Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:
«Метод і засіб виявлення фішингових інформаційних ресурсів»

Виконав: студент 2-го курсу, групи 1БС-20м
спеціальності 125 – Кібербезпека
_____ Думчиков С.А.
Керівник: к.т.н., ст. викл. каф. ЗІ
_____ Лукічов В.В.
Опонент: к.т.н., проф. каф . ОТ
_____ Азарова А.О.
«___» _____ 2021 р.

Допущено до захисту
Завідувач кафедри ЗІ
д.т.н., проф.
_____ Лужецький В.А.
«___» _____ 2021 р.

Вінниця ВНТУ – 2021 рік

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 Інформаційні технології
Спеціальність 125 Кібербезпека
Освітньо-професійна програма – Безпека інформаційних і комунікаційних систем

ЗАТВЕРДЖУЮ
Завідувач кафедри ЗІ,
д.т.н., проф.
_____ В.А. Лужецький
«___» _____ 2021 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Думчикову Станіславу Андрійовичу

1. Тема роботи: «Метод і засіб виявлення фішингових інформаційних ресурсів»
керівник роботи: Лукічов Віталій Володимирович, к.т.н., старший викладач каф. ЗІ,
затверджені наказом ректора ВНТУ від 24 вересня 2021 року №277.

2. Строк подання студентом роботи 20 грудня 2021 р.

3. Вихідні дані до роботи:
   - мова URL-адреси сайту – кирилиця, латиниця;
   - формат даних – текстове URL-посилання;
   - спосіб реалізації – програмний модуль.

4. Зміст текстової частини: Вступ. 1. Техніко-економічне та науково-технічне обґрунтування доцільності досліджень. 2. Аналіз методів виявлення фішингових інформаційних ресурсів. 3. Удосканалення методів виявлення фішингових інформаційних ресурсів. 4. Експериментальні дослідження. 5. Економічна частина. Висновки. Список використаних джерел. Додатки.

5. Перелік ілюстративного матеріалу: Порівняння методів виявлення фішингових інформаційних ресурсів (плакат, А4). Аналіз практичного застосування методів виявлення фішингових інформаційних ресурсів (плакат, А4). Модель процесу виявлення фішингових інформаційних ресурсів (плакат, А4). Алгоритм роботи засобу (плакат, А4). Результати тестування (плакат, А4).

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанти	Підпис, дата	
		завдання видав	виконання прийняв
1	Лукічов В.В., к.т.н., старш. викл. каф. ЗІ		
2	Лукічов В.В., к.т.н., старш. викл. каф. ЗІ		
3	Лукічов В.В., к.т.н., старш. викл. каф. ЗІ		
4	Лукічов В.В., к.т.н., старш. викл. каф. ЗІ		
5	Лесько О.Й., к.е.н., професор каф. ЕПВМ		

7. Дата видачі завдання 9 вересня 2021 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	01.09.2021 – 04.09.2021	
2	Розробка технічного завдання	05.09.2021 – 15.09.2021	
3	Аналіз інформаційних джерел за напрямком магістерської кваліфікаційної роботи	16.09.2021 – 22.09.2021	
4	Розробка рішень	23.09.2021 – 04.10.2021	
5	Практична реалізація, моделювання, експериментування, результати	05.10.2021 – 24.10.2021	
6	Розробка розділу економічного обґрунтування доцільності розробки	25.10.2021 – 17.11.2021	
7	Аналіз виконання ТЗ, висновки	18.11.2021 – 24.11.2021	
8	Оформлення пояснювальної записки	25.11.2021 – 30.11.2021	
9	Попередній захист та доопрацювання МКР	03.12.2021	
10	Представлення МКР до захисту	20.12.2021	
11	Захист МКР	21.12.2021	

Студент _____ С.А. Думчиков

Керівник роботи _____ В.В. Лукічов

АНОТАЦІЯ

УДК 004.056

Думчиков С. А. Метод і засіб виявлення фішингових інформаційних ресурсів. Магістерська кваліфікаційна робота зі спеціальності 125 – Кібербезпека, освітня програма – Безпека інформаційних і комунікаційних систем. Вінниця: ВНТУ, 2021. 95 с.

На укр. мові. Бібліогр.: 23 назв; рис.: 54; табл. 14.

Магістерська кваліфікаційна робота присвячена розробці методу та засобу для визначення фішингових інформаційних ресурсів. Для успішної розробки програмного засобу проведено дослідження основних методів визначення фішингових інформаційних ресурсів, розроблено власний метод визначення – метод визначення на основі бази даних справжніх URL-адрес та за допомогою машинного навчання. Ці методи виправляють основні недоліки вже існуючих методів визначення фішингових інформаційних ресурсів. В роботі обґрунтовано вибір власних методів, розроблено ряд схем і алгоритмів, здійснено програмну реалізацію. Засіб перевірено на коректність роботи, доведено ефективність виявлення фішингових ресурсів.

Графічна частина складається з 6 плакатів з демонстрацією результатів моделювання і проведених досліджень.

В економічному розділі оцінено витрати на розробку.

Ключові слова: програмний засіб, фішингова URL-адреса, фішинговий веб-сайт.

ABSTRACT

Dumchykov S. Method and means for detecting phishing information resources. Master's thesis in specialty 125 – Cybersecurity. Vinnytsia: VNTU, 2021. – 95 p.

In Ukrainian language. Bibliographer: 23 titles; fig.: 54; tabl.: 14.

Master's thesis is devoted to developing a means for identifying phishing information resources. For the successful development of software tool, research has been carried out on the main methods for identifying phishing information resources, and a proprietary method has been developed – a method for determining based on database real URLs and using machine learning. These methods correct the main disadvantages of existing methods for identifying phishing information resources. In the work was justified the choice of own methods, a number of schemes and algorithms were developed, software implementation was carried out. The tool has been tested for correct operation, and the effectiveness of detecting phishing resources has been proven.

The graphical part consists of 6 posters with simulation results.

The economic part estimates the development costs.

Keywords: software, phishing URL, phishing website.

ЗМІСТ

ВСТУП

Постановою Кабінету Міністрів України від 11 березня 2020 року №211 «Про запобігання поширенню на території України гострої респіраторної хвороби COVID-19, спричиненої коронавірусом SARS-CoV-2» пунктом 1 постановлено установити з 12 березня 2020 р. до 22 травня 2020 р. на всій території України карантин. З того моменту починає стрімко рости потреба у використанні комп'ютеру та інтернету для виконання щоденних задач та потреб. Відповідно, зростає попит на електронні фінанси, соціальні медіа, SaaS (програмне забезпечення як послуга), доставку товарів. Відбувається це все не завжди у офіційному додатку відповідного сервісу через те, що такого додатку могло і не бути на той час, люди не знали про його існування чи не хотіли заморочуватись над пошуком цього додатку аби скористатися його послугами раз в місяць. Через це збільшується активність у користування веб-сайтів з настільних чи мобільних пристроїв. На фоні цього підвищується шахрайська активність, щодо наживи на людях, які не приділяють уваги, яким ресурсом вони користуються чи бажанні людей до швидкого заробляння грошей легкими шляхами. Тому **актуальною** задачею є розробка методу і засобу визначення фішингових інформаційних ресурсів для зменшення страждань і матеріальних збитків користувачів, що використовують інтернет.

Засіб визначення фішингових інформаційних ресурсів дозволить зекономити час на пошук легітимного сервісу, підвищить безпеку даних, що зберігаються на пристрої користувача шляхом непотрібності їх використання на легітимному ресурсі та зменшить економічні збитки за таким же принципом.

Значний внесок у розвиток методів визначення фішингових інформаційних ресурсів зробили такі вчені: Алмомані А., Бергольц А., Паас Г., Стробель С., [1, 2, 3].

Об'єктом дослідження є процес розпізнавання фішингових інформаційних ресурсів.

Предметом є програмні засоби виявлення фішингових інформаційних ресурсів.

Метою магістерської кваліфікаційної роботи є покращення точності розпізнавання фішингових інформаційних ресурсів шляхом створення покращеного методу розпізнавання.

Для досягнення мети необхідно:

- проаналізувати відомі методи визначення фішингових інформаційних ресурсів;
- проаналізувати програмні засоби та сервіси визначення фішингових інформаційних ресурсів, що зараз представлені;
- розробити метод класифікації URL-адрес;
- розробити програмний засіб для визначення фішингових інформаційних ресурсів на основі вищезапропонованого методу;
- виконати перевірку коректності роботи системи.

Наукова новизна магістерської роботи полягає в тому, що:

- запропоновано удосконалений метод класифікації фішингових ресурсів на основі URL-адрес, що, на відміну від існуючих підходів, дозволяє здійснювати такий процес за семантичними, синтаксичними та прагматичними особливостями (полями) даних адрес.

Публікації результатів магістерської кваліфікаційної роботи. Результати магістерської роботи доповідалися на таких конференціях:

- Міжнародна наукова інтернет-конференція "Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 62)" / Збірник тез доповідей: випуск 62 (м. Тернопіль, 12 жовтня 2021 р.);
- Всеукраїнська науково-практична інтернет-конференція студентів аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи» (МН-2022).

За результатами магістерської кваліфікаційної роботи опубліковано дві тези у збірниках матеріалів конференції [4, 5].

1 АНАЛІЗ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1.1 Науково-технічне обґрунтування розробки методу і засобу виявлення фішингових інформаційних ресурсів

Фішинг – це популярна форма атаки соціальної інженерії, при якій зловмисник обманює жертву, видаючи себе за іншу особу. Фішер – це особа, яка обманом примушує користувачів Інтернету розкрити йому особисту чи конфіденційну інформацію, яку потім можна використати незаконно. Електронні листи і повідомлення зі шкідливими вкладеннями або отруєними URL-адресами, що перенаправляють на шкідливі веб-сайти, є одними з найпоширеніших векторів атак, які використовуються при фішингу. Технологічний прогрес надав фішерам більш досконалі інструменти для запуску небезпечних і витончених атак. У звіті Phishlabs про тенденції в області фішингу за 2018 рік [6] згадується, що цілі фішингових атак перемістилися з приватних осіб на підприємства. Майже половина всіх фішингових веб-сайтів в даний час використовує протокол HTTPS, який був одним з основних індикаторів справжності веб-сайтів [7]. В іншому звіті, опублікованому APWG в першому кварталі 2019 року, говориться, що кількість фішингових атак збільшилася на 30% в порівнянні з попереднім кварталом і що основними цілями були служби «Програмне забезпечення як послуга» (SaaS) і веб-пошта [8].

Фішинг став такою згубною загрозою, що з 2004 року дослідники приділяють все більше уваги боротьбі з ним. Запит «фішинг» на DBLP – сайт з відкритою бібліографією по інформатиці, – показує, що щорічно публікується в середньому близько 55 наукових робіт, присвячених фішинговим атакам і їх виявленню для електронної пошти, веб-сайтів та URL-адрес. Література по фішингу також включає кілька оглядів, наприклад [9], [10], в яких намагаються порівняти ці методи хоча б на теоретичному рівні. Однак дослідження виявлення фішингу піднімає кілька питань, які не отримали належної відповіді в попередніх інформаційних джерелах. Як вибрати найбільш підходящий метод виявлення фішингу в конкретному сценарії? Що в цьому контексті означає гарне рішення?

Більш того, чи є твердження про точність і інші показники, зроблені в дослідницьких роботах, правдивими і реалістичними? Як проблема безпеки, фішинг має кілька параметрів, які потребують особливої уваги для хорошого рішення [11, 12].

За даними опитування, проведеним OLX серед 55 тис. покупців у серпні 2021 року, з 01.01.2020 року по 01.07.2021 року відсоток українців, які звернулися б до правоохоронців через шахрайські дії, впав з 21% до 10%. Водночас 50% стикнулися зі зловмисниками в інтернеті, а 14% втратили кошти через фішинг. Мешканці Київської, Дніпропетровської та Харківської областей найбільше страждають від онлайн-шахраїв. Найчастіше фішинг-атаки відбуваються через популярні месенджери. У період з 01.01.2021 по 01.07.2021 45% з опитаних українців стикалися зі спробами онлайн-шахрайства. В той самий період 2020 року цей показник був 22%. Найчастіше з початку 2021-го атакують жителів великих міст – дві треті випадків. Кіберзлочинці намагаються активніше атакувати жіночу аудиторію, але їх жертвами часто стають і чоловіки (40% чоловіків та 60% жінок). Більше половини шахрайських інцидентів (54%) – продаж вигаданих товарів з передоплатою. На другій позиції (28%) – фішингова атака, коли у месенджерах користувачам надсилали посилання на підробну платіжну форму. На третій позиції (11%) – шахрайство з надсиланням фейкових скріншотів/квитанцій про оплату товару. Після фішингових атак шахраїв у 2021 році під час онлайн-шопінгу 14% жертв втратили свої кошти. Здебільшого (83% випадків) сторонні посилання на шкідливі сайти надходили у месенджери (Telegram, Viber, WhatsApp). 6% жертв отримали фішингове посилання через SMS, через пошту шахраї практично не атакують українців – таких спроб зафіксували лише 2%. У порівнянні зі схожим опитуванням OLX у 2020 році, значно знизилася частка українців, які звернуться до поліції та служби підтримки ресурсу у випадку шахрайства (табл. 1.1). У період з 01.01.2021 по 01.07.2021 6 тис. осіб із 55 тис. опитаних отримували фішингове посилання від шахраїв. 18% українців, які стикалися із шахрайством у 2021 році, не знають жодного способу захисту своїх платіжних даних. А про методи протидії

шахрайству 51% опитаних в цьому році вже дізналися з соцмереж, 31% – від друзів, 18% – з телебачення.

Таблиця 1.1 – Статистика реагування на шахрайські дії

Що ви робите, якщо стикаєтесь із шахрайством в інтернеті? (оберіть декілька варіантів)	2020 рік	2021 рік
Намагаюся розв’язувати свої питання самостійно	35%	41%
Розповідаю близьким	48%	32%
Звертаюся до Служби підтримки онлайн-ресурсу	58%	17%
Звертаюся до Кіберполіції	21%	10%
Не роблю нічого	17%	26%

Майже 40% українців уже знають, що надіслані посилання від малознайомих людей можуть бути шахрайською атакою, тому не відкривають їх і не обговорюють фінансові питання в месенджерах, якщо співрозмовник поводиться підозріло. 20% перевіряє користувача за номером телефону, 14% – уважно вчитується в зміст сторонніх SMS та email-листів, а 10% – перевіряє посилання сайтів, щоб не потрапити на шахрайський сайт-копію.

Багато методів використовують для шахраювання над українцями в інтернеті: пенсіонерів ошукують, наприклад, утричі частіше за інших – через SMS про виграш у лотерею, кожного четвертого підлітка після відкриття фішингового посилання. Мешканці сіл частіше натрапляють на продаж неіснуючого товару по передоплаті, а кияни – на фішингові атаки. Стать, до речі, практично не впливає на вірогідність зустріти шахрая в інтернеті. Такі дані наводить аналітична служба OLX, згідно з результатами опитування 25 тис. користувачів інтернету. Місце проживання значно впливає на тип шахрайств. Наприклад, дві треті жителів селищ міського типу та сіл стикаються зі спробами продати їм неіснуючий товар за передоплатою, а от жителі міст у 1,5 раза частіше отримують фішингові посилання. Жителі райцентрів на третину частіше за інших стикаються з підробленими квитанціями про оплату. У селах в цілому менше обізнаних з базовими правилами кібербезпеки. Якщо понад 42% містян не ведуть спілкування за межами платформи, де здійснюють угоду, та не відкривають посилання від незнайомих людей, то серед мешканців сіл – це

тільки 33% опитаних. Водночас жителі столиці найбільше потерпають від фішингу (36% від всіх видів онлайн-шахрайств у Києві), а от у Дніпрі, Одесі, Харкові та Львові половина випадків пов'язана з передоплатою неіснуючого товару. Найбільш відповідально до звернень у Кіберполіцію ставляться у Львівській області – кожний 8-ий ошуканий подасть заяву, а найменш – в Одеській – тут звернеться тільки кожний 14-ий. Серед всіх вікових категорій 80% випадків онлайн-шахрайств відбуваються у месенджерах (Viber, Telegram, WhatsApp). Пенсіонерам утричі частіше надсилають шахрайські SMS, у сім раз електронні листи про «виграш в лотерею», «нові умови тарифу» чи «правила карантину». Водночас 57% українців віком від 46 до 65 років натрапили на шахраїв, які просили зробити передоплату за неіснуючий товар. Молодь віком 18-25 років у кожному третьому випадку натрапляє на фішинг, а кожний четвертий неповнолітній втратив гроші після переходу за шкідливим посиланням на сайт підробку відомого бренду. Це пов'язано з тим, що молодь більше за старше покоління проводить час в онлайні, але так само слабо обізнана з базовими правилами кібербезпеки. Молодь більш обізнана з базовими правилами кібербезпеки, однак загальний рівень навичок онлайн-поведінки лишається невисоким (табл. 1.2).

Таблиця 1.2 – Статистика обізнаності правилам кібербезпеки

Правила кібербезпеки	до 18 років	після 65 років
Не відкривають посилання від незнайомців	48%	28%
Не розголошують платіжні дані банківської картки	30%	29%
Звертають увагу на оцінки і відгуки	38%	19%
Не обговорюють угоду поза платформою (у сторонніх месенджерах)	39%	38%
Перевіряють посилання на olx.ua	14%	7$
Перевіряють номер продавця/покупця в базах відгуків	26%	21%
Уважно оцінюють текст в SMS чи пошті	22%	14%
Звертають увагу на манеру спілкування (наприклад, чи є психологічний тиск або маніпуляції)	45%	38%

Водночас про кібербезпеку 65% неповнолітніх та 48% пенсіонерів дізнаються з соцмереж, 38% та 23% – від друзів та родичів, а от кожен третій

українець віком від 65 років – з освітніх проєктів. Стать практично не впливає на вірогідність натрапити на шахрая. І чоловіки (78%), і жінки (88%) отримують шкідливі посилання від шахраїв у Viber, Telegram чи WhatsApp. Чоловікам удвічі частіше за жінок надходять шахрайські SMS (виграш в лотерею, «ви виграли авто») – 9% проти 4% від всіх випадків відповідно.

За даними звіту PhishLabs за третій квартал 2021 року [13], кількість фішингових сайтів, виявлених у 2021 році, перевищила 2020 рік майже на 32% (рис. 1.1)

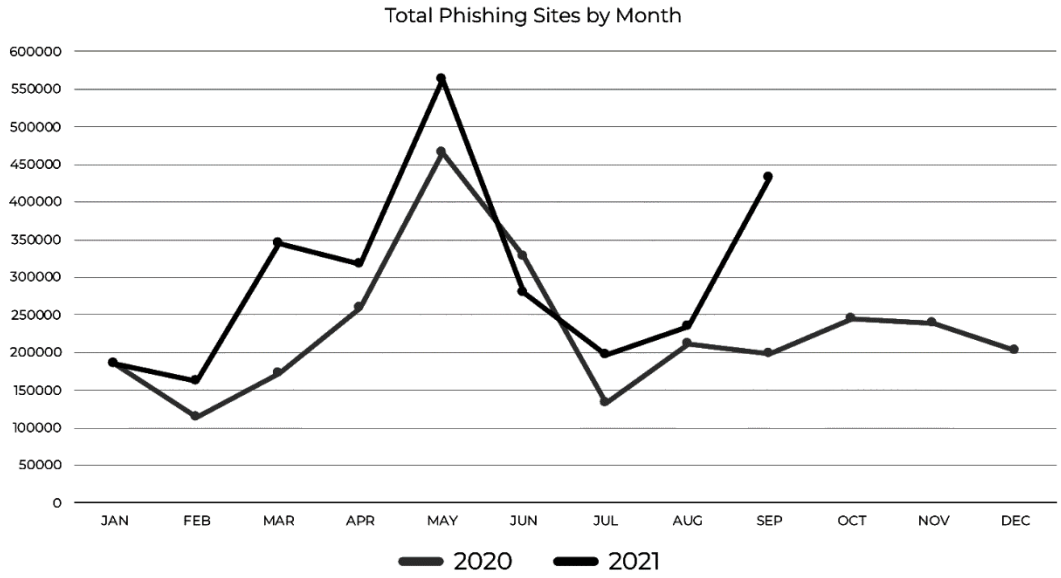


Рисунок 1.1 – Графік кількості фішингових сайтів у 2020 і 2021 роках

У третьому кварталі були помітні коливання в шести найбільших галузях, які зазнали фішинговим атакам. Фінансовий сектор продовжував залишатися найбільш цільовою галуззю у третьому кварталі і кількість фішингових атак зросла на 5,8%. У той час як компанії соціальних медіа зберегли друге місце, галузь продовжила зниження від найбільш цільових у першому кварталі, зі скороченням атак на 11,9% у третьому кварталі. Незважаючи на те, що соціальні медіа зазнали менше фішингу, облікові записи, пов’язані з системою єдиного входу для вторинних облікових записів, включаючи соціальні медіа, веб-пошту та онлайн-сервіси та електронну комерцію, все ще зазнали 37% фішингових атак (рис. 1.2) [13].

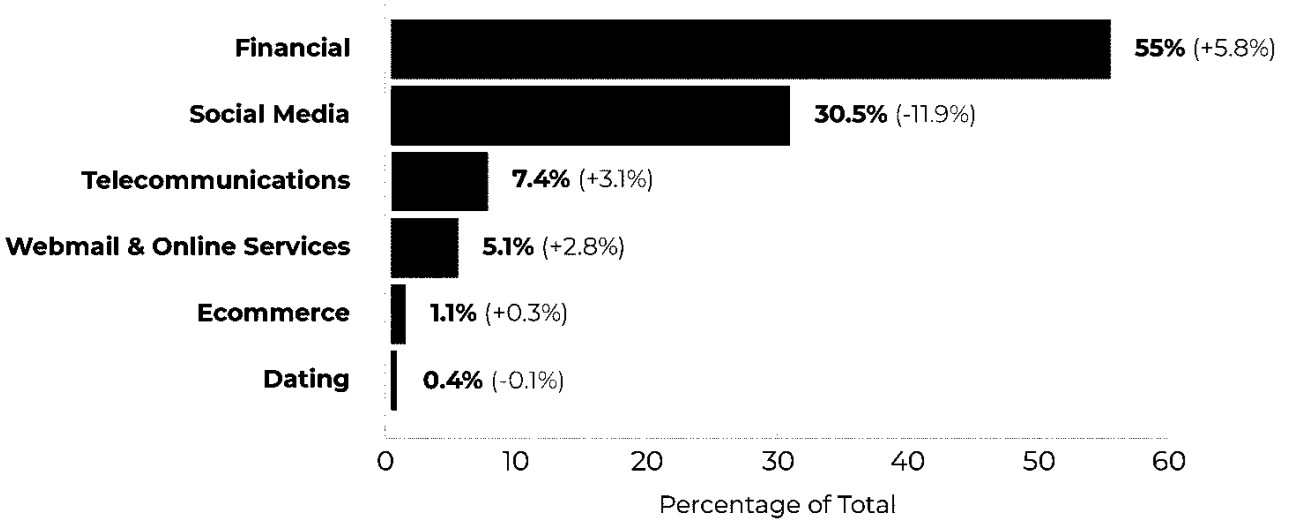


Рисунок 1.2 – Графік найбільш атакованих галузей фішинг-атаками

У третьому кварталі загрози продовжували покладатися на різноманітні методи для інсценування фішингових сайтів. У той час як незначна більшість (51%) фішингових сайтів продовжувала інсценуватися шляхом зловживання безкоштовними послугами та інструментами, у третьому кварталі відбувся відхід від цієї діяльності, оскільки учасники все частіше використовували скомпрометовані сайти для інсценування фішингових сайтів. Також зростало зловживання послугами платної реєстрації домену. Незважаючи на те, що зловживання безкоштовними послугами для інсценування фішингових сайтів продовжувало бути сильним, усі п’ять категорій безкоштовних послуг зазнали спаду активності порівняно з другим кварталом (рис. 1.3) [13].

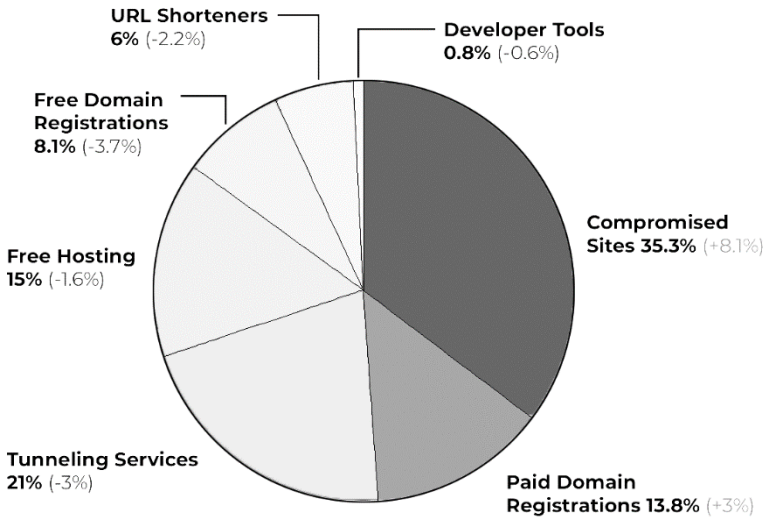


Рисунок 1.3 – Графік способів створення фішингових сайтів

Понад 65% усіх фішингових шахрайств, про які повідомлялося в третьому кварталі, використовували застарілі загальні домени верхнього рівня. Лідером став .com, який зріс на 15,2% у порівнянні з 2 кварталом і спричинив майже 55% усіх зловживань доменами верхнього рівня (далі – TLD, Top Level Domain). Код країни (далі – ccTLD) .ca також зазнав значного зростання в третьому кварталі, склавши 10,3% від загальної кількості TLD, якими зловживали. Зокрема, ccTLD .mx, .uz, .monster та .ae випали з 10 найкращих. Крім того, популярні раніше ccTLD, які пропонують безкоштовну реєстрацію доменів (.ml, .tk, .cf, .ga та .gq) зникли із списку найбільш популярних до використання. Хоча жоден з них не потрапив до першої десятки, вони все ще залишаються в 10% найбільших доменів верхнього рівня, якими зловживають під час фішингових атак. Разом ці безкоштовні домени становили 2,3% усього фішингу в третьому кварталі (рис. 1.4) [13].

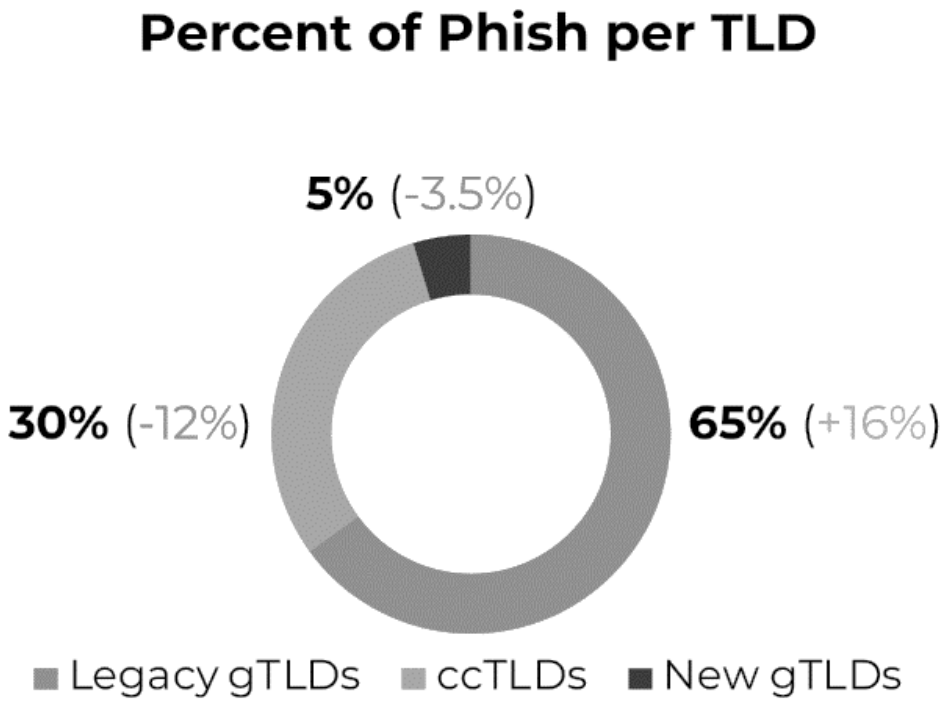


Рисунок 1.4 – Графік типів доменів, які використовували для фішинг-атак

У таблиці 1.3 наведено перші десять доменів серед найуживаніших за третій квартал 2021 року; Legacy gTLD – успадкований загальний домен верхнього рівня, ccTLD – національний домен верхнього рівня (домен країни), new gTLD – новий загальний домен верхнього рівня.

Таблиця 1.3 – Перші десять доменів, які використовують для фішингу

TLD	Тип	% Фішингу	+/-
.COM	Legacy gTLD	54,9%	+15,2%
.CA	ccTLD	10,3%	+6,2%
.ORG	Legacy gTLD	5,7%	-0,2%
.NET	Legacy gTLD	3,2%	-
.KE	ccTLD	1,7%	+1,6%
.IO	ccTLD	1,2%	-2,0%
.XYZ	new gTLD	1,2%	-0,2%
.LY	ccTLD	1,2%	+1,1%
.CO	ccTLD	1,1%	-1,3%
.US	ccTLD	1,0%	+0,7%

Фішингові сайти починали своє існування із найпростіших інтерфейсів для користувача. Наприклад, переказ грошей з карти на карту з комісією менше, ніж на ресурсі офіційного постачальника послуг. При створенні таких сайтів зазвичай використовується виділена IP-адреса. Для прикладу, на адресі 185.136.157.52 зареєстровано 1000 доменних імен (станом на 15:54 15.09.2021).

Категорії брендів для фішингових атак:

- маркетплейси та сайти об’яв;
- поштові та сервіси доставки;
- ресурси оренди нерухомості;
- торгові мережі;
- сервіси по перевезенню пасажирів;
- фінансові організації;
- електронні сервіси державних послуг;
- мережі піцерій;
- курортні готелі;
- соціальні мережі та месенджери;
- онлайн-сервіси;
- онлайн-ігри;
- онлайн-магазини;
- онлайн-послуги;
- ресурси, що продають відповідями до тестів/екзаменів;

Через карантинні обмеження споживання інтернетом зросло на 70%, користування стрімінговими платформами (платформи, що безперервно передають дані користувачам через інтернет) на 12% [14]. Режим самоізоляції збільшив попит на покупки через інтернет і послуги кур'єрської доставки. Тим часом фішинг, як одна з простих схем заробітку, привернув увагу великої аудиторії, що позбулася доходів. З'явилися шахрайські проекти з моделлю Phishing-as-a-Service (фішинг як послуга). Глибокі навички програмування стали менш потрібними, щоби розгорнути інфраструктуру для масштабної фішингової атаки. Шахраї швидко адаптувалися під пандемію і почали проводити фішингові атаки на бренди та сервіси, до яких раніше не мали інтересу. З'явилися спільноти, які пропонують допомогу в навчанні як правильно спілкуватися з жертвами, інколи, вони можуть пропонувати свій список жертв.

Ієрархія шахрайських груп являє собою умовну піраміду: адміністратори, працівники, ті хто дзвонять. Адміністратори відповідальні за набір нових учасників, створення шахрайських сторінок і нових облікових записів, забезпечення допомоги, коли банк блокує карту одержувача чи транзакцію. Адміністратори можуть отримувати 20-30% вкраденої суми. Працівники спілкуються з жертвами і надсилають їм фішингові посилання. Найбільш успішні працівники просуваються по кар'єрній драбині і стають впливовими учасниками проекту. Отримують 70-80% вкраденої суми. Ті хто дзвонять видають себе за спеціалістів технічної підтримки. Отримують 5-10% вкраденої суми [15].

Раніше зловмисники припиняли свої дії після блокування шахрайських ресурсів і перемикались на інші бренди; сьогодні – вони моментально виводять нові фішінгові сторінки на заміну заблокованим. Під кожний бренд використовується окремий бот, використовуються telegram-боти для формування підроблених посилань на основі посилання-оригіналу.

Використання одноразових посилань на фішинговий сайт: жертва фішингу отримує унікальну адресу сторінки, яка стає неактивною після першого відкриття.

Фішинговий вміст не віддається користувачу, якщо User-agent його пристрою відображає, що користувач не являється цільовою аудиторією. Наприклад, якщо проводиться фішингова атака на користувачів мобільних телефонів, у такому випадку відвідувачам ресурсу з браузером для настільного комп'ютеру фішинговий контент віддаватись не буде.

Злочинці використовують бази GeoIP. Наприклад, якщо фішингова сторінка направлена на користувача із України, то користувачам із США вона віддаватись не буде.

При невідповідності заданим параметрам, замість відображення фішингового контенту відвідувачі перенаправляються на офіційний сайт бренду, що атакується.

Злочинці намагаються одразу отримати фіксовану суму грошових коштів, а не доступ до рахунку атакованої жертви. Гроші можна використати відразу, у той час як доступ до рахунку більш ризикований і потребує додаткових дій.

У більшості фішингових ресурсів схожі ознаки:

- фішингові сторінки являються копією атакованої сторінки (копією на рівні коду, зі всіма динамічними елементами);
- сторінки оплати у різних фішингових ресурсів ідентичні (можливо, відрізняються логотипами).

Шахраям, що використовують маркетплейси та дошки оголошень байдуже чим займається площадка – орендою нерухомості чи продажом велосипедів. Для них важливо, щоб на ресурсі сервісу була:

- можливість внутрішньої комунікації між користувачами;
- «безпечні» угоди всередині самого сайту.

Це дозволяє підмінити кінцеве посилання на співзвучному домені для переказу коштів. В даному випадку, співзвучний домен – це домен, що виглядає як справжній домен бренду чи сервісу.

1.2 Аналіз відомих методів

Існують декілька основних схем шахраювання [15]. Схема перша: шахрай представляється продавцем і стимулює покупця перейти по фішинговому посиланню і заплатити за неіснуючий товар:

1. Шахрай розміщує на маркетплейсі оголошення-приманку з популярним товаром по вигідній ціні.
2. Знайшовши потенційного покупця, під будь-яким приводом пропонує йому змінити місце спілкування в месенджер для подальшого обговорення питання купівлі.
3. Шахрай стимулює покупця до проведення «Безпечної угоди». Якщо на маркетплейсі сервіс «Безпечної угоди» відсутній, переконує покупця в існуванні або появі послуги утримання коштів.
4. Шахрай відправляє посилання на фішингову сторінку «Безпечної угоди», де потрібно ввести дані карти для здійснення безпечної угоди. Сторінка за посиланням схожа на стандартну форму «Безпечної угоди» маркетплейсу.
5. Жертва, що атакується оплачує покупку на фішинговій сторінці, здійснивши p2p-переказ (переказ коштів з картки відправника на картку одержувача) на карту дропу. Далі кошти транзитом пересилаються по вибраній схемі переведення в готівку.
6. Шахрай реалізує схему повернення грошових засобів. Для повернення списаних грошових коштів шахрай/фіктивна служба підтримки маркетплейсу надсилає покупцю посилання на фішингову сторінку, де покупець вводить дані карти і суму попереднього списання, але насправді сума списується з карти ще раз. Сторінка повернення – це та сама сторінка оплати, але текст на кнопці змінений з «оплатити» на «повернути».

Схема друга: шахрай представляється покупцем і стимулює продавця перейти по фішинговому посиланню і, ввівши реквізити платіжної карти, отримати неіснуючу передоплату. Послідовність дій наступна:

1. Шахрай знаходить на маркетплейсі продавця. Представляється потенційним покупцем, контактує з ним в мессенджері і повідомляє, що готовий купити товар за передоплатою.
2. Шахрай говорить, що оформив покупку через сервіс «Безпечна угода». Якщо на маркетплейсі «Безпечна угода» відсутня, переконує продавця в існуванні або появі послуги утримання коштів.
3. Для отримання коштів надсилає продавцю посилання на підробну сторінку отримання передоплати, де продавцю потрібно ввести реквізити своєї банківської картки і код підтвердження. Якщо продавець має сумніви, то для маніпуляції шахрай надсилає йому підробні правила роботи послуги.
4. Коли продавець вводить реквізити своєї картки, шахрай отримує доступ до його рахунку.

Приклад фішингової сторінки, де шахрай представляється покупцем зображено на рисунку 1.5.

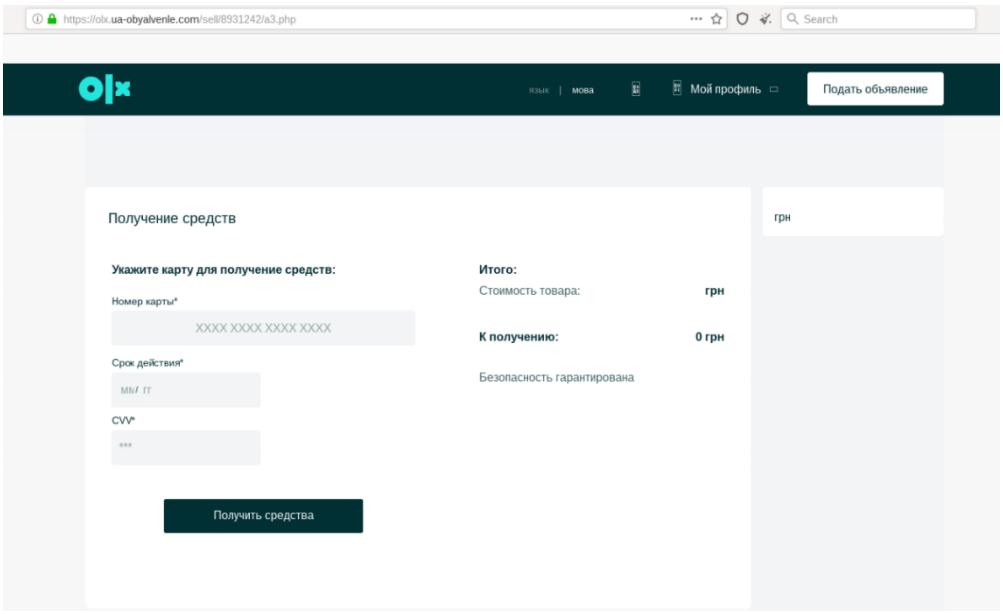


Рисунок 1.5 – Вигляд фішингової сторінки, створеної для передоплати товару

Приклади фішингових посилань, що виглядають як olx.ua: olx-ua.primepay24.ru, olx-ua.pay-vax.ru, olx-ua.pay-transfer.tk, olx-ua.pay-seruse.space,

olx-ua.pay-seruse.live, olx-ua.paysend24.com, olx-ua.pay-reg.xyz, olx-ua.pay-payment.space, olx-ua.payment-3ds.ga, olx-ua.pay-id62856.xyz, olx.ua.pay-get.online, olx-ua.pay-fsb.tk, olx-ua.paybank-3ds.gq, olx-ua.oplata-rf-2dc.cf, olx-ua.oplata-online.tk, olx-ua.oplata-mail.ml, olx-ua.oplata-inf24.tk, olxua.oplata-fcb.tk, olx-ua.oplatadostavka.online, olx-ua.eccomerce.ml, olx-ua.3dc-payment.online, olx-ua.ru, olx-ua-pay-transfers.tk, olx-ua.money-send.site, olx-ua.best, olx-ua.transfer-3ds.ru, olx-ua.send-order.ru, olx-ua.reg-pay.xyz, olx-ua.nsk-pay.xyz, olx-ua.msk-pay.xyz, olx-ua.lastpay24.com, olx-ua.idd55005.com, olx-ua.id827232.ml, olx-ua.id-41145.ml, olx-ua.id-41452.tk, olx-ua.id-41423.tk, olx-ua.id-2454.tk, olx-ua.id-24155.ml, olx-ua.free-pay.ru, olx-ua.ez-pays.ru, olx-ua.e-pay.cf, olx-ua.bankoplata.site, olx-ua.3ds-transition.xyz, olx-ua.3ds-transition.space, olx-ua.3dc-payment.ru, olx-ua.pay-seruse.xyz.

Приклади фішингових посилань, що виглядають як izi.ua: izi-ua.oplata-online.tk, izi-ua.id827232.ml, izi-ua.id-24155.ml, izi.3dc-payment.online, izi-ua.world-pay.ml, izi-ua.pay-transfer.tk, izi-ua.payment-3ds.ga, izi-ua.paybank-3ds.gq, izi-ua.eccomerce.ml, izi-ua.3dsx.su, izi.transfer-3ds.ru, izi-ua.gq.

Потреба під час пандемії послуг доставки призвела до вибухового зросту популярності «Кур’єрської схеми».

Кур’єрська схема №1:

1. Шахрай розміщує «лот-приманку» – об’яву про продаж популярного товару по заниженій ціні.
2. Після того, як покупець виходить на контакт, йому пропонується змінити місце обговорення покупки і доставки у месенджер.
3. В месенджері шахрай запитує у жертви ПІБ, адресу і номер телефону під приводом оформлення доставки через кур’єрську службу.
4. Шахрай присилає покупцю посилання на фішинговий ресурс, який повністю копіює сторінку популярної кур’єрської служби. За посиланням будуть представлені вказані раніше дані жертви, які вона нібито повинна повторити і після цього у тому ж місці оплатити.

5. Частину жертв обманюють повторно – ошукують на повернення коштів. Через деякий час після оплати товару покупцю сповіщають, що на «пошті» сталася надзвичайна ситуація. Легенда може бути будь-якою, наприклад, співробітник пошти нібито спійманий на крадіжці, а замовлений товар конфіскувала поліція, тому для компенсації перерахованої суми необхідно оформити «повернення коштів». В результаті з карти жертви виконується повторне списання такої ж суми.

Кур’єрська схема №2:

1. Шахраєм виконується пошук оголошень про продаж товарів на маркетплейсі. Обов’язкова умова для оголошення – продавець дозволив з собою зв’язуватись по мобільному телефону, обравши опцію показати свій номер телефону усім бажаючим.
2. Шахрай виходить на контакт з продавцем через чат стороннього сервісу, минаючи безпечний чат маркетплейсу і проявляє бажання придбати виставлений товар.
3. Шахрай стверджує, що зараз же оформить «покупку з доставкою кур’єрської служби», тим часом генеруючи за допомогою автоматизованого телеграм-боту фішингову сторінку кур’єрської служби, використовуючи реальну назву, фото і ціну лоту.
4. Шахрай пересилає готову фішингову сторінку жертві-продавцю в чат месенджеру, пояснюючи, що вже все сплатив і пропонує отримати грошові кошти. Для отримання коштів продавець вводить реквізити своєї платіжної картки і код підтвердження.

Приклад фішингової сторінки, де шахрай використовує кур’єрську схема зображено на рисунку 1.6.

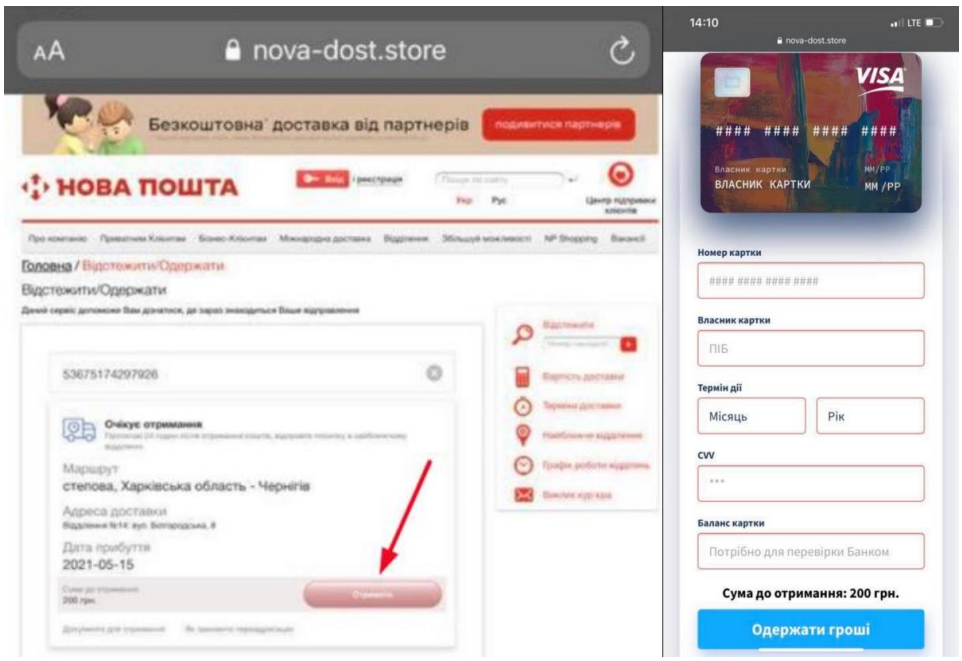


Рисунок 1.6 – Вигляд фішингової сторінки для кур’єрської схеми

Приклади фішингових посилань, що маскуються під бренд «Нова Пошта»: novaposhta.reg-pay.xyz, novaposhta.pay-seruse.xyz, novaposhta.pay-seruse.space, novaposhta.pay-reg.xyz, novaposhta.oplata-inf24.tk, novaposhta.idd55005.com, novaposhta.e-pay.cf.

Шахраї під виглядом орендодавців і орендаторів атакують користувачів дошок оголошень, використовуючи стандартні схеми обмана користувачів маркетплейсів і перенаправляють їх на підроблені сайти оренди квартир, що схожі на офіційні ресурси оренди нерухомості, де мотивуючи переказати/отримати передоплату, яка у випадку оренди житла може бути достатньо великою. Причини переводу уваги шахраїв на оренду житла ситуативні: високий сезон оренди житла, рекламна кампанія сервісу, конкуренція між злочинними угрупуваннями.

Як працюють шахраї у сфері оренди нерухомості:

1. Переводять розмову у месенджер. Шахрай зв’язується з орендодавцем під виглядом клієнта і бронює квартиру. Після цього пропонує перейти у месенджер. Варіант 1: шахрай в месенджері надсилає посилання нібито на сайт аренди нерухомості, де можна отримати оплату за бронювання. Варіант 2: шахрай надсилає смс від сайту повідомлення, а

в ньому – те саме посилання на нібито сайт, де можна отримати оплату за бронювання.

2. Маскують фішинговий сайт під відомий сайт оренди нерухомості. Посилання від шахрая веде на сайт, зовні дуже схожий на сайт оренди нерухомості, що імітує систему переказу коштів.
3. Маскують списання коштів під зарахування. Шахрай просить орендодавця ввести повні реквізити карти: номер, термін дії, CVV-код – нібито для зарахування коштів. Шахраї запевняють, що це вимоги банку, умови платіжної системи або особливість переказу.
4. Імітують чат з техпідтримкою. На фішинговому сайті може бути встановлений онлайн-чат з помічником. Це зроблено з розрахунком на те, що людина отримає повідомлення про списання коштів, запідозрить неладне і звернеться за допомогою. В онлайн-чаті шахрай запевняє, що для отримання коштів дійсно потрібно ввести повні реквізити і код із смс, а «списання» означає «зарахування».

Шахрай розміщає на дошці об’яв пропозицію великої знижки за дорогоцінні товари певної торгової мережі, представляючись її співробітником, і пропонує оплатити товар за допомогою своєї корпоративної знижки. Потенційному покупцю в месенджері розповідають, як оплатити товар через «кошик співробітника», що розміщенний на фішинговій сторінці, дуже схожій на сайт торгової мережі. В підсумку клієнт залишається без грошей і без покупки.

Як відбувається шахраювання через дошку оголошень:

1. Шахраї розміщують на маркетплейсі оголошення з пропозицією великої знижки за дорогоцінні товари. В описі вказується: «Нові товари з XXXX» (XXXX – назва торгової точки). Подробиці в ЛС».
2. Шахраї переводять у мессенджер всіх, хто цікавиться товаром. В месенджері представляються співробітниками торгової мережі і пропонують оплатити товар за допомогою своєї корпоративної знижки, а невелику націнку забрати собі при зустрічі і видачі товару.

3. В месенджері висилають зацікавленим покупцям посилання на фішингову сторінку, схожу на сайт торгової мережі і розповідають, як оплатити товар через «кошик співробітника», який розміщений на фішинговій сторінці. В підсумку покупець залишається без грошей і без покупки.

Шахраї реєструються в спільноті попутників, пропонують підвезти жертву і вимагають передоплату через фішинговий сайт.

Порядок дії, за якими працюють шахраї у сервісах перевезення:

1. Шахраї розміщують в сервісі по перевезенню пасажирів оголошення про вільні місця в автомобілі, оголошення нічим не відрізняються від справжніх і навіть ціни вказані у межах норми.
2. Коли користувач відгукується на оголошення, шахраї в чаті сервісу по перевезенню пасажирів просять його зв’язатися з ними у месенджері і відправляють номер телефону.
3. Шахраї охоче підтримують розмову про цікаві пасажиру подробиці і навіть самі задають доречні у контексті поїздки уточнюючі питання.
4. Коли розмова заходить про оплату поїздки, жертві пропонують «купити білет» по фішинговому посиланню, що нібито є сервісом по перевезенню пасажирів, пояснюючи, що «офіційні водії» готівкові гроші не підтримують або переконуючи жертву бонусами і кешбеком, які водій отримає при оплаті через сайт.
5. Якщо користувач переходить по фішинговому посиланню і вводить дані платіжної картки, то діло не обмежується лише втротою суми передоплати.

Приклад на рисунку 1.7. Так як політика сервісу BlaBlaCar забороняє пересилати у повідомленнях контактні дані, «водій» пише номер словами.

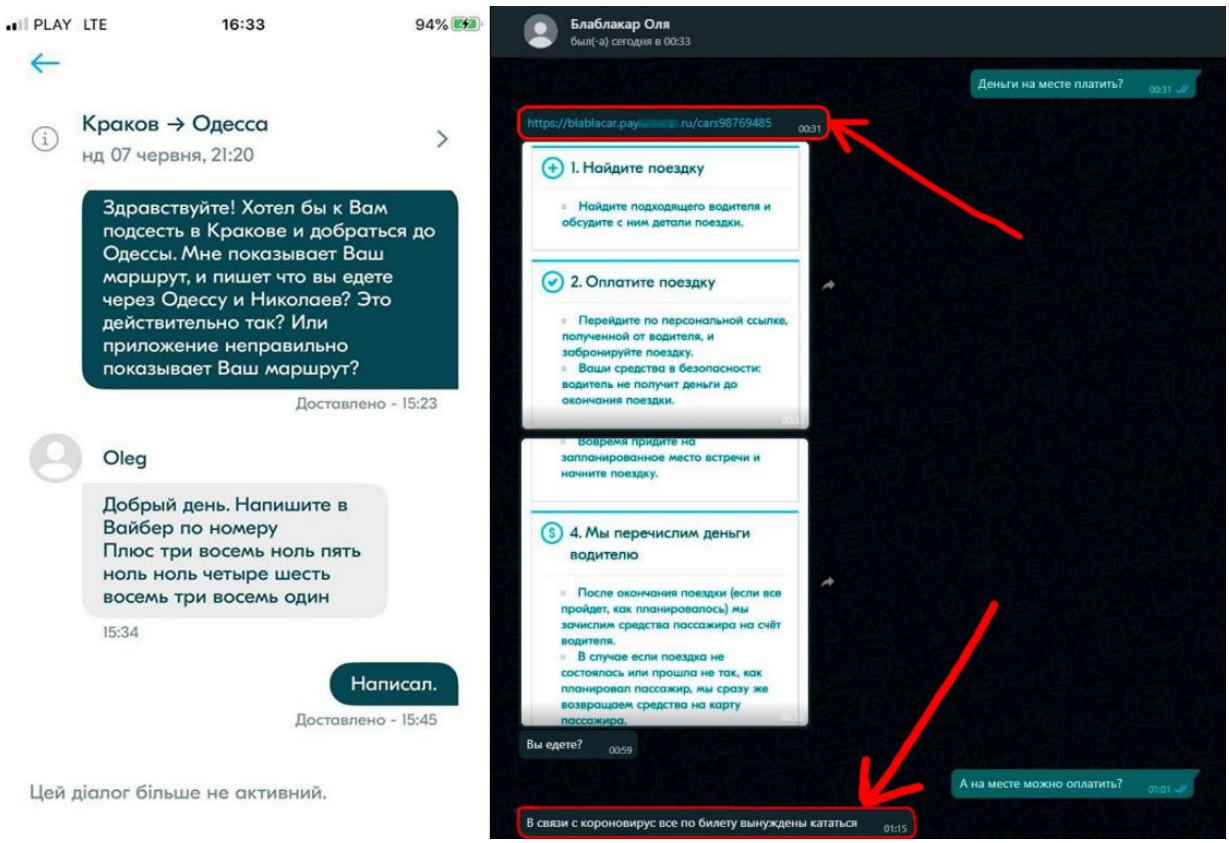


Рисунок 1.7 – Вигляд фішингової сторінки для кур’єрської схеми

Приклади фішингових посилань, що маскуються під бренд сервіс BlaBlaCar: bot.blablacar-oplata24.ru, blablacar.world-pay.ml, blablacar-ua.transfer-info.site, blablacar-ua.pay-payment.space, blablacar-ua.payment12783073.ru, blablacar-ua.nsk-pay.xyz, blablacar-ua.idd55005.com, blablacar-ua.id-24155.ml, blablacar-ua.3ds-transition.space, blablacar-ticket-pay.ru, blablacar.servis-payment.ru, blablacar.send-order.ru, blablacar.secutiry-service.ru, blablacar.pay-id62856.xyz, blablacar.pay-go.tk, blablacar.paybank-3ds.gq, blablacar.oplata-rf-2dc.cf.

Перенос даних в цифровий формат, а також розвантаження пунктів обслуговування населення викликав збільшення кількості користувачів сервісів «держава у смартфоні», де можна виконати певні адміністративні дії, заплатити автомобільні штрафи, зареєструвати народження дитини, пред’явити паспорт та інші можливо. Це також викликало інтерес шахраїв, наприклад для збору чутливих даних про користувачів. Для прикладу, на рисунку 1.8 зображено вигляд змісту сайту https://diia-gov.pp.ua/ станом на 27.05.2021.

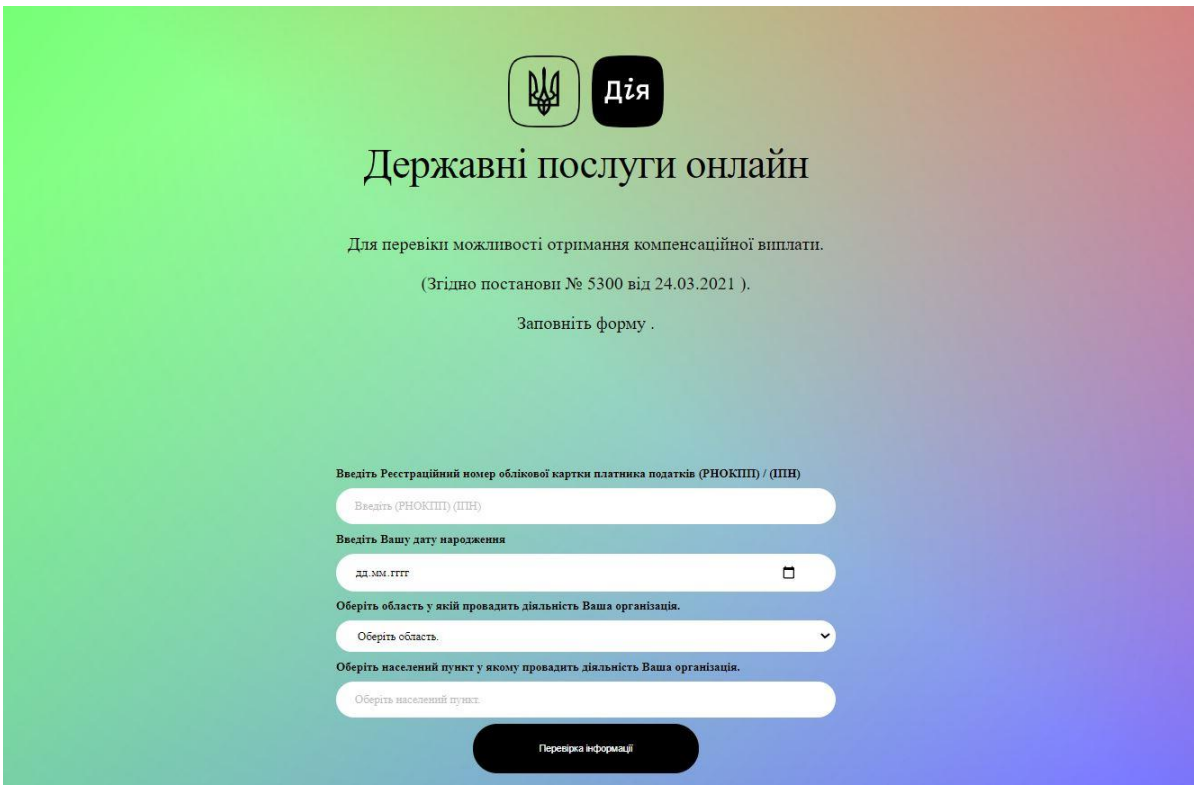


Рисунок 1.8 – Вигляд сторінки, замаскованої під сервіс «Дія»

4 грудня 2020 року Верховною Радою України був прийнятий закон проекту №4429 та підписаний Президентом України закон «Про соціальну підтримку застрахованих осіб та суб'єктів господарювання на період здійснення обмежувальних протиепідемічних заходів, запроваджених з метою запобігання поширенню на території України гострої респіраторної хвороби COVID-19, спричиненої коронавірусом SARS-CoV-2», згідно якого фізичні особи-підприємці(ФОПі) і наймані працівники можуть отримати одноразову матеріальну допомогу у застосунку для смартфонів та на порталі Дія. Після цього активізувалися шахраї, які побачили, що можна використати цю ситуацію на свою користь. Схема заключається у тому, що жертві потрібно ввести дані своєї банківської картки, як було показано у прикладах раніше, після чого кошти списувались на карту дропа чи інший підставний рахунок. Приклад фішингової сторінки, що імітує сторінку повернення 8000 гривень сервісу «Дія» зображено на рисунку 1.9. Дата фіксації URL-адреси https://diia8000gov.me/ 13 квітня 2021, станом на 01 вересня 2021 домен непрацюючий.

За час пандемії затребуваність відеоігор значно виросла. Загальна кількість активних гравців по всьому світі приближається до 3 млрд осіб, при цьому все більше користувачів під’єднує до Steam свої робочі пристрої.

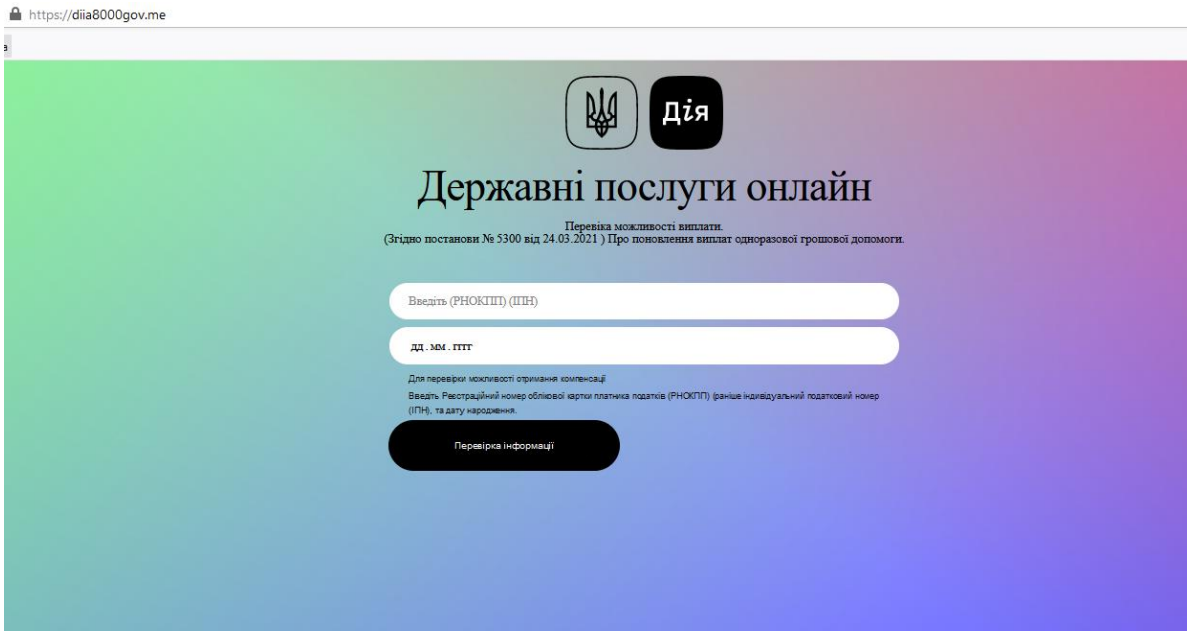


Рисунок 1.9 – Вигляд сервісу, що маскується під «Дія», що виплачує 8000 грн

Багато онлайн-ігор включають внутрішньоігрову валюту, внутрішньоігрові покупки і реальну валюту, що зберігається в гаманцях, тому вони представляють собою мішень для шахраїв. Крім виманювання облікових записів гравців, шахраї полюють за обліковими записами в соціальних мережах і обліковими даними користувачів Steam. Також шахраї мотивують користувачів здійснювати транзакції для покупок та винагороджень, яких вони не отримають.

Частіше за інших зловмисники використовують в якості приманки: Counter-Strike:Global Offensive (CS:GO), PUBG, Битву титанів, Brawl Stars, ігрові платформи Garena і створюють фішингові сторінки, імена яких експлуатують ігрову тематику. Шахрайські сайти маскуються під безкоштовні версії, оновлення, розширення для популярних ігор, чітерські програми.

Підхід «Freemium» особливо поширений в ігровій індустрії. Розробники та видавці ігор зазвичай пропонують користувачам незначні, але по-справжньому безкоштовні подарунки – очікуючи, що гравці будуть втягнуті і в кінцевому

підсумку, витратять їх на внутрішньоігрові покупки. Залежність від сиру «Freemium» – це те, чим користуються шахраї, заманюючи жертв на фішингові сторінки.

1.3 Аналіз засобів-аналогів для виявлення фішингових ресурсів

Для перевірки аналогів буде використовуватись фішинговий сайт https://atb-market.space/, що станом на 23.09.2021 є активним. Даний сайт маскується під мережу продуктових магазинів «АТБ», де після проходження опитування Ви можете виграти 5000 грн. Вигляд сайту https://atb-market.space/ зображено на рисунку 1.10.

Під час пошуку аналогів були виявлені такі:

1. Kaspersky Threat Intelligence Portal;
2. ScanURL;
3. PhishTank;
4. Google Transparency Report;
5. Онлайн-перевірка посилань за допомогою Dr.Web;
6. Norton Safe Web від Symantec;

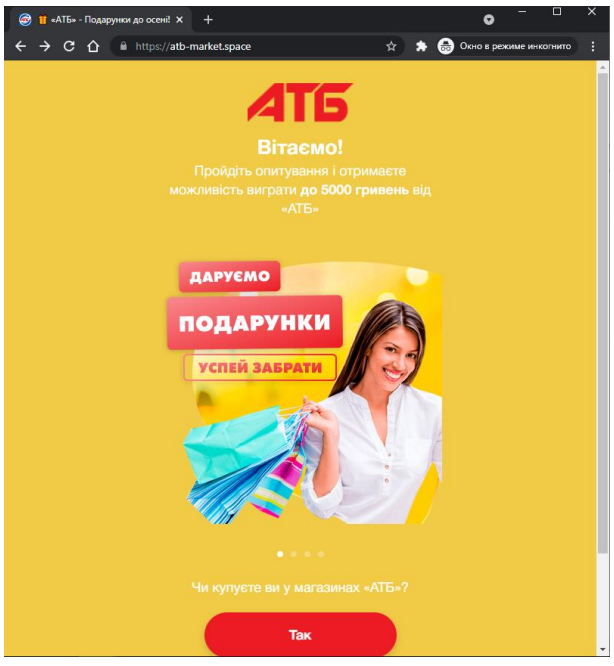


Рисунок 1.10 – Вигляд фішингового сайту https://atb-market.space/

Kaspersky Threat Intelligence Portal – це сервіс перевірки посилань [16]. Працює наступним чином: в стрічку запиту необхідно ввести URL-посилання. Далі треба натиснути «найти» і сервіс відобразить результат, в якому будуть такі дані:

– кількість IPv4-адрес;
– кількість файлів;
– домен;
– дата створення домену;
– на який термін оплачено домен;
– реєструюча організація;
– ім’я реєстратора;
– дані з сервісу WHOIS;
– певна інформація, яку можна побачити лише при наявності premium доступу.

Приклад видачі результату сервісом Kaspersky Threat Intelligence Portal показано на рисунку 1.11.

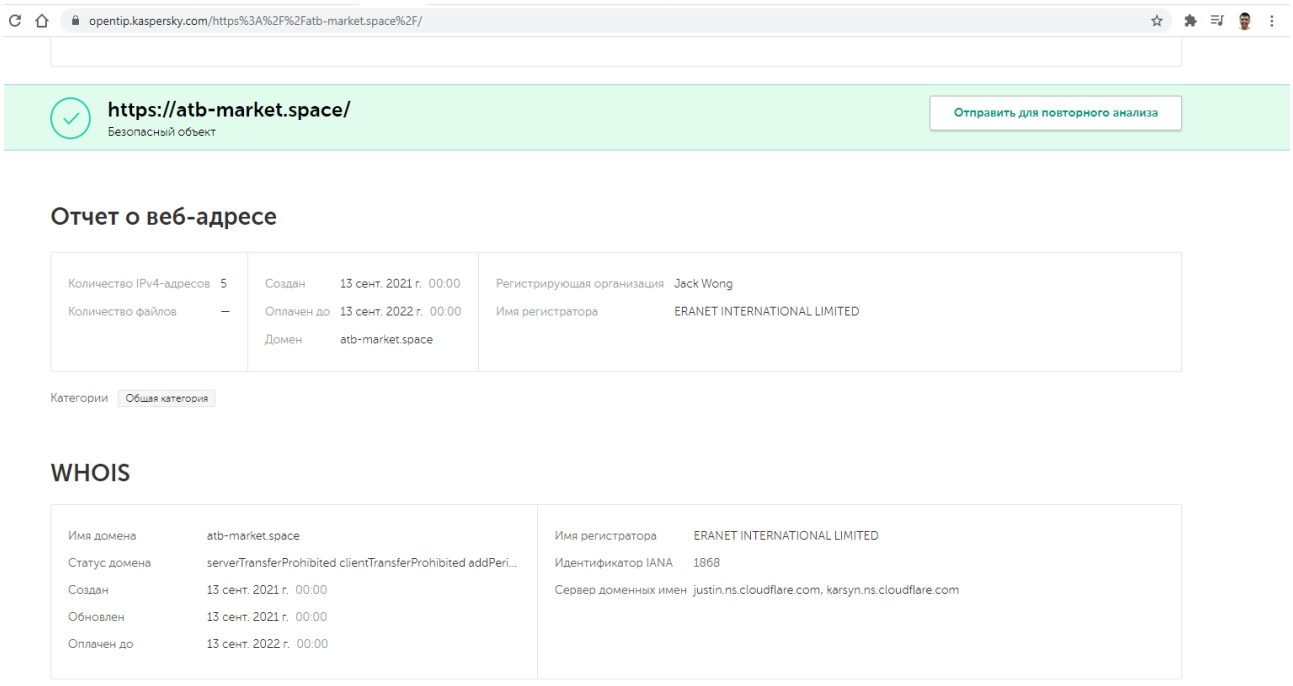


Рисунок 1.11 – Вигляд результату перевірки сервісом Kaspersky Threat Intelligence Portal

ScanURL – сервіс має мінімалістичний зрозумілий інтерфейс [17]. На сторінці розташовано поле для вводу URL-адреси, поле для вводу нікнейму, поле для коментаря до запиту; на даній сторінці розташовано повідомлення, що усі три поля мають бути обов'язково заповнені для виконання запиту. У даному випадку сервіс ScanURL результує, що адреса https://atb-market.space/ є безпечною для відвідування, жоден з сервісів, якими ScanURL перевіряв не сповістив про сайт, як про підозрілий (рис. 1.12). Слід зазначити, що сервіс ScanURL перевіряв сайт перш за все на наявність шкідливого програмного забезпечення, про що свідчить повідомлення «atb-market.space/ - Malware Scan Results», хоча на головній сторінці сервісу зазначалось «Введіть URL -адресу веб -сайту нижче, і побачимо, чи повідомлялося про це через фішинг, розміщення шкідливого програмного забезпечення/вірусів або погану репутацію. Перевіряємо з використанням таких авторитетних сторонніх сервісів, як Google Safe Browsing Diagnostic, PhishTank та Web of Trust (WOT).»

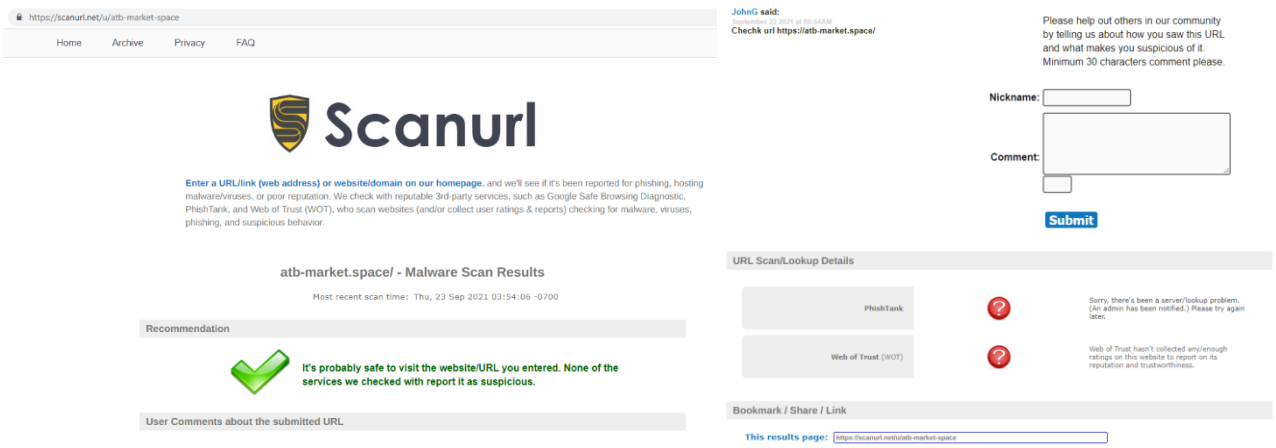


Рисунок 1.12 – Вигляд результату перевірки сервісом ScanURL

PhishTank – це сервіс, який перевіряє введений сайт на приналежність до фішингових ресурсів [18]. Умовно, головними елементами сайту даного сервісу є стрічка для введення запиту та список останніх URL-посилань, що користувачі самі подають як фішингові. Даний сервіс повідомив, що не має інформації про сайт https://atb-market.space/ (рис. 1.13).

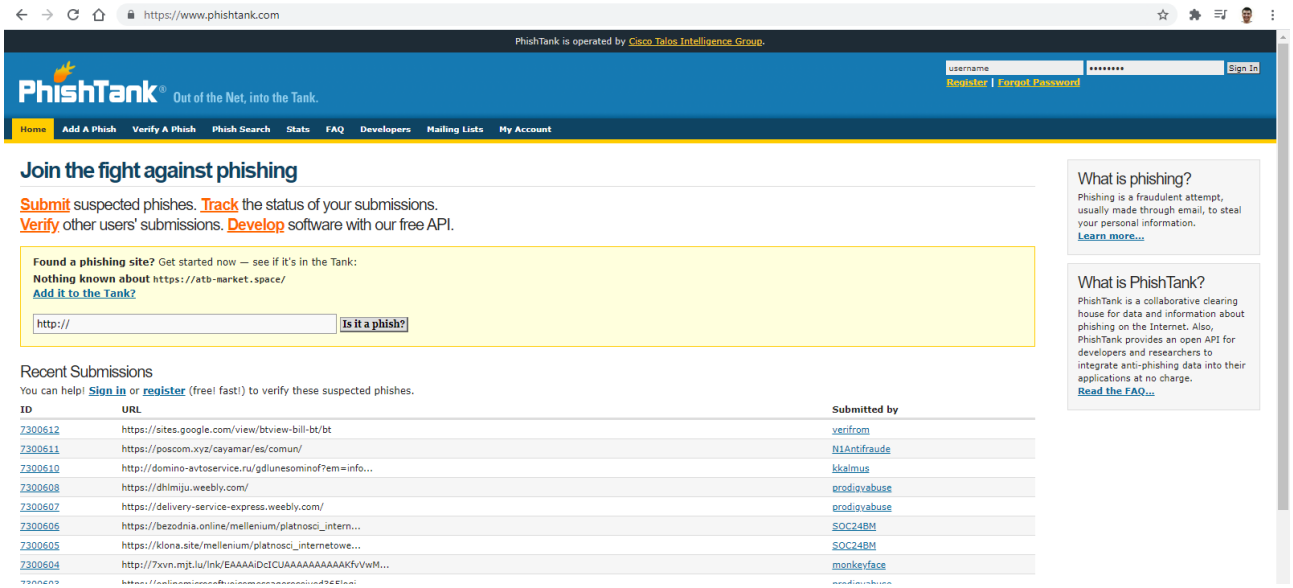


Рисунок 1.13 – Вигляд результату перевірки сервісом Phishtank

Google Transparency Report – сервіс для перевірки URL-посилань створений компанією Google [19]. Як зазначено на сайті «за допомогою Безпечного перегляду ми щоденно перевіряємо мільярди URL і знаходимо тисячі шкідливих веб-ресурсів, багато з яких створювались аж ніяк не зловмисниками, але потім були зламані. Для таких сайтів Google показує попередження в результатах пошуку або в браузері, щоб захистити користувачів». Даний сервіс повідомив, що на сайті https://atb-market.space/ не знайдено небезпечного вмісту (рис. 1.14). Чи є цей сайт фішинговим – у результаті не вказується.

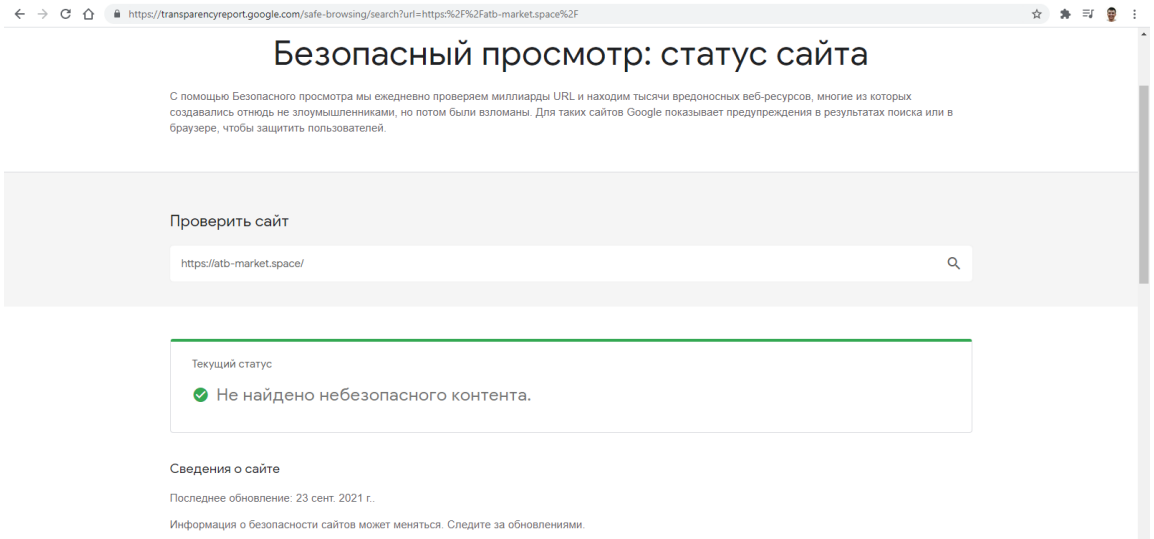


Рисунок 1.14 – Вигляд результату перевірки Google Transparency Report

Dr.Web – даний сервіс має інтерфейс стрічки для запиту, в яку необхідно вводити сайт, що буде перевірятись [20]. Згідно результату перевірки, сервіс Dr.Web не виявив чи є сайт https://atb-market.space/ фішинговим (рис. 1.15).

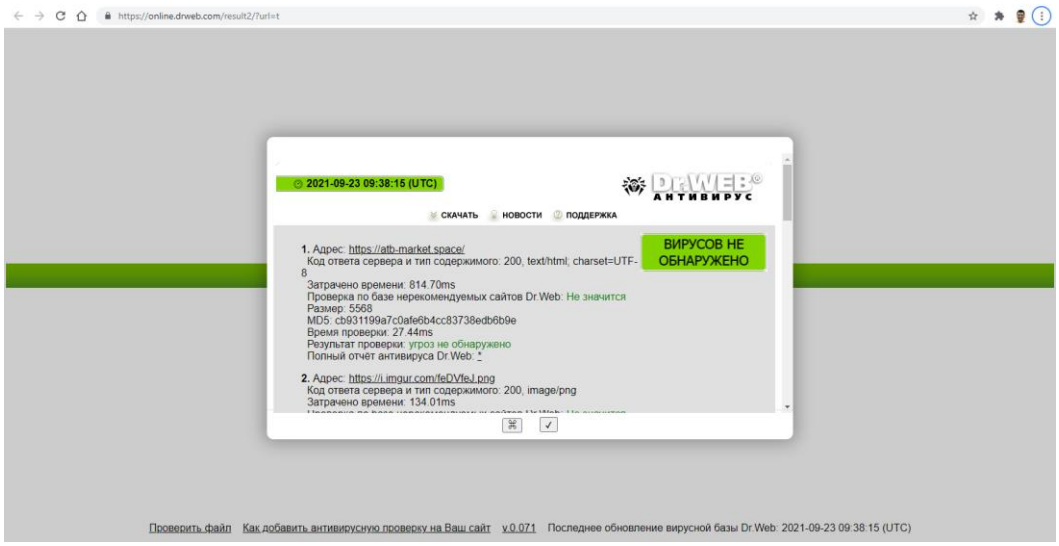


Рисунок 1.15 – Вигляд результату перевірки сервісом Dr.Web

Norton Safe Web – даний сервіс представляє собою стрічку куди треба вводити сайт для перевірки [21]. Також сервіс пропонує користувачу встановити розширення для безпечного пошуку та встановити у якості домашньої сторінки веб-браузеру посилання search.norton.com. Сервіс Norton Safe Web, як відповідь, видає дані, що сайт https://atb-market.space/ ще не має рейтингу, ніхто з користувачів ще не писав про нього відгук, статус перевірки – непротестовано (рис. 1.16).

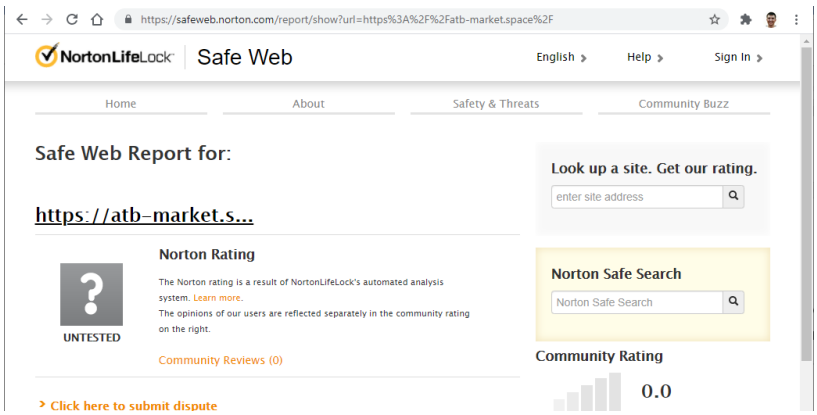


Рисунок 1.16 – Вигляд результату перевірки сервісом Norton Safe Web

У якості додаткової перевірки обрано сайт з умовновеликою аудиторією відвідувачів – vntu.edu.ua. Сервіс Norton Safe Web видав результат, що сайт vntu.edu.ua належить до категорій фішинг та освіта (рис. 1.17).

Аналіз обраних сервісів виявлення фішингових ресурсів дозволяє зрозуміти, що виявлення фішингу є наразі проблемою. Застосовуються різні методи та з'являються нові способи виявлення фішингових атак. В даний час запобігання фішингових атак вважається складним завданням в області безпеки систем. Ефективний засіб виявлення повинна мати можливість виявляти фішингові інформаційні ресурси з невеликою кількістю помилкових спрацювань. Можливим шляхом вирішення даної проблеми є інтелектуальний аналіз даних і евристика, машинне навчання.

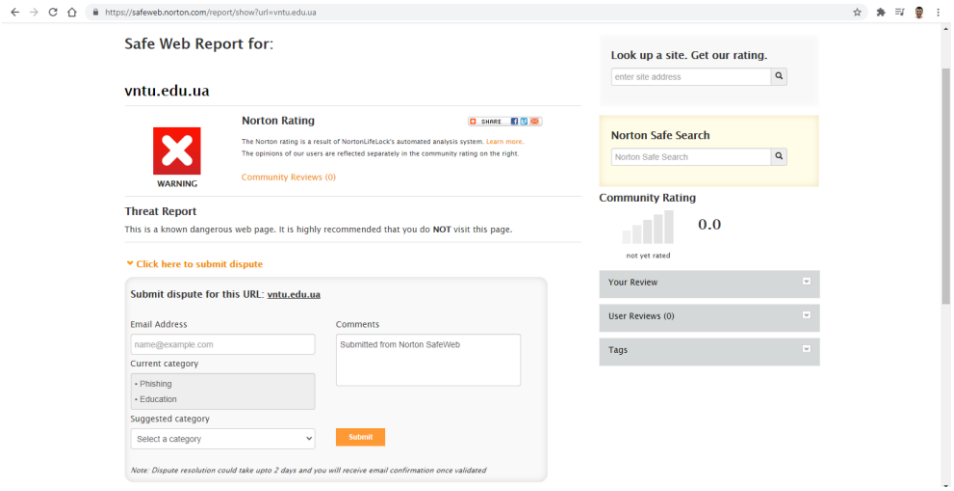


Рисунок 1.17 – Вигляд додаткового тестування сервісу Norton Safe Web

З високими обчислювальними витратами, евристичні методи і методи інтелектуального аналізу даних мають високі показники хибного спрацювання, проте краще розпізнають фішингові атаки. Процедури машинного навчання дають найкращі результати в порівнянні з іншими стратегіями. Частина процедур машинного навчання може ідентифікувати позитивного спрацювання до 99%. Оскільки шкідливі URL-адреси створюються через день і зловмисники використовують методи, щоб обдурити користувачів і змінити URL-адреси для атаки, є доцільним використати методи глибокого навчання і машинного навчання для виявлення фішингових атак.

1.4 Постановка завдання

Визначення фішингових інформаційних ресурсів – актуальна задача на сьогоднішній день, що не лише постала перед звичайними користувачами інтернету, але і негативно впливає на самі ресурси, що підроблюються через негативні наслідки. Фішингові інформаційні ресурси використовують зазвичай шахраї для виловлювання персональних даних довірливих, неуважних користувачів для збагачення незаконними шляхами. Все частіше фішингові інформаційні ресурси знаходяться у пошуковій видачі поряд з офіційними ресурсами з метою впливу на вибір користувачів, хоча вони не несуть практичної цінності, як результат видачі на шуканий запит.

Для того, зменшити можливість потрапляння без явного бажання на фішинговий інформаційний ресурс, необхідно розробити програмний засіб, що дозволить користувачам визначати фішингові ресурси та чітко розуміти з чим вони мають справу до переходу на ресурс, на якому вони бажають здійснити фінансову операцію чи ввести свої дані автентифікації.

Для подальшої розробки програмного засобу обрано метод машинного навчання. Метод машинного навчання дозволяє подати дані для навчання комп’ютеру і через набутий досвід – результат навчання – визначати чи є ресурс фішинговим.

Аналіз кожного елементу на веб-сайті поки не розробляється.

На основі сформованих вимог до програмного засобу розроблено технічне завдання.

Програмний засіб повинен відповідати таким вимогам:

– програмний засіб повинен складатися з двох частин: модуля зчитування та обробки інформації та модуля машинного навчання;

– модуль зчитування та обробки інформації повинен зчитувати дані про вказаний користувачем інформаційний ресурс, обробляти їх та перетворювати у масив чисел;

– модель повинна навчатися на навчальній вибірці, що знаходиться у окремому файлі, а також при подачі на вхід оброблених даних про

інформаційний ресурс видавати на вихід результат щодо справжньості даного ресурсу.

Отже, проаналізовано інформаційні джерела, розглянуто основні поняття інформаційних ресурсів, фішингових ресурсів та їх використання під час переходу роботу на дистанційну основу, а також під час звичайного користування. Розглянуто основні метрики інформаційних ресурсів. Розглянуто та проаналізовано методи аналізу інформаційних ресурсів за їх особливостями та обрано оптимальний метод для подальшого дослідження. Висунуто ряд вимог до розроблюваного програмного засобу.

2 РОЗРОБКА МЕТОДУ ВИЯВЛЕННЯ ФІШИНГОВОГО ВМІСТУ

2.1 Аналіз структури веб-сторінки

Частота термів – зворотна частота документу (TFIDF) є популярною статистичною ознакою. Це схема зважування термів, яка використовує частоту термів у документі та журнал зворотної популярності терма в колекції. Визначається формулою (2.1), де $n_{t,d}$ – кількість разів, коли термін t з'являється в документі d, N_d – загальна кількість термінів у d, D – загальна кількість документів, а d_t – кількість документів, що містять термін t. У даному виділенні функцій вектор TFIDF, вилучений з вмісту веб-сайту, розглядається як одна особливість.

$$TFIDF(t,d) = \frac{n_{t,d}}{N_d} \times \log_e\left(\frac{D}{1+d_t}\right) \tag{2.1}$$

Використаємо наведені нижче показники, щоб повідомити про результати.

Матриця плутанини: загальна кількість істинно позитивних тверджень (TP), справжніх негативних тверджень (TN), хибнопозитивних тверджень (FP) і хибнонегативних тверджень (FN) ідентифікованих класифікатором. Повідомляється як вихідні значення або відсотки. Також визначимо P = TP + FN і N = TN + FP.

Точність(Accuracy): відношення правильно класифікованих екземплярів до загальної кількості екземплярів (2.2).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.2}$$

Влучність(Precision): співвідношення екземплярів, правильно визначених класифікатором як відповідних до загальної кількості екземплярів, класифікованих як відповідні (2.3).

$$Precision = \frac{TP}{TP+FP} \tag{2.3}$$

Повнота(Recall): співвідношення екземплярів, правильно визначених класифікатором як відповідних до загальної кількості відповідних екземплярів (2.4).

$$Recall = \frac{TP}{TP+FN} \tag{2.4}$$

F_1-бал: середнє гармонічне значення точності та повноти (2.5).

$$F_1 = \frac{2*\text{Precission}*\text{Recall}}{\text{Precission}+\text{Recall}} \quad (2.5)$$

Геометричне середнє(Geometric Mean): середнє геометричне значення істинного від'ємного курсу (специфічність) та повноти (2.6).

$$G - Mean = \sqrt{\frac{TP}{TP+FN} * \frac{TN}{TN+FP}} \quad (2.6)$$

Збалансований коефіцієнт виявлення(BDR): метрика для вимірювання кількості екземплярів класів меншин, які були правильно класифіковані та відповідного покарання за використання неправильно класифікованих екземплярів класу більшості (2.7).

$$BDR = \frac{TP}{1+FP} \quad (2.7)$$

Площа під кривою(AUC): при використанні нормалізованих одиниць, площа під кривою дорівнює ймовірності того, що класифікатор оцінить випадково вибраний позитивний екземпляр вище, ніж випадково вибраний негативний.

Коефіцієнт кореляції Метью(MCC): він враховує істинні, хибнопозитивні та негативні показники і, як правило, розглядається як незбалансований показник, який можна використовувати, навіть якщо класи дуже різних розмірів (2.8).

$$MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TN+FN)(P)(N)}} \quad (2.8)$$

Виконавши попередній опис статистичних ознак у формульному вигляді, перейдемо до розробки методу виявлення фішингових ресурсів.

2.2 Розробка методу класифікації полів веб-сторінки

За останнє десятиліття дослідники ідентифікували і класифікували особливості, що проявляються з векторів фішингових атак різними способами. В деяких джерелах ознаки класифікуються з точки зору типу атаки, деякі класифікуються на основі того, як і де вони з'являються у векторі атаки [4], [5]. Однак, наскільки відомо, ніхто не дав систематичної класифікації, яка незалежна від підходів до виявлення і охоплює всі можливі особливості, які можуть бути

добуті з векторів атак. Вектори фішингу, наприклад веб-сайти, URL-адреси, представляють собою спеціалізовані стрічки. Тому, такі категорії мови чи логіки як синтаксис, семантика і прагматика можуть бути ефективно використані для класифікації їх атрибутів. Побудова класифікації потребує детального огляду інформаційних джерел, розуміння потенційних проблем та можливостей, а також систематичного і детального підходу. Класифікація, що представлена на рисунку 2.1, має основні компоненти веб-сторінки в якості других рівнів. Кожний компонент далі розбивається на такі групи: синтаксис, семантика і прагматика.

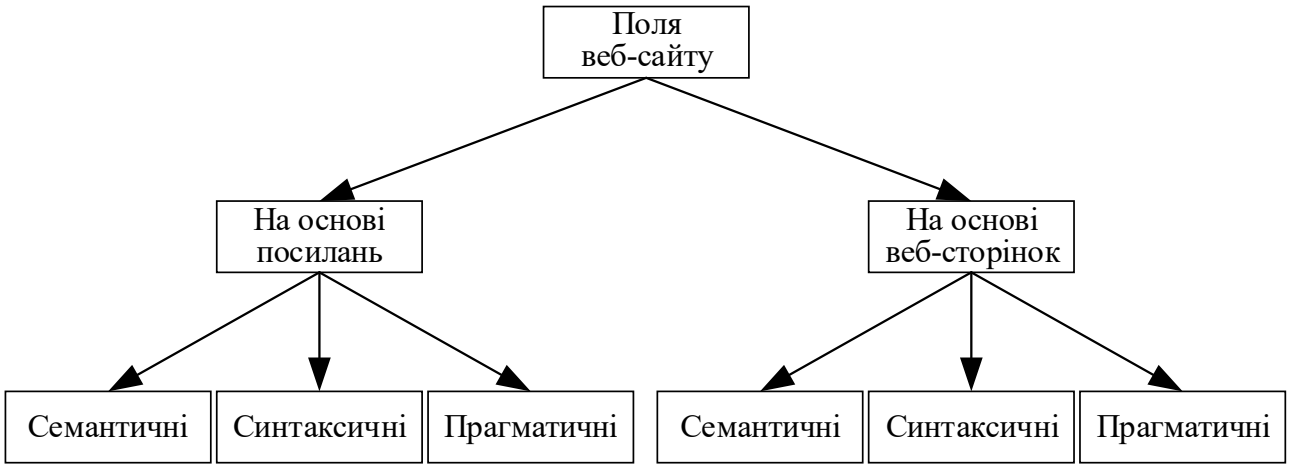


Рисунок 2.1 – Вигляд класифікації сайтів на основі посилань та веб-сторінок

Синтаксичні поля вектору залежать від формату і синтаксичної коректності вектору URL-адреси чи веб-сайту. Наприклад, у правильно побудованій URL-адресі домен верхнього рівня (.com, .net, .ua) з’являється лише один раз. Однак це не завжди так у випадку шкідливої URL-адреси. Ця проблема зв’язана з синтаксисом URL-адрес, тому розглядаємо позицію домену верхнього рівня як синтаксичну особливість. Досліджуючи вміст веб-сайту, можемо підрахувати усі теги, що являється ще одною синтаксичною особливістю.

Семантичні поля зосереджені на значенні та інтерпритації текстового вмісту в URL-адресах та веб-сайтах. Прикладом семантичного поля для веб-сайту може бути значення елементів HTML, наприклад кількість прихованих об’єктів.

Прагматичні поля не мають прямого відношення до синтаксису чи семантики URL-адреси чи веб-сайту. Наприклад, відключення натискання правої кнопки миші на веб-сайтах – це метод, що використовується зловмисниками для попередження перегляду і збереження вихідного коду користувачами. Це не має відношення ні до синтаксису вмісту HTML, ні до його семантики. Таким чином, це відноситься до категорії прагматичних особливостей. Іншими прикладами прагматичних особливостей є відомості про реєстрацію веб-сайту, термін дії веб-сайту і т.д.

Рисунок 2.1 ще можна поділити на два-три класи: лексичний, мережевий та сценарний.

Двома важливими «дійовими особами» фішингових атак на веб-сайти є веб-браузер і жертва. Під час побудови систематики розглядаємо, як веб -браузер та жертва інтерпретують URL-адресу або веб-сайт.

Реалізуємо URL-поля у даному модулі, які охоплюють усі типи основних класів атрибутів на основі URL-адрес: лексичні, мережеві та сценарні. Розглядаємо особливості, які оцінюються високо за допомогою методів ранжування ознак, таких як отримання інформації. Надамо приклади URL-адрес і функцій веб-сайту, щоб зробити кожну з цих категорій легшою для розуміння.

Класи функцій URL:

- лексичних особливостей URL включають у себе такі поля як, найдовший токен у домені (синтаксична), наявність цільової марки (семантична) та наявність слів із чорного списку, слів із білого списку (прагматична);
- мережевими особливостями є синтаксична категорія функцій, що включає наявність номера порту, чи є ім’я домену IP-адресою тощо. Під семантичною категорією входять такі функції, як наявність відповідності між портом і протоколом. Прагматична категорія мережевих функцій включає такі атрибути, як номер автономної системи AS, інформація WHOIS тощо;

– скриптовими є такі особливості, як ряд функцій типу escape, eval тощо. Наявність у шляху URL-адреси різних розширень файлів, таких як «.php», «.js» тощо (семантична), і час, необхідний для завантаження ресурсу сценарію (прагматична).

Класи функцій веб-сайту:

– лексичні: характеристики термінів із зворотною частотою документа (TF-IDF) із веб-сторінок (синтаксична), наявність назви брендів на веб-сторінці (семантична) та атрибути, такі як обробка підозрілого вмісту або вмісту з чорного списку (прагматична);
– мережеві: прикладом синтаксичного атрибута є довжина вмісту веб-сторінки (усі функції, вилучені з протоколу HTTP, вважаються мережевими функціями). Семантичні поля мережі включають час завантаження залежно від типу вмісту (зображення, аудіо, відео), тоді як прикладами прагматичних полів може бути номер автономної системи (AS);
– скриптові: кількість зовнішніх скриптів(синтаксична), чи є скрипт шкідливим або має певну поведінку, наприклад створення спливаючих вікон(семантична), чи скрипт завантажується асинхронно або відкладається (прагматична).

Схема системи порівняльного аналізу оцінки фішингу з усіма модулями наведена на рисунку 2.2.

Модуль вводу оброблює завантаження набору даних в пам’ять і має необхідні функції попередньої обробки, реалізовані для вилучення полів з різних типів введення. Він приймає список URL-адрес для фішингових веб-сайтів/URL-адрес. Для наборів даних URL, якщо вказано користувачем, модуль аналізує HTML-вміст веб-сайту, пов’язаного з кожною URL-адресою, і завантажує всі доступні метадані мережі, такі як інформація WHOIS та час відповіді HTTP. Схема системи порівняльного аналізу оцінки фішингу показано на рисунку 2.2.

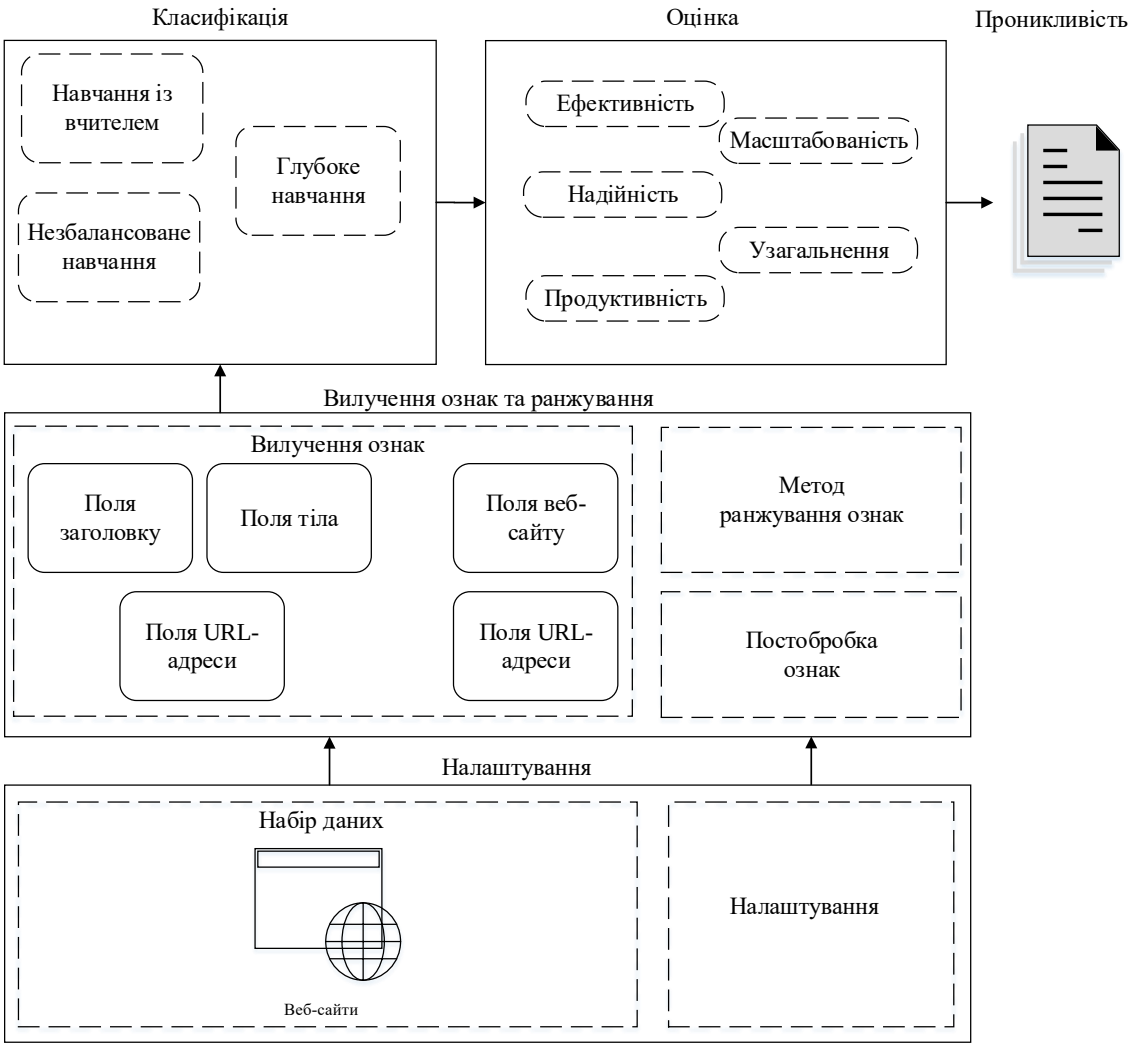


Рисунок 2.2 – Вигляд схеми запропонованої системи порівняльного аналізу оцінки фішингу

Модуль містить необхідні функції для вилучення полів. Модуль налаштовується, оскільки дозволяє вибирати функції, які потрібно витягти, і розширюється, оскільки нові функції можуть бути додані до фреймворку шляхом реалізації функції з необхідними входами та виходами. Гнучкість, надана цим модулем, допомагає порівнювати результати своїх нових полів з уже наявними в літературі. Він також повідомляє про час, необхідний для вилучення полів.

Основними функціями цього модуля є ранжування та нормалізація полів для використання їх як вхідних даних для класифікації. Використано бібліотеку Scikit-Learn для реалізації перерахованих нижче показників/алгоритмів рейтингу:

- приріст інформації: вибір на основі зменшення ентропії після того, як набір даних розділений на атрибут;
- індекс Джині: розбиває ознаку на різні класи, щоб вказати її ефективність для класифікації;
- метрика хі-квадрат: вимірює незалежність між появою конкретної ознаки та певного класу;
- рекурсивне видалення ознак: спочатку модель навчається за всіма поляит і обчислюється оцінка важливості для кожного поля. Тоді найнижчі рейтингові об’єкти видаляються після кожної ітерації.

Нормалізація виконується за допомогою таких методів: масштаб MaxAbsolute (масштабування кожної функції за її максимальним абсолютним значенням), масштабування min-max (віднімає мінімальне значення у об’єкті, а потім ділить на діапазон), середнє масштабування (віднімає середнє значення функція, а потім ділиться на діапазон).

Модуль виводить файл, що містить відсортований список функцій на основі результатів використовуваного алгоритму ранжування та розріджену матрицю найкращих полів, повернених алгоритмом. Кількість найкращих функцій визначає користувач.

Для модуля навчання із вчителем реалізуємо такі алгоритми навчання:

- Decision Tree (DT);
- Random Forest (RF);
- Multilayer Perceptrons;
- XGBoost;
- AutoEncoder;
- Support Vector Machines (SVM).

Модуль також повідомляє час роботи для кожного класифікатора. Він також є розширюваним, оскільки можна додавати нові класифікатори, реалізуючи нові методи або імпортуючи наявні бібліотеки.

Вибір належних показників оцінки є надзвичайно важливим. Дисбаланс класів у наборі даних оцінки може призвести до помилки базової ставки, коли

такий показник, як точність, може бути не найкращим вибором для оцінки ефективності класифікації.

Набір фішингових URL-адрес збирається з відкритого сервісу під назвою PhishTank. Ця служба надає набір фішингових URL-адрес у кількох форматах, таких як csv, json або інших. З цього набору даних обрано 5000 випадкових фішингових URL-адрес для навчання моделі.

Законні URL-адреси отримано з відкритих наборів даних Університету Нью-Брансвіка. Цей набір даних містить набір небезпечних, спамових, фішингових, зловмисних програм і URL-адрес, що сприяють зіпсуванню. З усіх цих типів для даної роботи обрано доброякісний набір даних URL-адрес у розмірі 5000, дані обрано випадковим чином.

Архітектура системи програмного забезпечення для виявлення фішингових інформаційних ресурсів складається з таких компонентів (рис. 2.3):

- програмного засобу, який складається з модуля зчитування та обробки інформації, а також моделі, яка має 16 входів та 1 вихід;
- URL-адреси досліджуваної веб-сторінки;
- зовнішніх файлів, де міститься допоміжна для роботи програми інформація.

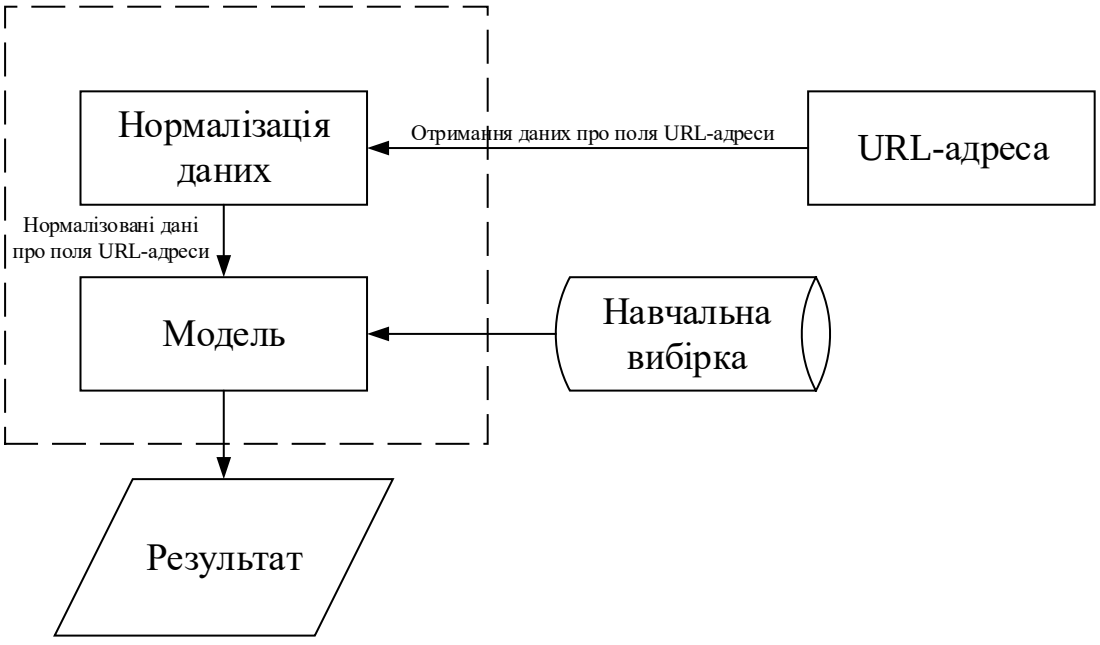


Рисунок 2.3 – Архітектура програмного засобу

Проаналізовано інформацію, що міститься в URL-адресі веб-сайту, а також виокремлено основні метрики інформаційних ресурсів та віднесено їх до відповідних категорій. Розподілено ступені справжньості інформаційни ресурсів у вигляді балів відповідно до значень метрик.

3 РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ

3.1 Дії з набором даних

Для цього проекту нам знадобиться набір URL-адрес типу 1 (фішингові) або 0 (законні).

Збирати фішингові URL-адреси досить легко завдяки службі з відкритим кодом під назвою PhishTank. Ця служба надає набір фішингових URL-адрес у кількох форматах, як-от csv, json тощо, які оновлюються щогодини.

Для законних URL-адрес знайдено джерело, яке містить набір небезпечних, спамових, фішингових, зловмисних і шкідливих URL-адрес. Джерело набору даних – Університет Нью-Брансвіка [22]. Кількість законних URL-адрес у цій колекції становить 35 378. Колекцію URL-адрес завантажено і з цього файлу, який нас цікавить, *'Benign_list_big_final.csv'*. Потім цей файл оброблюється для вилучення особливостей. Кожна стрічка файлу містить виключно URL-адресу.

Фішингові URL-адреси збираються з PhishTank за наданим посиланням. Файл csv фішингових URL-адрес отримується за допомогою команди wget. Після завантаження набору даних він завантажується у DataFrame.

```
import pandas as pd
!wget http://data.phishtank.com/data/online-valid.csv
```

Наведена вище команда завантажує файл фішингових URL-адрес *online-valid.csv* і зберігає його (рис. 3.1).

```
data0 = pd.read_csv("online-valid.csv")
data0.head()
```

	phish_id	url	phish_detail_url	submission_time	verified	verification_time	online	target
0	6557033	http://u1047531.cp.regruhosting.ru/acces-inges...	http://www.phishtank.com/phish_detail.php?phis...	2020-05-09T22:01:43+00:00	yes	2020-05-09T22:03:07+00:00	yes	Other
1	6557032	http://hoysalacreations.com/wp-content/plugins...	http://www.phishtank.com/phish_detail.php?phis...	2020-05-09T22:01:37+00:00	yes	2020-05-09T22:03:07+00:00	yes	Other
2	6557011	http://www.accsystemprblemhelp.site/checkpoint...	http://www.phishtank.com/phish_detail.php?phis...	2020-05-09T21:54:31+00:00	yes	2020-05-09T21:55:38+00:00	yes	Facebook
3	6557010	http://www.accsystemprblemhelp.site/login_atte...	http://www.phishtank.com/phish_detail.php?phis...	2020-05-09T21:53:48+00:00	yes	2020-05-09T21:54:34+00:00	yes	Facebook
4	6557009	https://firebasestorage.googleapis.com/v0/b/so...	http://www.phishtank.com/phish_detail.php?phis...	2020-05-09T21:49:27+00:00	yes	2020-05-09T21:51:24+00:00	yes	Microsoft

Рисунок 3.1 – Вигляд набору даних та його полів

Перевіримо розмірність набору online-valid.csv. Далі, коли будемо використовувати команду `shape` – це буде означати, що перевіряємо розмірність набору.

```
data0.shape
```

Розмірність набору валідних даних складає 14858 стрічок і 8 стовпців.

Отже, дані містять тисячі фішингових URL-адрес. Але проблема в тому, що ці дані оновлюються щогодини. Не ризикуючи дисбалансом даних, оперуємо значенням маржі в 10 000 фішингових URL-адрес і 5000 законних URL-адрес.

Таким чином, випадково відібрано 5000 зразків із вищевказаного фрейму даних (рис. 3.2).

```
phishurl = data0.sample(n = 5000, random_state = 12).copy()
phishurl = phishurl.reset_index(drop=True)
phishurl.head()
```

	phish_id	url	phish_detail_url	submission_time	verified	verification_time	online	target
0	6514946	http://confirmprofileaccount.com/	http://www.phishtank.com/phish_detail.php?phis...	2020-04-19T11:06:55+00:00	yes	2020-04-19T13:42:41+00:00	yes	Other
1	4927651	http://www.marreme.com/MasterAdmin/04mop.html	http://www.phishtank.com/phish_detail.php?phis...	2017-04-04T19:35:54+00:00	yes	2017-05-03T23:00:42+00:00	yes	Other
2	5116976	http://modsecpaststudents.com/review/	http://www.phishtank.com/phish_detail.php?phis...	2017-07-25T18:48:30+00:00	yes	2017-07-28T16:01:36+00:00	yes	Other
3	6356131	https://docs.google.com/forms/d/e/1FAIpQLScL6L...	http://www.phishtank.com/phish_detail.php?phis...	2020-01-13T20:13:37+00:00	yes	2020-01-17T01:55:38+00:00	yes	Other
4	6535965	https://oportunidadedasemana.com/americanas//?...	http://www.phishtank.com/phish_detail.php?phis...	2020-04-29T00:01:03+00:00	yes	2020-05-01T10:55:35+00:00	yes	Other

Рисунок 3.2 – Вигляд випадковообраних зразків

```
phishurl.shape
```

Розмірність складає 5000 стрічок на 8 стовпців.

На даний момент зібрано 5000 фішингових URL-адрес. Тепер потрібно зібрати законні URL-адреси.

Із завантаженого файлу *Benign_list_big_final.csv* URL-адреси завантажуються у фрейм даних (рис. 3.3).

```
data1 = pd.read_csv("Benign_list_big_final.csv")
data1.columns = ['URLs']
data1.head()
```

	URLs
0	http://1337x.to/torrent/1110018/Blackhat-2015-RUSSIAN-720p-WEB-DL-DD5-1-H264-RUFGT/
1	http://1337x.to/torrent/1122940/Blackhat-2015-x264-1080p-WEB-DL-eng-nl-subs-sharky/
2	http://1337x.to/torrent/1124395/Fast-and-Furious-7-2015-HD-TS-XVID-AC3-HQ-Hive-CM8/
3	http://1337x.to/torrent/1145504/Avengers-Age-of-Ultron-2015-CAM-New-Audio-x264-CPG/
4	http://1337x.to/torrent/1160078/Avengers-age-of-Ultron-2015-HQ-CAM-H264-AC3-MURD3R/

Рисунок 3.3 – Вигляд адрес у файлі Benign_list_big_final.csv

Як зазначено вище, 5000 законних URL-адрес випадковим чином вибираються з вищевказаного фрейму даних (рис. 3.4).

```
legiurl = data1.sample(n = 5000, random_state = 12).copy()
legiurl = legiurl.reset_index(drop=True)
legiurl.head()
```

	URLs
0	http://graphicriver.net/search?date=this-month&length_max=&length_min=&price_max=&price_min=&rating_min=&sales=&sort=sales&term=&view=list
1	http://ecnavi.jp/redirect/?url=http://www.cross-a.net/x.php?id=1845_3212_22061_26563&m=1004&pid=%user_id%
2	https://hubpages.com/signin?explain=follow+Hubs&url=%2Fhub%2FComfort-Theories-of-Religion
3	http://extratorrent.cc/torrent/4190536/AOMEI+Backupper+Technician+%2B+Server+Edition+2.8.0+%2B+Patch+%2B+Key+%2B+100%25+Working.html
4	http://icicibank.com/Personal-Banking/offers/offer-detail.page?id=offer-ezeego-domestic-airtravel-20141407112611060

Рисунок 3.4 – Вигляд випадкововідібраних законних адрес

```
legiurl.shape
```

Розмірність складає 5000 стрічок і 1 стовпець.

На цьому кроці особливості витягуються з набору даних URL-адрес. Вилучені ознаки поділяються на категорії:

1. функції на основі адресного рядка;
2. функції на основі домену;
3. функції на основі HTML і Javascript.

Особливості на основі адресного рядка. Можна витягти багато функцій, які можна вважати базовими функціями адресного рядка. З них, зазначені нижче, були розглянуті для цього проекту:

– домен URL-адреси;

- IP-адреса в URL;
- символ "@" в URL-адресі;
- довжина URL-адреси;
- глибина URL-адреси;
- перенаправлення "//" в URL;
- "http/https" в імені домену;
- використання служб скорочення URL-адрес «TinyURL»;
- префікс або суфікс «-» у домені.

Кожна з цих функцій пояснюється та кодиться нижче:

```
from urllib.parse import urlparse,urlencode
import ipaddress
import re
```

Домен URL-адреси. Тут витягуємо домен, присутній в URL-адресі. Ця особливість не має великого значення в навчанні. Може навіть бути скинутим під час навчання моделі.

```
def getDomain(url):
  domain = urlparse(url).netloc
  if re.match(r"^www.",domain):
            domain = domain.replace("www.","")
  return domain
```

IP-адреса в URL-адресі. Перевіряє наявність IP-адреси в URL-адресі. URL-адреси можуть мати IP-адресу замість доменного імені. Якщо IP-адреса використовується як альтернатива імені домену в URL-адресі, можемо бути впевнені, що хтось намагається вкрасти особисту інформацію за допомогою цієї URL-адреси.

Якщо доменна частина URL-адреси має IP-адресу, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
def havingIP(url):
  try:
    ipaddress.ip_address(url)
    ip = 1
  except:
    ip = 0
  return ip
```

Таким чином, алгоритм визначення значень кожного з полів для набору даних можна відобразити у вигляді блок-схеми (рис. 3.5). Символ "@" в URL-адресі. Перевіряє наявність символу «@» в URL-адресі. Використання символу «@» в URL-адресі змушує браузер ігнорувати все, що передує символу «@», а справжня адреса часто слідує за символом «@».

Якщо в URL-адресі є символ «@», цій функції призначається значення 1 (фішинг) або 0 (законне).

```
def haveAtSign(url):
  if "@" in url:
    at = 1
  else:
    at = 0
  return at
```

Перевірка довжина URL-адреси. Обчислюмо довжину URL-адреси, тому що шахраї можуть використовувати довгу URL-адресу, щоб приховати сумнівну частину в адресному рядку. У даному випадку, якщо довжина URL-адреси перевищує або дорівнює 54 символам, URL-адреса класифікується як фішингова.

Якщо довжина URL-адреси >= 54 , значення, призначене цій функції, дорівнює 1 (фішинг) або 0 (законне).

```
def getLength(url):
  if len(url) < 54:
    length = 0
  else:
    length = 1
  return length
```

Перевірка глибина URL-адреси. Обчислює глибину URL-адреси для перевірки кількості підсторінок у вказаній URL-адресі на основі «/».

Значення функції є числовим на основі URL-адреси.

```
def getDepth(url):
  s = urlparse(url).path.split('/')
  depth = 0
  for j in range(len(s)):
    if len(s[j]) != 0:
      depth = depth+1
  return depth
```

Перенаправлення "//" в URL-адресі. Наявність «//» в URL-шляху означає, що користувач буде перенаправлено на інший веб-сайт. Знаходимо розташування «//» в URL-адресі. Виявлено, що якщо URL-адреса починається з «HTTP», це означає, що «//» має з'явитися на шостій позиції. Однак, якщо в URL-адресі використовується «HTTPS», то «//» має з'явитися на сьомій позиції.

Якщо «//» знаходиться в будь-якому місці URL-адреси, крім протоколу, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
def redirection(url):
  pos = url.rfind('//')
  if pos > 6:
    if pos > 7:
      return 1
    else:
      return 0
  else:
    return 0
```

"http/https" в імені домену. Перевіряє наявність "http/https" у доменній частині URL-адреси. Фішери можуть додати токен «HTTPS» до домену URL-адреси, щоб обдурити користувачів.

Якщо URL-адреса містить "http/https" у частині домену, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
def httpDomain(url):
  domain = urlparse(url).netloc
  if 'https' in domain:
    return 1
  else:
    return 0
```

Використання служб скорочення URL-адрес «TinyURL». Скорочення URL-адреси – це метод, за допомогою якого URL-адреса може бути значно меншою за довжиною і все одно направляти на потрібну веб-сторінку. Це досягається за допомогою переспрямування на коротке доменне ім'я, яке посилається на веб-сторінку з довгою URL-адресою.

Якщо URL-адреса використовує послуги Shortening Services, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
shortening_services                                                    =
r"bit\.ly|goo\.gl|shorte\.st|go2l\.ink|x\.co|ow\.ly|t\.co|tinyurl|
tr\.im|is\.gd|cli\.gs|yfrog\.com|migre\.me|ff\.im|tiny\.cc|url4\.e
u|twit\.ac|su\.pr|twurl\.nl|snipurl\.com|short\.to|BudURL\.com|pin
g\.fm|post\.ly|Just\.as|bkite\.com|snipr\.com|fic\.kr|loopt\.us|do
iop\.com|short\.ie|kl\.am|wp\.me|rubyurl\.com|om\.ly|to\.ly|bit\.d
o|t\.co|lnkd\.in|db\.tt|qr\.ae|adf\.ly|goo\.gl|bitly\.com|cur\.lv|
tinyurl\.com|ow\.ly|bit\.ly|ity\.im|q\.gs|is\.gd|po\.st|bc\.vc|twi
tthis\.com|u\.to|j\.mp|buzurl\.com|cutt\.us|u\.bb|yourls\.org|x\.c
o|prettylinkpro\.com|scrnch\.me|filoops\.info|vzturl\.com|qr\.net|
1url\.com|tweez\.me|v\.gd|tr\.im|link\.zip\.net"


def tinyURL(url):
    match=re.search(shortening_services,url)
    if match:
        return 1
    else:
        return 0
```

Префікс або суфікс «-» у домені. Перевірка наявності символу "-" в доменній частині URL-адреси. Символ тире рідко використовується в законних URL-адресах. Шахраї, як правило, додають до імені домену префікси або суфікси, розділені (-), щоб користувачі відчували, що мають справу з законною веб-сторінкою.

Якщо URL-адреса містить символ «-» у доменній частині URL-адреси, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
def prefixSuffix(url):
    if '-' in urlparse(url).netloc:
        return 1
    else:
        return 0
```

Особливості на основі домену. Можна виділити багато особливостей, які підпадають під цю категорію. З них, зазначені нижче, були розглянуті у цій роботі:

- запис DNS;
- трафік веб-сайту;
- вік домену;
- кінцевий період домену.

Запис DNS. Для фішингових веб-сайтів або заявлена особистість не розпізнається базою даних WHOIS або не створено записи для імені хоста.

Якщо запис DNS порожній або його не знайдено, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
dns = 0
try:
    flags = 0
    flags = flags | whois.NICClient.WHOIS_QUICK
    domain_name = whois.whois(urlparse(url).netloc, flags=flags)
except:
    dns = 1
```

Значення із запису DNS отримується в самій функції featuresExtraction.

Трафік веб-сайту. Ця функція вимірює популярність веб-сайту, визначаючи кількість відвідувачів і кількість сторінок, які вони відвідують. Однак, оскільки фішингові веб-сайти існують протягом короткого періоду часу, вони можуть не розпізнаватися базою даних Alexa (Alexa the Web Information Company., 1996). Переглядаючи обраний набір даних, виявлено, що в найгірших сценаріях законні веб-сайти потрапляють до 100 000 найкращих. Крім того, якщо домен не має трафіку або не розпізнається базою даних Alexa, він класифікується як «фішинг».

Якщо ранг домену < 100000, значення цієї функції дорівнює 1 (фішинг), інакше 0 (легітимний).

```
def web_traffic(url):
  try:
    url = urllib.parse.quote(url)
  rank =
BeautifulSoup(urllib.request.urlopen("http://data.alexa.com/data?c
li=10&dat=s&url=" + url).read(), "xml").find(
        "REACH")['RANK']
    rank = int(rank)
  except TypeError:
        return 1
  if rank <100000:
    return 1
  else:
    return 0
```

Вік домену. Цю функцію можна отримати з бази даних WHOIS. Більшість фішингових веб-сайтів працюють протягом короткого періоду часу. Мінімальний вік законного домену для цього проекту становить 12 місяців. Вік тут не що інше, як різницю між створенням і часом закінчення.

Якщо вік домену > 12 місяців, значення цієї функції дорівнює 1 (фішинг), інакше 0 (законне).

```
def domainAge(domain_name):
  creation_date = domain_name.creation_date
  expiration_date = domain_name.expiration_date
           if          (isinstance(creation_date,str)          or
isinstance(expiration_date,str)):
    try:
      creation_date = datetime.strptime(creation_date,'%Y-%m-%d')
       expiration_date = datetime.strptime(expiration_date,"%Y-%m-
%d")
    except:
      return 1
  if ((expiration_date is None) or (creation_date is None)):
      return 1
  elif ((type(expiration_date) is list) or (type(creation_date) is
list)):
      return 1
  else:
    ageofdomain = abs((expiration_date - creation_date).days)
    if ((ageofdomain/30) < 6):
      age = 1
    else:
      age = 0
  return age
```

Кінцевий період домену. Цю функцію можна отримати з бази даних WHOIS. Для цієї функції час, що залишився в домені, розраховується шляхом знаходження різниці між часом закінчення терміну дії та поточним часом. Термін завершення, який розглядається для законного домену, становить 6 місяців або менше для цього проекту.

Якщо кінцевий період домену > 6 місяців, значення цієї функції дорівнює 1 (фішинг), інакше 0 (законне).

```
def domainEnd(domain_name):
  expiration_date = domain_name.expiration_date
  if isinstance(expiration_date,str):
    try:
```

```
        expiration_date = datetime.strptime(expiration_date,"%Y-%m-
%d")
    except:
      return 1
  if (expiration_date is None):
      return 1
  elif (type(expiration_date) is list):
      return 1
  else:
    today = datetime.now()
    end = abs((expiration_date - today).days)
    if ((end/30) < 6):
      end = 0
    else:
      end = 1
  return end
```

Функції на основі HTML і JavaScript. Можна виділити багато функцій, які підпадають під цю категорію. З них, зазначені нижче, були розглянуті для цього проекту:

- перенаправлення IFrame;
- налаштування рядка стану;
- відключення правою кнопкою миші;
- пересилання сайту.

Перенаправлення IFrame. IFrame – це тег HTML, який використовується для відображення додаткової веб-сторінки в тій, яка зараз відображається. Зловмисники можуть використовувати тег iframe і зробити його невидимим, тобто без меж рамки. У зв’язку з цим зловмисники використовують атрибут «frameBorder», який змушує браузер відображати візуальне розмежування.

Якщо iframe порожній або відповідь не знайдено, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
def iframe(response):
  if response == "":
      return 1
  else:
      if re.findall(r"[<iframe>|<frameBorder>]", response.text):
          return 0
      else:
          return 1
```

Налаштування рядка стану. Фішери можуть використовувати JavaScript, щоб показувати користувачам фальшиву URL-адресу в рядку стану. Щоб отримати цю функцію, необхідно отримати вихідний код веб-сторінки, зокрема подію «onMouseOver» і перевірити, чи він вносить будь-які зміни в рядку стану.

Якщо відповідь порожня або знайдено при наведенні курсора, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
def mouseOver(response):
  if response == "" :
    return 1
  else:
             if      re.findall("<script>.+onmouseover.+</script>",
response.text):
      return 1
    else:
      return 0
```

Відключення правою кнопкою миші. Фішери використовують JavaScript, щоб вимкнути функцію клацання правою кнопкою миші, щоб користувачі не могли переглядати та зберігати вихідний код веб-сторінки. Ця функція розглядається саме як «Використання onMouseOver, щоб приховати посилання». Тим не менш, для цієї функції ми шукатимемо подію «event.button==2» у вихідному коді веб-сторінки та перевіримо, чи вимкнено право клацання.

Якщо відповідь порожня або наведення миші не знайдено, цій функції призначається значення 1 (фішинг) або 0 (законне).

```
def rightClick(response):
  if response == "":
    return 1
  else:
    if re.findall(r"event.button ?== ?2", response.text):
      return 0
    else:
      return 1
```

Пересилання сайту. Тонка грань, яка відрізняє фішингові веб-сайти від легальних, полягає в тому, скільки разів веб-сайт був перенаправлений. У нашому наборі даних ми виявили, що законні веб-сайти були переспрямовані

максимум один раз. З іншого боку, фішингові веб-сайти, що містять цю функцію, були перенаправлені щонайменше 4 рази.

```
def forwarding(response):
  if response == "":
    return 1
  else:
    if len(response.history) <= 2:
      return 0
    else:
      return 1
```

Обчислення особливостей URL. Створено список і функцію, яка викликає інші функції та зберігає всі функції URL у списку. Витягнемо характеристики кожної URL-адреси та додамо до цього списку.

```
def featureExtraction(url,label):

  features = []
  features.append(getDomain(url))
  features.append(havingIP(url))
  features.append(haveAtSign(url))
  features.append(getLength(url))
  features.append(getDepth(url))
  features.append(redirection(url))
  features.append(httpDomain(url))
  features.append(tinyURL(url))
  features.append(prefixSuffix(url))

  dns = 0
  try:
    domain_name = whois.whois(urlparse(url).netloc)
  except:
    dns = 1

  features.append(dns)
  features.append(web_traffic(url))
  features.append(1 if dns == 1 else domainAge(domain_name))
  features.append(1 if dns == 1 else domainEnd(domain_name))

  try:
    response = requests.get(url)
  except:
    response = ""
  features.append(iframe(response))
  features.append(mouseOver(response))
  features.append(rightClick(response))
  features.append(forwarding(response))
  features.append(label)
  return features
```

Тепер вилучення функцій здійснюється на законних URL-адресах.

Розмірність набору законних адрес становить 5000 стрічок на 1 стовпець. Сформований датафрейм зображено на рисунку 3.5.

```
legi_features = []
label = 0
for i in range(0, 5000):
  url = legiurl['URLs'][i]
  legi_features.append(featureExtraction(url,label))
feature_names = ['Domain', 'Have_IP', 'Have_At', 'URL_Length',
'URL_Depth','Redirection',
                        'https_Domain', 'TinyURL', 'Prefix/Suffix',
'DNS_Record', 'Web_Traffic',
                                'Domain_Age', 'Domain_End', 'iFrame',
'Mouse_Over','Right_Click', 'Web_Forwards', 'Label']
legitimate = pd.DataFrame(legi_features, columns= feature_names)
legitimate.head()
```

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Domain_End	iFrame	Mouse_Over	Right_Click	Web_Forwards	Label
0	graphicriver.net	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	0
1	ecnavi.jp	0	0	1	1	1	0	0	0	0	1	1	1	1	1	1	1	0
2	hubpages.com	0	0	1	1	0	0	0	0	0	1	0	0	1	1	1	1	0
3	extratorrent.cc	0	0	1	3	0	0	0	0	0	0	0	1	1	1	1	1	0
4	icicibank.com	0	0	1	3	0	0	0	0	0	1	0	1	1	1	1	1	0

Рисунок 3.5 – Вигляд сформованого датафрейму законних адрес

Запишемо датафрейм у файл для подальшої роботи з ним.

```
legitimate.to_csv('legitimate.csv', index= False)
```

Тепер вилучення функцій виконується на фішингових URL-адресах. Вигляд датафрейму фішингових адрес зображено на рисунку 3.6.

```
phish_features = []
label = 1
for i in range(0, 5000):
  url = phishurl['url'][i]
  phish_features.append(featureExtraction(url,label))
feature_names = ['Domain', 'Have_IP', 'Have_At', 'URL_Length',
'URL_Depth','Redirection',
                        'https_Domain', 'TinyURL', 'Prefix/Suffix',
'DNS_Record', 'Web_Traffic',
                                'Domain_Age', 'Domain_End', 'iFrame',
'Mouse_Over','Right_Click', 'Web_Forwards', 'Label']

phishing = pd.DataFrame(phish_features, columns= feature_names)
phishing.head()
```

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Domain_End	iFrame	Mouse_Over	Right_Click	Web_Forwards	Label
0	confirmprofileaccount.com	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1	1
1	marreme.com	0	0	0	2	0	0	0	0	0	1	0	0	1	1	1	1	1
2	modsecpaststudents.com	0	0	0	1	0	0	0	0	0	1	1	1	1	1	1	1	1
3	docs.google.com	0	0	1	5	0	0	0	0	0	1	1	1	1	1	1	1	1
4	oportunidadedasemana.com	0	0	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1

Рисунок 3.6 – Вигляд сформованого датафрейму фішингових адрес

Збережемо у файл датафрейм фішингових адрес для подальшої роботи.

```
phishing.to_csv('phishing.csv', index= False)
```

Датафрейм має різні записи у своїх полях. Умови присвоювання значень полям у датафреймі у таблиці 3.1.

Таблиця 3.1 – Критерії оцінювання полів відповідно до значень метрик

Бали / Параметри	0	1
IP-адреса в URL	URL-адреса не є IP-адресою	URL-адреса є IP-адресою
Символ "@" в URL-адресі	в URL-адресі нема @	в URL-адресі є @
Довжина URL-адреси	довжина URL-адреси менша за 53 символи	довжина URL-адреси більша за 53 символи
Глибина URL-адреси	Проставляється число, що є глибиною даної URL-адреси	
Перенаправлення "//" в URL	якщо // немає будь-де, окрім після протоколу	якщо // знаходиться буде, окрім як після протоколу
"http/https" в імені домену	якщо http/https нема в самому імені домену	якщо є http/https в самому імені домену
Використання служб скорочення URL-адрес «TinyURL»	якщо URL-адреса не є скорочена	якщо URL-адреса є скороченням
Префікс або суфікс «-» у домені	якщо нема символу - в доменній частині адреси	якщо є символ - в доменній частині адреси
Запис DNS	якщо є DNS-запис про URL-адресу	якщо нема DNS-запису про URL-адресу
Трафік веб-сайту	якщо URL-адреса записана у БД Alexa	якщо про URL-адресу нема даних у БД Alexa
Вік домену	якщо вік домену URL-адреси більше 12 місяців	якщо вік домену URL-адреси менше 12 місяців
Кінцевий період домену	URL-адреса закінчується менше ніж за 6 місяців	URL-адреса закінчується більше ніж за 6 місяців
iFrame	є відповідь на запит	якщо iFrame пустий або нема відповіді
Mouse_Over	є негативна відповідь на подію onMouseOver	якщо відповідь пуста чи виявлено onMouseOver
Right_Click	є відповідь на подію event.button==2	якщо відповідь пуста чи виявлено onMouseOver
Web_Forwards	якщо перенапревлення менше 4 раз або його немає	якщо є перенаправлення мінімум 4 рази

У наведеному вище розділі сформовано два фрейми даних легітимних і фішингових URL-адрес. Тепер об’єднаємо їх в єдиний фрейм даних і експортуємо дані у файл csv для навчання моделі.

```
urldata = pd.concat([legitimate, phishing]).reset_index(drop=True)
urldata.head()
urldata.shape
```

Розмірність набору складає 10000 стрічок на 18 стовців.

Даними діями досягається мета підготовки набору даних. Нарешті отримано 18 особливостей для 10 000 URL-адрес, які мають 5000 фішингових і 5000 законних URL-адрес.

3.2 Дії по навчанню моделі

Для навчання моделі буде використовуватись набір даних urldata.csv.

Спочатку завантажимо дані файлу urldata.csv.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

data0 = pd.read_csv('5.urldata.csv')
data0.head()
```

Середовище розробки по замовчуванню відображає перших п’ять записів з набору даних. Структура набору даних файлу urldata.csv зображена на рисунку 3.7.

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Domain_End	iFrame	Mouse_Over	Right_Click	Web_Forwards	Label
0	graphicriver.net	0	0	1	1	0	0	0	0	0	1	1	1	0	0	1	0	0
1	ecnavi.jp	0	0	1	1	1	0	0	0	0	1	1	1	0	0	1	0	0
2	hubpages.com	0	0	1	1	0	0	0	0	0	1	0	1	0	0	1	0	0
3	extratorrent.cc	0	0	1	3	0	0	0	0	0	1	0	1	0	0	1	0	0
4	icicibank.com	0	0	1	3	0	0	0	0	0	1	0	1	0	0	1	0	0

Рисунок 3.7 – Вигляд даних файлу urldata.csv

Далі необхідно ознайомитися з даними. На цьому кроці використовується кілька методів фрейму даних, щоб розглянути дані та їх особливості.

```
data0.shape
```

```
data0.columns
```

Визначили, що розмір даних 10000 рядків і 18 стовбців, також визначили якими стовпцями будемо надалі оперувати (рис. 3.8).

```
data0.shape
```

```
(10000, 18)
```

```
data0.columns
```

```
Index(['Domain', 'Have_IP', 'Have_At', 'URL_Length', 'URL_Depth',
       'Redirection', 'https_Domain', 'TinyURL', 'Prefix/Suffix', 'DNS_Record',
       'Web_Traffic', 'Domain_Age', 'Domain_End', 'iFrame', 'Mouse_Over',
       'Right_Click', 'Web_Forwards', 'Label'],
      dtype='object')
```

Рисунок 3.8 – Вигляд розмірності та структури даних набору urldata.csv

Визначаємо як розподілені дані та як функції пов’язані одна з одною. Для легшого оперування даними представимо це у візуальні формі (рис. 3.9).

```
data0.hist(bins = 50,figsize = (15,15))
plt.show()
```

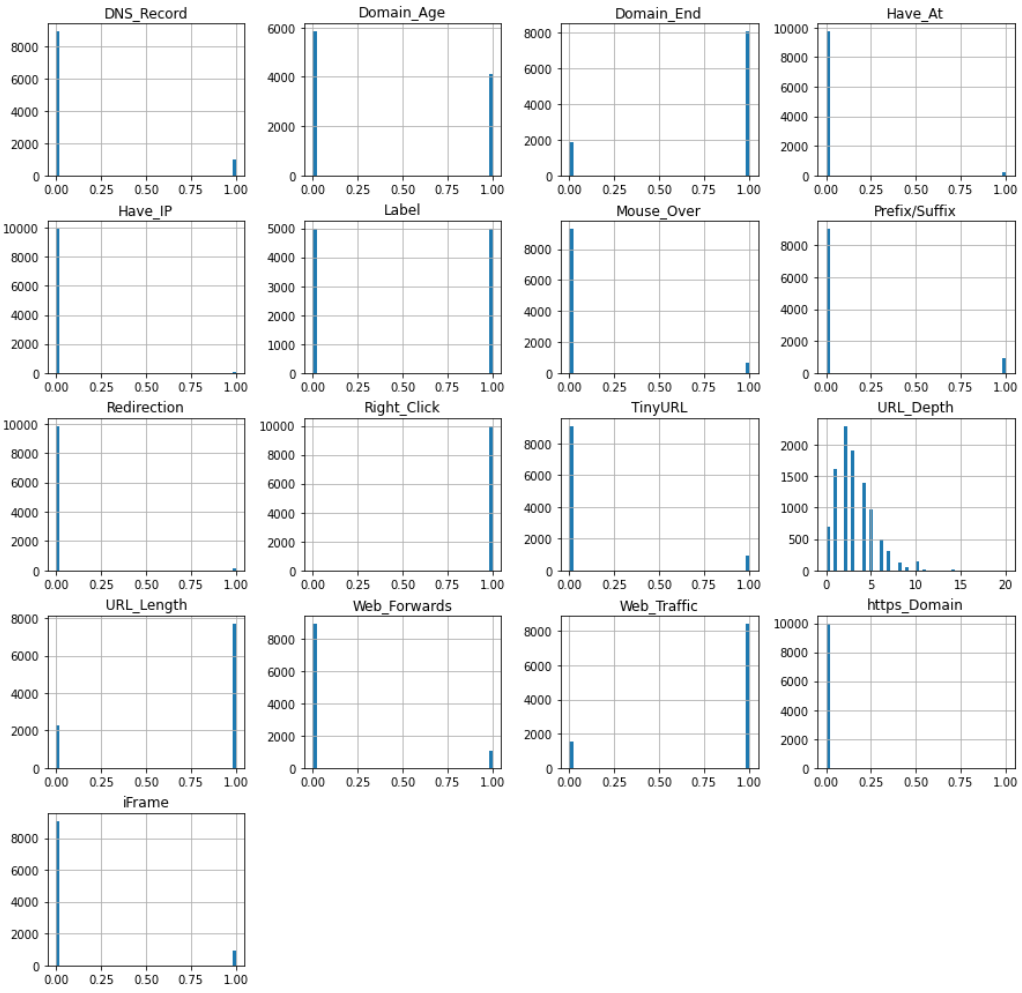


Рисунок 3.9 – Вигляд розподілення даних у наборі urldata.csv

Далі виконаємо попередню обробку даних. Очистимо дані, застосовуючи методи попередньої обробки даних і перетворимо дані, щоб використовувати їх у моделях (рис. 3.10).

```
data0.describe()
```

```
data0.describe()
```

Python

	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	0.005500	0.022600	0.773400	3.072000	0.013500	0.000200	0.090300	0.093200	0.100800
std	0.073961	0.148632	0.418653	2.128631	0.115408	0.014141	0.286625	0.290727	0.301079
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	1.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	1.000000	3.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	0.000000	0.000000	1.000000	4.000000	0.000000	0.000000	0.000000	0.000000	0.000000
max	1.000000	1.000000	1.000000	20.000000	1.000000	1.000000	1.000000	1.000000	1.000000

```
data0.describe()
```

Python

Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Domain_End	iFrame	Mouse_Over	Right_Click	Web_Forwards	Label
0000.000000	10000.000000	10000.000000	10000.000000	10000.0000	10000.000000	10000.00000	10000.00000	10000.000000	10000.000000
0.093200	0.100800	0.845700	0.413700	0.8099	0.090900	0.06660	0.99930	0.105300	0.500000
0.290727	0.301079	0.361254	0.492521	0.3924	0.287481	0.24934	0.02645	0.306955	0.500025
0.000000	0.000000	0.000000	0.000000	0.0000	0.000000	0.00000	0.00000	0.000000	0.000000
0.000000	0.000000	1.000000	0.000000	1.0000	0.000000	0.00000	1.00000	0.000000	0.000000
0.000000	0.000000	1.000000	0.000000	1.0000	0.000000	0.00000	1.00000	0.000000	0.500000
0.000000	0.000000	1.000000	1.000000	1.0000	0.000000	0.00000	1.00000	0.000000	1.000000
1.000000	1.000000	1.000000	1.000000	1.0000	1.000000	1.00000	1.00000	1.000000	1.000000

Рисунок 3.10 – Вигляд статистичних даних про стовпці в наборі даних urldata.csv

Отриманий вище результат показує, що більшість даних складається з 0 і 1, за винятком стовпців «Domain» і «URL_Depth». Стовпець Domain не має жодного значення для навчання моделі, тому вилучимо стовпець Domain з набору даних.

```
data = data0.drop(['Domain'], axis = 1).copy()
```

Далі перевіримо чи у даних, що залишились, немає пустих значень (рис. 3.11).

```
data.isnull().sum()
```

```
data.isnull().sum()
```

```
Have_IP          0
Have_At          0
URL_Length       0
URL_Depth        0
Redirection      0
https_Domain     0
TinyURL          0
Prefix/Suffix    0
DNS_Record       0
Web_Traffic      0
Domain_Age       0
Domain_End       0
iFrame           0
Mouse_Over       0
Right_Click      0
Web_Forwards     0
Label            0
dtype: int64
```

Рисунок 3.11 – Вигляд перевірки даних на пусті значення

У даному файлі поля наборів даних легітимних і фішингових URL-адрес просто об’єднуються без перемішування. Це призвело до 5000 верхніх рядків легітимних даних URL-адрес та 5000 нижчих рядків даних про фішингові URL-адреси. Щоб вирівняти розподіл, розділяючи дані на набори для навчання та тестування потрібно перемішати їх. Це навіть допоможе уникнути перенавчання під час навчання моделі. Перемішаємо рядки у наборі даних, щоб під час розділення на тренувальний і тестовий набір, дані були розподілені порівну (рис. 3.12).

```
data = data.sample(frac=1).reset_index(drop=True)
data.head()
```

```
data = data.sample(frac=1).reset_index(drop=True)
data.head()
```

	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Domain_End	iFrame	Mouse_Over	Right_Click	Web_Forwards	Label
0	0	0	1	2	0	0	0	1	0	1	0	1	0	0	1	0	1
1	0	0	1	5	0	0	0	0	0	1	0	1	0	0	1	1	0
2	0	0	1	1	0	0	0	0	0	1	1	0	0	0	1	0	0
3	0	0	1	1	0	0	1	0	0	1	1	1	0	0	1	0	0
4	0	0	0	0	0	0	0	1	0	1	0	1	0	0	1	0	1

Рисунок 3.12 – Вигляд перемішених даних

З наведеного вище виконання зрозуміло, що дані не мають жодних пропущених значень. Завдяки цьому дані ретельно обробляються і готові до використання.

Далі виконаємо розділення та призначення об’єктів і цільових стовпців X та y (рис. 3.13).

```
y = data['Label']
```

```
X = data.drop('Label',axis=1)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size = 0.2, random_state = 12)
X_train.shape, X_test.shape
```

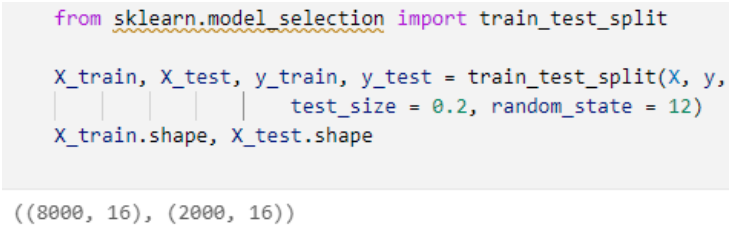


Рисунок 3.13 – Вигляд поділеного набору на тренувальний і тестовий

З набору даних вище видно, що це спосіб машинного навчання з вчителем. Існує два основних типи машинного навчання з вчителем, які називаються класифікацією та регресією. Цей набір даних піддається класифікації, оскільки вхідна URL-адреса класифікується як фішинг (1) або легітимна (0). Далі оберемо кращу класифікатора серед таких:

– Decision Tree;
– Random Forest;
– Multilayer Perceptrons;
– XGBoost;
– Autoencoder Neural Network;
– Support Vector Machines.

Одразу імпортуємо необхідні пакети для оцінювання класифікаторів.

```
from sklearn.metrics import accuracy_score

ML_Model = []
acc_train = []
acc_test = []

def storeResults(model, a,b):
  ML_Model.append(model)
  acc_train.append(round(a, 3))
  acc_test.append(round(b, 3))
```

Дерева рішень (Decision Tree) є широко використовуваними моделями для завдань класифікації та регресії. По суті, вони вивчають ієрархію запитань якщо/інакше, що веде до рішення. Вивчення дерева рішень означає вивчення послідовності запитань якщо/інакше, яка найшвидше приведе нас до істинної відповіді. На рисунку 3.14 зображено параметри, які має модель дерева рішень після навчання.

```
from sklearn.tree import DecisionTreeClassifier

tree = DecisionTreeClassifier(max_depth = 5)
tree.fit(X_train, y_train)
```

```
from sklearn.tree import DecisionTreeClassifier

# instantiate the model
tree = DecisionTreeClassifier(max_depth = 5)
# fit the model
tree.fit(X_train, y_train)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=5, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

Рисунок 3.14 – Вигляд параметрів дерева рішень

```
y_test_tree = tree.predict(X_test)
y_train_tree = tree.predict(X_train)
```

Оцінимо ефективність даного класифікатора (рис. 3.15).

```
acc_train_tree = accuracy_score(y_train,y_train_tree)
acc_test_tree = accuracy_score(y_test,y_test_tree)

print("Decision Tree: Accuracy on training Data:
{:.3f}".format(acc_train_tree))
print("Decision Tree: Accuracy on test Data:
{:.3f}".format(acc_test_tree))
```

```
acc_train_tree = accuracy_score(y_train,y_train_tree)
acc_test_tree = accuracy_score(y_test,y_test_tree)

print("Decision Tree: Accuracy on training Data: {:.3f}".format(acc_train_tree))
print("Decision Tree: Accuracy on test Data: {:.3f}".format(acc_test_tree))
```

```
Decision Tree: Accuracy on training Data: 0.810
Decision Tree: Accuracy on test Data: 0.826
```

Рисунок 3.15 – Вигляд оцінки дерева рішень

Перевіримо важливість функцій у моделі (рис. 3.16).

```
plt.figure(figsize=(9,7))
n_features = X_train.shape[1]
plt.barh(range(n_features), tree.feature_importances_,
align='center')
plt.yticks(np.arange(n_features), X_train.columns)
plt.xlabel("Feature importance")
plt.ylabel("Feature")
plt.show()
```

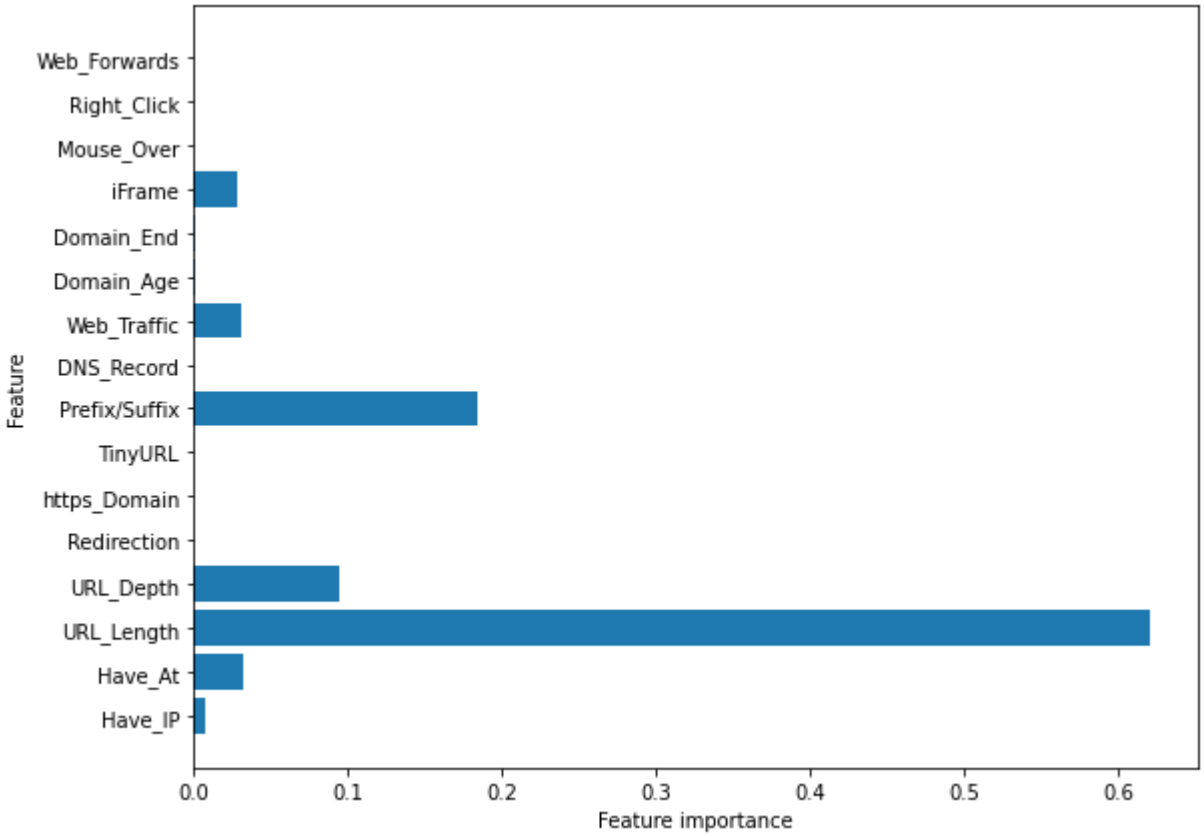


Рисунок 3.16 – Вигляд важливості стовпців для дерева рішень

Збережемо результати для подальшого порівняння.

```
storeResults('Decision Tree', acc_train_tree, acc_test_tree)
```

Випадкові ліси (Random Forest) для регресії та класифікації в даний час є одними з найбільш широко використовуваних методів машинного навчання. Випадковий ліс, по суті, є набір дерев рішень, де кожне дерево дещо відрізняється від інших. Ідея випадкових лісів полягає в тому, що кожне дерево може відносно добре передбачати, але, ймовірно, переповнює частину даних.

Якщо побудуємо багато дерев, усі з яких добре працюють і перенасичуються по-різному, можемо зменшити кількість перенасичення, усереднюючи їх результати (рис. 3.17).

```
from sklearn.ensemble import RandomForestClassifier

forest = RandomForestClassifier(max_depth=5)

forest.fit(X_train, y_train)
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=5, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

Рисунок 3.17 – Вигляд параметрів випадкового лісу

```
y_test_forest = forest.predict(X_test)
y_train_forest = forest.predict(X_train)
```

Виконаємо оцінку ефективності (рис. 3.18).

```
acc_train_forest = accuracy_score(y_train,y_train_forest)
acc_test_forest = accuracy_score(y_test,y_test_forest)

print("Random forest: Accuracy on training Data:
{:.3f}".format(acc_train_forest))
print("Random forest: Accuracy on test Data:
{:.3f}".format(acc_test_forest))
```

```
Random forest: Accuracy on training Data: 0.814
Random forest: Accuracy on test Data: 0.834
```

Рисунок 3.18 – Вигляд оцінки ефективності випадкового лісу

Перевірка важливості стовпців (рис. 3.19).

```
plt.figure(figsize=(9,7))
n_features = X_train.shape[1]
plt.barh(range(n_features), forest.feature_importances_,
align='center')
plt.yticks(np.arange(n_features), X_train.columns)
plt.xlabel("Feature importance")
plt.ylabel("Feature")
plt.show()
```

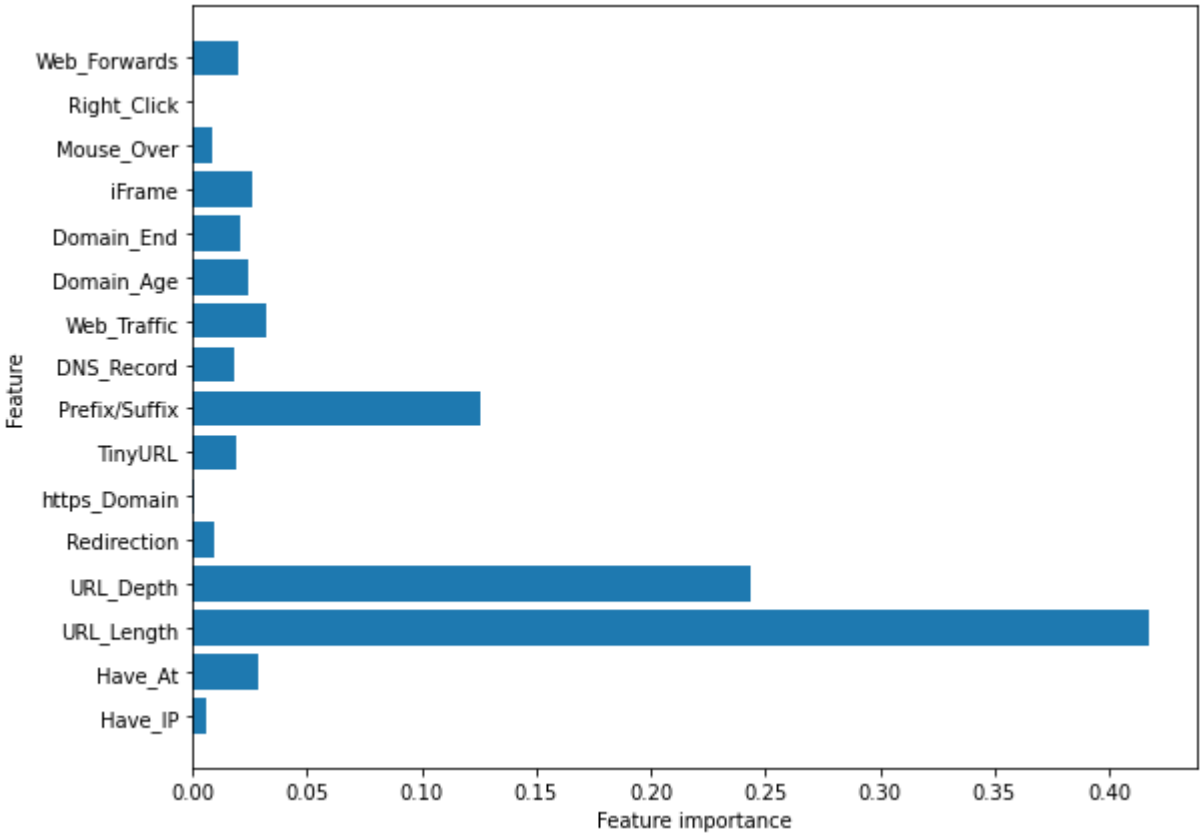


Рисунок 3.19 – Вигляд важливості стовпців для випадкового лісу

Збережемо результати оцінювання.

```
storeResults('Random Forest', acc_train_forest, acc_test_forest)
```

Багатошарові персептрони – це нейронні мережі з прямим зв’язком або іноді просто нейронні мережі. Багатошарові персептрони можна застосовувати як для задач класифікації, так і для задач регресії. MLP можна розглядати як узагальнення лінійних моделей, які виконують кілька етапів обробки для прийняття рішення (рис. 3.20).

```
from sklearn.neural_network import MLPClassifier

mlp = MLPClassifier(alpha=0.001,
hidden_layer_sizes=([100,100,100]))

mlp.fit(X_train, y_train)
```

```
MLPClassifier(activation='relu', alpha=0.001, batch_size='auto', beta_1=0.9,
              beta_2=0.999, early_stopping=False, epsilon=1e-08,
              hidden_layer_sizes=[100, 100, 100], learning_rate='constant',
              learning_rate_init=0.001, max_fun=15000, max_iter=200,
              momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True,
              power_t=0.5, random_state=None, shuffle=True, solver='adam',
              tol=0.0001, validation_fraction=0.1, verbose=False,
              warm_start=False)
```

Рисунок 3.20 – Вигляд параметрів багатошарового персептрону

Прогнозування цільового значення з моделі для зразків.

```
y_test_mlp = mlp.predict(X_test)
y_train_mlp = mlp.predict(X_train)
```

Оцінка ефективності багатошарового персептрону (рис. 3.21).

```
acc_train_mlp = accuracy_score(y_train,y_train_mlp)
acc_test_mlp = accuracy_score(y_test,y_test_mlp)

print("Multilayer Perceptrons: Accuracy on training Data:
{:.3f}".format(acc_train_mlp))
print("Multilayer Perceptrons: Accuracy on test Data:
{:.3f}".format(acc_test_mlp))
```

```
Multilayer Perceptrons: Accuracy on training Data: 0.859
Multilayer Perceptrons: Accuracy on test Data: 0.863
```

Рисунок 3.21 – Вигляд оцінки багатошарового персептрону

Збережемо результати

```
storeResults('Multilayer Perceptrons', acc_train_mlp,
acc_test_mlp)
```

XGBoost є одним з найпопулярніших алгоритмів машинного навчання сьогодні. XGBoost означає eXtreme Gradient Boosting. Незалежить від типу передбачуваного завдання регресія або класифікація. XGBoost – це реалізація дерев рішень із підсилюваними градієнтами, розробленими для швидкості та продуктивності (рис. 3.22).

```
from xgboost import XGBClassifier

xgb = XGBClassifier(learning_rate=0.4,max_depth=7)
xgb.fit(X_train, y_train)
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0,
              learning_rate=0.4, max_delta_step=0, max_depth=7,
              min_child_weight=1, missing=None, n_estimators=100, n_jobs=1,
              nthread=None, objective='binary:logistic', random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
              silent=None, subsample=1, verbosity=1)
```

Рисунок 3.22 – Вигляд параметрів XGBoost

Прогнозування цільового значення з моделі для зразків.

```
y_test_xgb = xgb.predict(X_test)
y_train_xgb = xgb.predict(X_train)
```

Оцінювання ефективності XGBoost (рис. 3.23).

```
acc_train_xgb = accuracy_score(y_train,y_train_xgb)
acc_test_xgb = accuracy_score(y_test,y_test_xgb)

print("XGBoost: Accuracy on training Data:
{:.3f}".format(acc_train_xgb))
print("XGBoost : Accuracy on test Data:
{:.3f}".format(acc_test_xgb))
```

```
XGBoost: Accuracy on training Data: 0.866
XGBoost : Accuracy on test Data: 0.864
```

Рисунок 3.23 – Вигляд оцінки XGBoost

Збереження результатів XGBoost.

```
storeResults('XGBoost', acc_train_xgb, acc_test_xgb)
```

Автокодер – це нейронна мережа, яка має таку ж кількість вхідних нейронів, як і вихідних. Приховані шари нейронної мережі матимуть менше нейронів, ніж нейрони входу/виходу. Оскільки нейронів менше, автокодер повинен навчитися кодувати вхід до меншої кількості прихованих нейронів. Предиктори (x) і вихід (y) абсолютно однакові в автоматичному кодері.

```
import keras
from keras.layers import Input, Dense
from keras import regularizers
import tensorflow as tf
from keras.models import Model
from sklearn import metrics
```

Побудова моделі автокодеру (рис. 3.24).

```
input_dim = X_train.shape[1]
encoding_dim = input_dim

input_layer = Input(shape=(input_dim, ))
encoder = Dense(encoding_dim, activation="relu",
                activity_regularizer=regularizers.l1(10e-
4))(input_layer)
encoder = Dense(int(encoding_dim), activation="relu")(encoder)

encoder = Dense(int(encoding_dim-2), activation="relu")(encoder)
code = Dense(int(encoding_dim-4), activation='relu')(encoder)
decoder = Dense(int(encoding_dim-2), activation='relu')(code)

decoder = Dense(int(encoding_dim), activation='relu')(encoder)
decoder = Dense(input_dim, activation='relu')(decoder)
autoencoder = Model(inputs=input_layer, outputs=decoder)
autoencoder.summary()
```

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 16)                0
_________________________________________________________________
dense_1 (Dense)              (None, 16)                272
_________________________________________________________________
dense_2 (Dense)              (None, 16)                272
_________________________________________________________________
dense_3 (Dense)              (None, 14)                238
_________________________________________________________________
dense_6 (Dense)              (None, 16)                240
_________________________________________________________________
dense_7 (Dense)              (None, 16)                272
=================================================================
Total params: 1,294
Trainable params: 1,294
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 3.24 – Вигляд моделі автокодеру

```
autoencoder.compile(optimizer='adam',
                    loss='binary_crossentropy',
                    metrics=['accuracy'])
history = autoencoder.fit(X_train, X_train, epochs=10,
batch_size=64, shuffle=True, validation_split=0.2)
```

Епохи тренувань автокодеру зображені на рисунку 3.25.

```
Train on 6400 samples, validate on 1600 samples
Epoch 1/10
6400/6400 [==============================] - 0s 51us/step - loss: 1.3997 - accuracy: 0.7132 - val_loss: -0.3941 - val_accuracy: 0.7890
Epoch 2/10
6400/6400 [==============================] - 0s 24us/step - loss: -0.4269 - accuracy: 0.7821 - val_loss: -0.5190 - val_accuracy: 0.7812
Epoch 3/10
6400/6400 [==============================] - 0s 24us/step - loss: -1.0514 - accuracy: 0.7908 - val_loss: -1.3147 - val_accuracy: 0.8149
Epoch 4/10
6400/6400 [==============================] - 0s 24us/step - loss: -1.3118 - accuracy: 0.8200 - val_loss: -1.3532 - val_accuracy: 0.8128
Epoch 5/10
6400/6400 [==============================] - 0s 25us/step - loss: -1.3789 - accuracy: 0.8168 - val_loss: -1.4710 - val_accuracy: 0.8190
Epoch 6/10
6400/6400 [==============================] - 0s 25us/step - loss: -1.4435 - accuracy: 0.8187 - val_loss: -1.5160 - val_accuracy: 0.8204
Epoch 7/10
6400/6400 [==============================] - 0s 25us/step - loss: -1.4951 - accuracy: 0.8215 - val_loss: -1.5601 - val_accuracy: 0.8240
Epoch 8/10
6400/6400 [==============================] - 0s 23us/step - loss: -1.5208 - accuracy: 0.8192 - val_loss: -1.5912 - val_accuracy: 0.8236
Epoch 9/10
6400/6400 [==============================] - 0s 25us/step - loss: -1.5044 - accuracy: 0.8140 - val_loss: -1.5868 - val_accuracy: 0.8191
Epoch 10/10
6400/6400 [==============================] - 0s 25us/step - loss: -1.5554 - accuracy: 0.8214 - val_loss: -1.6153 - val_accuracy: 0.8205
```

Рисунок 3.25 – Вигляд тренування автокодеру

Оцінювання ефективності автокодеру (рис. 3.26).

```
acc_train_auto = autoencoder.evaluate(X_train, X_train)[1]
acc_test_auto = autoencoder.evaluate(X_test, X_test)[1]

print('\nAutoencoder: Accuracy on training Data: {:.3f}'
.format(acc_train_auto))
print('Autoencoder: Accuracy on test Data: {:.3f}'
.format(acc_test_auto))
```

```
8000/8000 [==============================] - 0s 18us/step
2000/2000 [==============================] - 0s 20us/step

Autoencoder: Accuracy on training Data: 0.819
Autoencoder: Accuracy on test Data: 0.818
```

Рисунок 3.26 – Вигляд оцінки автокодеру

Збереження результату автокодеру.

```
storeResults('AutoEncoder', acc_train_auto, acc_test_auto)
```

У машинному навчанні машини опорних векторів (SVM, також мережі опорних векторів) – це моделі навчання з вчителем з відповідними алгоритмами навчання, які аналізують дані, що використовуються для класифікації та регресійного аналізу. З огляду на набір навчальних прикладів, кожен із яких позначено як належний до однієї чи іншої з двох категорій, навчальний алгоритм SVM будує модель, яка призначає нові приклади до тієї чи іншої категорії, роблячи її неімовірнісним бінарним лінійним класифікатором (рис. 3.27).

```
from sklearn.svm import SVC

svm = SVC(kernel='linear', C=1.0, random_state=12)
svm.fit(X_train, y_train)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
    max_iter=-1, probability=False, random_state=12, shrinking=True, tol=0.001,
    verbose=False)
```

Рисунок 3.27 – Вигляд класифікатору SVM

Прогнозування цільового значення з моделі для зразків.

```
y_test_svm = svm.predict(X_test)
y_train_svm = svm.predict(X_train)
```

Оцінка ефективності SVM (рис. 3.28).

```
acc_train_svm = accuracy_score(y_train,y_train_svm)
acc_test_svm = accuracy_score(y_test,y_test_svm)

print("SVM: Accuracy on training Data:
{:.3f}".format(acc_train_svm))
print("SVM : Accuracy on test Data: {:.3f}".format(acc_test_svm))
```

```
SVM: Accuracy on training Data: 0.798
SVM : Accuracy on test Data: 0.818
```

Рисунок 3.28 – Вигляд оцінки SVM

Збереження результатів SVM.

```
storeResults('SVM', acc_train_svm, acc_test_svm)
```

Порівняємо наведені класифікатори. Для порівняння продуктивності моделей створимо фрейм даних. Стовпці цього фрейму даних є списками, створеними для зберігання результатів моделі.

```
results = pd.DataFrame({ 'ML Model': ML_Model,
    'Train Accuracy': acc_train,
    'Test Accuracy': acc_test})
results
```

	ML Model	Train Accuracy	Test Accuracy
0	Decision Tree	0.810	0.826
1	Random Forest	0.814	0.834
2	Multilayer Perceptrons	0.858	0.863
3	XGBoost	0.866	0.864
4	AutoEncoder	0.819	0.818
5	SVM	0.798	0.818

Рисунок 3.29 – Вигляд оцінки класифікаторів

Відсортуємо по спаданню (рис. 3.30).

```
results.sort_values(by=['Test Accuracy', 'Train Accuracy'], ascending=False)
```

	ML Model	Train Accuracy	Test Accuracy
3	XGBoost	0.866	0.864
2	Multilayer Perceptrons	0.858	0.863
1	Random Forest	0.814	0.834
0	Decision Tree	0.810	0.826
4	AutoEncoder	0.819	0.818
5	SVM	0.798	0.818

Рисунок 3.30 – Відсортовані оцінки класифікаторів

З наведеного вище порівняння зрозуміло, що класифікатор XGBoost найкраще працює з цим набором даних. Отже, зберігаємо модель для подальшого використання.

```
import pickle
pickle.dump(xgb, open("XGBoostClassifier.pickle.dat", "wb"))
```

Перевіримо збережену модель (рис. 3.30).

```
loaded_model = pickle.load(open("XGBoostClassifier.pickle.dat", "rb"))
loaded_model
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0,
              learning_rate=0.4, max_delta_step=0, max_depth=7,
              min_child_weight=1, missing=nan, n_estimators=100, n_jobs=1,
              nthread=None, objective='binary:logistic', random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
              silent=None, subsample=1, verbosity=1)
```

Рисунок 3.31 – Вигляд збереженої моделі класифікатора XGBoost

Таким чином, алгортим роботи програми можна відобразити у вигляді блок-схеми на (рис. 3.31)

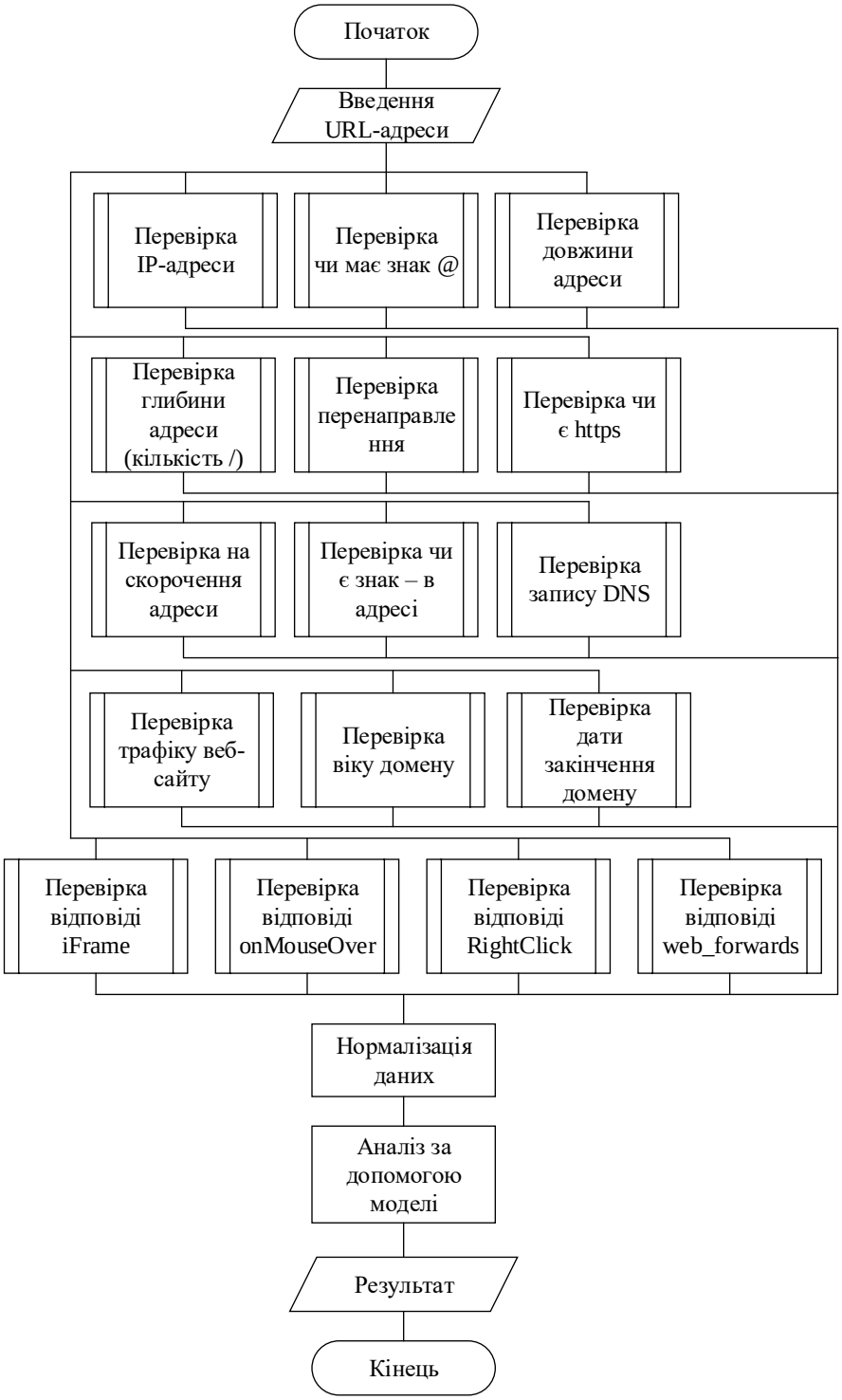


Рисунок 3.32 – Схема роботи програмного засобу

У даному розділі були опрацьовані набори даних справжніх та фішингових адрес для приведення їх у потрібний стан для навчання. Протестовано п’ять моделей для машинного навчання та по результатам сортування обрано XGBoost як модель, на основі якої буде створено програмний засіб виявлення фішингових інформаційних ресурсів.

4 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ

4.1 Тестування програмного засобу

Для тестування роботи програмного засобу розроблено тестові випадки для перевірки поведінки програми під час виключних випадків. До таких випадків належать введення справжньої та фішингової URL-адреси, реакція засобу на присутність та відсутність XGBoostClassifier.pickle.dat, а також перевірка роботи з коректними та некоректними даними у файлу XGBoostClassifier.pickle.dat (табл. 4.1).

Таблиця 4.1 – Тестові випадки відпрацювання засобу

№	Назва тестового випадку	Очікувана дія програми	Чи пройшов засіб тест
1	Введення коректної URL-адреси	Розбиття за полями	Так
2	Введення некоректної URL-адреси	Повідомлення про помилку	Так
3	Навчання моделі, коли файл XGBoostClassifier.pickle.dat існує	Навчання моделі	Так
4	Аналіз URL-адреси, коли файл XGBoostClassifier.pickle.dat не існує	Повідомлення про помилку	Так
5	Навчання моделі за тестовою вибіркою, що містить коректні дані	Навчання моделі	Так
6	Навчання моделі за тестовою вибіркою, що містить некоректні дані	Повідомлення про помилку	Так

Засіб має вигляд командної стрічки. На рисунку 4.1 зображено вікно програми після перевірки URL-адреси. URL-адреса є легальною. У результаті перевірки програмний засіб розпізнав URL-адресу як легальну. Про це свідчить зелена група (рис. 4.1).

```
PS D:\Prjs\projectd> & C:/Users/G/AppData/Local/Programs/Python/Python310/python.exe d:/Prjs/projectd/demo.py
########################################################################
URL: https://vntu.edu.ua/
Ймовірність шкідливої URL-адреси для фішингу: 0.243
Сайт https://vntu.edu.ua/ знаходиться у зелена/ій групі
PS D:\Prjs\projectd>
```

Рисунок 4.1 – Вигляд засобу під час перевірки легальноїURL-адреси

Також для перевірки роботи програмного засобу обрано фішингову URL-адресу, що має URL https://atb-market.space/ (рис. 4.2).

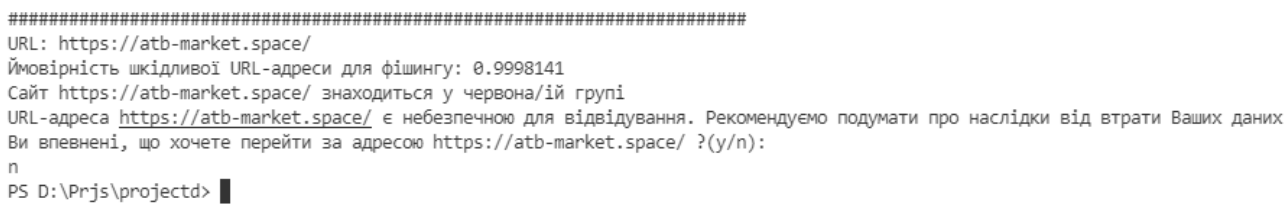


Рисунок 4.2 – Вигляд засобу під час перевірки фішингової URL-адреси

Програмний засіб аналізує такі параметри: чи є введена URL-адреса IP-адресою, чи містить знак @, чи не перевищує довжина URL-адреси 54 символи, чи не є URL-адреса занадто глубокою, чи нема перенаправлень у URL-адресі, чи вказано https у доменній частині адреси, чи не є URL-адреса скороченою, чи розділяється адреса символом -, чи має адреса запис у базі DNS, чи є веб-трафік за цією адресою, перевірка дати існування URL-адреси, перевірка датb закінчення існування URL-адреси, перевірка відповіді iFrame, перевірка відповіді на подію onMouseOver, перевірка на натискання правої кнопки миші, перевірка відповіді webforwards.

Отже, перевірено роботу функціоналу програмного засобу та протестовано його роботу під час виникнення виключних ситуацій та помилок, таких як відсутність навчального файлу для моделі, а також під час введення некоректної URL-адреси.

Ефективність розробленого програмного засобу оцінюється за правильністю виявлення легальних та фішингових URL-адрес та за кількістю помилок першого та другого роду. Для цього перевірено сто URL-адрес. Результати аналізу десяти URL-адрес наведено у таблиці 4.2.

Таблиця 4.2 – Результати аналізу URL-адрес

URL-адреса	Статус адреси	Висновок програми
https://vntu.edu.ua/	легальна	легальна
https://atb-market.space/	фішингова	фішингова
https://paypal.com.us.home.login.page.loginchecks.xyz/	фішингова	фішингова

Продовження таблиці 4.2

https://secure.login.aliexpress.com.coin-balance.com/entscheidung-817277406681628&cgi3-viewkontakt-817277406681628-007acctpagetype-817277406681628=33445-gesendet&amp	фішингова	фішингова
https://evernote.com/register?upgrade=premium&itemCode=premium-1year&offer=www_pricing_CTA	фішингова	фішингова
https://itc.ua/news/monobank-vzhe-vidkriv-15-mln-kartok/	легальна	фішингова
https://www.overclockers.ua/	легальна	легальна
https://www.millionsfree.site/	фішингова	легальна
https://www.prom.ua.rybachok.online/	фішингова	фішингова
https://letiletiavia.online/	фішингова	легальна

Протестувавши засіб на різни URL-адресах, потрібно визначити помилки першого та другого порядку.

4.2 Перевірка достовірності роботи засобу

Після аналізу необхідно перевірити достовірність роботи програми, визначивши помилки першого та другого роду. Для цього застосовано перевірку гіпотез та введено два поняття: нульова гіпотеза H_0 та альтернативна гіпотеза H_1. Нульова гіпотеза H_0 – це гіпотеза про те, що URL-адреса є легальною, а альтернативна гіпотеза H_1 – URL-адреса є фішинговою. Таким чином, якщо приймається нульова гіпотеза H_0, то робиться висновок, що URL-адреса легальна. Якщо приймається альтернативна гіпотеза H_1, то робиться висновок про те, що URL-адреса є фішинговою (табл. 4.3).

Таблиця 4.3 – Помилки першого і другого роду під час прийняття рішень

	H_0 приймається	H_0 не приймається, H_1 приймається
H_0 вірна	Правильно визначена легальна URL-адреса	Фішингова URL-адреса визначена як легальна
H_0 не вірна, H_1 вірна	Легальна URL-адреса визначена як фішингова	Правильно визначена фішингова URL-адреса

При перевірці гіпотези експериментальні дані можуть суперечити гіпотезі H_0, тоді ця гіпотеза відхиляється. В іншому випадку, якщо експериментальні дані не суперечать гіпотезі H_0, то ця гіпотеза не відхиляється.

Зрозуміло, що при перевірці гіпотез система прийняття рішень може з деякою імовірністю прийняти помилкове рішення. Існують такі помилки: помилка першого роду та помилка другого роду.

Помилка першого роду означає, що приймається рішення відхилити гіпотезу H_0, хоча в дійсності вона є вірною. Помилка другого роду означає, що приймається рішення не відхиляти гіпотезу H_0, хоча в дійсності вона буде невірна.

Помилки першого та другого роду для системи виявлення фейкових облікових записів становлять:

– помилка першого роду β = 0,05;

– помилка другого роду α = 0,08.

Тому, для оцінки роботи системи виявлення фейкових облікових записів у соціальних мережах доцільно визначити імовірності правильного визначення фейкових та справжніх облікових записів, а також імовірності появи помилок першого та другого роду.

Імовірність правильно визначеного справжнього облікового запису становить $P_{H_0}(H_0) = 1 - \alpha$ і дорівнює 92%.

Імовірність правильно визначеного фейкового облікового запису становить $P_{H_1}(H_1) = 1 - \beta$ і дорівнює 95%.

Імовірність появи помилки 1-го роду становить $P_{H_1}(H_0) = \beta$ і дорівнює 5%. Імовірність появи помилки 2-го роду становить $P_{H_0}(H_1) = \alpha$ і дорівнює 8%.

Перевірка точності роботи засобу виявлення фішинових інформаційних ресурсів показала, що помилки першого та другого роду дорівнюють 0,05 і 0,08 відповідно, а отже – система доволі точно виявляє легальні та фішингові облікові записи. Загальна достовірність системи виявлення виявлення фішинових інформаційних відповідно до отриманих результатів становить $P(H) = 1 - \alpha - \beta$ = 0,87 = 87%.

Очевидно, що система підтримки прийняття рішень може помилитися при ухваленні рішення. Виключити помилки повністю при прийнятті гіпотез

неможливо, тому необхідно мінімізувати можливі наслідки прийняття неправильного рішення (невірної статистичної гіпотези). Тому, для мінімізації можливих помилок у подальших дослідженнях необхідно збільшити обсяг вибірки.

Отже, розроблено програмний засіб, який реалізує процес виявлення фішинових інформаційних ресурсів. Проект складається з двох файлів, які, у свою чергу, складаються з класів і методів, а також з допоміжних файлів, у яких містититься навчальна вибірка для моделі та файл з моделлю. У результаті тестування виявлено, що програмний засіб майже завжди правильно виявляє легальні та фішингові URL-адреси, про що свідчить достовірність прийняття рішення, яка становить 87%. Із 100 досліджуваних URL-адрес програма правильно визначила статус для 87 URL-адрес. При цьому, помилки першого та другого роду становлять 5% і 8% відповідно. Для поліпшення результату роботи програмного засобу необхідно збільшити розмір навчальної вибірки, розширити кількість параметрів, що перевіряються.

5 ЕКОНОМІЧНА ЧАСТИНА

Метою економічної частини магістерської кваліфікаційної роботи є обґрунтування економічної доцільності розробки методу та засобу визначення фішингових інформаційних ресурсів. Економічна частина виконувалась згудно до методичних вказівок [23].

Для цього необхідно виконати такі етапи робіт:

- оцінити комерційний потенціал розробки;
- спрогнозувати витрати на виконання наукової роботи та впровадження її результатів;
- спрогнозувати комерційний ефект від реалізації результатів розробки;
- розрахувати ефективність вкладених інвестицій та період їх окупності.

5.1 Проведення комерційного та технологічного аудиту розробки

Об’єктом дослідження магістерської кваліфікаційної роботи є метод і засіб виявлення фішингових інформаційних ресурсів.

Для проведення технологічного аудиту було залучено трьох незалежних експертів: Каплун В.А., Баришев Ю. В., Войтович О. П.. Кожен з експертів повинен ознайомитися з запропонованою розробкою та заповнити таблицю, яка визначає рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можливу оцінку в балах. Після виконання цього, підраховується середньоарифметична сума балів та визначається який рівень комерційного потенціалу має нова розробка. Здійснюємо оцінювання комерційного потенціалу розробки за дванадцяттю критеріями, наведеними в таблиці 5.1.

Таблиця 5.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Критерій	0	1	2	3	4
Технічна здійсненність концепції:					

Продовження таблиці 5.1.

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Критерій	0	1	2	3	4
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї

Продовження таблиці 5.1.

9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання комерційного потенціалу розробки наведено в таблиці 5.2.

Таблиця 5.2 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали експерта		
	Каплун В. А.	Баришев Ю. В.	Войтович О. П.
	Бали, виставлені експертами:		
1	3	2	3
2	2	2	2

Продовження таблиці 5.2.

3	3	3	3
4	2	2	3
5	2	3	2
6	1	1	2
7	3	2	3
8	3	2	3
9	3	3	3
10	3	3	3
11	3	3	3
12	3	3	3
Сума балів	$СБ_1$=31	$СБ_2$=29	$СБ_3$=33
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_{1}^{i} СБ_i}{i} = \frac{31+29+33}{3} = 31$		

Отже, з отриманих даних таблиці 5.2 видно, що середньоарифметична сума балів дорівнює 31, тобто нова розробка має рівень комерційного потенціалу вище середнього. Використовувались рекомендації наведені в табл. 5.3.

Таблиця 5.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41...48	Високий
31...40	Вищий середнього
21...30	Середній
11...20	Нижчий середнього
0...10	Низький

На ринку є достатньо засобів для визначення фішингових інформаційних ресурсів, але кожен з них не може із 100% точністю визначити ресурси, що не є тривалоіснуючими і, здебільшого, правильно відпрацьовують на загальноговідомих ресурсах. Тому є необхідність в розробці нових методів та засобів визначення фішингових інформаційних ресурсів. Такий науково-технічний рівень та комерційний потенціал досягнуто за рахунок розширення функціональних можливостей нової розробки, шляхом аналізу інформаційного ресурсу у реальному часі, а не по існуючим даним, яких може і не бути.

5.2 Прогнозування витрат на виконання науково-дослідної та конструкторсько-технологічної роботи

Прогнозування витрат на виконання науково-дослідної роботи, дослідно-конструкторської та конструкторсько-технологічної роботи може складатися з таких етапів:

1. Розрахунок витрат, які безпосередньо стосуються виконавців даного розділу роботи;
2. Розрахунок загальних витрат на виконання роботи даної роботи;
3. Прогнозування загальних витрат на виконання та впровадження результатів даної роботи.

Розрахунок витрат на розробку методики дослідження здійснюється по статтях калькуляції, а саме:

- основна заробітна плата розробників;
- додаткова заробітна плата розробників;
- нарахування на заробітну плату розробників;
- амортизація персонального комп'ютеру;
- витрати на матеріали, що були використані в процесі розробки програмного продукту;
- витрати на електроенергію;
- інші витрати.

У розробці даного програмного продукту брав участь один спеціаліст та один науковий керівник. Основна заробітна плата для спеціаліста визначається за формулою:

$$З_О = \sum_{i=1}^{k} \frac{М_{ni} \cdot t_i}{Т_р}, \quad (5.1)$$

де k – кількість посад дослідників залучених до процесу досліджень;

$М_{ni}$ – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

$Т_р$ – середнє число робочих днів у місяці, $Т_р$ = 22 дні.

Оплата за робочий день розробника проекту становить 1111 грн, а наукового керівника – 1111 грн.

Заробітна плата наукового керівника:

$$З_{О_{НК}} = \frac{24200}{22} \times 7 = 1100 \times 7 = 7700 \text{ (грн)}$$

Заробітна плата розробника:

$$З_{О_Р} = \frac{24200}{22} \times 15 = 1100 \times 15 = 16500 \text{ (грн)}$$

Витрати на оплату праці, основна заробітна плата:

$$З_О = З_{О_{НК}} + З_{О_Р} = 7700 + 16500 = 24200 \text{ (грн)}$$

Розрахунки заробітної плати для дослідників наведені в таблиці 5.4.

Таблиця 5.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи, t	Витрати на оплату праці, грн
Науковий керівник проекту	24200	1100	7	7700
Розробник	24200	1100	15	16500
Всього:				24200

Витрати на основну заробітну плату робітників ($З_р$) за відповідним найменуваннями робіт розраховано за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i , \quad (5.2)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i визначено за формулою:

$$C_i = \frac{M_M \cdot K_i}{T_р \cdot t_{зм}} , \quad (5.3)$$

де M_M – розмір прожиткового мінімуму працездатної особи або мінімальної заробітної плати (в залежності від діючого законодавства), грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду;

T_p – середнє число робочих днів у місяці, T_p = 22 дні.

$t_{зм}$ – тривалість зміни, год.

Розрахунки заробітної плати робітників наведено в таблиці 5.5.

Таблиця 5.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника, грн
Розробка	8	4	1,5	55,40	443,18
Тестування	8	3	1,35	49,86	396,86
Впровадження	2	2	1,1	40,63	81,25
Всього					923,30

Розрахуємо додаткову заробітну плату. Додаткова заробітна плата розраховується як 10...12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (0{,}1...0{,}12) \times З_О . \quad (5.4)$$

Додаткова заробітна плата:

$$З_{дод} = 0{,}1 \times (З_{О_{НК}} + З_{О_Р}) = 0{,}1 \times (24200 + 923{,}30) = 2512{,}33\,(\text{грн})$$

Нарахування на заробітну плату дослідників та робітників розраховується як 22% від суми основної та додаткової заробітної плати за формулою:

$$Н_{ЗП} = (З_о + З_р + З_{дод}) \times \frac{\beta}{100}, \quad (5.5)$$

де $З_о$ – основна заробітна плата розробників, грн;

$З_{дод}$ – додаткова заробітна плата розробників, грн;

$З_р$ – заробітна плата робітників, грн;

β – ставка єдиного внеску на загальнообов'язкове державне страхування.

Єдиний соціальний внесок на загальнообов'язкове державне соціальне страхування (ЄСВ) – об'єднаний страховий внесок в Україні, збір якого здійснюється в системі загальнообов'язкового державного страхування в

обов'язковому порядку та на регулярній основі. Відповідно до даних офіційного сайту міністерства фінансів в 2021 році ЄСВ становить 22%.

$$H_{ЗП} = (24200 + 923{,}30 + 2512{,}33) \times \frac{22}{100} = \times 0{,}22 = 6079{,}84 \text{ (грн)}.$$

Розрахунок амортизаційних витрат може бути виконаний з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \times \frac{t_{вик}}{12}, \quad (5.6)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$– термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

Виконання розробки проводилось на ноутбуц вартістю 5000 грн. Амортизаційні відрахування наведені у таблиці 5.6.

Таблиця 5.6 – Амортизаційні відрахування

Найменування	Ціна, грн	Корисний час використання, роки	Термін використання, міс.	Сума амортизації, грн
Ноутбук	5000	4	0,73	76,04
Всього	76,04			

Витрати на матеріали розраховуються за формулою:

$$M = \sum_{1}^{n} H_i \times Ц_i \times K_i - \sum_{1}^{n} B_i \times Ц_B, \quad (5.7)$$

де H_i – витрати матеріалу *i*-го найменування;

$Ц_i$ – вартість матеріалу *i*-го найменування, грн/кг;

K_i – коефіцієнт транспортних витрат, $К_і$ = (1,1...1,15);

B_i – маса відходів матеріалу *i*-го найменування, кг;

$Ц_в$ – ціна відходів матеріалу *i*-го найменування, грн/кг;

n – кількість видів матеріалів.

Вартість матеріалів, що були використані на розробку системи зведені до таблиці 5.7.

$M = 355{,}1 \times 1{,}1 = 390{,}61\,(\text{грн})$

Таблиця 5.7 – Вартість матеріалів, що були використані для розробки алгоритму тестування.

Найменування матеріалу	Одиниці виміру	Ціна, грн	Витрачено	Вартість витрачених матеріалів, грн.
Флеш-накопичувач	Шт.	150	1	150
Пачка паперу	Уп	95	2	190
Ручка	Шт	7,55	2	15,1
Всього				355,1

Таким чином, витрати на матеріали, що були використані в процесі розробки алгоритму тестування складають М = 390,61 грн.

Розрахуємо витрати на комплектуючі.

Витрати на силову електроенергію розраховуються за формулою:

$$B_E = B \times П \times Ф \times K_П, \qquad (5.7)$$

де B – вартість 1кВт-години електроенергії (B = 4,62 грн/кВт);

$П$ – установлена потужність комп'ютеру ($П$ = 0,2 кВт);

$Ф$ – фактична кількість годин роботи комп'ютеру ($Ф$ = 22 дні х 8 год = 176 год);

$K_П$ – коефіцієнт використання потужності ($K_П$<1, $K_П$ = 0,8).

$B_E = 4{,}62 \times 0{,}2 \times 176 \times 0{,}8 = 130{,}10$ (грн)

Розрахуємо інші витрати $B_{ін}$.

Інші витрати $B_{ін}$ можна прийняти як (100-300)% від суми основної заробітної плати розробників, які виконували дану роботу, тобто:

$$B_{ін} = (1...3) \cdot (З_о + З_р), \qquad (5.7)$$

Отже, розрахуємо інші витрати:

$B_{ін} = 1 \cdot (24200 + 923{,}30) = 25123{,}30$ (грн).

Усі витрати складають:

$B = 24200 + 923{,}30 + 2512{,}33 + 6079{,}84 + 130{,}10 + 25123{,}30 + 76{,}04 = 36433{,}94$ (грн) (5.8)

Розрахуємо загальну вартість наукової розробки $B_{заг}$ за формулою:

$$B_{заг} = \frac{B}{\alpha}, \tag{5.9}$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відносних одиницях.

$$B_{заг} = \frac{36433,94}{1} = 36433,94 \text{ (грн)}$$

Прогнозування загальних витрат $ЗВ$ на виконання та впровадження результатів виконаної наукової роботи здійснюється за формулою:

$$ЗВ = \frac{B_{заг}}{\beta}, \tag{5.10}$$

Отже, розрахуємо прогнозовані загальні витрати:

$$ЗВ = \frac{36433,94}{0,7} = 52048,48 \text{ (грн)}.$$

5.3 Розрахунок економічної ефективності розробки за її можливої комерціалізації потенційним інвестором

Магістерська кваліфікаційна робота передбачає створення методу і засобу виявлення фішингових інформаційних ресурсів, інакше кажучи – розробка програмного засобу для використання масовим споживачам.

Для даного випадку можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із років, протягом яких очікується отримання позитивних результатів між можливого впровадження та комерціалізації науково-технічної розробки, розраховується за формулою:

$$\Delta П_i = (\pm \Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \tag{5.11}$$

де $\pm \Delta Ц_o$ – зміна основного якісного показника від впровадження результатів науково-технічної розробки у році що аналізується. Зазвичай, таким показником може бути зміна ціни реалізації одиниці нової розробки у році що аналізується (відносно року до впровадження цієї розробки); $\pm \Delta Ц_o$ може мати як додатне, так

і від'ємне значення (від'ємне – при зниженні ціни відносно року до впровадження цієї розробки, додатне – при зростанні ціни);

N – основний кількісний показник, який визначає величину попиту на аналогічні чи подібні розробки у році до впровадження результатів нової науково-технічної розробки;

$Ц_о$ – основний якісний показник, який визначає ціну реалізації нової науково-технічної розробки у році що аналізується, $Ц_о = Ц_б \pm \Delta Ц_о$;

$Ц_б$ – основний якісний показник, який визначає ціну реалізації існуючої (базової) науково-технічної розробки у році до впровадження результатів;

ΔN – зміна основного кількісного показника від впровадження результатів науково-технічної розробки у році що аналізується. Зазвичай таким показником може бути зростання попиту на науково-технічну розробку у році що аналізується (відносно року до впровадження цієї розробки);

λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2021 році ставка податку на додану вартість складає 20%, а коефіцієнт λ = 0,8333;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту (послуги). Рекомендується приймати ρ = 0,2...0,5;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2021 році ϑ = 18%.

$$\Delta П_i = (50 \cdot 5000 + (200 + 50) \cdot 1500) \cdot 0{,}8333 \cdot 0{,}3 \cdot (1 - \frac{18}{100}) = 128119{,}9 \text{ (грн)}$$

Далі розрахуємо приведену вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{ш=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.12)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,05…0,15;

t – період часу (в роках) від моменту початку впровадження науковотехнічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = \frac{128119,9}{(1+0,15)^1} + \frac{128119,9}{(1+0,15)^2} + \frac{128119,9}{(1+0,15)^3} = 292526,5\,(\text{грн})$$

Далі розрахуємо величину початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науковотехнічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} \cdot ЗВ, \qquad (5.13)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай інв $k_{інв}$ =2...5, але може бути і більшим;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$$PV = 2 \cdot 52048,48 = 104096,96\,(\text{грн})$$

Тоді абсолютний економічний ефект $E_{абс}$ або чистий приведений дохід (NPV, Net Present Value) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV, \qquad (5.13)$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, грн;

PV – теперішня вартість початкових інвестицій, грн.

$$E_{абс} = 292526,5 - 104096,96 = 188429,6$$

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність $E_в$ або показник внутрішньої

норми дохідності (IRR, Internal Rate of Return) вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Внутрішня економічна дохідність інвестицій Ев , які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науковотехнічної розробки, розраховується за формулою:

$$E_B = \sqrt[T_ж]{1+\frac{E_{абс}}{PV}} - 1, \tag{5.14}$$

де $E_{абс}$ – абсолютна ефективність вкладених інвестицій, грн;

PV – теперішня вартість інвестицій, грн;

$T_ж$ – життєвий цикл наукової розробки, роки.

$$E_B = \sqrt[4]{1+\frac{292526,5}{104096,96}} - 1 = \sqrt[3]{1+1,81} - 1 = 1,41 - 1 = 0,41$$

Далі визначаємо бар'єрну ставку дисконтування мін τ, тобто мінімальну внутрішню економічну дохідність інвестицій, нижче якої кошти у впровадження науково-технічної розробки та її комерціалізацію вкладатися не будуть.

Мінімальна внутрішня економічна дохідність вкладених інвестицій мін τ визначається за формулою:

$$\tau_{\min} = d + f, \tag{5.15}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2020 році в Україні d = 0,9...0,12;

f – показник, що характеризує ризикованість вкладення інвестицій; зазвичай величина f =0,05...0,5, але може бути і значно вищою.

$$\tau_{\min} = 0,2 + 0,2 = 0,4$$

Якщо величина $E_B > \tau_{\min}$, то потенційний інвестор може бути зацікавлений у фінансуванні впровадження науково-технічної розробки та виведенні її на ринок, тобто в її комерціалізації. У даному випадку $E_B > \tau_{\min}$, 0,41>0,4.

Далі розраховуємо період окупності інвестицій $T_{ОК}$ (DPP, Discounted Payback Period), які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ОК} = \frac{1}{E_B}, \tag{5.16}$$

де E_B – внутрішня економічна дохідність вкладених інвестицій.

$$T_{ОК} = \frac{1}{0{,}41} = 2{,}432$$

$T_{ОК}$ < 3-х років – це свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження цієї розробки та виведення її на ринок.

В розділі економічної частини виконано обґрунтування економічної доцільності проведення наукового дослідження та розробки методу та засобу автентифікації користувачів із нульовим знанням.

Рівень комерційного потенціалу розробки методу та засобу визначення фішингових інформаційних ресурсів має середньоарифметичну сума балів, розраховану на основі висновків експертів 31, що відповідає «вище середнього» науково-технічному рівню та комерційному потенціалу розробки.

Загальні витрати, що стосуються виконавців розробки склали 52048,48 грн; внутрішня економічна дохідність інвестицій E_B 0,41.

Термін окупності продукції вкладених інвестицій складає 2,432 роки, що свідчить про доцільність фінансування розробки.

ВИСНОВКИ

Розглянуто основні поняття фішингових інформаційних ресурсів. Проаналізовано основні методи аналізу інформації, отриманої з вигляду URL-адреси, а також розглянуто теоретичні відомості про параметри URL-адреси – метрики, що використовують для подальшого аналізу отриманих даних про URL-адресу. Поставлено завдання на розробку методу і засобу виявлення фішингових інформаційних ресурсів, базуючись на проаналізованих джерелах.

Проаналізувавши структуру інформаційного ресурсу, виділено інформацію з полів, що містяться у ресурсі. Розглянуто та структуровано метрики за лексичними, синтаксичними та прагматичними особливостями інформаційного ресурсу. Для зручності подальшої роботи всі метрики були віднесені до певних категорій.

Вдосконалено метод класифікації фішингових ресурсів на основі URL-адрес, що, на відміну від існуючих підходів, дозволяє здійснювати такий процес за семантичними, синтаксичними та прагматичними особливостями (полями) даних адрес.

Розглянуто та проаналізовано існуючі класифікатори прийняття рішень та обрано для подальшого дослідження XGBoost. На основі обраного класифікатору реалізовано засіб виявлення фішингових інформаційних ресурсів.

Розроблено архітектуру та схему роботи програми, а також основні компоненти. Проведено ряд експерементальних досліджень для тестування роботи програми шляхом перевірки на справжність URL-адрес веб-сайтів. Експериментальні дослідження показали достовірність роботи системи виявлення фішингових інформаційних ресурсів на рівні 87%, при цьому помилки першого та другого роду становлять 5% і 8% відповідно.

Здійснено економічне обґрунтування та доведено доцільність розробки виявлення фішингових інформаційних ресурсів. У ході дослідження виявлено, що прибуток у інвестора буде сягати 128119,9 грн, а термін окупності становить 2,432 року.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. D. Almomani, T.-C. Wan, A. Manasrah, A. Taha, M. Baklizi, and S. Ramadass *An enhanced online phishing e-mail detection framework based on evolving connectionist system*. Int. J. Innov. Comput. Inf. Control, vol. 9, no. 3, 2012, pp. 169–175.
2. Bergholz, J. Chang, G. Paaß, F. Reichartz, and S. Strobel *Improved phishing detection using model-based features.* In Proc. CEAS, Aug. 2008, pp. 1–10.
3. Bergholz, J. De Beer, S. Glahn, M.-F. Moens, G. Paaß, and S. Strobel, *New filtering approaches for phishing email*. J. Comput. Secur., vol. 18, no. 1, Jan. 2010, pp. 7–35.
4. Думчиков С.А., Лукічов В.В. Виявлення фішингових інформаційних ресурсів. *Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення*: зб. тез доп. міжн. наук.-інтер. конф., м. Тернопіль, 12 жовтня 2021 р. Тернопіль 2016 р. с. 24-26.
5. Думчиков С.А., Лукічов В.В., Статистика фішингових інцидентів в україні за 2021 рік. «*Молодь в науці: дослідження, проблеми, перспективи (МН-2022)*». URL: https://conferences.vntu.edu.ua/index.php/mn/mn2022/paper/viewFile/14069/11918/.
6. The 2018 Phishing Trends & Intelligence Report. URL: https://info.phishlabs.com/2018_phishing_trends_and_-intelligence_report-0 (дата звернення 07.09.2021).
7. Patrick Nohe. HTTPS Phishing: 49% of Phishing Websites Now Sport The Green Padlock. URL: https://www.thesslstore.com/blog/https-phishing-green-padlock/ (дата звернення 07.09.2021).
8. A.-P. W. Group. Phishing Activity Trends Report–1st Quarter 2019. URL: https://docs.apwg.org/reports/apwg_trends_report_q1_2019.pdf (дата звернення 07.09.2021).
9. K. L. Chiew, K. S. C. Yong, and C. L. Tan, «A survey of phishing attacks: Their types, vectors and technical approaches,» Expert Syst. Appl., vol. 106, Sep. 2018,

pp. 1-20.
URL: http://www.sciencedirect.com/science/article/pii/S0957417418302070 (дата звернення 07.09.2021).

10. Z. Dou, I. Khalil, A. Khreishah, A. Al-Fuqaha, and M. Guizani *Systematization of knowledge (SOK): A systematic review of software-based Web phishing detection*. IEEE Commun. Surveys Tuts., vol. 19, no. 4, , Sep. 2017, pp. 2797–2819.
11. R. Sommer and V. Paxson *Outside the closed world: On using machine learning for network intrusion detection.* In Proc. IEEE Symp. Secur. Privacy, Berleley, CA, USA, May 2010, pp. 305–316.
12. R. Verma, M. Kantarcioglu, D. Marchette, E. Leiss, and T. Solorio *Security analytics: Essential data analytics knowledge for cybersecurity professionals and students.* IEEE Secur. Privacy, vol. 13, no. 6, Nov. 2015, pp. 60–65.
13. Quarterly Threat Trends & Intelligence – November 2021. URL: https://info.phishlabs.com/quarterly-threat-trends-and-intelligence-november-2021 (дата звернення 07.09.2021).
14. COVID-19 Pushes Up Internet Use 70% And Streaming More Than 12%, First Figures Reveal. URL: https://www.forbes.com/sites/markbeech/2020/03/25/covid-19-pushes-up-internet-use-70-streaming-more-than-12-first-figures-reveal/ (дата звернення 08.09.2021).
15. Українська міжбанківська Асоціація членів платіжних систем «ЄМА». URL: https://www.ema.com.ua/ (дата звернення 02.09.2021).
16. Kaspersky Threat Intelligence Portal. URL: https://opentip.kaspersky.com/ (дата звернення 23.09.2021).
17. Website/URL/Link Scanner Safety Check for Phishing, Malware, Viruses - ScanURL.net. URL: https://scanurl.net/ (дата звернення 23.09.2021).
18. PhishTank. URL: https://www.phishtank.com/ (дата звернення 23.09.2021).
19. Google Transparency Report. URL: https://transparencyreport.google.com/safe-browsing/search (дата звернення 23.09.2021).
20. Онлайн-проверка антивирусом Dr.Web. URL: https://online.drweb.com/result2/?url=t (дата звернення 23.09.2021).

21. Is This Website Safe | Website Security | Norton Safe Web . URL: https://safeweb.norton.com/ (дата звернення 23.09.2021).
22. URL dataset (ISCX-URL2016). URL: https://www.unb.ca/cic/datasets/url-2016.html
23. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт. URL: http://ir.lib.vntu.edu.ua/bitstream/handle/123456789/34589/90807.pdf?sequence=2&isAllowed=y.

ДОДАТКИ

Додаток А

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

Затверджую
д. т. н., проф.
_____ В. А. Лужецький
«____» _____ 2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ
на виконання магістерської кваліфікційної роботи

на тему: «Метод і засіб виявлення фішингових інформаційних ресурсів»
08-20.МКР.002.00.000 ТЗ

Керівник магістерської кваліфікаційної роботи
к.т.н., ст. викл. каф. ЗІ
_____ В. В. Лукічов

Розробив студент групи 1БС-20м
_____ С. А. Думчиков

Вінниця 2021

1 Підстави для проведення робіт

Робота проводиться на підставі наказу ректора ВНТУ від 24 вересня 2021 року № 277.

Дата початку роботи 01.09.21 р.

Дата закінчення роботи 20.11.21 р.

2 Мета та призначення МКР

Мета – покращення безпеки веб-серфінгу за рахунок збільшення точності розпізнавання фішингового ресурсу.

Об'єктом дослідження є процес розпізнавання фішингових ресурсів.

Предметом є засоби розпізнавання фішингових ресурсів.

Актуальність теми. Більшість людей у сучасному світі користуються мережею Інтернет для перегляду веб-сторінок. Ці сторінки можуть бути пізнавальним чи розважальним змістом; сторінки з важливою комерційною або персональною інформацією, що не підлягає передачі третім особам.

Для того щоб запобігти втраті конфіденційної інформації через серфінг веб-сторінок потрібно перевіряти чи не є веб-сторінка підробленою.

Для перевірки веб-сторінок використовують автоматизовану та ручну перевірку URL-адреси. Існує багато методів проведення автоматизованої перевірки URL-адресв. Перед відправкою даних в засіб, який піддається тестуванню, відбувається його навчання. Саме від правильності навчання моделі залежить ефективність точного виявлення фішинговиого ресурсу. Оцінка формується за певними критеріями і базуються на значення полів URL-адреси.

3 Вихідні дані для проведення МКР

МКР проводиться вперше і вихідними даними для проведення МКР є:

3.1 D. Almomani, T.-C. Wan, A. Manasrah, A. Taha, M. Baklizi, and S. Ramadass, ‘‘An enhanced online phishing e-mail detection framework based on evolving connectionist system,’’ Int. J. Innov. Comput. Inf. Control, vol. 9, no. 3, 2012. pp. 169–175.

3.2 A. Bergholz, J. Chang, G. Paaß, F. Reichartz, and S. Strobel, ‘‘Improved phishing detection using model-based features,’’ in Proc. CEAS, Aug. 2008, pp. 1–10.

3.3 A. Bergholz, J. De Beer, S. Glahn, M.-F. Moens, G. Paaß, and S. Strobel, ‘‘New filtering approaches for phishing email,’’ J. Comput. Secur., vol. 18, no. 1, Jan. 2010, pp. 7–35.

3.4 K. L. Chiew, K. S. C. Yong, and C. L. Tan, «A survey of phishing attacks: Their types, vectors and technical approaches,» Expert Syst. Appl., vol. 106, pp. 1–20, Sep. 2018. URL: http://www.sciencedirect.com/science/article/pii/S0957417418302070 (дата звернення 07.09.2021)

3.5 Z. Dou, I. Khalil, A. Khreishah, A. Al-Fuqaha, and M. Guizani, «Systematization of knowledge (SOK): A systematic review of software-based

Web phishing detection,» IEEE Commun. Surveys Tuts., vol. 19, no. 4, Sep. 2017, pp. 2797–2819.

4 Виконавці МКР

Студент групи 1 БС-20м Думчиков Станіслав Андрійович

5 Вимоги до виконання МКР

Для покращення точності виявлення фішингових інформаційних ресурсів за рахунок застосування машинного навчання неохідно розв’язати такі задачі:

- науково-досліджене обґрунтування необхідності дослідження та розробки нового методу розпізнавання фішингового вмісту;
- аналіз наукової літератури щодо існуючих методів та засобів виявлення фішингового вмісту веб-сторінок з метою виявлення потенційних нових способів виявлення фішингових ресурсів;
- удосконалення методу виявлення фішингових інформаційних ресурсів, шляхом використання методу поділу ресурсу на синтаксичні, семантичні та прагматичні поля;
- розробка алгоритмів функціонування модулів програмного засобу;
- розробка програмного засобу згідно розроблених алгоритмів функціонування;
- експериментальне дослідження удосконалених методів визначення фішингових ресурсів.

6 Вимоги до супровідної документації

Графічна і текстова документація повинна відповідати діючим стандартам України – ДСТУ 3008:2015.

7 Етапи МКР

Робота з теми виконується у 7 етапів.

Зміст етапу	Початок - закінчення	Очікувані результати	Звітна документація
Аналіз завдання. Вступ	01.09.2021 – 04.09.2021	Вступ	Чернетка вступу
Розробка технічного завдання	05.09.2021 – 15.09.2021	Технічне завдання	Проект технічного завдання
Аналіз літературних джерел за напрямком магістерської кваліфікаційної роботи	16.09.2021 – 22.09.2021	Аналіз існуючих аналогів. Вибір напрямку дослідження Аналіз відомих методів. Постановка завдання	Чернетка першого розділу

Удосконалення методів виявлення фішингу	23.09.2021 – 04.10.2021	Удосконалений метод виявлення фішингу на основі нової класифікації полів веб-сторінки	Чернетка другого розділу
Експериментальні дослідження	05.10.2021 – 14.11.2021	Програмний засіб, який реалізує розроблювані методи	Чернетка третього і четвертого розділу
Розробка економічного розділу	15.11.2021 – 24.11.2021	Економічні показники дослідження	Чернетка з економічного розділу
Оформлення пояснювальної записки	25.11.2021 – 30.11.2021	Пояснювальна записка	Пояснювальна записка

8 Очікувані результати та порядок реалізації МКР

Передбачається розробка нових (удосконалення існуючих) методів які спрямовані на покращення точності розпізнавання фішунги. Заплановане створення програмного засобу, який може бути використаний у навчальному процесі.

9 Матеріали які подаються після закінчення МКР

По завершенню роботи подається пояснювальна записка та ілюстративна частина.

10 Порядок приймання МКР та її етапів

Апробація на науково-технічних конференціях та семінарах. Результати роботи будуть розглядатися на засіданні ДЕК із захисту магістерських кваліфікаційних робіт.
Попередній захист та доопрацювання МКР – 3 грудня 2021 р.
Представлення МКР до захисту – 20 грудня 2021 р.
Захист МКР – 21.12.21.

11 Вимоги до розроблення документації

Документація буде виконуватись за допомогою комп’ютерного набору у відповідності вимог ДСТУ 3008:2015 «Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання».

12 Вимоги щодо технічного захисту інформації з обмеженим доступом

У зв’язку з тим, що дана робота не містить інформації, що потребує захисту у відповідності до законів України, заходи з її технічного захисту не передбачаються.

Додаток Б

Текст програмного засобу

Файл demo.py

```
import pickle
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from url_parser import featureExtraction, feature_names
import webbrowser
from threading import Timer
import time

# URL to check
# test_url = "https://vntu.edu.ua/"
test_url = "https://atb-market.space/"
# test_url = "https://paypal.com.us.home.login.page.loginchecks.xyz/"
# test_url = "https://secure.login.aliexpress.com.coin-balance.com/entscheidung-
817277406681628&amp;cgi3-viewkontakt-817277406681628-007acctpagetype-817277406681628=33445-
gesendet&amp;amp"
# test_url = "https://evernote.com/register?upgrade=premium&itemCode=premium-
1year&offer=www_pricing_CTA"

xgboost_model = pickle.load(open("models/222XGBoostClassifier.pickle.dat", "rb"))

webbrowser.register('chrome', None,  webbrowser.BackgroundBrowser("C://Program
Files//Google//Chrome//Application//chrome.exe"))

def countdown(t):

    while t:
        mins, secs = divmod(t, 60)
        timer = '{:02d}:{:02d}'.format(mins, secs)
        print("Зачекайте", timer, end="\r")
        time.sleep(1)
        t -= 1

cols_when_model_builds = xgboost_model.get_booster().feature_names
feature_list = [featureExtraction(test_url)]
df = pd.DataFrame(feature_list, columns=feature_names)
result_1 = xgboost_model.predict_proba(df)

df.plot(kind = 'bar')
plt.yticks(np.arange(0, 2, 1))
plt.show()

color_of_group = "невизначеній"
if result_1[0,1] <= 0.49:
    color_of_group = "зелена/ій"
elif result_1[0,1] >= 0.5 and result_1[0,1]<=0.79:
    color_of_group = "жовта/ій"
else:
    color_of_group = "червона/ій"

results = pd.DataFrame({'ML Model': ["XGBoost"],
                        'Result': [result_1]})
print("#########################################################################")
print("URL:", test_url)
# print(results)

print("Ймовірність шкідливої URL-адреси для фішингу:", result_1[0,1]) #Тут нормально
print("Сайт", test_url, "знаходиться у", color_of_group, "групі")
if result_1[0,1] <= 0.49:
    webbrowser.get('chrome').open(test_url)
elif result_1[0,1] >= 0.5 and result_1[0,1]<=0.79:
    print("Увага! Потенційно небезпечне посилання. Бажаєте перейти?")
    answer = input()
    if answer == "y":
        webbrowser.get('chrome').open(test_url)
    elif answer == "n":
        quit()
    else:
        print("Оберіть відповідь")
```

```
    webbrowser.get('chrome').open(test_url)
else:
    print("URL-адреса", test_url, "є небезпечною для відвідування. Рекомендуємо подумати про
наслідки від втрати Ваших даних")
    countdown(10)
    print("Ви впевнені, що хочете перейти за адресою", test_url, "?(y/n):")
    answer = input()
    if answer == "y":
        webbrowser.get('chrome').open(test_url)
    elif answer == "n":
        quit()
    else:
        print("Оберіть відповідь")
```

Файл url_parser.py

```
from datetime import datetime
import requests
import urllib.request
import urllib
import whois
from bs4 import BeautifulSoup
import pandas as pd

from urllib.parse import urlparse, urlencode
import ipaddress
import re


# 1.Checks for IP address in URL (Have_IP)
def havingIP(url):
    try:
        ipaddress.ip_address(url)
        ip = 1
    except:
        ip = 0
    return ip


# 2.Checks the presence of @ in URL (Have_At)
def haveAtSign(url):
    if "@" in url:
        at = 1
    else:
        at = 0
    return at


# 3.Finding the length of URL and categorizing (URL_Length)
def getLength(url):
    if len(url) < 54:
        length = 0
    else:
        length = 1
    return length


# 4.Gives number of '/' in URL (URL_Depth)
def getDepth(url):
    s = urlparse(url).path.split('/')
    depth = 0
    for j in range(len(s)):
        if len(s[j]) != 0:
            depth = depth + 1
    return depth


# 5.Checking for redirection '//' in the url (Redirection)
def redirection(url):
    pos = url.rfind('//')
    if pos > 6:
        if pos > 7:
            return 1
        else:
            return 0
    else:
        return 0
```

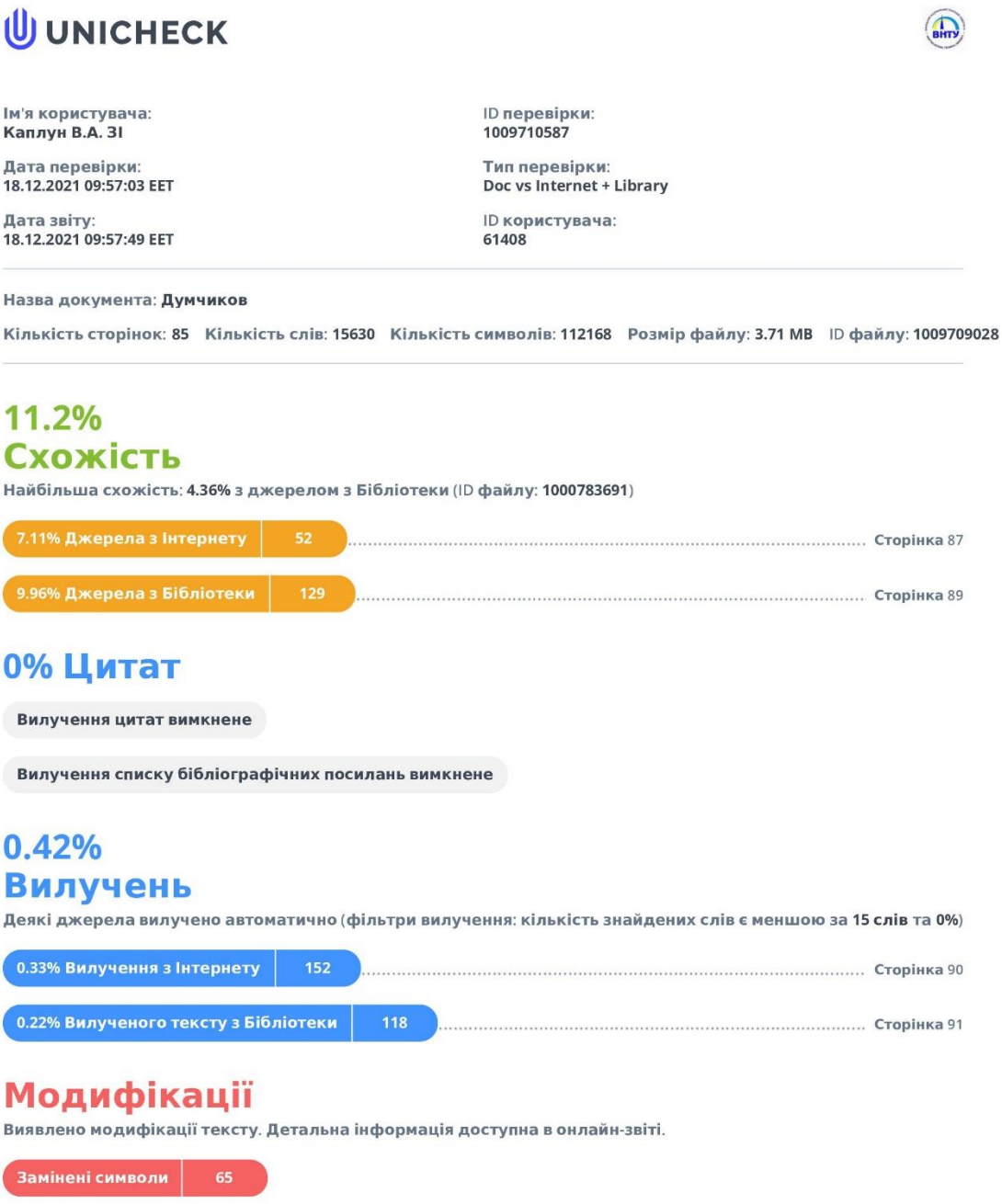
```
# 6.Existence of “HTTPS” Token in the Domain Part of the URL (https_Domain)
def httpDomain(url):
    domain = urlparse(url).netloc
    if 'https' in domain:
        return 1
    else:
        return 0


# listing shortening services
shortening_services =
r"bit\.ly|goo\.gl|shorte\.st|go2l\.ink|x\.co|ow\.ly|t\.co|tinyurl|tr\.im|is\.gd|cli\.gs|yfrog\.com|m
igre\.me|ff\.im|tiny\.cc|url4\.eu|twit\.ac|su\.pr|twurl\.nl|snipurl\.com|short\.to|BudURL\.com|ping\
.fm|post\.ly|Just\.as|bkite\.com|snipr\.com|fic\.kr|loopt\.us|doiop\.com|short\.ie|kl\.am|wp\.me|rub
yurl\.com|om\.ly|to\.ly|bit\.do|t\.co|lnkd\.in|db\.tt|qr\.ae|adf\.ly|goo\.gl|bitly\.com|cur\.lv|tiny
url\.com|ow\.ly|bit\.ly|ity\.im|q\.gs|is\.gd|po\.st|bc\.vc|twitthis\.com|u\.to|j\.mp|buzurl\.com|cut
t\.us|u\.bb|yourls\.org|x\.co|prettylinkpro\.com|scrnch\.me|filoops\.info|vzturl\.com|qr\.net|1url\.
com|tweez\.me|v\.gd|tr\.im|link\.zip\.net"


# 7. Checking for Shortening Services in URL (Tiny_URL)
def tinyURL(url):
    match = re.search(shortening_services, url)
    if match:
        return 1
    else:
        return 0


# 8.Checking for Prefix or Suffix Separated by (-) in the Domain (Prefix/Suffix)
def prefixSuffix(url):
    if '-' in urlparse(url).netloc:
        return 1  # phishing
    else:
        return 0  # legitimate


# importing required packages for this section

# 9. Domain based features (4)
dns = 0
try:
    flags = 0
    flags = flags | whois.NICClient.WHOIS_QUICK
    domain_name = whois.whois(urlparse(url).netloc, flags=flags)
except:
    dns = 1

# 10.Web traffic (Web_Traffic)

# import csv
#
# with open('top-1m.csv') as f:
#     reader = csv.reader(f)
#     alexa = list(reader)


def web_traffic(url):
    try:
        # Filling the whitespaces in the URL if any
        url = urllib.parse.quote(url)
        rank = BeautifulSoup(urllib.request.urlopen(
            "http://data.alexa.com/data?cli=10&dat=s&url=" + url).read(),
"xml").find("REACH")['RANK']
        rank = int(rank)
    except TypeError:
        return 1
    if rank < 100000:
        return 1
    else:
        return 0


# 11.Survival time of domain: The difference between termination time and creation time (Domain_Age)
def domainAge(domain_name):
    creation_date = domain_name.creation_date
    expiration_date = domain_name.expiration_date
    if (isinstance(creation_date, str) or isinstance(expiration_date, str)):
```

```
        try:
            creation_date = datetime.strptime(creation_date, '%Y-%m-%d')
            expiration_date = datetime.strptime(expiration_date, "%Y-%m-%d")
        except:
            return 1
    if ((expiration_date is None) or (creation_date is None)):
        return 1
    elif ((type(expiration_date) is list) or (type(creation_date) is list)):
        return 1
    else:
        ageofdomain = abs((expiration_date - creation_date).days)
        if ((ageofdomain / 30) < 12):
            age = 1
        else:
            age = 0
    return age


# 12.End time of domain: The difference between termination time and current time (Domain_End)
def domainEnd(domain_name):
    expiration_date = domain_name.expiration_date
    if isinstance(expiration_date, str):
        try:
            expiration_date = datetime.strptime(expiration_date, "%Y-%m-%d")
        except:
            return 1
    if (expiration_date is None):
        return 1
    elif (type(expiration_date) is list):
        return 1
    else:
        today = datetime.now()
        end = abs((expiration_date - today).days)
        if ((end / 30) < 6):
            end = 1
        else:
            end = 0
    return end

# 13 iFrame


def iframe(response):
    if response == "":
        return 1
    else:
        if re.findall(r"[<iframe>|<frameBorder>]", response.text):
            return 0
        else:
            return 1

# 14 MouseOver


def mouseOver(response):
    if response == "":
        return 1
    else:
        if re.findall("<script>.+onmouseover.+</script>", response.text):
            return 1
        else:
            return 0

# 15 Rightclick


def rightClick(response):
    if response == "":
        return 1
    else:
        if re.findall(r"event.button ?== ?2", response.text):
            return 0
        else:
            return 1

# 16 Web_Forwards


def forwarding(response):
    if response == "":
```

```
        return 1
    else:
        if len(response.history) <= 2:
            return 0
        else:
            return 1


# Function to extract features
def featureExtraction(url):
    features = []
    # Address bar based features (10)
    # features.append(getDomain(url))
    features.append(havingIP(url))
    features.append(haveAtSign(url))
    features.append(getLength(url))
    features.append(getDepth(url))
    features.append(redirection(url))
    features.append(httpDomain(url))
    features.append(tinyURL(url))
    features.append(prefixSuffix(url))
    features.append(dns)
    features.append(web_traffic(url))
    features.append(1 if dns == 1 else domainAge(domain_name))
    features.append(1 if dns == 1 else domainEnd(domain_name))

    # HTML & Javascript based features
    try:
        response = requests.get(url)
    except:
        response = ""
    features.append(iframe(response))
    features.append(mouseOver(response))
    features.append(rightClick(response))
    features.append(forwarding(response))

    return features


# converting the list to dataframe
feature_names = ['Have_IP', 'Have_At', 'URL_Length', 'URL_Depth', 'Redirection',
                 'https_Domain', 'TinyURL', 'Prefix/Suffix', 'DNS_Record', 'Web_Traffic',
                 'Domain_Age', 'Domain_End', 'iFrame', 'Mouse_Over', 'Right_Click', 'Web_Forwards']
```

Додаток В

Протокол перевірки на наявність плагіату



UNICHECK

Ім'я користувача:
Каплун В.А. ЗІ

ID перевірки:
1009710587

Дата перевірки:
18.12.2021 09:57:03 EET

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
18.12.2021 09:57:49 EET

ID користувача:
61408

Назва документа: Думчиков

Кількість сторінок: 85 Кількість слів: 15630 Кількість символів: 112168 Розмір файлу: 3.71 MB ID файлу: 1009709028

11.2%
Схожість

Найбільша схожість: 4.36% з джерелом з Бібліотеки (ID файлу: 1000783691)

7.11% Джерела з Інтернету 52 ..... Сторінка 87

9.96% Джерела з Бібліотеки 129 ..... Сторінка 89

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0.42%
Вилучень

Деякі джерела вилучено автоматично (фільтри вилучення: кількість знайдених слів є меншою за 15 слів та 0%)

0.33% Вилучення з Інтернету 152 ..... Сторінка 90

0.22% Вилученого тексту з Бібліотеки 118 ..... Сторінка 91

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 65

ІЛЮСТРАТИВНА ЧАСТИНА

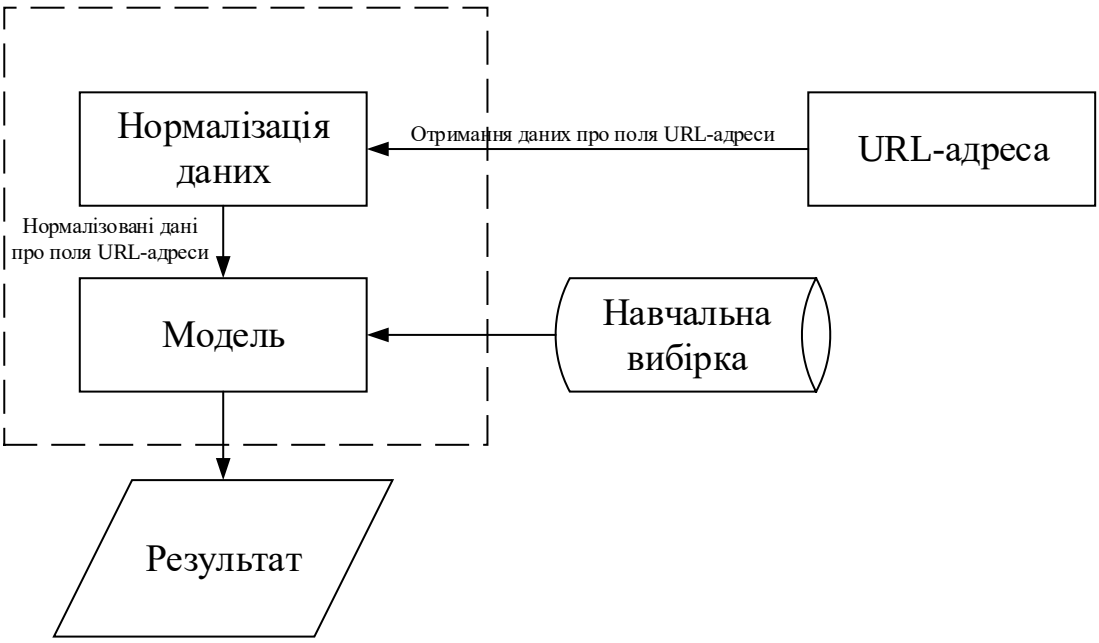
МЕТОД І ЗАСІБ ВИЯВЛЕННЯ ФІШИНГОВИХ ІНФОРМАЦІЙНИХ РЕСУРСІВ

Порівняння засобів-аналогів

Назва засобу-аналога	Результат виконання
Kaspersky Threat Intelligence Portal	Фішинг не виявлено, сайт безпечений
ScanURL	Фішинг не виявлено, сайт безпечений
PhishTank	Нема даних про сайт
Google Transparency Report	Фішинг не виявлено, сайт безпечений
Онлайн-перевірка посилань за допомогою Dr.Web	Фішинг не виявлено, сайт безпечений
Norton Safe Web від Symantec	Нема даних про сайт

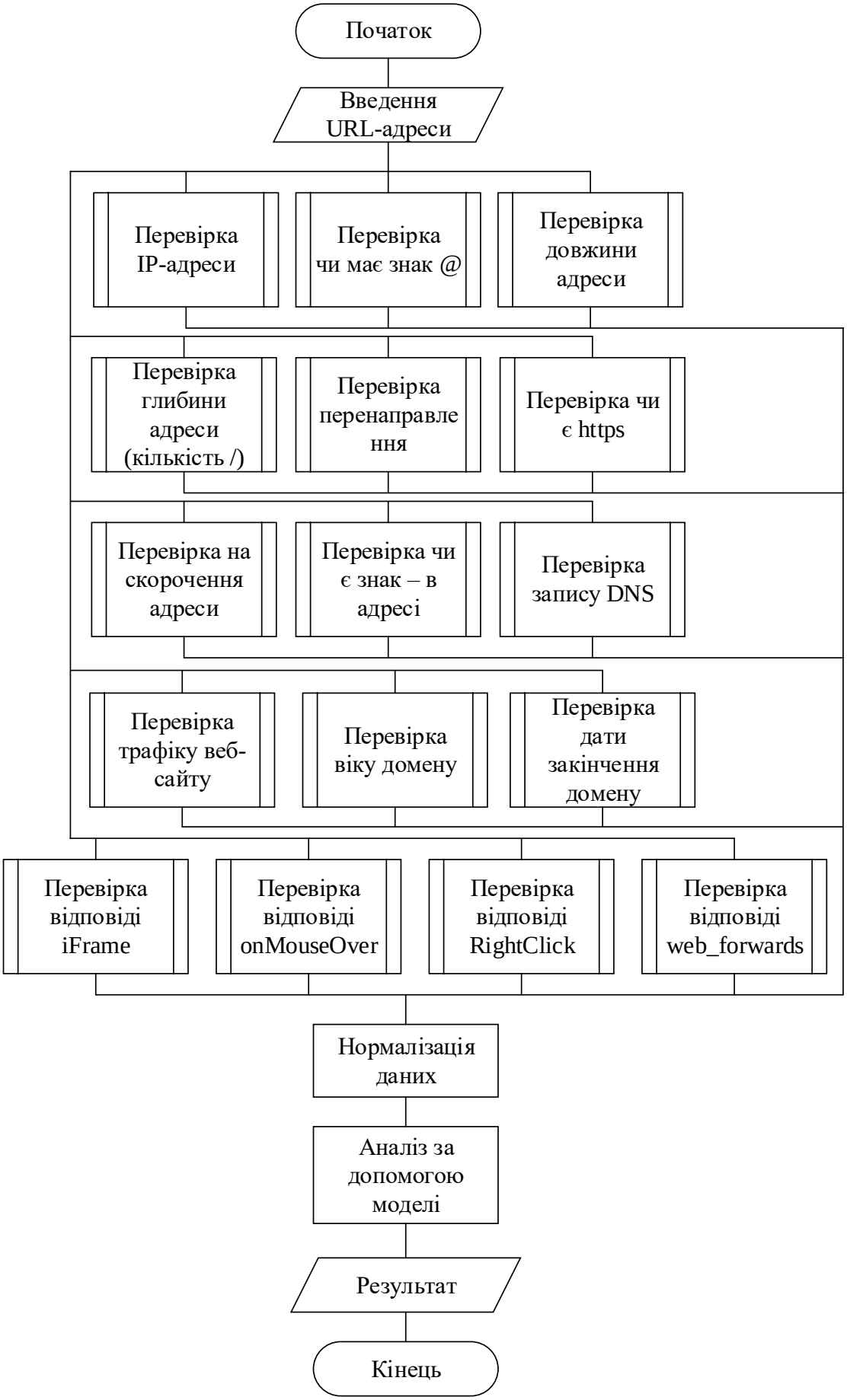
					08-20.МКР.002.00.000 ПЗ		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Думчиков С. А.			Метод і засіб виявлення фішингових інформаційних ресурсів	Літ.	Арк. 1 / Аркушів 1
Перевір.		Лукічов В. В.					
Рецензент		Азарова А. О.				ВНТУ гр. 1БС-20м	
Н. Контр.		Лукічов В. В.					
Затверд.		Лужецький В. А.					

Архітектура програмного засобу



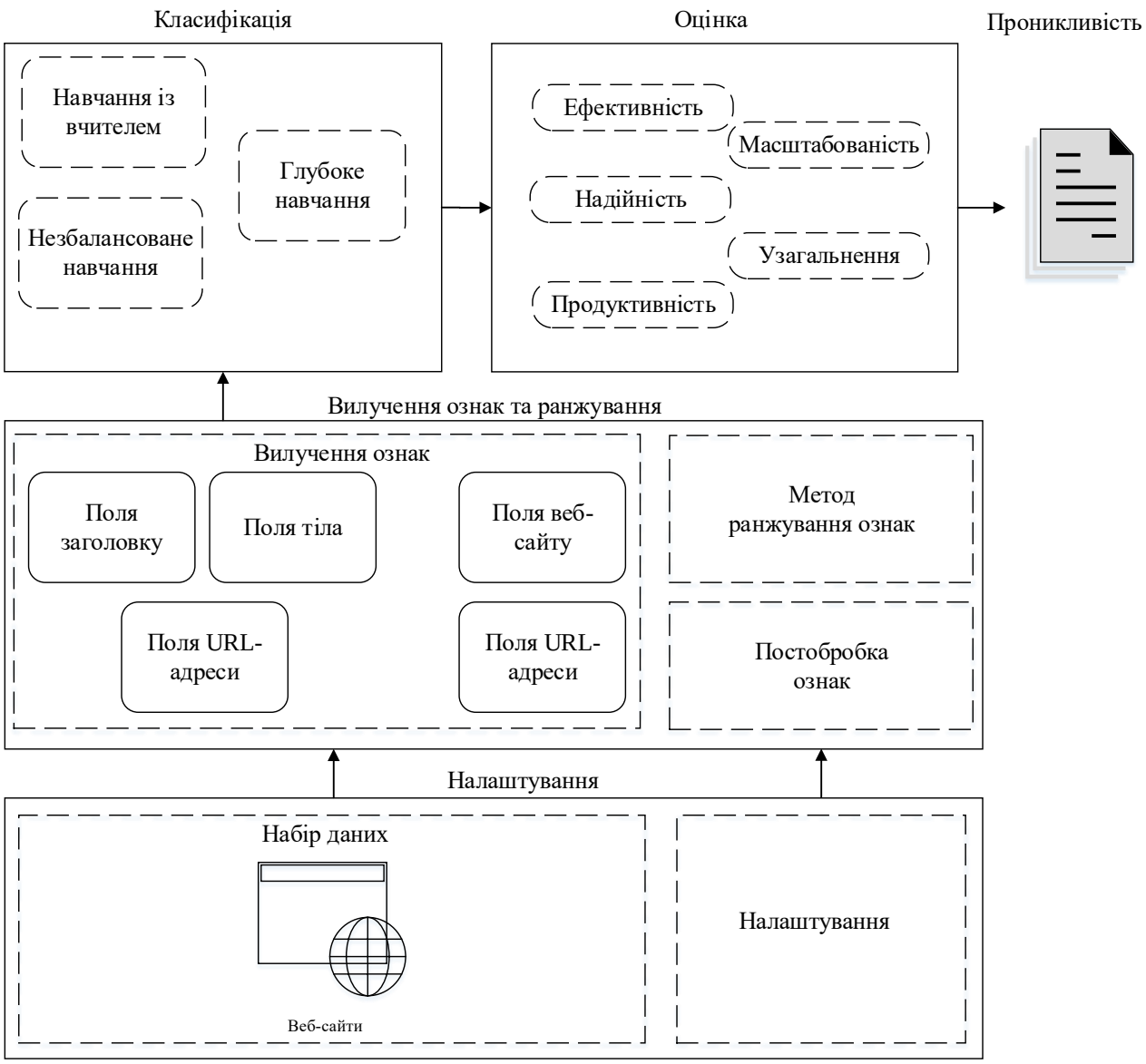
					08-20.МКР.002.00.000 ІЧ2			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Думчиков С. А.			Метод і засіб виявлення фішингових інформаційних ресурсів	Літ.	Арк.	Аркушів
Перевір.		Лукічов В. В.					1	1
Рецензент		Азарова А. О.				ВНТУ гр. 1БС-20м		
Н. Контр.		Лукічов В. В.						
Затверд.		Лужецький В. А.						

Схема роботи програмного засобу



					08-20.МКР.002.00.000 ПЗ		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Думчиков С. А.			Метод і засіб виявлення фішингових інформаційних ресурсів	Літ.	Арк. 1 / Аркушів 1
Перевір.		Лукічов В. В.					
Рецензент		Азарова А. О.				ВНТУ гр. 1БС-20м	
Н. Контр.		Лукічов В. В.					
Затверд.		Лужецький В. А.					

Схема запропонованої основи та системи порівняльного аналізу оцінки фішингу



					08-20.МКР.002.00.000 І44			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Думчиков С. А.			Метод і засіб виявлення фішингових інформаційних ресурсів	Літ.	Арк.	Аркушів
Перевір.		Лукічов В. В.					1	1
Рецензент		Азарова А. О.				ВНТУ гр. 1БС-20м		
Н. Контр.		Лукічов В. В.						
Затверд.		Лужецький В. А.						

Критерії оцінювання полів відповідно до значень метрик

Бали / Параметри	0	1
IP-адреса в URL	URL-адреса не є IP-адресою	URL-адреса є IP-адресою
Символ "@" в URL-адресі	в URL-адресі нема @	в URL-адресі є @
Довжина URL-адреси	довжина URL-адреси менша за 53 символи	довжина URL-адреси більша за 53 символи
Глибина URL-адреси	Проставляється число, що є глибиною даної URL-адреси	
Перенаправлення "//" в URL	якщо // немає будь-де, окрім після протоколу	якщо // знаходиться буде, окрім як після протоколу
"http/https" в імені домену	якщо http/https нема в самому імені домену	якщо є http/https в самому імені домену
Використання служб скорочення URL-адрес «TinyURL»	якщо URL-адреса не є скорочена	якщо URL-адреса є скороченням
Префікс або суфікс «-» у домені	якщо нема символу - в доменній частині адреси	якщо є символ - в доменній частині адреси
Запис DNS	якщо є DNS-запис про URL-адресу	якщо нема DNS-запису про URL-адресу
Трафік веб-сайту	якщо URL-адреса записана у БД Alexa	якщо про URL-адресу нема даних у БД Alexa
Вік домену	якщо вік домену URL-адреси більше 12 місяців	якщо вік домену URL-адреси менше 12 місяців
Кінцевий період домену	URL-адреса закінчується менше ніж за 6 місяців	URL-адреса закінчується більше ніж за 6 місяців
iFrame	є відповідь на запит	якщо iFrame пустий або нема відповіді
Mouse_Over	є негативна відповідь на подію onMouseOver	якщо відповідь пуста чи виявлено onMouseOver
Right_Click	є відповідь на подію event.button==2	якщо відповідь пуста чи виявлено onMouseOver
Web_Forwards	якщо перенапревлення менше 4 раз або його немає	якщо є перенаправлення мінімум 4 рази

					08-20.МКР.002.00.000 ПЗ		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Думчиков С. А.			Метод і засіб виявлення фішингових інформаційних ресурсів	Літ.	Арк. 1 / Аркушів 1
Перевір.		Лукічов В. В.					
Рецензент		Азарова А. О.				ВНТУ гр. 1БС-20м	
Н. Контр.		Лукічов В. В.					
Затверд.		Лужецький В. А.					

Робота програмного засобу

```
PS D:\Prjs\projectd> & C:/Users/G/AppData/Local/Programs/Python/Python310/python.exe d:/Prjs/projectd/demo.py
########################################################################
URL: https://vntu.edu.ua/
Ймовірність шкідливої URL-адреси для фішингу: 0.243
Сайт https://vntu.edu.ua/ знаходиться у зелена/ій групі
PS D:\Prjs\projectd>
```

```
########################################################################
URL: https://atb-market.space/
Ймовірність шкідливої URL-адреси для фішингу: 0.9998141
Сайт https://atb-market.space/ знаходиться у червона/ій групі
URL-адреса https://atb-market.space/ є небезпечною для відвідування. Рекомендуємо подумати про наслідки від втрати Ваших даних
Ви впевнені, що хочете перейти за адресою https://atb-market.space/ ?(y/n):
n
PS D:\Prjs\projectd>
```

```
################################################################################
URL: https://itc.ua/news/monobank-vzhe-vidkriv-15-mln-kartok/
Result [0]: Probability of Benign URL
Result [1]: Probability of Malicious URL for phishing
################################################################################
  ML Model                          Result
0  XGBoost  [[0.89188623, 0.108113736]]
Ймовірність шкідливої URL-адреси для фішингу: 0.108113736
Сайт https://itc.ua/news/monobank-vzhe-vidkriv-15-mln-kartok/ знаходиться у зелена/ій групі
PS D:\Prjs\projectd>
```

```
########################################################################
URL: https://evernote.com/register?upgrade=premium&itemCode=premium-1year&offer=www_pricing_CTA
Result [0]: Probability of Benign URL
Result [1]: Probability of Malicious URL for phishing
########################################################################
  ML Model                    Result
0  XGBoost  [[0.47405428, 0.5259457]]
Ймовірність шкідливої URL-адреси для фішингу: 0.5259457
Сайт https://evernote.com/register?upgrade=premium&itemCode=premium-1year&offer=www_pricing_CTA знаходиться у жовта/ій групі
Увага! Потенційно небезпечне посилання. Бажаєте перейти?
n
```

					08-20.МКР.002.00.000 ПЗ		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Думчиков С. А.			Метод і засіб виявлення фішингових інформаційних ресурсів	Літ.	Арк. 1 / Аркушів 1
Перевір.		Лукічов В. В.					
Рецензент		Азарова А. О.				ВНТУ гр. 1БС-20м	
Н. Контр.		Лукічов В. В.					
Затверд.		Лужецький В. А.					