Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

БАКАЛАВРСЬКА ДИПЛОМНА РОБОТА

на тему:

«Розробка програмного модуля виявлення зловмисних посилань для онлайн ресурсів»

Виконав: студент 4 курсу, гр. 2КІТС-20б
спеціальності 125– Кібербезпека
Освітня програма – Кібербезпека
інформаційних технологій та систем

Олексюк Є. М.

Керівник: к.т.н., проф., проф. каф. МБІС
Азарова А. О.

«6» червня 2024 р.

Рецензент: к.т.н., доц., доцент каф. ОТ
Тарновський М. Г.

«6» червня 2024 р.

Допущено до захисту
Голова секції УБ кафедри МБІС
д.т.н., проф. Яремчук Ю. Є.
"6" червня 2024 р.

Вінниця ВНТУ – 2024

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Рівень вищої освіти І-й (бакалаврський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма - Кібербезпека інформаційних технологій та систем

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС
д.т.н., проф. Яремчук Ю.Є.
“ 22 ” березня 2024 р.

З А В Д А Н Н Я
на бакалаврську дипломну роботу студенту
Олексюк Євгеній Михайлович

1. Тема роботи «Розробка програмного модуля виявлення зловмисних посилань для онлайн ресурсів»
Керівник роботи проф. Азарова Анжеліка Олексіївна
затверджені наказом вищого навчального закладу від “ 11 ” березня 2024року № 80
2. Термін подання студентом роботи 10.06.2024 р.
3. Вихідні дані до роботи Набір даних з безпечними та зловмисними URL-адресами для навчання моделі машинного навчання.
4. Зміст текстової частини:
1. Аналіз існуючих підходів до виявлення зловмисних посилань на онлайн ресурси
2. Розроблення методу до виявлення зловмисних посилань засобами методів чорних і білих списків та к-найближчих сусідів
3. Розроблення програмного модуля до виявлення зловмисних посилань на онлайн ресурси
5. Перелік ілюстративного матеріалу (з точним зазначенням обов`язкових креслень)
Презентація у форматі PowerPoint з ілюстраціями архітектури програмного модуля, скріншотами інтерфейсу та результатами тестування.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Розділ I	проф. Азарова А. О.		
Розділ II	проф. Азарова А. О.		
Розділ III	проф. Азарова А. О.		

7. Дата видачі завдання 22 березня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Ознайомлення з темою та постановкою завдання	22.03.2024	24.03.2024	
2	Пошук та вивчення літератури за темою дослідження	25.03.2024	31.03.2024	
3	Аналіз існуючих підходів до виявлення зловмисних посилань	01.04.2024	07.04.2024	
4	Обґрунтування вибору методу KNN та розроблення підходу	08.04.2024	14.04.2024	
5	Вибір мови програмування та середовища розробки	15.04.2024	21.04.2024	
6	Розроблення інтерфейсу програмного модуля	22.04.2024	28.04.2024	
7	Реалізація функціоналу виявлення зловмисних посилань	29.04.2024	05.05.2024	
8	Тестування програмного модуля	06.05.2024	12.05.2024	
9	Оформлення текстової частини роботи	13.05.2024	22.05.2024	
10	Переддипломний захист			
11	Захист бакалаврської дипломної роботи			

Студент (підпис) Олексюк Є. М. (прізвище та ініціали)

Керівник роботи (підпис) Азарова А. О. (прізвище та ініціали)

АНОТАЦІЯ

Бакалаврська дипломна робота складається з 65 сторінок формату А4, на яких є 17 рисунків, 2 таблиці, 3 додатки, список використаних джерел має 26 найменувань.

Актуальність теми. Зловмисні посилання на онлайн ресурси є серйозною загрозою для безпеки користувачів Інтернету, оскільки вони можуть призвести до завантаження шкідливого програмного забезпечення, крадіжки конфіденційних даних або фінансових втрат. Тому розробка ефективних методів для виявлення та блокування таких посилань є актуальним завданням.

Мета та завдання дослідження. Метою роботи є розроблення програмного модуля для виявлення зловмисних посилань на онлайн ресурси з використанням методу машинного навчання K-найближчих сусідів та чорних, білих списків. Основними завданнями дослідження є аналіз існуючих підходів до виявлення зловмисних посилань, обґрунтування вибору методу KNN, розроблення методу виявлення зловмисних посилань на базі KNN та білих/чорних списків, створення програмного модуля з графічним інтерфейсом користувача.

Об'єкт дослідження: процес виявлення зловмисних посилань на онлайн ресурси. Предмет дослідження: методи та інструменти для виявлення зловмисних посилань на онлайн ресурси. Методи дослідження: аналітичний огляд літератури, методи машинного навчання KNN, об'єктно-орієнтоване програмування.

Наукова новизна: запропоновано новий підхід до виявлення зловмисних посилань на онлайн ресурси, який поєднує метод машинного навчання KNN та чорні, білі списки для підвищення точності та ефективності виявлення.

Практичне значення: розроблений програмний модуль може бути використаний організаціями та користувачами для швидкого та ефективного виявлення зловмисних посилань на онлайн ресурси, що сприятиме підвищенню рівня кібербезпеки.

Ключові слова: машинне навчання, k-найближчих сусідів, чорні списки, білі списки, зловмисні посилання.

ABSRTACT

Bachelor`s thesis consists of 65 pages in A4 format, containing 17 figures, 2 tables, 3 appendices, and a list of 26 references.

Relevance of the topic. Malicious links to online resources pose a serious threat to the security of Internet users, as they can lead to the download of malicious software, theft of confidential data, or financial losses. Therefore, the development of effective methods for detecting and blocking such links is an urgent task.

The purpose and objectives of the study. The aim of the work is to develop a software module for detecting malicious links to online resources using the K-nearest neighbors machine learning method and blacklists, whitelists. The main objectives of the study are to analyze existing approaches to detecting malicious links, justify the choice of the KNN method, develop a method for detecting malicious links based on KNN and white/black lists, and create a software module with a graphical user interface.

Object of research: the process of detecting malicious links to online resources.

Subject of research: methods and tools for detecting malicious links to online resources.

Research methods: analytical literature review, KNN machine learning methods, object-oriented programming.

Scientific novelty: a new approach to detecting malicious links to online resources is proposed, which combines the KNN machine learning method and blacklists, whitelists to improve the accuracy and effectiveness of detection.

Practical significance: the developed software module can be used by organizations and users for quick and effective detection of malicious links to online resources, which will contribute to improving cybersecurity.

Keywords: machine learning, k-nearest neighbors, blacklists, whitelists, malicious links.

ЗМІСТ

ВСТУП

У сучасному світі онлайн-ресурси стали невід'ємною частиною повсякденного життя людей. Однак разом з їхнім зростанням зростає й ризик кіберзлочинності. Зловмисні посилання є одним із найпоширеніших методів кібератак, які використовуються для крадіжки особистих даних, поширення шкідливого програмного забезпечення та інших злочинних цілей.

Розроблення ефективних методів виявлення зловмисних посилань є актуальною науково-практичною проблемою, яка має значний вплив на розвиток інформаційних технологій та кібербезпеку. Вирішення цієї проблеми має важливе значення для захисту користувачів онлайн-ресурсів, особливо в Україні, де кібератаки стають все більш поширеними.

Опрацьовано значний теоретичний доробок по темі дослідження завдяки роботам провідних закордонних та вітчизняних авторів таких як : Brownlee J., Buelta J., Géron A., Vidales A., W. Shang, Zhang S., Данилюк І. І., Карпінець В. В., Приймак А. В., Яремчук Ю. Є. [1–5].

На сьогоднішній день існує багато різних підходів до виявлення зловмисних посилань на онлайн ресурси, зокрема долучення сайту до чорного та білого списків URL, аналіз контенту веб-сторінки, вивчення поведінки користувача, дослідження структури і компонентів URL-адреси, методи машинного навчання, а також аналіз DNS, аналіз трафіку, біометрична автентифікація. Проте вони не є самодостатніми і вимагають комплексного застосування для результативного пошуку, тому актуальним є розвиток існуючих підходів до виявлення зловмисних посилань на онлайн ресурси.

Мета цього дослідження полягає у покращенні процесу ідентифікації зловмисних посилань на онлайн ресурси із застосуванням чорних, білих списків та методу К-найближчих сусідів.

Практична новизна дослідження полягає в розробленні програмного модуля, який можна використовувати для захисту онлайн-ресурсів від зловмисних

посилань. Програмний модуль може бути інтегрований у веб-сайти, мобільні додатки та інші онлайн-платформи.

Об'єктом дослідження бакалаврської роботи є процес виявлення зловмисних посилань в інформаційних середовищах.

Предметом дослідження є автоматизований підхід для виявлення зловмисних посилань на онлайн ресурси.

Публікації: Азарова А. О., Олексюк Є. М., Азарова Л. Є., Комп’ютерна програма «Розробка програмного модуля виявлення зловмисних посилань для онлайн ресурсів»

1 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ДО ВИЯВЛЕННЯ ЗЛОВМИСНИХ ПОСИЛАНЬ НА ОНЛАЙН РЕСУРСИ

У цьому розділі проводиться аналіз різних методів виявлення зловмисних посилань для онлайн ресурсів: проведено детальний огляд існуючих підходів до виявлення зловмисних посилань, зокрема, методів чорних і білих списків URL, аналіз контенту веб-сторінок, поведінковий аналіз користувачів та методів машинного навчання. Проаналізовано ефективність кожного підходу, а саме його точність, швидкість виявлення та здатність адаптуватися до нових видів атак. Визначено переваги та недоліки кожного підходу, у тому числі обмеження та потенційні ризики, пов’язані з їхнім застосуванням. Це уможливило обґрунтувати використання методів чорних і білих списків та машинного навчання для подальших досліджень.

1.1 Базові поняття процесу виявлення зловмисних посилань

Зловмисне посилання – це веб-сайт або ресурс, який спеціально створений для вчинення шкідливих дій або виконання злочинних дій. Такі веб-сайти можуть бути використані для розповсюдження вірусів, шкідливого програмного забезпечення або для здійснення атак фішингу з метою викрадення особистої інформації користувачів [6].

Важливо розуміти, що зловмисні веб-сайти можуть мати різні цілі. Деякі з них можуть намагатися викликати шкоду комп'ютерам або пристроям користувачів, використовуючи вразливості програмного забезпечення або перехоплюючи конфіденційну інформацію. Інші можуть намагатися обманути користувачів, наприклад, шляхом імітації авторитетних веб-сайтів або послуг, щоб отримати їхні паролі, номери кредитних карток або іншу особисту інформацію.

Поширення шкідливого програмного забезпечення через зловмисні посилання є одним з найпоширеніших методів кібератак. Зловмисники використовують такі посилання для того, щоб заманити користувачів на заражені

веб-сайти або змусити їх завантажити та встановити шкідливе програмне забезпечення на свої пристрої.

Як відбувається поширення шкідливого ПЗ через зловмисні посилання:

1. Зловмисники створюють зловмисні посилання: Ці посилання можуть бути розміщені в електронних листах, повідомленнях у соцмережах, на веб-сайтах або в рекламі. Їх часто маскують під легітимні посилання, щоб заманити користувачів натиснути на них.

2. Користувач натискає на зловмисне посилання: Коли користувачем було натиснено на зловмисне посилання, людину може бути перенаправлено на заражений веб-сайт або йому може бути автоматично запропоновано завантажити та встановити шкідливе ПЗ.

3. Шкідливе ПЗ інсталюється на пристрій: Якщо користувач завантажує та встановлює шкідливе ПЗ, воно може виконати різні шкідливі дії, наприклад:

– крадіжка особистих даних: шкідливе ПЗ може красти особисті дані користувача, такі як паролі, номери кредитних карток або банківські реквізити;

– шпигунство: шкідливе ПЗ може шпигувати за діяльністю користувача в Інтернеті, записувати натискання клавіш або робити знімки екрана;

– шифрування даних: шкідливе ПЗ може зашифрувати дані користувача і вимагати викуп за їх розшифрування;

– використання комп'ютера для майнінгу криптовалюти: шкідливе ПЗ може використовувати комп'ютер користувача для майнінгу криптовалюти без його згоди;

– поширення на інші пристрої: шкідливе ПЗ може поширюватися на інші пристрої в локальній мережі користувача;

– перенаправлення на небажані сайти: деякі зловмисні посилання можуть перенаправляти на веб-сайти, які містять неприємний або шкідливий контент. Вони також можуть використовуватися для показу вам небажаної реклами;

– відмова в обслуговуванні (DDoS): зловмисники можуть використовувати велику кількість комп'ютерів, заражених шкідливим ПЗ, для

одночасного надсилання запитів на цільовий веб-сайт. Це може перевантажити веб-сайт і зробити його недоступним для законних користувачів [6].

Процес виявлення зловмисних посилань – це процес автоматизованого або ручного визначення посилань, які є зловмисними. Цей процес може ґрунтуватися на різних факторах, таких як:

– URL-адреса посилання: аналіз URL-адреси посилання може допомогти визначити, чи є воно зловмисним. Наприклад, URL-адреси, які містять підозрілі слова або фрази, або які ведуть до невідомих веб-сайтів, можуть бути зловмисними;

– контент веб-сайту: аналіз контенту веб-сайту, на який веде посилання, може допомогти визначити, чи є він зловмисним. Наприклад, веб-сайти, які містять шкідливе програмне забезпечення, фішингові форми або інші підозрілі елементи, можуть бути зловмисними;

– поведінка користувача: аналіз поведінки користувача при відвідуванні веб-сайту може допомогти визначити, чи є він зловмисним. Наприклад, якщо користувач швидко залишає веб-сайт або натискає на підозрілі посилання, це може бути ознакою того, що веб-сайт є зловмисним;

– репутація веб-сайту: репутація веб-сайту, на який веде посилання, може допомогти визначити, чи є він зловмисним. Наприклад, веб-сайти, які мають негативні відгуки або які були включені до чорних списків, можуть бути зловмисними;

– методи машинного навчання: методи машинного навчання можна використовувати для автоматичного аналізу великих обсягів даних і виявлення зловмисних посилань. Ці методи можуть враховувати різні фактори, такі як URL-адреса посилання, контент веб-сайту, поведінка користувача та інші, щоб визначити, чи є посилання зловмисним [6].

Існує два основних типи методів виявлення зловмисних посилань:

– статичні методи: ці методи ґрунтуються на аналізі URL-адреси посилання, контенту веб-сайту або інших статичних даних;

– динамічні методи: ці методи ґрунтуються на аналізі поведінки користувача або інших динамічних даних.

1.2 Аналіз переваг та недоліків існуючих методів та інструментів для ідентифікації зловмисних посилань та визначення обраного підходу дослідження

Існує багато методів ідентифікації зловмисних посилань, кожен з яких має свої переваги та недоліки. Важливо зазначити, що жоден метод не є досконалим. Тому рекомендується використовувати комбінацію методів для досягнення найкращих результатів.

1. Чорні списки URL – це списки веб-сайтів, які вважаються зловмисними або небезпечними. Ці списки використовуються різними програмами та службами, такими як антивірусне програмне забезпечення, веб-браузери та брандмауери, для блокування доступу користувачів до цих веб-сайтів [7].

Чорні списки URL зазвичай складаються зі списку URL-адрес або доменних імен веб-сайтів, які вважаються зловмисними. Ці списки можуть бути створені вручну експертами з безпеки або автоматично генеруватися за допомогою програмного забезпечення.

Коли користувач намагається відвідати веб-сайт, URL-адреса або доменне ім'я цього веб-сайту порівнюється зі списком чорних списків. Якщо URL-адреса або доменне ім'я знаходиться в списку, доступ до веб-сайту блокується.

2. Білі списки URL – це списки веб-сайтів, які вважаються безпечними та надійними. Ці списки використовуються різними програмами та службами, такими як веб-браузери, брандмауери та системи контролю контенту, щоб дозволити доступ лише до веб-сайтів, які включені до списку.

Білі списки URL зазвичай складаються зі списку URL-адрес або доменних імен веб-сайтів, які вважаються безпечними. Ці списки можуть бути створені вручну експертами з безпеки або автоматично генеруватися за допомогою програмного забезпечення.

Коли користувач намагається відвідати веб-сайт, URL-адреса або доменне ім'я цього веб-сайту порівнюється зі списком білих списків. Якщо URL-адреса або доменне ім'я знаходиться в списку, доступ до веб-сайту дозволяється. Якщо URL-адреса або доменне ім'я не знаходиться в списку, доступ до веб-сайту може бути заблокований або потребувати додаткової перевірки.

3. Аналіз контенту веб-сторінок – це метод виявлення зловмисних посилань, який полягає у вивченні тексту, коду та інших елементів веб-сторінки, щоб визначити, чи є вона шкідливою [7]. Цей метод може використовуватися для виявлення різних типів зловмисних посилань, включаючи:

- посилання на шкідливі веб-сайти: ці посилання можуть вести на веб-сайти, які містять шкідливе програмне забезпечення, фішингові атаки або інші онлайн-загрози;
- посилання на неякісний контент: ці посилання можуть вести на веб-сайти, які містять спам, дезінформацію або інший неякісний контент;
- посилання на веб-сайти, які не відповідають темі: ці посилання можуть вести на веб-сайти, які не мають нічого спільного з темою веб-сторінки, на якій вони розміщені.

Аналіз контенту веб-сторінок зазвичай використовує методи машинного навчання для аналізу тексту, коду та інших елементів веб-сторінки. Ці методи можуть вивчати різні фактори, такі як:

- ключові слова: слова та фрази, які використовуються на веб-сторінці;
- посилання: посилання, які ведуть на інші веб-сайти;
- код: код, який використовується для створення веб-сторінки;
- структура: структура веб-сторінки, включаючи заголовки, зображення та інші елементи.

На основі цих факторів методи машинного навчання можуть визначити, чи ймовірно, що веб-сторінка є шкідливою.

4. Аналіз поведінки користувачів – це метод виявлення зловмисних посилань, який полягає у відстеженні дій користувачів на веб-сайтах, щоб визначити, чи

демонструють вони підозрілу або шкідливу поведінку [7]. Цей метод може використовуватися для виявлення різних типів зловмисних посилань, включаючи:

– посилання на шкідливі веб-сайти: користувачі, які переходять за шкідливими посиланнями, можуть демонструвати підозрілу поведінку, наприклад, швидко клацати на посилання, завантажувати файли або вводити особисті дані;

– посилання на неякісний контент: користувачі, які переходять за посиланнями на неякісний контент, можуть швидко покидати веб-сайти або повертатися до пошукової системи;

– посилання на веб-сайти, які не відповідають темі: користувачі, які переходять за посиланнями на веб-сайти, які не відповідають темі, можуть витрачати мало часу на цих веб-сайтах або не взаємодіяти з їхнім контентом.

Аналіз поведінки користувачів зазвичай використовує різні методи для відстеження дій користувачів на веб-сайтах [8]. Ці методи можуть включати:

– файли cookie – це невеликі текстові файли, які зберігаються на комп'ютері користувача, коли він відвідує веб-сайт [8]. Файли cookie можуть використовуватися для відстеження того, які сторінки відвідує користувач, скільки часу він проводить на кожній сторінці та які дії він виконує;

– скрипти – це невеликі програми, які можуть виконуватися на веб-сайтах. Скрипти можуть використовуватися для відстеження рухів миші користувача, натискань клавіш та інших дій;

– інструменти аналітики веб-сайтів, такі як Google Analytics, можуть використовуватися для збору даних про поведінку користувачів на веб-сайтах. Ці дані можуть включати те, звідки приходять користувачі, які сторінки вони відвідують та скільки часу вони проводять на веб-сайті [8].

5. Аналіз URL-адрес – це метод виявлення зловмисних посилань, який полягає в аналізі структури та компонентів URL-адреси, щоб визначити, чи є вона шкідливою. Аналіз URL-адрес зазвичай використовує правила та алгоритми для аналізу різних компонентів URL-адреси, таких як:

– домен: ім'я веб-сайту, на який веде посилання;

– субдомен: частина домену, яка розташована перед основним ім'ям домену;

– шлях: частина URL-адреси, яка вказує на розташування конкретної сторінки або ресурсу на веб-сайті;

– параметри: додаткова інформація, яка передається веб-сайту за допомогою URL-адреси.

На основі цих компонентів правила та алгоритми можуть визначити, чи є URL-адреса шкідливою.

6. Методи машинного навчання стають все більш потужними інструментами для виявлення зловмисних посилань в Інтернеті [9]. Ці методи можуть аналізувати великі обсяги даних, виявляти складні закономірності та робити точні прогнози, що робить їх ідеально придатними для цієї задачі.

Ось деякі з поширених методів машинного навчання, які використовуються для виявлення зловмисних посилань:

– класифікація: цей метод використовується для навчання моделі методу машинного навчання розрізняти шкідливі та безпечні посилання. Модель навчається на наборі даних, який містить приклади обох типів посилань, а потім використовується для класифікації нових посилань;

– аномальне виявлення: цей метод використовується для виявлення посилань, які відрізняються від нормальних. Модель машинного навчання навчається на наборі даних нормальних посилань, а потім використовується для виявлення посилань, які відхиляються від цієї норми;

– видобуток тексту: цей метод використовується для виявлення шкідливих посилань на основі тексту, який їх оточує. Модель машинного навчання навчається на наборі даних тексту та посилань, а потім використовується для виявлення тексту, який свідчить про те, що посилання є шкідливим;

– аналіз графів: цей метод використовується для виявлення шкідливих посилань на основі їх зв'язків з іншими посиланнями в Інтернеті. Модель машинного навчання будує граф, де вузли представляють посилання, а ребра представляють зв'язки між ними, а потім використовується для виявлення посилань, які є частиною шкідливої підмережі.

Переваги та недоліки методів наведено в табл. 1.1.

Таблиця 1.1 – Порівняння методів виявлення зловмисних посилань

Метод	Переваги	Недоліки
Чорні списки URL	Прості, ефективні для виявлення відомих зловмисних посилань, не потребують значних обчислювальних ресурсів	Не можуть виявити нові зловмисні посилання; призводять до блокування легітимних веб-сайтів
Білі списки URL	Гарантують, що користувачі відвідують лише безпечні веб-сайти	Не спроможні виявити зловмисні веб-сайти, які не включені до списку відповідно є незручними для користувачів
Аналіз контенту веб-сторінок	Здатний побачити зловмисні веб-сайти, які не можна виявити за допомогою чорних або білих списків; враховує контекст веб-сторінки	Є обчислювально складним та призводить до помилок
Аналіз поведінки користувачів	Здібний до виявлення зловмисних веб-сайтів, також враховує поведінку користувача на веб-сайті	Надокучливий для користувачів, призводить до помилок
Аналіз URL-адрес	Розкриває зловмисні посилання, які не включені до чорних або білих списків; не потребує значних обчислювальних ресурсів	Точність є недостатньою, призводить до блокування легітимних веб-сайтів
Методи машинного навчання	Адаптування до нових видів атак, також є більш точними, ніж інші методи	Потенційно складний для реалізації, потребують великих обсягів даних для навчання

Крім вищезазначених методів, існують й інші, менш поширені методи, такі як:

– аналіз DNS: цей метод використовується для аналізу запитів DNS, щоб виявити зловмисні веб-сайти;

– аналіз трафіку: цей метод використовується для аналізу трафіку мережі, щоб виявити зловмисні посилання;

– біометрична аутентифікація: цей метод використовується для аутентифікації користувачів на основі їхніх біометричних даних, щоб запобігти доступу до зловмисних веб-сайтів.

Вибір методу виявлення зловмисних посилань також залежить від конкретних потреб та вимог організації.

Отже, для подальшого дослідження з урахуванням різних критеріїв вибору, а саме функціональності, точності методів, простоті їх реалізації, спроможності до навчання автор бакалаврської роботи пропонує застосовувати надалі саме методи чорних та білих списків у поєднанні із методом машинного навчання.

У бакалаврській роботі пропонується враховувати п'ять інструментів веб-безпеки, що є популярними у сфері оцінки безпеки веб-сайтів. Кожен з них відіграє важливу роль у виявленні та уникненні потенційно шкідливих веб-ресурсів, а саме:

1. VirusTotal – це онлайн-сервіс, який дозволяє аналізувати URL-адреси та файлові хеші за допомогою декількох антивірусних движків, щоб виявити потенційно шкідливі програми та веб-сайти [10].

2. Web of Trust (WOT): WOT – це розширення для веб-переглядачів, яке надає рейтинги безпеки веб-сайтів на основі відгуків користувачів та інших джерел даних [11].

3. URLVoid цей сервіс дозволяє аналізувати URL-адреси, щоб визначити їхню шкідливість за допомогою різних джерел та критеріїв [12].

4. McAfee SiteAdvisor – це додаток, який надає рейтинги безпеки для веб-сайтів, щоб користувачі могли уникнути потенційних загроз [13].

5. Norton Safe Web – це інструмент, що надає інформацію про безпеку веб-сайтів на основі власної бази даних загроз та відгуків користувачів [14].

Розглянемо методи ідентифікації зловмисних посилань, які використовують різні інструменти веб-безпеки:

VirusTotal застосовує такі:

– сканування антивірусними двигунами: VirusTotal сканує URL-адреси, файли та хеші за допомогою десятків антивірусних двигунів, щоб визначити, чи були вони раніше позначені як зловмисні [10];

– аналіз URL-адрес: VirusTotal аналізує структуру URL-адреси, щоб виявити підозрілі характеристики, такі як використання незвичайних доменів, субдоменів або шляхів;

– аналіз контенту веб-сторінок: VirusTotal може аналізувати контент веб-сторінок, щоб виявити зловмисний код, скрипти або інші загрози;

– аналіз поведінки: VirusTotal може відстежувати поведінку веб-сайту, щоб виявити підозрілу активність, таку як перенаправлення на інші веб-сайти або спроби завантаження зловмисного програмного забезпечення;

– звернення до спільноти: VirusTotal використовує дані, надані користувачами, щоб позначити веб-сайти та файли як зловмисні або безпечні.

Web of Trust використовує такі засоби:

– аналіз відгуків користувачів: Web of Trust збирає відгуки користувачів про веб-сайти та посилання, щоб оцінити їхню надійність;

– аналіз соціальних мереж: Web of Trust аналізує дані з соціальних мереж, щоб виявити підозрілу активність, пов'язану з веб-сайтами або посиланнями;

– аналіз контенту веб-сторінок: Web of Trust може аналізувати контент веб-сторінок, щоб виявити зловмисний код, скрипти або інші загрози;

– аналіз поведінки: Web of Trust може відстежувати поведінку веб-сайту, щоб виявити підозрілу активність, таку як перенаправлення на інші веб-сайти або спроби завантаження зловмисного програмного забезпечення [11].

URLVoid використовує такі:

– аналіз URL-адрес: URLVoid аналізує структуру URL-адреси, щоб виявити підозрілі характеристики, такі як використання незвичайних доменів, субдоменів або шляхів [12];

– перевірка чорних списків: URLVoid перевіряє URL-адресу проти списків відомих зловмисних веб-сайтів;

– аналіз IP-адрес: URLVoid може аналізувати IP-адресу, з якою пов'язаний веб-сайт, щоб виявити підозрілу активність;

– звернення до сторонніх ресурсів: URLVoid може використовувати дані з інших джерел, таких як VirusTotal, щоб отримати додаткову інформацію про веб-сайт.

McAfee SiteAdvisor використовує такі:

– аналіз URL-адрес: McAfee SiteAdvisor аналізує структуру URL-адреси, щоб виявити підозрілі характеристики, такі як використання незвичайних доменів, субдоменів або шляхів [13];

– перевірка чорних списків: McAfee SiteAdvisor перевіряє URL-адресу проти списків відомих зловмисних веб-сайтів;

– аналіз контенту веб-сторінок: McAfee SiteAdvisor може аналізувати контент веб-сторінок, щоб виявити зловмисний код, скрипти або інші загрози;

– аналіз поведінки: McAfee SiteAdvisor може відстежувати поведінку веб-сайту, щоб виявити підозрілу активність, таку як перенаправлення на інші веб-сайти або спроби завантаження зловмисного програмного забезпечення;

– звернення до хмарного сервісу McAfee: McAfee SiteAdvisor може використовувати дані з хмарного сервісу McAfee, щоб отримати додаткову інформацію про веб-сайт.

Norton Safe Web застосовує такі:

– аналіз URL-адрес: Norton Safe Web аналізує структуру URL-адреси, щоб виявити підозрілі характеристики, такі як використання незвичайних доменів, субдоменів або шляхів;

– перевірка чорних списків: Norton Safe Web перевіряє URL-адресу проти списків відомих зловмисних веб-сайтів, які підтримуються Norton;

– аналіз контенту веб-сторінок: Norton Safe Web може аналізувати контент веб-сторінок, щоб виявити зловмисний код, скрипти або інші загрози;

– аналіз поведінки: Norton Safe Web може відстежувати поведінку веб-сайту, щоб виявити підозрілу активність, таку як перенаправлення на інші веб-сайти або спроби завантаження зловмисного програмного забезпечення;

– звернення до хмарного сервісу Norton: Norton Safe Web може використовувати дані з хмарного сервісу Norton, щоб отримати репутацію веб-сайту на основі агрегованих даних від користувачів Norton та інших джерел [14];

– машинне навчання: Norton Safe Web використовує машинне навчання для аналізу великих обсягів даних, щоб виявити складні закономірності, які можуть допомогти ідентифікувати нові та невідомі зловмисні веб-сайти.

Таблиця 1.2 – Порівняння деяких популярних веб-сайтів для ідентифікації зловмисних посилань

Веб-сайт	Переваги	Недоліки
VirusTotal	Простий у використанні; безкоштовний; підтримує багато форматів файлів; сканує за допомогою десятків антивірусних двигунів	Інколи дає помилкові спрацьовування; не завжди виявляє нові загрози
Web of Trust	Співпрацює з користувачами, щоб оцінити веб-сайти та посилання; пропонує розширення для браузера	Упередженість до думок більшості, не завжди точний
URLVoid	Безкоштовний; швидкий; простий інтерфейс; аналізує URL-адреси, домени та IP-адреси	Не такий детальний, як інші інструменти; не завжди виявляє нові загрози
McAfee SiteAdvisor	Безкоштовний, пропонує розширення для браузера, оцінює веб-сайти та посилання в режимі реального часу	Надокучливий; не завжди точний
Norton Safe Web	Безкоштовний, пропонує розширення для браузера, оцінює веб-сайти та посилання в режимі реального часу	Нав’язливий; не завжди точний

Отже, найкращим серед розглянутих інструментів для подальшого дослідження варто обрати саме VirusTotal як простий у використанні, безкоштовний, такий, що підтримує багато форматів файлів та сканує за допомогою десятків антивірусних двигунів.

Онлайн-ресурси об’єднані спільними перевагами такими як швидкість, простий інтерфейс та головне безкоштовність для користувачів, щоб із легкістю перевіряти сайти або файли для забезпечення безпеки персональних даних.

1.3 Обґрунтування вибору методу машинного навчання для виявлення зловмисних посилань

Використання методу К-найближчих сусідів для виявлення зловмисних посилань є перспективним методом, який демонструє високу ефективність та здатність працювати з великими обсягами даних. Завдяки своїм перевагам, таким як простота реалізації, гнучкість та здатність до навчання, KNN може стати цінним інструментом для забезпечення безпеки онлайн ресурсів.

Метод К-найближчих сусідів є непараметричним методом керованого навчання. Він використовується як для класифікації, так і для регресії. Основна ідея полягає в тому, що для вхідних даних обчислюється відстань до k найближчих навчальних прикладів у наборі даних [15].

При класифікації KNN визначає належність об’єкта до певного класу. Об’єкт класифікується за допомогою голосів його сусідів, при цьому він належить до класу, який є найпоширенішим серед його k найближчих сусідів. Якщо k = 1, то об’єкт просто приписується до класу єдиного найближчого сусіда.

При KNN регресії результатом є числове значення властивості об’єкта. Це значення обчислюється як середнє із значень k найближчих сусідів.

Оскільки KNN апроксимує функцію локально, важливо враховувати масштаби ознак. Нормалізація навчальних даних може підвищити точність алгоритму.

Для кращої точності можна враховувати ваговий внесок сусідів, де вага залежить від відстані до сусіда. Сусіди беруться з навчального набору, де відомий клас (для класифікації) або значення властивості (для регресії). Навчання не потрібно, оскільки алгоритм використовує лише відомі дані.

У дослідженні розглянуто метод KNN у контексті класифікації об'єктів з векторами багатовимірних ознак. Цей метод базується на навчанні з урахуванням набору векторів ознак та відповідних міток класів навчальних прикладів.

На етапі класифікації використовується константа k, визначена користувачем, для призначення мітки неналежного вектора, або тестової точки. Ця мітка визначається шляхом ідентифікації найбільш поширеної серед k навчальних прикладів, що найближчі до тестової точки.

Для вимірювання відстані між об'єктами застосовується різні метрики залежно від типу змінних. Наприклад, для неперервних змінних застосовується евклідова відстань, в той час як для дискретних, таких як текст, може використовуватися метрика перекриття або відстань Геммінга.

Проблема нерівномірного розподілу класів може виникнути при застосуванні методу "голосування більшості". Це може призвести до перекосу в сторону більш представлених класів у передбаченні нових прикладів. Для подолання цієї проблеми можуть використовуватися методи зважування класифікації або абстрагування представлення даних, наприклад, за допомогою самоорганізаційних карт Кохонена [15].

Вибір оптимального значення параметру k у методі k-найближчих сусідів залежить від конкретних даних. Зазвичай більші значення k сприяють зменшенню впливу шуму на класифікацію, але можуть призвести до змазування меж між класами. Оптимальне значення k можна вибрати за допомогою різних евристичних методів, таких як оптимізація гіперпараметрів.

Випадок, коли k = 1, відомий як метод найближчого сусіда, представляє окремий випадок методу KNN.

Точність методу KNN може значно знизитися через наявність шуму в даних, невідповідність ознак або різницю в масштабах ознак. Багато досліджень

спрямовані на вибір та масштабування ознак для покращення класифікації. Зокрема, еволюційні алгоритми часто використовуються для оптимізації масштабування ознак. Інший підхід полягає у масштабуванні ознак з використанням взаємної інформації між ознаками та класами навчальних даних.

У випадку бінарної класифікації корисно обирати непарне значення k, щоб уникнути ситуації, коли голоси для кожного класу рівні. Це можна досягти за допомогою методу бутстрепінгу для визначення емпірично оптимального значення параметру k.

Класифікатор k найближчих сусідів можна розглядати як систему, де кожному з k найближчих сусідів призначається вага 1/k, а всім іншим – нульова вага. Це підходить для не зваженої моделі. Але зважені класифікатори найближчих сусідів розглядаються як система, де кожному із k найближчих сусідів призначається вага w_{ni}, з умовою, що сума всіх ваг дорівнює 1. При цьому застосовується узагальнення, яке враховує ваги [15].

Тут C_n^{wnn} позначає зважений класифікатор k найближчих сусідів з вагами $\{w_{ni}\}_{i=1}^{n}$. Якщо передбачити, що розподіл класів регулярний, то надмірний ризик має наступне асимптотичне розширення (формула 1.1).

$$R_R(C_n^{wnn}) - R_R(C^{Bayes}) = (B_1 s_n^2 + B_2 t_n^2)\{1 + o(1)\} \quad (1.1)$$

Існує оптимальна схема зважування $\{w_{ni}^*\}_{i=1}^{n}$яка збалансовує два зазначені вище члени. Кількість k^* обчислюється за формулою $k^* = \left\lfloor Bn^{\frac{4}{d+4}} \right\rfloor$і вага кожного з k* найближчих сусідів визначається як (формула 1.2).

$$w_{ni}^* = \frac{1}{k^*}\left[1 + \frac{d}{2} - \frac{d}{2k^{*\frac{2}{d}}}\left\{i^{1+\frac{2}{d}} - (i-1)^{1+\frac{2}{d}}\right\}\right] \text{ для } i = 1,2,\dots,k^*$$

$$\text{і } w_{ni}^* = 0, \text{для } i = k^* + 1, \dots, n \quad (1.2)$$

При таких оптимальних вагах домінуючим членом у асимптотичному розкладанні надлишкового ризику є $O(n^{-\frac{4}{d+4}})$. Подібні результати відповідні й при застосуванні класифікатора найближчого сусіда для агрегації [16].

Приклад KNN класифікації (рис. 1.1). Тестовий зразок (зелене коло) повинен бути класифікований як синій квадрат або як червоний трикутник. Якщо k = 3, то

класифікується як червоний трикутник, тому що всередині меншого кола 2 трикутника і тільки 1 квадрат. Якщо k = 5, то він буде класифікований як синій квадрат (3 квадрата проти 2-ох трикутників всередині більшого кола).

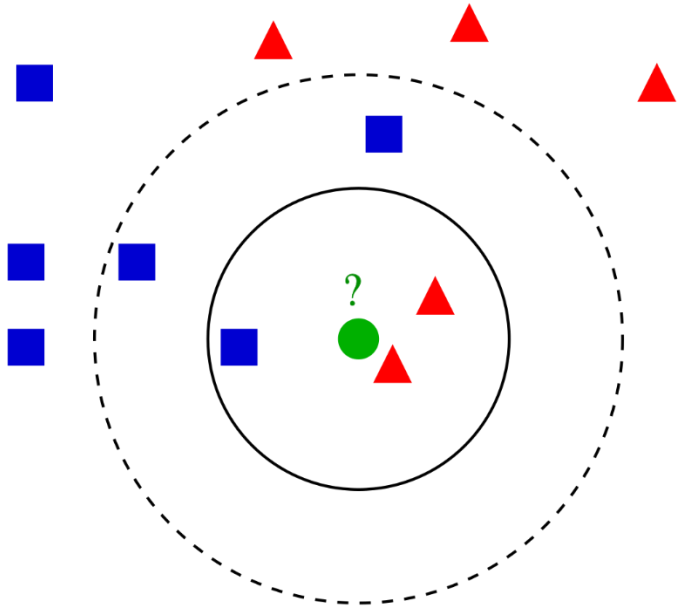


Рисунок 1.1 – Приклад KNN класифікації

Метод K-найближчих сусідів є простим, але потужним методом машинного навчання, який може бути ефективно використаний для виявлення зловмисних посилань. Ось деякі з ключових причин, чому KNN є придатним вибором для цієї задачі:

Простота: KNN є простим для розуміння та реалізації методом, що робить його доступним для широкого кола користувачів, навіть без глибоких знань машинного навчання.

Ефективність: KNN є обчислювально ефективним методом, що робить його придатним для обробки великих наборів даних, які часто використовуються для виявлення зловмисних посилань.

Гнучкість: KNN може бути налаштований за допомогою різних параметрів, таких як кількість сусідів (K) та міра відстані, що дозволяє адаптувати його до конкретних потреб та характеристик даних.

Інтерпретованість: KNN є методом, який легко інтерпретувати, що означає, що можна зрозуміти, чому певні посилання класифіковані як зловмисні. Це може бути корисно для аналізу результатів та вдосконалення моделі.

Ефективність у виявленні невідомих зловмисних посилань: KNN добре справляється з виявленням нових, невідомих зловмисних посилань, які не були явно включені до набору даних для навчання. Це важлива характеристика для виявлення нових кіберзагроз.

Додаткові переваги: KNN не потребує попередньої обробки даних, може працювати з даними, що мають змішані типи (категоріальні та чисельні), і може бути використаний для інших завдань машинного навчання, окрім виявлення зловмисних посилань.

Приклади використання KNN для виявлення зловмисних посилань:

– класифікація URL-адрес: KNN можна використовувати для класифікації URL-адрес як зловмисних або безпечних на основі їх характеристик, таких як домен, шлях, параметри та текст;

– виявлення аномальних посилань: KNN можна використовувати для виявлення посилань, які відрізняються від нормальних посилань у наборі даних, що може свідчити про те, що вони є зловмисними;

– аналіз поведінки користувачів**:** KNN можна використовувати для аналізу поведінки користувачів на веб-сайті та виявлення користувачів, які демонструють підозрілу поведінку, що може свідчити про те, що вони намагаються перейти за зловмисними посиланнями [16].

Важливо зазначити, що KNN має й певні обмеження:

– чутливість до шуму: KNN може бути чутливим до шуму в даних, що може призвести до помилок класифікації;

– вибір параметрів: вибір правильних значень для параметрів KNN, таких як K та міра відстані, може бути складним завданням і може суттєво вплинути на результати;

– проблема розмірності: KNN може бути неефективним при роботі з даними високої розмірності, що може потребувати використання методів зменшення розмірності.

Загалом, KNN є потужним та гнучким методом машинного навчання, який може бути ефективно використаний для виявлення зловмисних посилань. Його простота, ефективність та інтерпретованість роблять його придатним вибором для широкого кола користувачів.

Однак, важливо враховувати обмеження KNN та ретельно налаштовувати його параметри для досягнення найкращих результатів.

1.4 Висновки до першого розділу

У цьому розділі було проаналізовано існуючі методи виявлення зловмисних посилань на онлайн ресурси та обґрунтовано на базі цього оптимальний підхід до розв’язку поставленої задачі. Серед недоліків наведених методів варто зазначити такі, як: обчислювально складні, надокучливість, точність не є достатньою призводить до блокування легітимних веб-сайтів.

Було обґрунтовано методи чорних та білих списків, машинного навчання як такі, що є:

– простими у реалізації та використанні;

– ефективними для блокування відомих зловмисних посиланнь;

– точними у ідентифікації, навіть нових або невідомих посилань;

– спроможнимми навчатися та вдосконалюватися з часом, коли їм надається більше даних.

Для подальшого дослідження з урахуванням різних критеріїв вибору, а саме функціональності, точності методів, простоті їх реалізації, спроможності до навчання автор бакалаврської роботи пропонує застосовувати надалі саме методи чорних та білих списків у поєднанні із методом машинного навчання.

У результатів вивчення наявних інструментів для ідентифікації зловмисних посилань було доведено доцільність використання саме VirusTotal як простий у використанні, безкоштовний, такий, що підтримує багато форматів файлів та сканує за допомогою десятків антивірусних двигунів.

Було встановлено, що існуючі антивірусні програми та методи аналізу підозрілої поведінки програмного забезпечення мають свої обмеження, особливо у виявленні нових, раніше невідомих загроз.

Використання методу K-найближчих сусідів для виявлення зловмисних посилань є перспективним підходом, який демонструє високу ефективність та здатність працювати з великими обсягами даних. Завдяки своїм перевагам, таким як простота реалізації, гнучкість та здатність до навчання, KNN може стати цінним інструментом для забезпечення безпеки онлайн-ресурсів.

На основі цього аналізу було обґрунтовано вибір методу K-найближчих сусідів для виявлення зловмисних посилань. Цей метод відзначається високою ефективністю та здатністю працювати з великими обсягами даних.

2 РОЗРОБЛЕННЯ МЕТОДУ ДО ВИЯВЛЕННЯ ЗЛОВМИСНИХ ПОСИЛАНЬ ЗАСОБАМИ МЕТОДІВ ЧОРНИХ І БІЛИХ СПИСКІВ ТА К-НАЙБЛИЖЧИХ СУСІДІВ

У цьому розділі описано розроблення підходу до виявлення зловмисних посилань методами К-найближчих сусідів та чорних і білих списків.

Визначено цілі та вимоги до процесу виявлення зловмисних посилань.

Здійснено вибір мови програмування та середовища розробки.

2.1 Розроблення методу до виявлення зловмисних посилань на онлайн ресурсах

Сформуємо підхід до виявлення зловмисних посилань на основі методу KNN та чорних і білих списків.

На першому етапі застосовуємо метод KNN для класифікації URL-адрес шляхом пошуку «найближчого сусіда». Для цього необхідно спочатку навчити модель на наборі даних зловмисних та безпечних посилань, які беруться з білих та чорних списків.

На другому етапі реалізовуємо навчену модель для класифікації URL-адрес шляхом порівняння зі вмістом адрес із білих та чорних списків. У разі збігу посилання, відбувається його відповідна класифікація за двома класами (1-зловмисне посилання; 0 – не зловмисне посилання).

На третьому етапі здійснюється оцінювання точності та перевірка ефективності запропонованого методу.

На четвертому етапі необхідно інтегрувати до веб-браузерів, веб-сайтів або інших онлайн-платформ.

Представимо візуально кожен із описаних етапів (рис. 2.1).

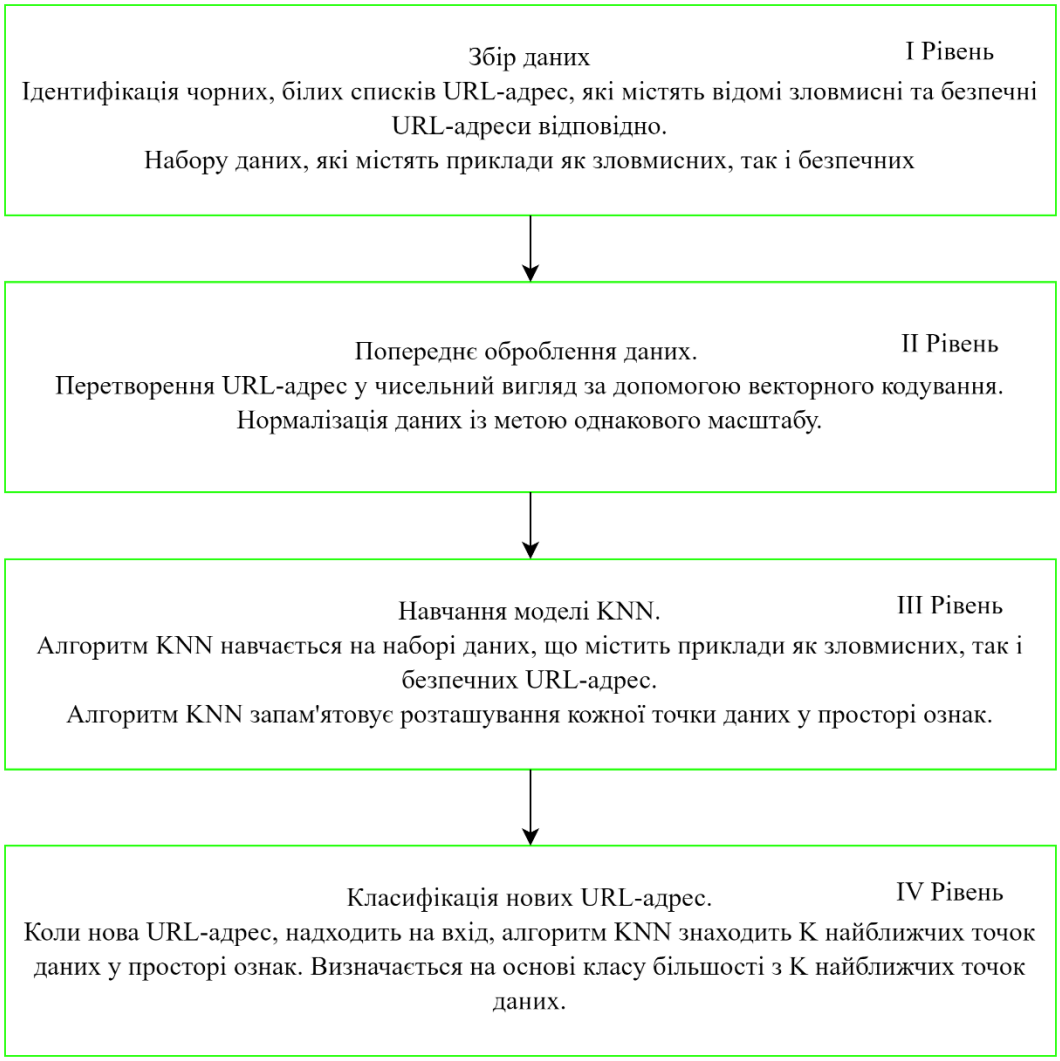


Рисунок 2.1 – Структура моделі виявлення зловмисних посилань з використанням методу KNN, чорних та білих списків

Етап збирання даних є важливою частиною методу виявлення зловмисних посилань, оскільки він відповідає за збирання інформації, необхідної для класифікації URL-адрес як шкідливих або легітимних.

Існує багато різних джерел даних, які можна використовувати для збирання інформації про URL-адреси. Деякі з найпоширеніших джерел включають:

– чорні списки URL-адрес: це списки веб-сайтів, які відомі як шкідливі або небезпечні. Чорні списки URL-адрес часто складаються компаніями з кібербезпеки та урядовими органами;

– результати пошукових систем: можна використовувати результати пошукових систем для виявлення веб-сайтів, які пов'язані з шкідливою діяльністю.

Це можна зробити, шукаючи ключові слова, пов'язані з шкідливим програмним забезпеченням, фішингом або іншими кіберзагрозами;

– звіти про кіберзагрози: державні органи та компанії з кібербезпеки часто публікують звіти про кіберзагрози, які містять інформацію про нові шкідливі веб-сайти та методи атак;

– веб-сайти соціальних мереж: шкідливі посилання часто поширюються через веб-сайти соціальних мереж. Можна використовувати API веб-сайтів соціальних мереж для збору даних про посилання, які публікуються та поширюються користувачами.

Етап збирання даних залежить від джерел даних. Наприклад, чорні списки URL-адрес і звіти про кіберзагрози зазвичай доступні у форматі CSV або JSON, який можна легко завантажити та обробити. З іншого боку, для збору даних із результатів пошукових систем або веб-сайтів соціальних мереж може знадобитися використовувати веб-скрапінг або API [16].

Наступним етапом є оброблення даних оскільки він відповідає за підготовку даних для використання методом KNN, зокрема він полягає в очищенні даних, перетворення даних та масштабуванні даних.

Очищення даних – це процес видалення неповних, дублюючих або помилкових даних. Це важливо, оскільки неякісні дані можуть призвести до неточних результатів класифікації. Деякі поширені методи очищення даних включають:

– видалення неповних даних: видалення записів, які мають відсутні значення;

– видалення дублюючих даних: видалення записів, які є ідентичними іншим записам у наборі даних;

– виявлення та виправлення помилок: виявлення та виправлення помилок, таких як орфографічні помилки та помилки введення даних.

Перетворення даних – це процес перетворення категоріальних даних у числовий формат, який може бути оброблений методом KNN. Категоріальні дані –

це дані, які складаються з дискретних категорій, наприклад, "шкідливий" або "легітимний" [17]. Деякі поширені методи перетворення даних включають:

– кодування гарячого однієї категорії: цей метод перетворює кожну категорію в новий бінарний атрибут. Наприклад, категорія "шкідливий" може бути перетворена в бінарний атрибут "шкідливий = 1", а категорія "легітимний" може бути перетворена в бінарний атрибут "шкідливий = 0";

– масштабування даних: цей метод перетворює значення даних до спільного діапазону. Це важливо, оскільки метод KNN може бути чутливим до масштабу даних. Деякі поширені методи масштабування даних включають стандартизацію та нормалізацію min-max.

Масштабування даних – це процес нормалізації значень даних до спільного діапазону. Це важливо, оскільки метод KNN може бути чутливим до масштабу даних. Деякі поширені методи масштабування даних включають:

– стандартизація: цей метод перетворює значення даних так, щоб їх середнє значення було дорівнює 0, а стандартне відхилення – 1;

– нормалізація min-max: цей метод перетворює значення даних так, щоб вони лежали в діапазоні від 0 до 1.

Також не менш важливим є етап класифікації є основним компонентом методу виявлення зловмисних посилань, оскільки він відповідає за класифікацію URL-адрес як шкідливих або легітимних. У цьому модулі використовується метод K-Nearest Neighbors (KNN) для класифікації URL-адрес на основі їх схожості з відомими шкідливими або легітимними URL-адресами.

У контексті виявлення зловмисних посилань метод KNN використовується для класифікації нової URL-адреси на основі її схожості з відомими шкідливими або легітимними URL-адресами. Відстань між URL-адресами можна обчислити за допомогою різних методів, наприклад, за допомогою метрики Левенштейна або метрики Хаммінга [17].

Реалізація оптимізованого алгоритму пошуку відстані Левенштейна мовою Python 3:

```
def levenstein(s1,s2):
```

```
    n = range(0,len(s1)+1)
    for y in range(1,len(s2)+1):
        l,n = n,[y]
        for x in range(1,len(s1)+1):
            n.append(min(l[x]+1,n[-1]+1,l[x-1]+((s2[y-1]!=s1[x-1]) and 1 or 0)))
    return n[-1]
```

Реалізація оптимізованого алгоритму пошуку відстані Хеммінга мовою Python 3:

```
def hamming_distance(string1, string2):
    if (len(string1) != len(string2)):
        raise Exception('Strings must be of equal length.')
    dist_counter = 0
    for n in range(len(string1)):
        if string1[n] != string2[n]:
            dist_counter += 1
    return dist_counter
```

Для навчання моделі KNN необхідний набір даних, який містить URL-адреси та їх відповідні класи (шкідливий або легітимний). Набір даних можна створити, використовуючи різні джерела даних, такі як чорні списки URL-адрес, звіти про кіберзагрози та результати пошукових систем [17].

Модель KNN має X1…Xn кількість зловмисних посилань та Y1…Yn кількість безпечних посилань, навчання використовує ці дані класифікує їх як «1 - зловмисне посилання», «0 - безпечне посилання», вводячи нову URL-адресу метод визначає до якого списку посилань її віднести та виводить результат R який показує інформацію про небезпеку чи безпеку даної адреси (рис. 2.2).

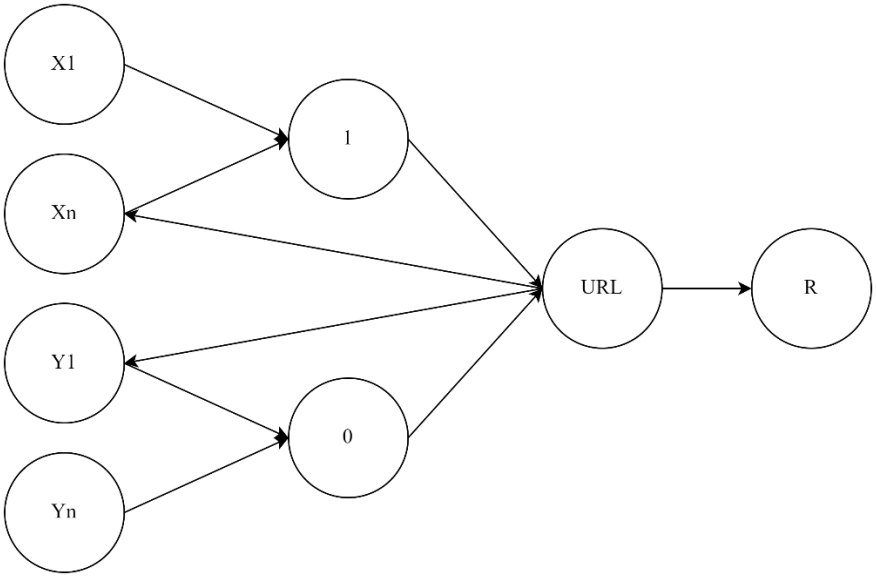


Рисунок 2.2 – Архітектура навчання моделі KNN

За основу для навчання моделі KNN було взято 100 зловмисних та 100 безпечних сайтів, які отримані з JSON та CSV файлів відповідно (рис. 2.3, 2.4).

Після створення набору даних його можна використовувати для навчання моделі KNN. Це включає вибір значення K та обчислення відстаней між усіма точками даних у наборі навчання.


```
{
    "2849997": [
        {
            "dateadded": "2024-05-14 14:04:25 UTC",
            "url": "http://117.204.206.144:39375/Mozi.m",
            "url_status": "online",
            "last_online": "2024-05-14 14:04:25 UTC",
            "threat": "malware_download",
            "tags": [
                "elf",
                "Mozi"
            ],
            "urlhaus_link": "https://urlhaus.abuse.ch/url/2849997/",
            "reporter": "lrz_urlhaus"
        }
    ],
    "2849996": [
        {
            "dateadded": "2024-05-14 14:04:10 UTC",
            "url": "http://219.157.166.134:60575/Mozi.m",
            "url_status": "online",
            "last_online": "2024-05-14 14:04:10 UTC",
            "threat": "malware_download",
            "tags": [
                "elf",
                "Mozi"
            ],
            "urlhaus_link": "https://urlhaus.abuse.ch/url/2849996/",
            "reporter": "lrz_urlhaus"
        }
    ],
```


Рисунок 2.3 – Приклад зловмисних посилань які використовувались для навчання моделі KNN

1,1,google.com,com,474186,2160824,google.com,com,1,1,473836,2158231									
2,2,facebook.com,com,466535,2249894,facebook.com,com,2,2,466120,2246772									
3,3,youtube.com,com,423073,1859246,youtube.com,com,3,3,422679,1856031									
4,4,twitter.com,com,404966,1746758,twitter.com,com,4,4,404641,1744509									
5,5,instagram.com,com,366589,1604619,instagram.com,com,5,5,366111,1602288									
6,6,linkedin.com,com,340119,1326114,linkedin.com,com,6,6,339764,1324264									
7,7,apple.com,com,264801,805961,apple.com,com,7,7,264477,804356									
8,8,microsoft.com,com,257643,712094,microsoft.com,com,8,8,257428,711006									
9,9,googletagmanager.com,com,256060,871434,googletagmanager.com,com,9,9,255628,870073									
10,1,wikipedia.org,org,247555,829132,wikipedia.org,org,10,1,247253,827630									
11,2,wordpress.org,org,230402,828836,wordpress.org,org,11,2,230295,828343									
12,1,youtu.be,be,221435,700003,youtu.be,be,12,1,221099,698109									
13,10,github.com,com,205981,497566,github.com,com,13,10,205828,497162									
14,3,en.wikipedia.org,org,204878,613513,en.wikipedia.org,org,14,3,204599,612522									
15,11,play.google.com,com,201621,545170,play.google.com,com,15,11,201308,544003									
16,12,pinterest.com,com,197281,720972,pinterest.com,com,16,12,197095,720062									
17,1,goo.gl,gl,192304,618567,goo.gl,gl,17,1,191953,617591									
18,13,maps.google.com,com,189705,637655,maps.google.com,com,18,13,189494,637092									
19,14,vimeo.com,com,186372,602806,vimeo.com,com,19,14,185976,601578									
20,15,docs.google.com,com,172671,483208,docs.google.com,com,20,15,172240,482143									

Рисунок 2.4 – Приклад безпечних посилань які використовувались для навчання моделі KNN

Після того, як модель KNN навчена, її можна використовувати для класифікації нових URL-адрес. Для цього необхідно обчислити відстань між новою URL-адресою та кожною URL-адресою в наборі навчання, а потім визначити клас нової URL-адреси на основі K найближчих сусідів.

Набір даних (рис. 2.4) та карта класифікацій з K = 5 (рис. 2.5), яка відображає розуміння роботи KNN у просторі.

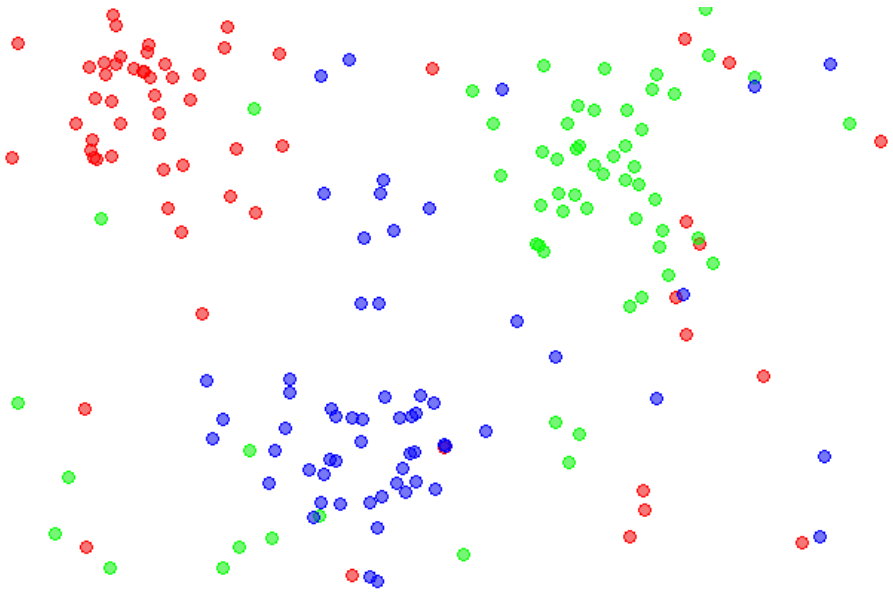


Рисунок 2.4 – Приклад набору даних у просторі

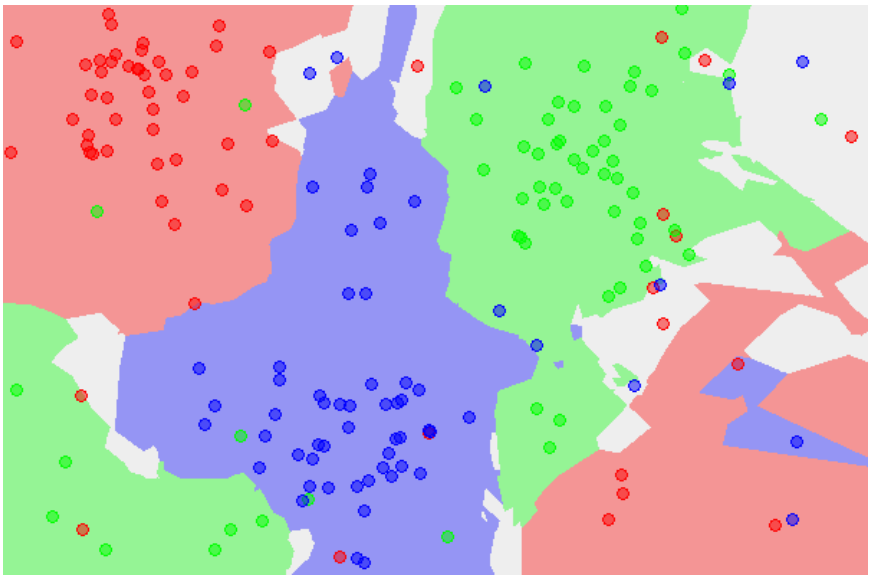


Рисунок 2.5 – Карта класифікації з K = 5

Покроковий алгоритм роботи модуля виявлення зловмисних посилань на онлайн ресурси. Крок перший, модуль починає роботу з завантаження навченої моделі та векторизатора. Навчена модель – це модель машинного навчання, яка була навчена на наборі даних, що містить приклади безпечних та зловмисних посилань. Векторизатор – це інструмент, який перетворює текст посилань у числовий формат, який може бути оброблений моделлю.

Крок другий, модуль відображає головне вікно, яке містить інтерфейс користувача. Інтерфейс користувача дозволяє користувачам вводити URL-адреси посилань, які вони хочуть проаналізувати.

Крок третій, користувач вводить URL-адресу, яке він хоче проаналізувати.

Крок четвертий, модуль використовує алгоритм KNN для попередньої класифікації посилання. Алгоритм KNN порівнює URL-адресу з набором даних навчальних прикладів і знаходить K найближчих прикладів. Класифікація посилання визначається на основі класу більшості з K найближчих прикладів.

Крок п’ятий, якщо URL-адреса класифікована як зловмисна за допомогою KNN, модуль переходить до кроку шість. Якщо URL-адреса класифікована як безпечна, модуль переходить до кроку сім.

Крок шостий, модуль використовує API для перевірки URL-адреси на наявність у чорному списку зловмисних посилань. API – це інтерфейс

програмування додатків, який дозволяє модулю отримувати доступ до даних з інших джерел.

Крок сьомий, модуль оновлює журнал, який містить інформацію про проаналізовані посилання, включаючи їх URL-адреси, класифікацію та дату аналізу (рис. 2.6).

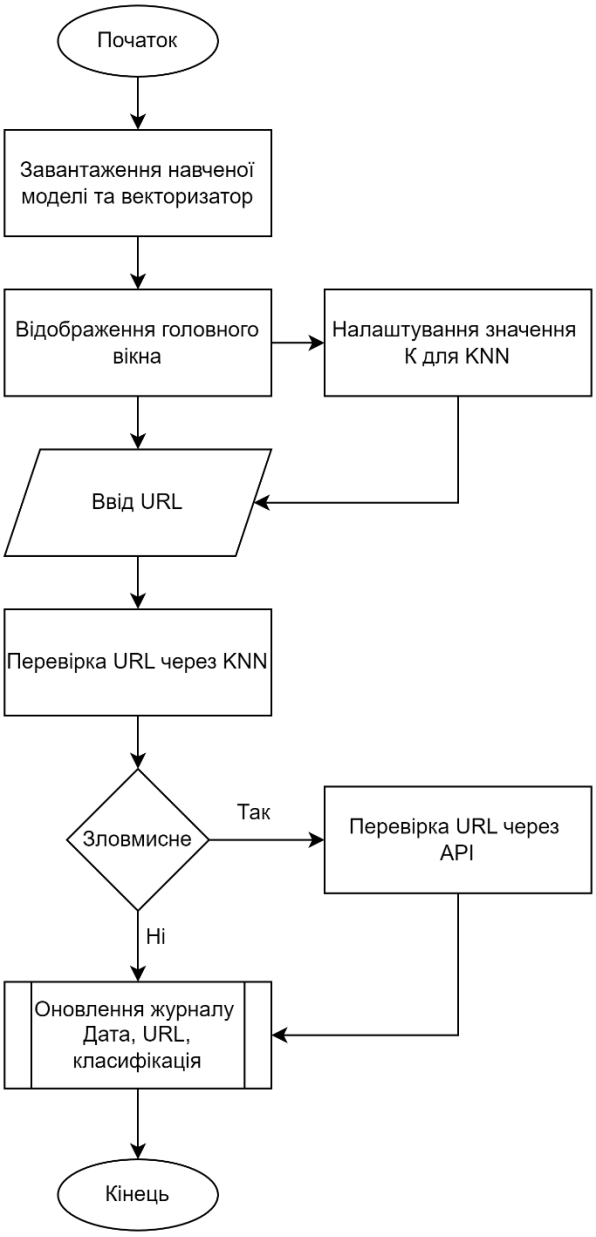


Рисунок 2.6 – Блок-схема роботи модулю

Важливо оцінити продуктивність моделі KNN, щоб переконатися, що вона може точно класифікувати URL-адреси. Це можна зробити за допомогою різних методів, таких як перехресна перевірка та точність.

Існує кілька способів покращити продуктивність моделі KNN, наприклад:

– вибір оптимального значення K: значення K може мати значний вплив на продуктивність моделі. Важливо експериментувати з різними значеннями K, щоб знайти оптимальне значення для конкретного набору даних [17];

– вибір відповідної метрики відстані: метрика відстані, яка використовується для обчислення відстаней між URL-адресами, може також впливати на продуктивність моделі. Важливо вибрати метрику відстані, яка добре підходить для даних;

– використання методів зменшення розмірності: якщо набір даних має високу розмірність, це може призвести до прокляття розмірності, що може погіршити продуктивність моделі. Методи зменшення розмірності можна використовувати для зменшення розмірності набору даних, що може покращити продуктивність моделі.

2.2 Цілі та вимоги до процесу виявлення зловмисних посилань на базі запропонованого методу

Визначимо основні цілі методу виявлення зловмисних посилань на основі машинного навчання засобами KNN:

– виявлення та класифікація зловмисних посилань на веб-сторінках. Зловмисні посилання на веб-сторінках становлять серйозну загрозу для користувачів Інтернету. які містять шкідливе програмне забезпечення, Ці посилання можуть вести до веб-сайтів, фішингові атаки, спам та інші онлайн-загрози [18]. Виявлення та класифікація цих посилань є важливим завданням для забезпечення безпеки користувачів та загального досвіду користування Інтернетом;

– забезпечення безпеки користувачів від шкідливого контенту, фішингу, спаму та інших онлайн-загроз. Інтернет є цінним інструментом для отримання

інформації, спілкування та ведення бізнесу. Однак він також містить безліч онлайн-загроз, які можуть поставити під загрозу безпеку та конфіденційність користувачів;

– покращення загального досвіду користування Інтернетом.

Вимоги до вхідних даних:

– набір даних зловмисних та безпечних URL-адрес;

– характеристики URL-адрес, такі як доменне ім'я, IP-адреса, шлях, параметри, анкорний текст тощо;

– додаткова інформація про веб-сайти, наприклад, рейтинг безпеки, відгуки користувачів, звіти про шкідливе програмне забезпечення тощо.

Вимоги до якості та структури даних. Дані повинні бути максимально точними та актуальними. Помилки в даних можуть призвести до неточних результатів класифікації. Набір даних повинен містити достатню кількість прикладів зловмисних та безпечних URL-адрес для представлення різноманіття онлайн-загроз [18]. Дані повинні бути очищені від помилок, пропусків, дублікатів та аномальних значень. Дані повинні бути представлені в узгодженому форматі, щоб їх можна було легко обробляти методом KNN.

Збір та підготовка даних є критично важливим етапом у розроблення методу виявлення зловмисних посилань. На цьому етапі необхідно забезпечити наявність достатньої кількості якісних даних, які будуть використовуватися для навчання та тестування моделі.

Джерела даних. Для отримання необхідних даних використовувалися різні джерела:

1. Відкриті бази даних. Одним з основних джерел даних стали відкриті бази даних зловмисних URL, такі як PhishTank, OpenPhish, та бази даних, надані компаніями, що спеціалізуються на кібербезпеці [18]. Ці ресурси регулярно оновлюють свої списки зловмисних посилань, що дозволяє отримати актуальну та різноманітну інформацію про загрози.

2. Реальні приклади зловмисних та безпечних посилань. Крім відкритих баз даних, дані були отримані від компаній та організацій, які стикаються з фішингом

та іншими видами кібератак у своїй діяльності. Ці реальні приклади є цінним джерелом, оскільки вони відображають поточні тенденції та методи, які використовуються зловмисниками.

Методи збору даних. Для збору даних використовувалися кілька методів:

1. Веб-скрейпінг. Веб-скрейпінг був використаний для автоматичного збору даних з різних веб-сайтів та ресурсів, які публікують списки зловмисних посилань. За допомогою спеціально розроблених скриптів на Python, зокрема з використанням бібліотек BeautifulSoup та Scrapy, вдавалося регулярно збирати нові дані. Цей метод дозволив швидко та ефективно отримувати велику кількість посилань з мінімальними витратами часу.

2. API. Деякі ресурси, такі як VirusTotal, надають API для доступу до їхніх баз даних. Використання API дозволило інтегрувати процес збору даних безпосередньо в метод, що забезпечило автоматизацію оновлення списків зловмисних та безпечних посилань. API забезпечують структурований доступ до даних і зручний спосіб їх оновлення та обробки.

3. Ручний збір. В окремих випадках, коли автоматизовані методи збору були неможливі або недостатні, використовувався ручний збір даних. Це включало аналіз звітів про кібербезпеку, форуми та блоги, де обговорювалися конкретні інциденти зловмисних атак. Хоча цей метод є більш трудомістким, він дозволяє отримати унікальні дані, які не завжди доступні автоматизованими методами.

Після збору даних необхідно було провести їх підготовку для подальшого використання в моделі:

Очистка даних. Зібрані дані часто містили дублікати, пошкоджені або неповні записи. Було проведено очистку даних, яка включала видалення дублікатів, заповнення або видалення пропущених значень та фільтрацію нерелевантних записів. Цей процес дозволив забезпечити якість даних та підвищити точність моделі.

Нормалізація даних. Для того щоб модель могла ефективно працювати з різними характеристиками URL, дані були нормалізовані. Це включало масштабування числових значень до певного діапазону, наприклад від 0 до 1, та

перетворення категоріальних змінних у числові формати, які може обробляти модель [19]. Нормалізація забезпечила рівнозначність різних характеристик та полегшила процес навчання моделі.

Вибір характеристик (features). Визначення ключових характеристик URL, які мають бути використані для навчання моделі, було важливим етапом підготовки даних. Для цього були обрані характеристики, що мали найбільший вплив на класифікацію посилань, такі як довжина URL, кількість піддоменів, використання спеціальних символів, наявність HTTPS, та інші [19]. Вибір цих характеристик базувався на аналізі попередніх досліджень та експертних рекомендаціях у сфері кібербезпеки.

Таким чином, збір та підготовка даних були проведені з використанням різноманітних джерел та методів, що дозволило створити якісну та репрезентативну базу для навчання та тестування методу виявлення зловмисних посилань на основі KNN.

Підбір оптимального значення K у методі K-найближчих сусідів є важливим етапом, який впливає на точність моделі. Однією з основних методик для вибору цього параметра є крос-валідація.

Кратна крос-валідація (K-fold cross-validation): Цей метод розбиває весь набір даних на K підмножин або "folds". Модель навчається на K-1 підмножинах, а тестується на залишеній підмножині [19]. Процес повторюється K разів, і кожна підмножина використовується рівно один раз як тестова. Середнє значення показників (наприклад, точності) всіх K випробувань використовується для оцінки моделі. Кратна крос-валідація допомагає уникнути проблеми перенавчання та дає змогу отримати більш надійну оцінку продуктивності моделі.

Leave-One-Out Cross-Validation (LOOCV): Спеціальний випадок K-fold крос-валідації, де K дорівнює кількості прикладів у наборі даних [19]. Для кожного прикладу модель навчається на всіх інших прикладах і тестується на цьому конкретному прикладі. LOOCV забезпечує максимально можливе використання даних для навчання, але є обчислювально інтенсивним для великих наборів даних.

Підбір параметра через пошук по сітці (Grid Search). Підбір параметра через пошук по сітці є ще одним ефективним підходом, який часто використовується разом із крос-валідацією. В цьому методі обирається набір можливих значень K і проводиться крос-валідація для кожного значення. Потім обирається значення K, яке забезпечує найкращі показники продуктивності.

Вплив значення K на точність моделі можна описати наступним чином:

– малі значення K:

Коли K мале (наприклад, K = 1 або K = 3), модель може бути схильна до перенавчання, оскільки вона буде чутливою до шуму у навчальних даних. Це може призвести до високої варіативності та низької узагальнювальної здатності. При K = 1, модель просто використовує найближчого сусіда, що може призвести до нестабільних рішень у випадку шумних даних.

– великі значення K:

Коли K велике (наприклад, K = 20 або більше), модель стає більш стабільною і менш схильною до перенавчання. Проте, надмірно велике значення K може призвести до зниження точності, оскільки модель буде більше орієнтована на загальні тенденції даних і може не вловлювати важливі деталі. У цьому випадку модель може втратити здатність до належного класифікування прикладів, що належать до менш поширених класів.

– оптимальне значення K:

Оптимальне значення K часто знаходиться десь посередині. Для його визначення зазвичай проводять експерименти з різними значеннями K, застосовуючи методики крос-валідації. Наприклад, значення K від 5 до 10 часто рекомендується для багатьох задач, але конкретне оптимальне значення залежить від специфіки даних і задачі.

Навчання методу K-найближчих сусідів включає кілька ключових кроків, які забезпечують правильну роботу моделі на підготовлених даних.

Дані повинні бути попередньо оброблені та нормалізовані для забезпечення однорідності. Це включає очистку, нормалізацію та підготовку наборів тренувальних і тестових даних.

Визначення та вибір релевантних характеристик URL для використання в моделі. Ці характеристики можуть включати довжину URL, кількість піддоменів, наявність спеціальних символів, використання HTTPS тощо.

Набір даних розділяється на тренувальний та тестовий набори. Тренувальний набір використовується для навчання моделі, а тестовий — для оцінки її продуктивності.

Вибір оптимального значення K є важливим етапом, який впливає на продуктивність моделі. Зазвичай використовується метод крос-валідації для визначення найкращого значення K.

У процесі навчання модель зберігає тренувальні дані. Метод KNN не має явного етапу навчання, оскільки це метод "ледачого" навчання, де всі обчислення виконуються під час класифікації нових даних.

Для класифікації нового URL обчислюються відстані між цим URL та всіма іншими URL у тренувальному наборі. Найчастіше використовується евклідова відстань, але можливі й інші метрики (манхеттенська, косинусна тощо) [20].

Визначається K найближчих сусідів, і новий URL класифікується на основі більшості класів серед цих сусідів.

Крос-валідація є потужним інструментом для налаштування гіперпараметрів моделі, зокрема параметра K в методі KNN. Ось як цей процес виглядає на практиці:

Набір даних розбивається на K рівних підмножин. Одна підмножина використовується як тестовий набір, а решта K-1 — як тренувальний набір. Процес повторюється K разів, при цьому кожна підмножина використовується один раз як тестовий набір.

Для кожного значення K модель навчається на тренувальному наборі і тестується на тестовій підмножині. Процес повторюється для всіх K підмножин.

Результати кожної ітерації зберігаються, і обчислюється середня точність або інша метрика продуктивності для кожного значення K. Цей підхід дозволяє отримати надійну оцінку ефективності моделі для різних значень K.

Значення K, яке забезпечує найвищу середню точність або найкращу метрику продуктивності на крос-валідації, вибирається як оптимальне для моделі.

Для реалізації методу K-найближчих сусідів у Python зручно використовувати бібліотеку Scikit-learn [21], яка надає готові реалізації методів машинного навчання та інструменти для роботи з даними. Нижче наведені основні етапи реалізації методу KNN: підготовка даних, навчання моделі та оцінка її продуктивності.

Для реалізації методу KNN у роботі пропонується використовувати такі бібліотеки:

– Scikit-learn: для реалізації методів машинного навчання;

– Pandas: для обробки і аналізу даних;

– NumPy: для виконання числових операцій;

– Matplotlib та Seaborn: для візуалізації даних.

2.3 Вибір мови програмування та середовища розробки

Вибір мови програмування та середовища розробки є важливим кроком у проектуванні програмного модуля для виявлення зловмисних посилань. Необхідно врахувати декілька факторів, таких як:

– Функціональні можливості: Мова програмування повинна мати можливість реалізувати метод K-найближчих сусідів та інші необхідні функції, такі як обробка даних, мережеві комунікації та візуалізація.

– продуктивність: програмний модуль повинен мати високу продуктивність, щоб обробляти великі обсяги даних у режимі реального часу;

– простота використання: мова програмування та середовище розробки повинні бути простими у використанні, щоб розробники могли швидко та ефективно створювати та обслуговувати програмний модуль.

З огляду на ці фактори, було вирішено переглянути наступні мови програмування та середовища розробки:

Мови програмування:

Python – це популярна мова програмування загального призначення, яка має широкий спектр бібліотек та інструментів для машинного навчання та обробки даних [22]. Вона також відома своєю простотою використання та читабельністю коду. Python це потужна мова програмування загального призначення, яка водночас є простою у вивченні. Вона характеризується:

– інтерпретованість: код Python не потрібно компілювати перед запуском, що робить розробку швидшою;

– об'єктно-орієнтованістю: Python підтримує парадигму об'єктно-орієнтованого програмування, що дозволяє розбивати програми на модульні блоки;

– простим синтаксисом: Python відомий своїм лаконічним та читабельним кодом, який більше схожий на природну мову;

– стандартною бібліотекою: Python має великий набір вбудованих функцій та модулів, що спрощує розробку;

– широкою сферою застосування: Python використовують для створення вебзастосунків, аналізу даних, машинного навчання та багатьох інших задач.

Завдяки своїм особливостям, Python є популярним вибором як для початківців, так і для досвідчених програмістів.

Іншим варіантом середовища розробки може бути Java. Java – це ще одна популярна мова програмування загального призначення, яка відома своєю надійністю, масштабованістю та портативністю. Вона має велику активну спільноту та широкий спектр бібліотек та інструментів для розробки програмного забезпечення. Java це об'єктно-орієнтована мова програмування загального призначення, створена компанією Sun Microsystems у 1995 році. Вона є однією з найпопулярніших мов програмування у світі завдяки своїм особливостям, таким як:

– портативність: код Java можна компілювати в байт-код, який потім може виконуватися на будь-якій платформі з віртуальною машиною Java (JVM);

– об'єктно-орієнтованість: парадигма програмування в Java дозволяє розробникам створювати модульні програми, що полегшує їх організацію та обслуговування;

– безпека: Java має вбудовані функції безпеки, які допомагають запобігти поширенню шкідливого коду;

– велика спільнота: завдяки великій та активній спільноті розробників, існує багато навчальних ресурсів та бібліотек для Java.

Java широко використовується для розробки різних типів програмного забезпечення, включаючи:

– веб-застосунки;

– підприємницькі додатки;

– вбудовані системи;

– Android-додатки.

Проте, не зважаючи на переваги, мова не уможливлює виконання поставленої задачі через відсутність відповідних бібліотек для реалізації методу машинного навчання.

C++ – це потужна мова програмування загального призначення з багатою історією. Вона була розроблена Бьорном Страуструпом у Bell Labs на початку 1980-х років на основі мови C [24]. C++ увібрав у себе кращі практики C, додавши при цьому функціональність для:

– об'єктно-орієнтованого програмування (ООП): що дозволяє розробникам створювати програми з використанням об'єктів, C++ підтримує ООП парадигму, які представляють реальні сутності світу;

– узагальненого програмування: за допомогою шаблонів можна створювати гнучкий код, який працює з різними типами даних;

– процедурного програмування: C++ підтримує традиційні процедурні підходи до розробки, де програма розбивається на функції.

C++ відомий своєю:

– швидкістю та ефективністю: код C++ компілюється в машинний код, що робить його дуже швидким. Також програміст має детальний контроль над управлінням пам'яттю;

– складністю: потужність C++ може бути двосічним мечем, оскільки складний синтаксис і управління пам'яттю наближають його до мови C. Це вимагає від програмістів більшої дисципліни для запобігання помилкам.

Широкий спектр застосувань робить C++ незамінним для задач, що потребують високої продуктивності:

– системне програмування: операційні системи, драйвери пристроїв та інше системне програмне забезпечення часто пишуться на C++;

– ігри розробки: багато ігрових рушіїв використовують C++ для забезпечення графічної потужності та швидкої реакції;

– вбудовані системи: C++ застосовується для програмування мікроконтролерів та інших вбудованих систем завдяки ефективному управлінню ресурсами;

– фінансові програми: програмне забезпечення для торгівлі на біржі та інші фінансові інструменти часто покладаються на швидкість C++.

Отже, C++ є потужною та універсальною мовою програмування, яка добре підходить для розробки високопродуктивних додатків, але не підходить для поставленої задачі із за складності та відсутності бібліотек.

Середовища розробки:

Visual Studio Code – це безкоштовний редактор коду з відкритим кодом, який підтримує майже всі мови програмування. Він має широкий спектр функцій, таких як підсвічування синтаксису, автодоповнення та інтеграція з системами контролю версій [25]. Він працює на Windows, macOS, Linux та навіть у веб-браузерах. VS Code популярний завдяки своїм функціям, таким як:

– підтримка багатьох мов програмування: VS Code підтримує синтаксис виділення та інтелектуальне автозаповнення коду для широкого спектра мов програмування;

– розширення: існує величезна кількість безкоштовних розширень, які додають нові функції до VS Code, наприклад, відладку, підтримку конкретних фреймворків або мов, та покращують інтерфейс;

– Git інтеграція: VS Code має вбудовану підтримку Git, що дозволяє вам легко керувати вашим кодом;

– легкий та швидкий: VS Code працює швидко і не вимагає потужного комп'ютера.

VS Code є гарним варіантом як для початківців, так і для досвідчених програмістів. Для початківців він пропонує зручний інтерфейс та інтуїтивне керування, а досвідчені програмісти можуть розширити його функціональність за допомогою розширень.

Вибір мови програмування та середовища розробки зрештою залежить від конкретних потреб та вподобань розробників. Однак Python та Java з Visual Studio Code є хорошими вибором для розробки програмного модуля для виявлення зловмисних посилань.

Окрім вищезазначених, є ще кілька факторів, які слід врахувати при виборі мови програмування та середовища розробки:

– досвід розробників: якщо розробники мають досвід роботи з певною мовою програмування або середовищем розробки, це може бути фактором, який слід врахувати;

– вимоги до розгортання: програмний модуль може бути розгорнутий у веб-середовищі, хмарній платформі або локальному сервері. Мова програмування та середовище розробки повинні підтримувати обрану платформу розгортання;

– бюджет: деякі мови програмування та середовища розробки є безкоштовними, тоді як інші потребують платної ліцензії. Необхідно врахувати бюджет проекту при виборі мови програмування та середовища розробки.

Python – відмінний вибір для розроблення модуля виявлення шкідливих посилань на основі методу KNN. Ось короткий виклад того, чому Python добре підходить для цього завдання:

– простота та читабельність: синтаксис Python відомий тим, що він чіткий і лаконічний, що полегшує написання та підтримку коду. Це особливо корисно для проєктів, де співпраця та розуміння коду мають вирішальне значення;

– бібліотеки машинного навчання: Python може похвалитися багатою екосистемою бібліотек машинного навчання, таких як scikit-learn, яка забезпечує ефективну реалізацію методу KNN та інших інструментів аналізу даних;

– можливості обробки даних: такі бібліотеки, як Pandas і NumPy, чудово справляються з маніпулюванням даними та чисельними обчисленнями, що важливо для обробки великих наборів даних URL-адрес і функцій;

– активна спільнота та ресурси: Python має величезну спільноту розробників, пропонуючи великі онлайн-підручники, документацію та форуми для усунення неполадок і пошуку рішень.

– бібліотеки для мережевого зв'язку: хоча Python сильний в аналізі даних, може знадобитися використовувати додаткові бібліотеки, такі як Scrapy або Beautiful Soup, якщо модуль передбачає отримання та аналіз вмісту веб-сайту;

– візуалізація: для візуалізації результатів аналізу KNN, будуть корисними бібліотеки, такі як Matplotlib або Seaborn.

2.4 Висновки до другого розділу

У цьому розділі було сформовано підхід до виявлення зловмисних посилань на основі методу KNN та чорних і білих списків. На першому етапі застосовуємо метод KNN для класифікації URL-адрес шляхом пошуку «найближчого сусіда». Для цього необхідно спочатку навчити модель на наборі даних зловмисних та безпечних посилань, які беруться з білих та чорних списків. На другому етапі реалізовуємо навчену модель для класифікації URL-адрес шляхом порівняння зі вмістом адрес із білих та чорних списків. У разі збігу посилання, відбувається його відповідна класифікація за двома класами (1-зловмисне посилання; 0 - безпечне посилання). На третьому етапі здійснюється оцінювання точності та перевірка

ефективності запропонованого методу. На четвертому етапі необхідно інтегрувати до веб-браузерів, веб-сайтів або інших онлайн-платформ.

Визначено цілі та вимоги до процесу виявлення зловмисних посилань на базі запропонованого підходу, що включають точність, швидкість та надійність моделі. Описано детально кожен з етапів реалізації запропонованого методу, починаючи з підготовки даних, вибору оптимального значення K, навчання моделі, оцінки її ефективності та порівняння з іншими методами.

Було обґрунтовано мову програмування та середовища розробки. Доведено, що Python є засобом для реалізації методу машинного навчання та аналізу даних завдяки своїй простоті, гнучкості та широкому набору бібліотек. Scikit-learn – це популярна бібліотека машинного навчання в Python, яка пропонує реалізацію методу KNN, а також для обробки та аналізу даних, що значно полегшує процес розробки та впровадження моделі.

3 РОЗРОБЛЕННЯ ПРОГРАМНОГО МОДУЛЯ ДО ВИЯВЛЕННЯ ЗЛОВМИСНИХ ПОСИЛАНЬ НА ОНЛАЙН РЕСУРСИ

Зловмисні посилання на онлайн ресурси становлять серйозну загрозу для користувачів, адже вони можуть спричинити крадіжку особистих даних, зараження комп'ютера шкідливим програмним забезпеченням або інші негативні наслідки.

Розробка програмного модуля для виявлення таких посилань є актуальним та важливим завданням, яке може допомогти захистити користувачів від кіберзагроз.

3.1 Розроблення інтерфейсу програмного модуля

У рамках бакалаврської роботи було розроблено програмний модуль для виявлення зловмисних посилань на онлайн ресурси з використанням методу машинного навчання KNN та інтеграцією з чорними та білими списками.

Програма має зручний графічний інтерфейс користувача, створений за допомогою бібліотеки PyQt5. Інтерфейс складається з поля для введення URL-адреси, кнопки для перевірки введеної URL-адреси, кнопки для перевірки через API VirusTotal, кнопки налаштувань та журналу дій.

Для класифікації URL-адрес як зловмисних або безпечних використовується навчена модель KNN. Модель була навчена на наборі даних, що містить як зловмисні, так і безпечні URL-адреси. Для векторизації URL-адрес використовується метод TF-IDF.

Крім класифікації за допомогою моделі KNN, програма також інтегрована з чорними та білими списками URL-адрес. Якщо введена URL-адреса присутня в чорному списку, вона автоматично класифікується як зловмисна. Якщо URL-адреса присутня в білому списку, вона класифікується як безпечна.

Для додаткової перевірки програма також може сканувати введену URL-адресу за допомогою API VirusTotal. Результати сканування VirusTotal відображаються в журналі дій разом з результатами класифікації моделлю KNN.

Програма дозволяє налаштовувати значення параметра K для моделі KNN через діалогове вікно налаштувань. Це дає можливість експериментувати з різними значеннями K для покращення точності класифікації.

Усі результати класифікації та сканування зберігаються в журналі дій, який відображається в інтерфейсі програми. Журнал містить часову мітку, введену URL-адресу, результат класифікації моделлю KNN та результат сканування VirusTotal (якщо доступний).

Програма була розроблена з використанням мови програмування Python та середовища розробки PyCharm. Для роботи з моделями машинного навчання та векторизацією даних використовувалася бібліотека scikit-learn.

Розроблений програмний модуль може бути корисним для організацій та користувачів, які стикаються з проблемою зловмисних посилань на онлайн ресурси, оскільки він допомагає швидко та ефективно ідентифікувати потенційно небезпечні URL-адреси.

Одним з головних етапів є інтерфейс, він важлива частина програмного модуля для виявлення зловмисних посилань, оскільки він забезпечує інтерфейс для користувачів для взаємодії з модулем. Користувачі зможуть вводити URL-адреси для класифікації, переглядати результати класифікації та налаштовувати параметри методу KNN.

Інтерфейс повинен бути простим у використанні та зрозумілим для користувачів з різним рівнем технічних знань. Він повинен бути надійним і стійким до помилок, а також мати можливість обробляти великі обсяги трафіку.

Інтерфейс повинен надавати наступні функції:

– введення URL-адреси: користувачі мають можливість вводити URL-адреси для класифікації (рис 3.1);

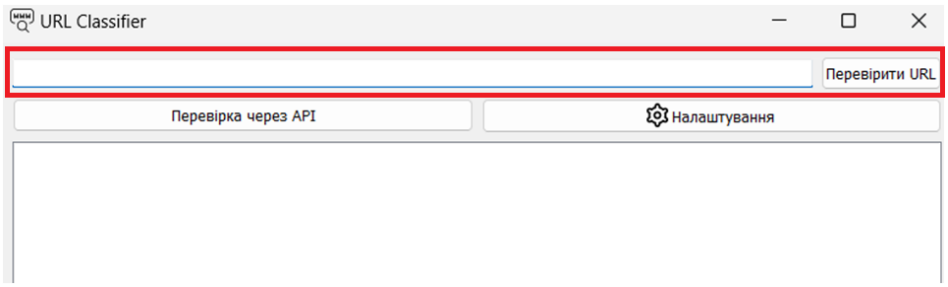


Рисунок 3.1 – Поле введення URL-адрес

– класифікація URL-адрес: інтерфейс відображає результати класифікації URL-адрес, включаючи те, чи класифікована URL-адреса як шкідлива або легітимна (рис 3.2);

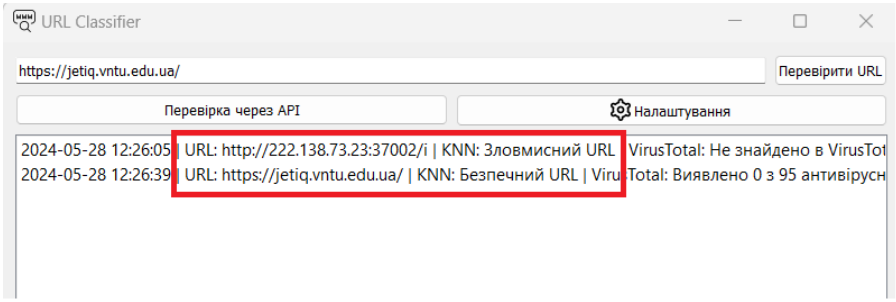


Рисунок 3.2 – Відображення класифікації URL-адреси

– налаштування параметрів: користувачі мають можливість налаштовувати параметри методу KNN, такі як значення K (рис. 3.3) і метрику відстані;

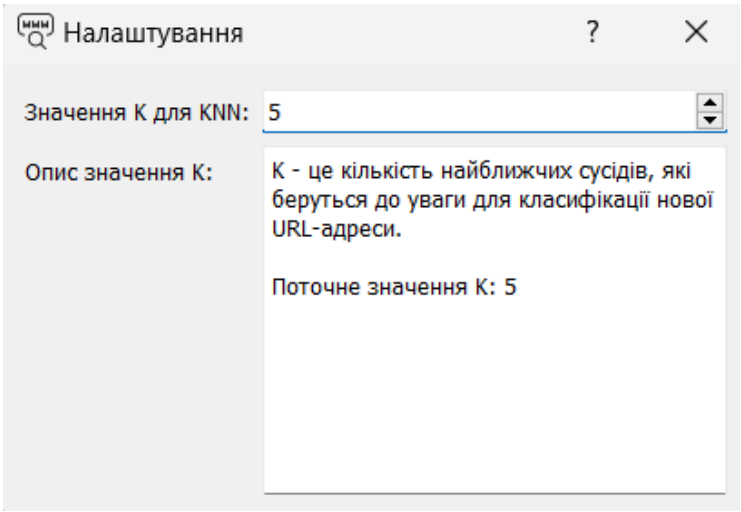


Рисунок 3.3 – Відображення налаштувань KNN

– журнал дій: інтерфейс включає журнал дій, який записує історію класифікації URL-адрес (рис 3.4).

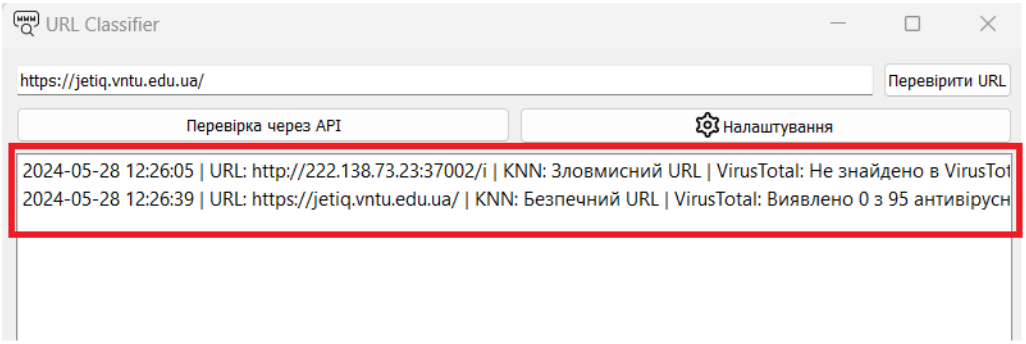


Рисунок 3.4 – Відображення журналу дій

Щоб почати розробляти інтерфейс потрібно навчити модель KHH, для цього використано бібліотеку Sklearn. Частина створеного програмного коду (додаток Б) мовою Python, який демонструє завантаження даних, додавання зловмисних, безпечних посилань.

```
with open('malicious_urls.json', 'r') as f:
    malicious_data = json.load(f)
benign_data = pd.read_csv('benign_urls.csv')
urls = []
labels = []
for url_data in malicious_data.values():
    for entry in url_data:
        urls.append(entry['url'])
        labels.append(1)
for _, row in benign_data.iterrows():
    urls.append(row['Domain'])
    labels.append(0)
```

Також для навчання моделі KHH потрібно векторизувати URL за допомогою TF-IDF, розділити дані на навчальні та тестові набори, зберегти навчену модуль та векторизатор.

```
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(urls)
X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.2, random_state=42)
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
with open('knn_model.pkl', 'wb') as f:
    pickle.dump(knn, f)
with open('vectorizer.pkl', 'wb') as f:
    pickle.dump(vectorizer, f)
```

Для розроблення інтерфейсу програмного модуля (додаток В) було використано бібліотеку PyQt5, яка є одним з найпопулярніших інструментів для створення графічних інтерфейсів користувача для додатків Python [26]. Інтерфейс програмного модуля складається з головного вікна, яке містить наступні елементи:

1. Поле для введення URL-адреси, яку потрібно перевірити.

2. Кнопка "Перевірити URL" для запуску класифікації введеної URL-адреси за допомогою навченої моделі KNN.

```
def check_url(self):
    url = self.url_input.text()
    knn_result, url_vector = classify_url(url)
    self.log_entries.append(LogEntry(url, knn_result, "", url_vector))
    self.update_log_list()
```

3. Кнопка "Перевірка через API" для сканування введеної URL-адреси за допомогою API VirusTotal.

```
def check_api(self):
    url = self.url_input.text()
    api_key = "YOUR_API_KEY"
    params = {
        "url": url,
        "apikey": api_key
    }
    try:
        response = requests.post("https://www.virustotal.com/vtapi/v2/url/scan", data=params)
        response.raise_for_status()
        data = response.json()
        if data["response_code"] == 1:
            scan_id = data["scan_id"]
            report_params = {
                "resource": scan_id,
                "apikey": api_key
            }
            report_response = requests.get("https://www.virustotal.com/vtapi/v2/url/report",
params=report_params)
            report_response.raise_for_status()
            report_data = report_response.json()
            if report_data["response_code"] == 1:
                positives = report_data["positives"]
                total = report_data["total"]
                vt_result = f"Виявлено {positives} з {total} антивірусних програм"
            else:
                vt_result = "Не знайдено в VirusTotal"
        else:
            vt_result = "Помилка сканування URL"
        self.update_log_entry(url, vt_result)
    except requests.exceptions.RequestException as e:
        vt_result = f"Помилка запиту до VirusTotal: {e}"
        self.update_log_entry(url, vt_result)
```

4. Кнопка "Налаштування" для відкриття діалогового вікна налаштувань.

```
    def __init__(self, parent):
        super().__init__(parent)
        self.setWindowTitle("Налаштування")
        self.k_label = QLabel("Значення K для KNN:")
        self.k_spinbox = QSpinBox()
        self.k_spinbox.setMinimum(1)
        self.k_spinbox.setValue(knn.n_neighbors)
        self.k_description = QTextEdit()
        self.k_description.setReadOnly(True)
        self.k_description.setPlainText(f"K - це кількість найближчих сусідів, які беруться до уваги для класифікації нової URL-адреси.\n\nПоточне значення K: {knn.n_neighbors}")
        layout = QFormLayout()
        layout.addRow(self.k_label, self.k_spinbox)
        layout.addRow(QLabel("Опис значення K:"), self.k_description)
        self.setLayout(layout)
        self.k_spinbox.valueChanged.connect(self.update_k)
```

5. Журнал дій, який відображає результати класифікації та сканування URL-адрес.

```
def update_log_entry(self, url, vt_result):
    for entry in self.log_entries:
        if entry.url == url:
            entry.vt_result = vt_result
            break
    self.update_log_list()
def update_log_list(self):
    self.log_list.clear()
    for entry in self.log_entries:
        item = QListWidgetItem(str(entry))
        self.log_list.addItem(item)
```

Інтерфейс реалізований за допомогою візуалізації модулю мовою програмування Python, в подальшому можна створити окрему програму щоб користувачі не встановлювали потрібні бібліотеки.

3.2 Тестування програмного засобу

Тестування програмного засобу проводилося шляхом введення різних URL-адрес у поле введення та перевірки коректності результатів класифікації та сканування. Було протестовано наступні випадки:

1. Введення безпечних URL-адрес, які були правильно класифіковані як "Безпечний URL" моделлю KNN та не виявлені як зловмисні VirusTotal. Приклад відповіді програмного модулю:

URL: https://jetiq.vntu.edu.ua/ | KNN: Безпечний URL | VirusTotal: Виявлено 0 з 95 антивірусних програм

URL: https://www.youtube.com/ | KNN: Безпечний URL | VirusTotal: Виявлено 0 з 95 антивірусних програм

URL: https://app.diagrams.net/ | KNN: Безпечний URL | VirusTotal: Виявлено 0 з 95 антивірусних програм

URL: https://uk.wikipedia.org/ | KNN: Безпечний URL | VirusTotal: Виявлено 0 з 95 антивірусних програм

URL: https://mail.google.com/ | KNN: Безпечний URL | VirusTotal: Виявлено 0 з 95 антивірусних програм

2. Введення зловмисних URL-адрес, які були правильно класифіковані як "Зловмисний URL" моделлю KNN та виявлені як зловмисні VirusTotal.

URL: http://112.248.176.194:52463/Mozi.m | KNN: Зловмисний URL | VirusTotal: Виявлено 11 з 95 антивірусних програм

URL: http://93.123.39.63/mipsel | KNN: Зловмисний URL | VirusTotal: Виявлено 2 з 95 антивірусних програм

URL: http://117.217.39.38:51349/bin.sh | KNN: Зловмисний URL | VirusTotal: Виявлено 7 з 95 антивірусних програм

URL: https://dukeenergyltd.top/louud.scr | KNN: Зловмисний URL | VirusTotal: Виявлено 20 з 95 антивірусних програм

URL: https://covid19help.top/loudzx.exe | KNN: Зловмисний URL | VirusTotal: Виявлено 21 з 95 антивірусних програм

3. Введення URL-адрес, для яких результати класифікації моделлю KNN та сканування VirusTotal не збігалися. У таких випадках перевірялася коректність навчання моделі KNN на основі неправильно класифікованих даних.

URL: https://www.southstar.com.tw/south/download/software/UpdateTool_858.exe | KNN: Безпечний URL | VirusTotal: Виявлено 2 з 95 антивірусних програм

URL: http://bots.gxz.me:8000/skt.sh4 | KNN: Безпечний URL | VirusTotal: Виявлено 10 з 95 антивірусних програм

Всі тестування дали свої результати (рис. 3.5), які дали змогу побачити спроможність кваліфікації URL-адрес, помилки які виникли під час тестувань.

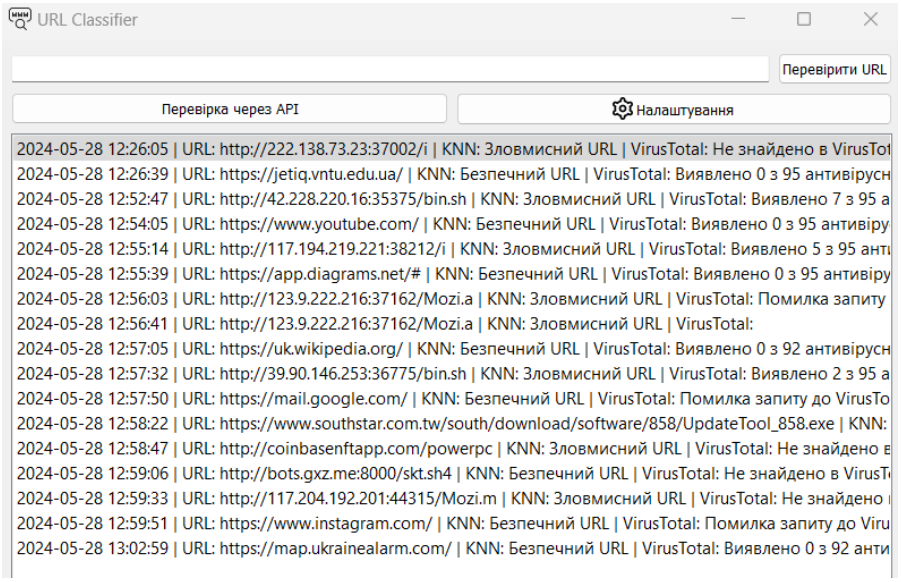


Рисунок 3.5 – Загальний вигляд тестувань програмного модулю

4. Перевірка коректності налаштування значення K для моделі KNN у діалоговому вікні налаштувань (рис. 3.6).

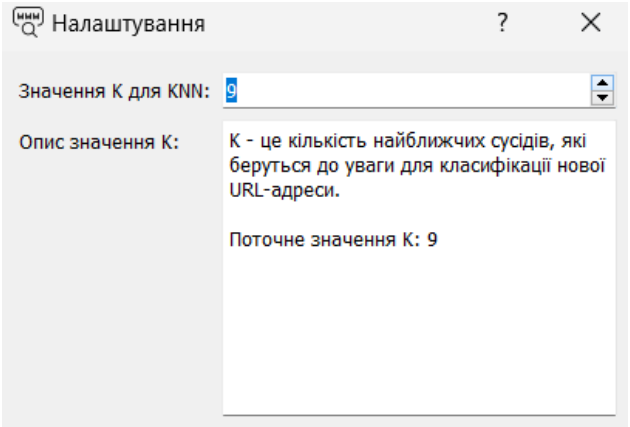


Рисунок 3.6 – Перевірка коректності налаштувань

Під час тестування було виявлено та виправлено кілька помилок у коді, пов'язаних з обробкою винятків та оновленням журналу дій.

3.3 Розроблення інструкції для користувача

Для зручності використання програмного модулю було розроблено інструкцію для користувача, яка містить наступні розділи:

1. Вступ. Короткий опис програми та її призначення.

Програмний модуль для виявлення зловмисних посилань на онлайн ресурси створений з метою забезпечення безпеки користувачів під час перегляду веб-сторінок та користування інтернетом. Основною функцією програми є аналіз введених URL-адрес та виявлення потенційно небезпечних або шкідливих посилань. Цей програмний модуль призначена для:

- перевірки URL-адрес на наявність зловмисного контенту або поведінки;
- відображення результатів перевірки для інформування користувача про потенційні загрози;
- надання рекомендацій щодо безпечного використання інтернет-ресурсів.

Основні функції програмного модулю:

- введення URL: користувач може ввести одну або кілька URL-адрес для перевірки;
- перевірка безпеки: програмний модуль аналізує введені URL-адреси, використовуючи бази даних відомих зловмисних сайтів та алгоритми виявлення шкідливого контенту;
- відображення результатів: результати перевірки виводяться у вигляді списку;
- налаштування: користувач може налаштувати параметри перевірки такі як К для більш точного аналізу.

Цей програмний модуль стане корисним для будь-якого користувача, який хоче підвищити свою інтернет-безпеку, а також для ІТ-фахівців, які потребують інструменту для швидкої перевірки веб-ресурсів на наявність загроз.

2. Системні вимоги: Перелік вимог до операційної системи, версії Python та необхідних бібліотек.

Для коректної роботи програмного модуля для виявлення зловмисних посилань на онлайн ресурси необхідно забезпечити наступні системні вимоги:

- Windows 7 або новіша версія;
- macOS 10.10 або новіша версія;
- Linux (будь-яка сучасна дистрибуція, наприклад, Ubuntu 18.04 або новіша).

Версія Python:

- Python 3.6 або новіша версія

Необхідні бібліотеки Python:

- PyQt5 (5.15 або новіша версія);
- scikit-learn (0.24 або новіша версія);
- pandas (1.2 або новіша версія);
- requests (2.25 або новіша версія);
- pickle (стандартна бібліотека Python).

Додаткові вимоги:

- підключення до Інтернету для використання API VirusTotal;
- ключ API VirusTotal (безкоштовний або платний).

3. Інсталяція: Інструкції з встановлення програми та її залежностей.

Для успішної інсталяції та використання програми "URL Classifier" необхідно виконати наступні кроки:

Крок перший переконайтеся, що на Вашому комп'ютері встановлено Python версії 3.6 або новіша. Ви можете перевірити це, запустивши команду в терміналі (командному рядку):

```
python --version
```

Якщо команда виконується успішно і виводиться інформація про версію Python, то все гаразд.

Крок другий переконайтеся, що Ви маєте права адміністратора на вашому комп'ютері для запуску програми. Це необхідно для того, щоб програма могла коректно працювала.

Крок третій використайте команду для встановлення бібліотек:

```
pip install -r requirements.txt
```

Крок четвертий запустіть програму "URL Classifier" з терміналу (командного рядку). Для цього:

а) Відкрийте термінал (командний рядок) на вашому комп'ютері.

б) Перейдіть до директорії програми, використовуючи команду:

```
cd <шлях до директорії програми>
```

в) Виконайте команду для запуску програми:

```
python main.py
```

Ця команда запустить програму "URL Classifier"

4. Використання програми: Детальний опис інтерфейсу програми та функціоналу кожного елемента інтерфейсу.

Після успішного запуску програми "URL Classifier" Ви побачите головне вікно програми (рис. 3.7).

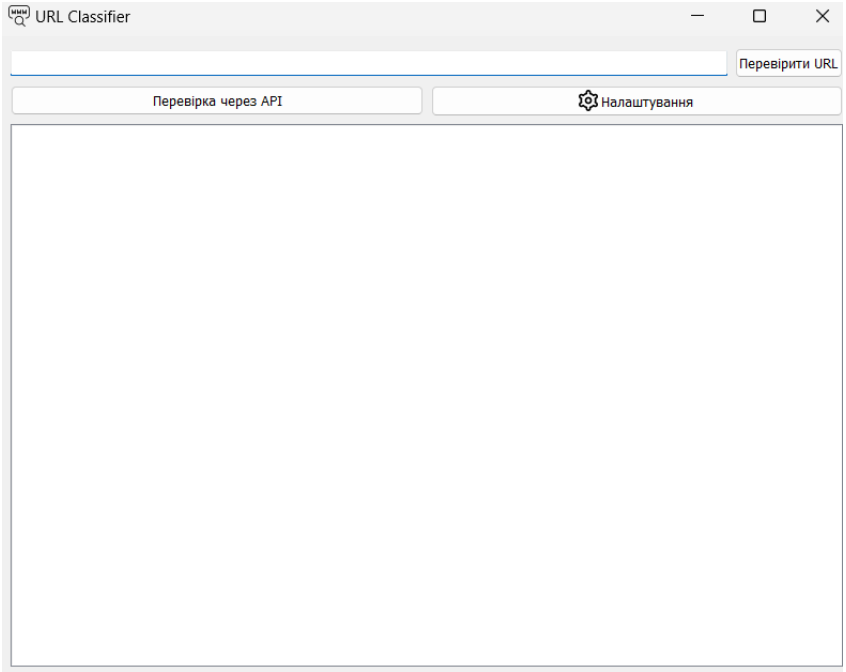


Рисунок 3.7 – Головне вікно програми

Інтерфейс програми складається з наступних елементів:

Поле для введення URL-адреси. У це поле потрібно ввести URL-адресу, яку ви бажаєте перевірити на наявність зловмисного контенту.

Кнопка "Перевірити URL". Після введення URL-адреси в поле, натисніть цю кнопку для запуску процесу класифікації введеної URL-адреси за допомогою навченої моделі K-найближчих сусідів. Результат класифікації буде відображено у журналі дій.

Кнопка "Перевірка через API". Натисніть цю кнопку для сканування введеної URL-адреси за допомогою API VirusTotal. Результат сканування буде відображено у журналі дій.

Кнопка "Налаштування". Натисніть цю кнопку для відкриття діалогового вікна налаштувань. У вікні налаштувань Ви можете змінити значення параметра K для моделі KNN.

Журнал дій. У цьому вікні відображаються результати класифікації та сканування URL-адрес. Кожен запис у журналі містить таку інформацію: часова мітка, введена URL-адреса, результат класифікації моделлю KNN, результат сканування VirusTotal.

Для перевірки URL-адреси на наявність зловмисного контенту виконайте наступні кроки:

- введіть URL-адресу в поле для введення;
- натисніть кнопку "Перевірити URL" для класифікації введеної URL-адреси за допомогою моделі KNN;
- натисніть кнопку "Перевірка через API" для додаткового сканування введеної URL-адреси за допомогою API VirusTotal;
- перегляньте результати класифікації та сканування у журналі дій.

Для зміни значення параметра K для моделі KNN натисніть кнопку "Налаштування", встановіть бажане значення K у діалоговому вікні налаштувань та натисніть "OK".

5. Приклади використання: Кілька прикладів використання програми з поясненнями.

Приклад 1: Перевірка безпечної URL-адреси. Введіть безпечну URL-адресу в поле для введення, наприклад: https://www.google.com. Натисніть кнопку "Перевірити URL". У журналі дій з'явиться новий запис з результатом класифікації

моделлю KNN. Очікуваний результат: "Безпечний URL". Натисніть кнопку "Перевірка через API". У журналі дій з'явиться оновлений запис з результатом сканування VirusTotal. Очікуваний результат: "Не знайдено в VirusTotal" або "Виявлено 0 з X антивірусних програм" (рис. 3.8).

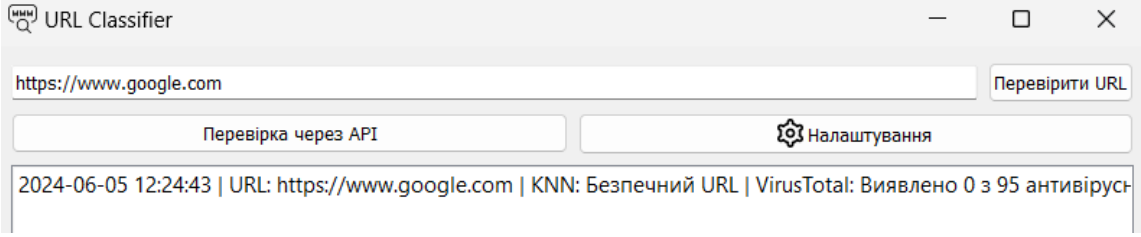


Рисунок 3.8 – Перевірка безпечної URL-адреси

Приклад 2: Перевірка зловмисної URL-адреси. Введіть зловмисну URL-адресу в поле для введення, наприклад: http://malicious.example.com/malware.exe. Натисніть кнопку "Перевірити URL". У журналі дій з'явиться новий запис з результатом класифікації моделлю KNN. Очікуваний результат: "Зловмисний URL". Натисніть кнопку "Перевірка через API". У журналі дій з'явиться оновлений запис з результатом сканування VirusTotal. Очікуваний результат: "Виявлено X з Y антивірусних програм" (де X - кількість антивірусних програм, які виявили зловмисний контент) (рис. 3.9).

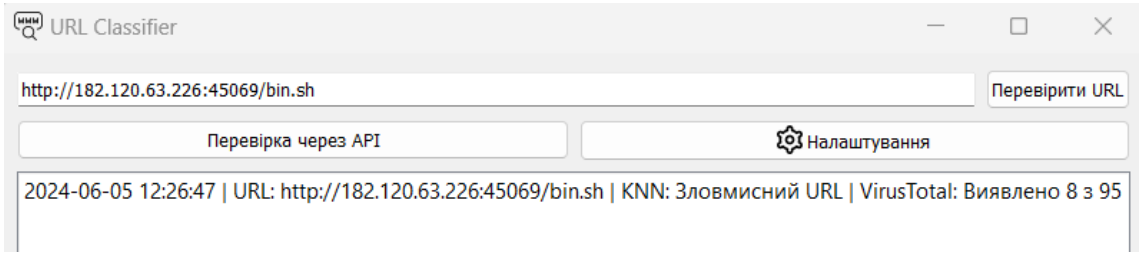


Рисунок 3.9 – Перевірка зловмисної URL-адреси

Приклад 3: Зміна значення K для моделі KNN. Натисніть кнопку "Налаштування". У діалоговому вікні налаштувань Ви побачите поточне значення K для моделі KNN. Змініть значення K. Натисніть "OK" для застосування нового значення K. Тепер модель KNN буде використовувати нове значення K для класифікації URL-адрес.

3.4 Висновки до третього розділу

У цьому розділі було розглянуто процес розроблення програмного модуля для виявлення зловмисних посилань на онлайн ресурси. Основна увага була приділена створенню інтерфейсу програмного модуля, його тестуванню та розробці інструкції для користувача.

Створений інтерфейс забезпечує зручний та інтуїтивно зрозумілий спосіб взаємодії користувача з програмним модулем. Тестування підтвердило коректність роботи всіх функцій інтерфейсу, а розроблена інструкція для користувача допоможе швидко освоїти використання програмного модуля.

У третьому розділі було описано процес розроблення програмного модуля для виявлення зловмисних посилань на онлайн ресурси. Було розроблено графічний інтерфейс користувача за допомогою бібліотеки PyQt5, який складається з головного вікна з полем для введення URL-адреси, кнопками для перевірки URL-адреси та налаштувань, а також журналом дій.

Програмний засіб було протестовано на різних наборах даних, включаючи безпечні та зловмисні URL-адреси, а також випадки, коли результати класифікації моделлю KNN та сканування VirusTotal не збігалися. Під час тестування було виявлено та виправлено кілька помилок у коді.

Для полегшення використання програмного засобу було розроблено інструкцію для користувача, яка містить детальний опис інтерфейсу, інструкції з інсталяції та використання програми, а також поради щодо вирішення можливих проблем.

Загалом, розроблений програмний модуль забезпечує зручний та ефективний спосіб виявлення зловмисних посилань на онлайн ресурси за допомогою навченої моделі KNN та сканування URL-адрес через API VirusTotal.

ВИСНОВКИ

Розроблення програмного модуля виявлення зловмисних посилань для онлайн-ресурсів є актуальною науково-практичною проблемою, яка має значний вплив на розвиток інформаційних технологій та кібербезпеку. Вирішення цієї проблеми допоможе захистити користувачів онлайн-ресурсів від кіберзлочинності та сприятиме розвитку інформаційного суспільства в Україні.

У даній роботі було розроблено програмний модуль для виявлення зловмисних посилань на онлайн ресурси за допомогою методу машинного навчання K-найближчих сусідів та чорних, білих списків URL.

У першому розділі було проведено аналіз існуючих підходів до виявлення зловмисних посилань, розглянуто базові поняття та переваги й недоліки різних методів та інструментів. На основі проведеного аналізу було обґрунтовано вибір методу машинного навчання KNN для виявлення зловмисних посилань.

У другому розділі було розроблено метод до виявлення зловмисних посилань на онлайн ресурси, який поєднує метод KNN та чорні, білі списки. Були визначені цілі та вимоги до процесу виявлення зловмисних посилань, а також обрано мову програмування Python та середовище розробки Visual Studio Code для реалізації програмного модуля.

У третьому розділі було описано процес розроблення програмного модуля, включаючи створення графічного інтерфейсу користувача. Програмний модуль було протестовано на різних наборах даних, а також було розроблено інструкцію для користувача.

Розроблений програмний модуль забезпечує зручний та ефективний спосіб виявлення зловмисних посилань на онлайн ресурси за допомогою навченої моделі KNN та сканування URL-адрес через API VirusTotal. Модуль має інтуїтивно зрозумілий інтерфейс з полем для введення URL-адреси, кнопками для перевірки URL-адреси та налаштувань, а також журналом дій.

Одною з ключових переваг розробленого програмного модуля є можливість навчання моделі KNN на основі неправильно класифікованих даних, що дозволяє

покращувати точність класифікації з часом. Крім того, інтеграція з API VirusTotal забезпечує додатковий рівень перевірки URL-адрес, що підвищує загальну ефективність виявлення зловмисних посилань.

Програмний модуль буде корисним для різних організацій та користувачів, які стикаються з проблемою зловмисних посилань на онлайн ресурси, оскільки він допомагає швидко та ефективно ідентифікувати потенційно небезпечні URL-адреси.

Подальші напрямки розвитку програмного модуля можуть включати інтеграцію з додатковими джерелами даних про зловмисні посилання, вдосконалення алгоритмів машинного навчання та розширення функціональності інтерфейсу користувача.

СПИСОК ЛІТЕРАТУРИ

1. Приймак А. В., Карпінець В.В., Яремчук Ю.Є. Метод автоматизованого пошуку несанкціонованого майнінгу криптовалюти у контейнерах серверних ОС. *Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні*. 2019. Випуск № 2(38). С. 18–26.

2. Данилюк І.І., Карпінець В.В., Приймак А. В., Яремчук Ю. Є., Костюченко О.І. Метод ідентифікації користувача за клавіатурним почерком на основі нейромереж. *Матеріали ІІІ-ої Міжнародної науково-технічної конференції "Інформаційна безпека в сучасному суспільстві"*. 2018. С. 50–51.

3. Яремчук Ю. Є. Криптографічні методи та засоби шифрування інформації на основі рекурентних послідовностей : Монографія. Вінниця : Кн.-Вега, 2002. 136 с.

4. Підвищення стійкості криптографічних алгоритмів у багатокористувацьких web-ресурсах на основі генераторів випадкових чисел, що враховують ентропію поведінки користувача / О. Салієва та ін. *Measuring and computing devices in technological processes*. 2023. № 1. С. 167–173.

5. Богом'я В. І. Особливості використання штучного інтелекту та машинного навчання для виявлення та запобігання кібератак. *Vodnij transport*. 2023. № 2(38). С. 335–343.

6. Kohonen R. Hydraulic network simulation. Espoo, Finland : Valtion teknillinen tutkimuskeskus, 1989. 115 p.

7. An Improved kNN Algorithm – Fuzzy kNN / W. Shang та ін. *Computational Intelligence and Security*. Berlin, Heidelberg, 2005. С. 741–746. URL: https://doi.org/10.1007/11596448_109

8. M. L. and ML and AI Academy. Machine Learning with Python: The Ultimate Guide to Learn Machine Learning Algorithms. Includes a Useful Section about Analysis, Data Mining and Artificial Intelligence in Business Applications. Independently Published, 2020.

9. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, 2019. 856 p.

10. VirusTotal. *VirusTotal*. URL: https://www.virustotal.com/gui/home/url (дата звернення: 20.05.2024).

11. Website Safety Check & Phishing Protection | WOT. *Website Safety Check & Phishing Protection | WOT*. URL: https://www.mywot.com/ (дата звернення: 20.05.2024).

12. Check if a Website is Malicious/Scam or Safe/Legit | URLVoid. *URLVoid.com*. URL: https://www.urlvoid.com/ (дата звернення: 13.04.2024).

13. Browse safely and steer clear of online dangers | McAfee WebAdvisor. *McAfee*. URL: https://www.mcafee.com/en-us/safe-browser/mcafee-webadvisor.html (дата звернення: 14.04.2024).

14. Safeweb. *Safeweb*. URL: https://safeweb.norton.com/ (дата звернення: 15.04.2024).

15. Zhang S. Challenges in KNN Classification. *IEEE Transactions on Knowledge and Data Engineering*. 2021. С. 1. URL: https://doi.org/10.1109/tkde.2021.3049250

16. kNN Query. *Encyclopedia of Database Systems*. Boston, MA, 2009. P. 1590. URL: https://doi.org/10.1007/978-0-387-39940-9_2922

17. kNN Classification: a review / P. K. Syriopoulos et al. *Annals of Mathematics and Artificial Intelligence*. 2023. URL: https://doi.org/10.1007/s10472-023-09882-x

18. Vidales A. Machine Learning with Matlab: Supervised Learning Using. Independently Published, 2019.

19. Brownlee J. K-Nearest Neighbors for Machine Learning - MachineLearningMastery.com. *MachineLearningMastery.com*. URL: https://machinelearningmastery.com/k-nearest-neighbors-for-machine-learning/ (дата звернення: 17.04.2024).

20. Krause E. F. Taxicab Geometry. *The Mathematics Teacher*. 1973. Vol. 66, no. 8. P. 695–706.

21. Applications for Python. *Python.org*. URL: https://www.python.org/about/apps/ (дата звернення: 19.04.2024).

22. Python Tutorial: Data Cleaning and Preprocessing for ML. *Your Path to Mastering AI*. URL: https://machinelearningmodels.org/python-tutorial-data-cleaning-and-preprocessing-for-ml/ (дата звернення: 19.04.2024).

23. Buelta J. Python Automation Cookbook: 75 Python Automation Ideas for Web Scraping, Data Wrangling, and Processing Excel, Reports, Emails, and More, 2nd Edition. Packt Publishing, Limited, 2020. 526 p.

24. C++ Language Reference. *Microsoft Learn: Build skills that open doors in your career*. URL: https://learn.microsoft.com/uk-ua/cpp/cpp/cpp-language-reference?view=msvc-170 (дата звернення: 25.04.2024).

25. Microsoft. Visual Studio Code - Code Editing. Redefined. *Visual Studio Code - Code Editing. Redefined*. URL: https://code.visualstudio.com/ (дата звернення: 15.04.2024).

26. Азарова А. О., Олексюк Є. М., Азарова Л. Є., Комп'ютерна програма «Розробка програмного модуля виявлення зловмисних посилань для онлайн ресурсів» Свідоцтво про реєстрацію авторського права на твір № 214529. Дата реєстрації 10.06.2024 р. Заявка с202405782

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ

Голова секції "Управління інформаційною безпекою" кафедри МБІС
д.т.н., професор

_____ Юрій ЯРЕМЧУК

" 22 " березня 2024 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до бакалаврської дипломної роботи на тему:

«Розробка програмного модуля виявлення зловмисних посилань для онлайн ресурсів»

08-42. БДР.017.00.065.ТЗ

Керівник бакалаврської дипломної роботи
к.т.н., проф., проф., каф., МБІС:

_____ Азарова А. О.

" _____ " _____

Вінниця – 2024 р.

1. Найменування та область застосування

Програмний модуль виявлення зловмисних посилань для онлайн ресурсів. Область застосування: захист користувачів від кіберзагроз на онлайн ресурсах.

2.Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ № 80 від 11.03.2024 р.

3. Мета та призначення розробки

3.1 Мета розробки: розробка програмного модуля для виявлення зловмисних посилань на онлайн ресурси з використанням методу машинного навчання K-найближчих сусідів та чорних, білих списків.

3.2 Призначення: розроблений програмний модуль попереджає користувачів про зловмисні посилання.

4. Джерела розробки

4.1. Kohonen R. Hydraulic network simulation. Espoo, Finland : Valtion teknillinen tutkimuskeskus, 1989. 115 p.

4.2. M. L. and ML and AI Academy. Machine Learning with Python: The Ultimate Guide to Learn Machine Learning Algorithms. Includes a Useful Section about Analysis, Data Mining and Artificial Intelligence in Business Applications. Independently Published, 2020.

4.3. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, 2019. 856 p.

4.4. Богом’я В. І. Особливості використання штучного інтелекту та машинного навчання для виявлення та запобігання кібератак. Vodnij transport. 2023. № 2(38). С. 335–343.

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Програмний модуль повинен мати зручний, легкий у використанні інтерфейс користувача;

5.1.2 Реалізація модулю не повинна вимагати спеціальних ліцензійних програмних додатків;

5.1.3 Програмний модуль повинен виконувати процес виявлення зловмисних посилань.

5.2 Вимоги до надійності:

5.2.1 Програмний модуль повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень;

5.2.2 Бази даних повинні бути налаштовані на навчання моделі КНН;

5.2.3 Програмний модуль повинен виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– процесор – Pentium 1500 МГц і подібні до них;

– оперативна пам’ять – не менше 512 Mb;

– середовище функціонування – операційна система сімейство Windows;

– вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

6. Вимоги до програмної документації

6.1 Обов’язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.3

7. Вимоги до технічного захисту інформації

7.1 Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2 Можливість отримання доступу користувачів до інформаційних ресурсів таких як чорні та білі списки.

8. Техніко-економічні показники

8.1 Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

8.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

9. Стадії та етапи розробки

№ з/п	Назва етапів бакалаврської дипломної роботи	Початок	Закінчення
1.	Визначення напрямку бакалаврської роботи, формулювання теми		
2.	Аналіз предметної області обраної теми		
3.	Розробка алгоритму роботи		
4.	Написання бакалаврської роботи на основі розробленої теми		
5.	Передзахист бакалаврської дипломної роботи		
6.	Виправлення, уточнення, корегування бакалаврської дипломної роботи		
7.	Захист бакалаврської дипломної роботи		

10. Порядок контролю та прийому

10.1 До приймання бакалаврської дипломної роботи надається:

- ПЗ до бакалаврської дипломної роботи;
- демонстрація результату бакалаврської дипломної роботи;
- презентація;
- відзив керівника роботи;
- відзив рецензента.

Технічне завдання до виконання прийняв _____ Олексюк Є. М.

Додаток Б. Лістинг коду для навчання KHH

```
import json
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import pickle
with open('malicious_urls.json', 'r') as f:
    malicious_data = json.load(f)
benign_data = pd.read_csv('benign_urls.csv')
urls = []
labels = []
for url_data in malicious_data.values():
    for entry in url_data:
        urls.append(entry['url'])
        labels.append(1)  # Зловмисний URL
for _, row in benign_data.iterrows():
    urls.append(row['Domain'])
    labels.append(0)  # Безпечний URL
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(urls)
X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.2, random_state=42)
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Точність моделі KNN: {accuracy * 100:.2f}%")
with open('knn_model.pkl', 'wb') as f:
    pickle.dump(knn, f)
with open('vectorizer.pkl', 'wb') as f:
    pickle.dump(vectorizer, f)
```

Додаток В. Лістинг програмного модулю до виявлення зловмисних посилань

```
import sys
import pickle
import requests
from PyQt5.QtWidgets import QApplication, QMainWindow, QVBoxLayout, QHBoxLayout, QLineEdit,
QPushButton, QWidget, QListWidgetItem, QListWidget, QDialog, QLabel, QSpinBox, QFormLayout, QTextEdit
from PyQt5.QtGui import QIcon
from datetime import datetime
with open('knn_model.pkl', 'rb') as f:
    knn = pickle.load(f)
with open('vectorizer.pkl', 'rb') as f:
    vectorizer = pickle.load(f)
def classify_url(url):
    url_vector = vectorizer.transform([url])
    prediction = knn.predict(url_vector)
    if prediction[0] == 1:
        return "Зловмисний URL", url_vector
    else:
        return "Безпечний URL", url_vector
class LogEntry:
    def __init__(self, url, knn_result, vt_result, url_vector):
        self.timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        self.url = url
        self.knn_result = knn_result
        self.vt_result = vt_result
        self.url_vector = url_vector
    def __str__(self):
        return f"{self.timestamp} | URL: {self.url} | KNN: {self.knn_result} | VirusTotal: {self.vt_result}"
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("URL Classifier")
        self.setGeometry(100, 100, 800, 600)
        self.setWindowIcon(QIcon('app_icon.png'))
        self.url_input = QLineEdit()
        self.check_button = QPushButton("Перевірити URL")
        self.api_button = QPushButton("Перевірка через API")
        self.settings_button = QPushButton("Налаштування")
        self.settings_button.setIcon(QIcon('settings_icon.png'))
        self.log_entries = []
        self.log_list = QListWidget()
        main_layout = QVBoxLayout()
        input_layout = QHBoxLayout()
        button_layout = QHBoxLayout()
        input_layout.addWidget(self.url_input)
        input_layout.addWidget(self.check_button)
        button_layout.addWidget(self.api_button)
        button_layout.addWidget(self.settings_button)
        main_layout.addLayout(input_layout)
        main_layout.addLayout(button_layout)
        main_layout.addWidget(self.log_list)
        central_widget = QWidget()
        central_widget.setLayout(main_layout)
        self.setCentralWidget(central_widget)
        self.check_button.clicked.connect(self.check_url)
        self.api_button.clicked.connect(self.check_api)
        self.settings_button.clicked.connect(self.show_settings)
```

```
    def check_url(self):
        url = self.url_input.text()
        knn_result, url_vector = classify_url(url)
        self.log_entries.append(LogEntry(url, knn_result, "", url_vector))
        self.update_log_list()
    def check_api(self):
        url = self.url_input.text()
        api_key = "YOUR_API_KEY"
        params = {
            "url": url,
            "apikey": api_key
        }
        try:
            response = requests.post("https://www.virustotal.com/vtapi/v2/url/scan", data=params)
            response.raise_for_status()
            data = response.json()
            if data["response_code"] == 1:
                scan_id = data["scan_id"]
                report_params = {
                    "resource": scan_id,
                    "apikey": api_key
                }
                report_response = requests.get("https://www.virustotal.com/vtapi/v2/url/report", params=report_params)
                report_response.raise_for_status()
                report_data = report_response.json()
                if report_data["response_code"] == 1:
                    positives = report_data["positives"]
                    total = report_data["total"]
                    vt_result = f"Виявлено {positives} з {total} антивірусних програм"
                else:
                    vt_result = "Не знайдено в VirusTotal"
            else:
                vt_result = "Помилка сканування URL"
            self.update_log_entry(url, vt_result)
        except requests.exceptions.RequestException as e:
            vt_result = f"Помилка запиту до VirusTotal: {e}"
            self.update_log_entry(url, vt_result)
    def update_log_entry(self, url, vt_result):
        for entry in self.log_entries:
            if entry.url == url:
                entry.vt_result = vt_result
                break
        self.update_log_list()
    def update_log_list(self):
        self.log_list.clear()
        for entry in self.log_entries:
            item = QListWidgetItem(str(entry))
            self.log_list.addItem(item)
    def show_settings(self):
        settings_dialog = SettingsDialog(self)
        settings_dialog.exec_()
class SettingsDialog(QDialog):
    def __init__(self, parent):
        super().__init__(parent)
        self.setWindowTitle("Налаштування")
        self.k_label = QLabel("Значення K для KNN:")
        self.k_spinbox = QSpinBox()
        self.k_spinbox.setMinimum(1)
        self.k_spinbox.setValue(knn.n_neighbors)
        self.k_description = QTextEdit()
        self.k_description.setReadOnly(True)
```

```
            self.k_description.setPlainText(f"K - це кількість найближчих сусідів, які беруться до уваги для класифікації нової URL-адреси.\n\nПоточне значення K: {knn.n_neighbors}")
            layout = QFormLayout()
            layout.addRow(self.k_label, self.k_spinbox)
            layout.addRow(QLabel("Опис значення K:"), self.k_description)
            self.setLayout(layout)
            self.k_spinbox.valueChanged.connect(self.update_k)
        def update_k(self, value):
            global knn
            knn.n_neighbors = value
            self.k_description.setPlainText(f"K - це кількість найближчих сусідів, які беруться до уваги для класифікації нової URL-адреси.\n\nПоточне значення K: {value}")
    if __name__ == "__main__":
        app = QApplication(sys.argv)
        window = MainWindow()
        window.show()
        sys.exit(app.exec_())
```

Додаток Г. Ілюстративний матеріал

Вінницький національний технічний університет

Розробка програмного модуля виявлення зловмисних посилань для онлайн ресурсів

Виконав:
студент 4 курсу
групи 2КІТС-20б
Олексюк Є. М.

Вінниця-2024

Мета: покращення процесу ідентифікації зловмисних посилань на онлайн ресурси із застосуванням чорних, білих списків та методу К-найближчих сусідів.

Об'єкт: процес виявлення зловмисних посилань в інформаційних середовищах.

Предмет: автоматизований підхід для виявлення зловмисних посилань на онлайн ресурси.

Актуальність теми: В епоху цифрових технологій та активного використання Інтернету зловмисні посилання на онлайн ресурси становлять серйозну загрозу для безпеки користувачів. Такі посилання можуть призводити до завантаження шкідливого програмного забезпечення, крадіжки конфіденційних даних, фінансових втрат або поширення шахрайських схем. Зловмисники використовують різноманітні методи, такі як фішинг, спам-розсилки та приховані вразливості веб-сайтів, для розповсюдження зловмисних посилань.

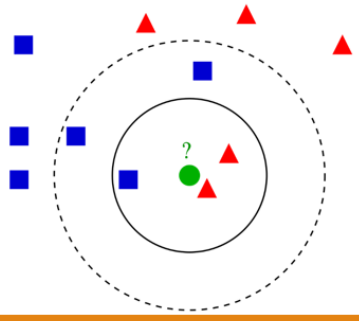
Аналіз існуючих методів

Метод	Переваги	Недоліки
Чорні списки URL	Прості, ефективні для виявлення відомих зловмисних посилань, не потребують значних обчислювальних ресурсів	Не можуть виявити нові зловмисні посилання; призводять до блокування легітимних веб-сайтів
Білі списки URL	Гарантують, що користувачі відвідують лише безпечні веб-сайти	Не спроможні виявити зловмисні веб-сайти, які не включені до списку відповідно є незручними для користувачів
Аналіз контенту веб-сторінок	Здатний побачити зловмисні веб-сайти, які не можна виявити за допомогою чорних або білих списків; враховує контекст веб-сторінки	Є обчислювально складним та призводить до помилок
Аналіз поведінки користувачів	Здібний до виявлення зловмисних веб-сайтів, також враховує поведінку користувача на веб-сайті	Надокучливий для користувачів, призводить до помилок
Аналіз URL-адрес	Розкриває зловмисні посилання, які не включені до чорних або білих списків; не потребує значних обчислювальних ресурсів	Точність є недостатньою, призводить до блокування легітимних веб-сайтів
Методи машинного навчання	Адаптування до нових видів атак, також є більш точними, ніж інші методи	Потенційно складний для реалізації, потребують великих обсягів даних для навчання

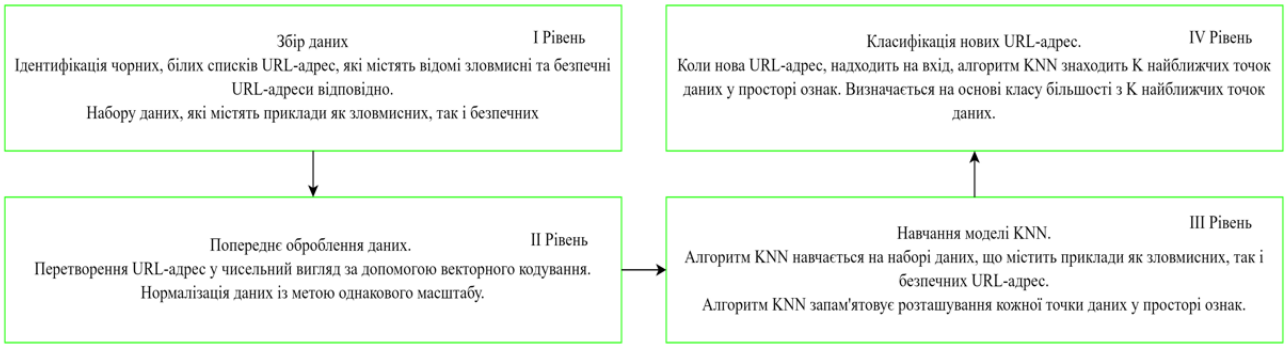
Обґрунтування вибору методу K-найближчих сусідів (KNN)

Використання методу K-найближчих сусідів для виявлення зловмисних посилань є перспективним методом, який демонструє високу ефективність та здатність працювати з великими обсягами даних. Завдяки своїм перевагам, таким як простота реалізації, гнучкість та здатність до навчання, KNN може стати цінним інструментом для забезпечення безпеки онлайн ресурсів.

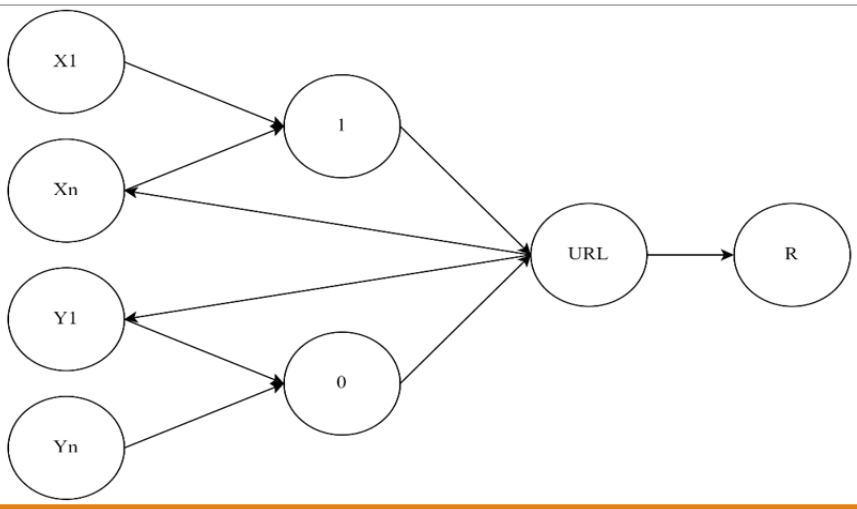
Метод K-найближчих сусідів є непараметричним методом керованого навчання. Він використовується як для класифікації, так і для регресії. Основна ідея полягає в тому, що для вхідних даних обчислюється відстань до k найближчих навчальних прикладів у наборі даних.



Структура моделі виявлення зловмисних посилань з використанням методу KNN, чорних та білих списків

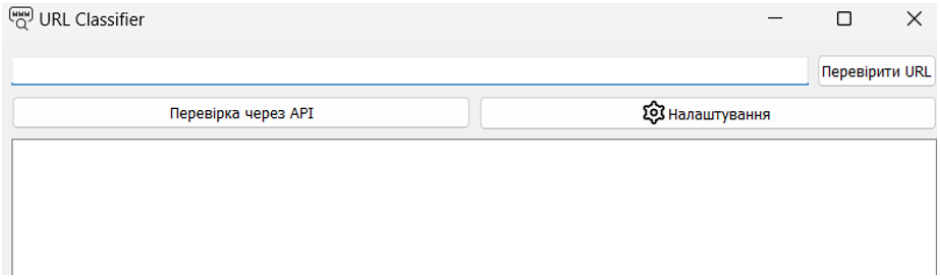


Архітектура навчання моделі KNN

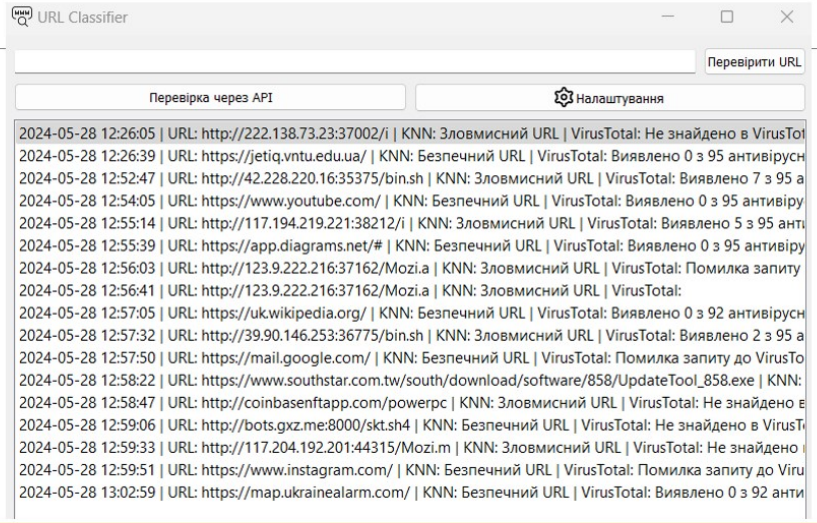


Для класифікації URL-адрес як зловмисних або безпечних використовується навчена модель KNN. Модель була навчена на наборі даних, що містить як зловмисні, так і безпечні URL-адреси. Для векторизації URL-адрес використовується метод TF-IDF.

Крім класифікації за допомогою моделі KNN, програма також інтегрована з чорними та білими списками URL-адрес. Якщо введена URL-адреса присутня в чорному списку, вона автоматично класифікується як зловмисна. Якщо URL-адреса присутня в білому списку, вона класифікується як безпечна.



Загальний вигляд тестувань програмного модулю



Висновок

У даній роботі було розроблено програмний модуль для виявлення зловмисних посилань на онлайн ресурси. Модуль поєднує метод машинного навчання K-найближчих сусідів та інтеграцію з чорними і білими списками URL. Це забезпечує високу точність виявлення зловмисних посилань та адаптивність до нових загроз.
Програмний модуль має зручний графічний інтерфейс з полем для введення URL, кнопками для перевірки та налаштувань, а також журналом дій. Модуль класифікує URL за допомогою навченої моделі KNN та додатково сканує їх через API VirusTotal. Можливість навчання моделі на неправильно класифікованих даних дозволяє покращувати точність з часом.
Розроблений програмний модуль є корисним інструментом для організацій та користувачів, які прагнуть підвищити рівень кібербезпеки та захиститися від зловмисних посилань. Він допомагає швидко та ефективно ідентифікувати потенційно небезпечні URL-адреси, мінімізуючи ризики, пов'язані з відвідуванням шкідливих онлайн ресурсів.

ДЯКУЮ ЗА УВАГУ!

Додаток Д. Протокол перевірки на антиплагіат

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Розробка програмного модуля виявлення зловмисних посилань для онлайн ресурсів

Тип роботи: бакалаврська дипломна робота
(БДР, МКР)

Підрозділ: Кафедра менеджменту та безпеки інформаційних систем
Факультет менеджменту та інформаційної безпеки
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 97 % Схожість 3 %

Аналіз звіту подібності (відмітити потрібне):
1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.
2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.
3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Коваль Н.П.
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Олексюк Є.М.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Азарова А.О.
(підпис) (прізвище, ініціали)