

$$ROSI > EH$$

При $ROSI<E_H$ варіант є збитковим і більш економічним визнається відмова від його реалізації.

Розрахунок бажаного значення коефіцієнта ефективності виконується за узгодженням з консультантом економічної частини.

Для вибраного варіанта визначається розрахунковий строк окупності капітальних інвестицій $T_p$.

***Термін окупності*** капітальних інвестицій $T_p$ показує, за скільки років капітальні інвестиції окупляться за рахунок загального ефекту від впровадження системи інформаційної безпеки:

$$To = \frac{K}{E} = \frac{1}{ROSI} \text{ років} \quad (3.19)$$

Якщо варіанти економічно рівноцінні, то приймається варіант, що забезпечує більш високу надійність, поліпшення умов праці.

$$T_o = 56339/5985354 = 0.0094 \text{років}$$

Таким чином, термін окупності капітальних інвестицій становить приблизно 0.00094 року. Це дуже короткий термін, що свідчить про високу ефективність інвестицій у вказану систему інформаційної безпеки. Це число можна перевести в дні, помноживши на 365 (кількість днів у році):

$0.00094 \text{ року} \times 365 \text{ днів/рік} \approx 3.43 \text{дні}$

Тобто окупність відбудеться за 3 дні.

## 3.4 Висновок

Згідно з наведеними розрахунками можливо зробити висновок, що розробка алгоритму на основі машинного навчання для сканування та аналізу соціальних мереж співробітників є економічно доцільною.

Капітальні витрати, які складають 56339 грн, дозволяють отримати ефект величиною 5985354 грн. Відповідно до отриманих значень показників економічної можна зазначити, що запропонований підхід дозволить отримувати 106 економічного ефекту на 1 грн. капітальних витрат (коефіцієнт повернення інвестицій ROSI складає 106 частка одиниці). Термін окупності при цьому складатиме 3 дні.

# ВИСНОВКИ

Було проведено огляд та оцінка ефективності різних класифікаційних моделей машинного навчання, таких як MLPClassifier, RandomForestClassifier, SVC, і GradientBoostingClassifier, CNN, RNN для ідентифікації та аналізу соціальних мереж співробітників, пошуку користувачів і тексту їх повідомлень. Серед цих алгоритмів, MLPClassifier виявився найбільш ефективним для виявлення витоків інформації у соціальних мережах.

Розроблена модель демонструє високу здатність до ефективного розпізнавання необхідних слів у тексті та пошуку конкретних користувачіва, що є ключовим аспектом для своєчасного виявлення каналів витоку. Ця модель дозволяє не тільки виявляти можливі витоки в реальному часі, але й швидко реагувати на них, запобігаючи та мінімізуючи можливі збитки від атак соціальної інженерії.

Дане дослідження підтверджує, що використання машинного навчання для виявлення каналів витоку соціальної інженерії є перспективним напрямком. Це може стати в нагоді для підприємст стратегіної галузі, у яких завжди є необхідність в посиленні інформаційної безпеки. Важливо враховувати необхідність регулярного оновлення та доопрацювання моделей для забезпечення їх актуальності та ефективності у відповідь на постійно еволюціонуючи патерни атак.

Згідно з наведеними розрахунками можливо зробити висновок, що розробка алгоритму на основі машинного навчання для сканування та аналізу соціальних мереж співробітників є економічно доцільною.

Капітальні витрати на розробку алгоритму на основі машинного навчання для сканування та аналізу соціальних мереж співробітників складають 56339 грн. а коефіцієнт повернення інвестицій (ROSI) складає 106 частка одиниці, що свідчить про економічну доцільність запропонованого підходу. Окупність інвестицій у розробку алгоритму виявлення витоків соціальної інженерії з використанням машинного навчання становить 3 дні, що також підкреслює її економічну ефективність.

# ПЕРЕЇЛК ПОСИЛАНЬ

1. Фішинг [Електронний ресурс] – Режим доступу до ресурсу: https://www.eset.com/ua/support/information/entsiklopediya-ugroz/fishing/

2. Що таке вішинг? І як від нього захиститися [Електронний ресурс]– Режим доступу до ресурсу: https://www.scamadviser.com/uk/articles/shcho-take-vishynh-i-iak-vid-nyoho-zakhystytysia.

3. Smishing: SMS Phishing Explained [Електронний ресурс] – Режим доступу до ресурсу: https://www.twilio.com/blog/smishing-sms-phishing.

4. IBM\'s 2023 Cost of a Data Breach report [Електронний ресурс]. 2023. – Режим доступу до ресурсу: https://www.ibm.com/account/reg/us-en/signup?formid=urx-52258.

5. Phishing - Statistics & Facts [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: https://www.statista.com/topics/8385/phishing/#editorsPicks.

6. FBI\'s Internet Crimes Report (2022) [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://www.ic3.gov/Media/PDF/AnnualReport/2021_IC3Report.pdf.

7. 2022 Data Breach Investigations Report [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://www.verizon.com/business/en-gb/resources/2022-data-breach-investigations-report-dbir.pdf.

8. Spear Phishing [Електронний ресурс] – Режим доступу до ресурсу: https://assets.barracuda.com/assets/docs/dms/spear-phishing_report_vol6.pdf.https://pages.sift.com/rs/526-PCC-974/images/Sift-2023-Q3-Index-Report_ATO.pdf

9. Social Engineering Statistics to Know in 2023 [Електронний ресурс].– 2023. – Режим доступу до ресурсу: https://www.resmo.com/blog/social-engineering-statistics.

10. Chio C. Machine Learning and Security / C. Chio, D. Freeman. Boston: O*Reilly, 2020.

11. Google Colaboratory [Електронний ресурс] // Google – Режим доступу до ресурсу: https://colab.research.google.com/?hl=ru.

12. Bishom C. M. Pattern Recognition and Machine learning / Christopher Bishom., 2006.

13. Support Vector Machines [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: https://scikit-learn.org/stable/modules/svm.html#mathematical-formulation.

14. Chris A. Machine Learning with Python Cookbook Practical Solutions from Preprocessing to Deep Learning / Albon Chris., 2018.

# ДОДАТОК А. Відомість матеріалів кваліфікаційної роботи

| № | Формат | Найменування | Кількість | Примітки |
|---|---|---|---|---|
| Документація | | | | |
| 1 | А4 | Реферат | 2 | |
| 2 | А4 | Список умовних скорочень | 1 | |
| 3 | А4 | Зміст | 2 | |
| 4 | А4 | Вступ | 1 | |
| 5 | А4 | Стан Питання. Постановка задачі | 21 | |
| 6 | А4 | Спеціальна частина | 43 | |
| 7 | А4 | Економічний розділ | 14 | |
| 8 | А4 | Висновки | 1 | |
| 9 | А4 | Перелік посилань | 2 | |
| 10 | А4 | Додаток А | 1 | |
| 11 | А4 | Додаток Б | 1 | |
| 12 | А4 | Додаток В | 1 | |
| 13 | А4 | Додаток Г | 1 | |

# ДОДАТОК Б. Перелік документів на оптичному носії

ХричовОВ_125м-22-2_ПЗ.docx
ХричовОВ_125м-22-2_ПЗ.pdf
ХричовОВ_125м-22-2_П.pptx

ДОДАТОК В. Відгук керівника кваліфікаційної роботи

В І Д Г У К

на кваліфікаційну роботу студента групи 125м-22-2 Хричова Олександра Вадимовича

на тему: « Виявлення каналів витоку інформації соціальної інженерії за допомогою машинного навчання.»

Пояснювальна записка складається зі вступу, трьох розділів і висновків, розташованих на 93 сторінках.

Мета роботи є актуальною, оскільки вона спрямована на підвищення освітлення проблем захисту від атак соціальної інженерії які є великою проблемою сьогодення.

При виконанні роботи автор продемонстрував добрий рівень теоретичних знань і практичних навичок. На основі аналізу різноманітних методів та моделей машинного навчання для виявлення витоків, був сформульований найбільш відповідний метод захисту від витоку інформації.

Практична цінність роботи полягає у тому, що представлений метод захисту є повноцінною можливістю уникнення атак та забезпечення доступності інформації.

Рівень запозичень у кваліфікаційній роботі не перевищує вимог «Положення про систему виявлення та запобігання плагіату».

В цілому робота задовольняє усім вимогам, а її автор Хричов О.В. заслуговує на оцінку «відмінно» та присвоєння кваліфікації «Магистр з кібербезпеки» за спеціальністю 125 Кібербезпека.

| | |
|---|---|
| Керівик кваліфікаційної роботи к. ф-м. н., професор | Магро В.І. |
| Керівник спеціального розділу ст. викладач | Святошенко В.О. |

ДОДАТОК Г. Відгук керівника економічного розділу

Економічний розділ виконаний відповідно до вимог, які ставляться до кваліфікаційних робіт, та заслуговує на оцінку 90 б. («відмінно»).

Керівник розділу _____ Доцент. к.е.н. Пілова Д.П.

(підпис) (ініціали, прізвище)