Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ДОПУСТИТИ ДО ЗАХИСТУ:
В.о. завідувача кафедри
кібербезпеки та захисту
інформації
_____ Іван ПАРХОМЕНКО
«___» червня 2025 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

кваліфікаційної роботи

галузь знань 12 Інформаційні технології
(шифр і назва галузі знань)

спеціальність 125 Кібербезпека
(код і назва спеціальності)

освітній ступень бакалавр

освітня програма Кібербезпека
(назва освітньо-професійної програми)

на тему: «Метод детектування фішингових URL на основі принципу самоорганізації»

Виконавець: студентка IV курсу, групи КБ-41

_____ **Анастасія ШАБАНОВА**
(підпис) (ім'я, прізвище)

	Підпис	**Ім'я ПРІЗВИЩЕ**
Керівник		Сергій БУЧИК
Нормоконтроль		Леся БАРАНОВСЬКА

Київ 2025

Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
В.о. завідувача кафедри
кібербезпеки
та захисту інформації
_____ Іван ПАРХОМЕНКО
«29» листопада 2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності 125 Кібербезпека
(код і назва спеціальності)

освітньої програми Кібербезпека
(назва освітньо-професійної програми)

Студентці **КБ-41** **Шабановій Анастасії Олексіївні**
(група) (прізвище ім'я по батькові)

Тема кваліфікаційної роботи Метод детектування фішингових URL на основі принципу самоорганізації

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Тема кваліфікаційної роботи затверджена на засіданні кафедри кібербезпеки та захисту інформації протокол №6 від 28.11.2024 р.

2. ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Структуровані URL-адреси, розмічені за ознаками потенційної фішингової активності, на основі яких формуються вектори чіткого портрета для подальшої класифікації

3. ЗМІСТ РОЗРАХУНКОВО-ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Теоретичне обґрунтування проблеми; аналіз сучасних фішингових атак і методів їх виявлення; формалізація чіткого портрета URL-адреси; реалізація моделей класифікації (логістична регресія, GMDH, SOM); експериментальне дослідження ефективності методу; аналіз результатів та рекомендації

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Практична цінність Реалізація структурного методу виявлення фішингових URL-адрес на основі чіткого портрета адреси та моделей класифікації (логістична регресія, GMDH, SOM), який може бути інтегрований у веб-фільтри та системи захисту задля виявлення нових фішингових шаблонів.

5. ДАТА ВИДАЧІ ЗАВДАННЯ

Дата видачі завдання: 29 листопада 2024 року

Завдання видав _____ (підпис) Сергій БУЧИК (ім'я, прізвище)

Завдання прийняла до виконання _____ (підпис) Анастасія ШАБАНОВА (ім'я, прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів робіт	Строки виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки завдання	29.11.2024 – 05.12.2024	*виконано*
2	Аналіз літератури та актуальних проблематик фішингових загроз	06.12.2024 – 29.12.2024	*виконано*
3	Теоретичне обґрунтування формалізація портрета URL-адреси	15.01.2025 – 29.01.2025	*виконано*
4	Реалізація моделей класифікації (логістична регресія, GMDH) та їх тестування	30.01.2025 – 14.02.2025	*виконано*
5	Реалізація модуля Self-Organizing Map для дослідження структури ризику	15.02.2025 – 28.02.2025	*виконано*
6	Експериментальна перевірка методу, оцінка якості класифікації та інтерпретованості рішень	29.02.2025 – 29.03.2025	*виконано*
7	Написання розрахунково-пояснювальної записки	01.04.2025 – 15.05.2025	*виконано*
9	Підготовка до захисту кваліфікаційної роботи	24.05.2025 – 13.06.2025	*виконано*

Завдання видав _____ (підпис) Сергій БУЧИК (ім'я, прізвище)

Завдання прийняла до виконання _____ (підпис) Анастасія ШАБАНОВА (ім'я, прізвище)

Термін подання кваліфікаційної роботи до ЕК 13 червня 2025 року

РЕФЕРАТ

Кваліфікаційна робота складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел, 10 додатків, має 84 сторінок основного тексту, 1 таблицю та 24 рисунків. Список використаних джерел містить 28 найменуваннь і займає 4 сторінки.

Метою роботи є реалізація методу детектування фішингових URL-адрес на основі чіткого портрета адреси та принципу самоорганізації моделей, який здатний виявляти сучасні фішингові шаблони з переважною точністю та пояснюваністю рішень.

Досягнення мети потребує розв'язання таких задач:

– сформувати формальну портретну модель URL-адреси, що уніфікує її структурні, технічні та семантичні ознаки, та обґрунтувати її застосування для задач класифікації;

– розробити архітектуру методу виявлення фішингових шаблонів, яка включає модулі логістичної регресії як інтерпретованої базової моделі, GMDH для самоорганізованої побудови нелінійних залежностей та Self-Organizing Map для дослідження кластерів ризику;

– здійснити програмну реалізацію запропонованої архітектури у середовищі Python та провести експериментальні дослідження з використанням реальних датасетів фішингових та легітимних URL-адрес;

– провести багатовимірний аналіз точності класифікації та виявити ключові фактори, що впливають на ефективність методу, зокрема показники F1-score, ROC AUC та показники стабільності;

– розробити рекомендації щодо впровадження методу у практичні сценарії кіберзахисту, зокрема у веб-фільтри, поштові шлюзи та центри моніторингу кібербезпеки.

Об'єктом дослідження є процеси виявлення та класифікації фішингових URL-адрес у веб-просторі.

Предметом дослідження є методи та моделі детектування фішингових URL-адрес, зокрема побудова портрета адреси, реалізація алгоритмів класифікації (логістична регресія, GMDH), а також використання Self-Organizing Map для дослідження ризикових шаблонів.

Практична значущість отриманих результатів полягає у розробці методу інтелектуального виявлення фішингових URL-адрес, який характеризується високим рівнем прецизійності та інтерпретованості прийнятих рішень. Запропонований підхід може бути інтегрований у складні системи кібербезпеки, зокрема веб-фільтри, поштові шлюзи та модулі аналізу трафіку, з метою забезпечення автоматизованого виявлення і блокування фішингових атак у режимі реального часу, що суттєво підвищує загальний рівень захищеності інформаційного середовища.

Ключові слова: фішингові URL-адреси, чіткий портрет URL, принцип самоорганізації, логістична регресія, GMDH, Self-Organizing Map, XAI.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

URL	–	Uniform Resource Locator – уніфікований ідентифікатор ресурсу в Інтернеті;
TLD	–	Top-Level Domain – домен верхнього рівня;
HTTPS	–	HyperText Transfer Protocol Secure – протокол захищеної передачі даних;
XAI	–	Explainable Artificial Intelligence – пояснювана штучна інтелектуальна система;
GMDH	–	Group Method of Data Handling – груповий метод обробки даних;
SOM	–	Self-Organizing Map – самоорганізуюча карта;
LR	–	Logistic Regression – логістична регресія;
ROC AUC	–	Receiver Operating Characteristic Area Under the Curve – площа під кривою ROC;
F1-score	–	інтегральна метрика точності та повноти у задачах класифікації;
WHOIS	–	сервіс перевірки інформації про доменне ім’я;
PhishTank	–	відкрита база даних фішингових URL-адрес;
IP	–	Internet Protocol – протокол мережевої адресації;
CV	–	Cross-Validation – крос-валідація;
API	–	Application Programming Interface – інтерфейс програмування додатків;
X_train / X_test	–	навчальна / тестова вибірки для побудови та оцінки моделей
FP	–	хибно позитивне спрацювання класифікатора;
FN	–	хибно негативне спрацювання класифікатора;
TP	–	правильне позитивне спрацювання класифікатора;
TN	–	правильне негативне спрацювання класифікатора.

ВСТУП

У сучасній кіберінформаційній парадигмі фішингові атаки залишаються однією з найбільш адаптивних та стійких форм загроз інформаційній безпеці. Їх еволюція супроводжується впровадженням складних механізмів маскування, що базуються не лише на соціотехнічних прийомах, а й на глибокій трансформації структури веб-ідентифікаторів. Зокрема, URL-адреси, як первинні вектори доступу до ресурсів – дедалі частіше виступають основним інструментом фішингових кампаній, що зумовлює потребу в розробці методів, здатних здійснювати структурно-семантичний аналіз адрес з метою виявлення прихованих ознак аномальної активності.

Існуючі системи протидії фішинговим загрозам, як правило, спираються на статичні моделі – зокрема, чорні списки, евристичні правила або обмежені в узагальнюваності алгоритми машинного навчання. Такий підхід не забезпечує належної ефективності в умовах швидкої ротації атакувальних шаблонів, короткоживучих доменів та псевдолегітимних Top-Level Domain(TLD). Крім того, у практиці застосування систем безпеки актуалізується вимога до пояснюваності класифікаційних рішень, особливо у сферах, де результати мають бути прозорими для аудиту (наприклад, у банківському або державному секторі).

З огляду на зазначене, особливого значення набуває створення інтелектуальних систем виявлення фішингових шаблонів, які б поєднували точність класифікації, адаптивність до нових атак та можливість інтерпретації рішень XAI (Explainable AI).

Метою даної роботи є розробка методу детектування фішингових URL-адрес на основі портретної ознакової моделі та самоорганізованих алгоритмів класифікації, що забезпечують баланс між точністю, стійкістю до нових шаблонів та пояснюваністю результатів.

Об’єктом дослідження виступають процеси виявлення та класифікації фішингових адрес у веб-просторі, а предметом – методологія побудови ознакових моделей URL, алгоритми логістичної регресії, Group Method of Data

Handling(GMDH) та Self-Organizing Maps(SOM), а також механізми їх інтеграції в системи кіберзахисту.

Методи дослідження охоплюють: формальний структурний аналіз URL; побудову векторизованих портретів; реалізацію та тренування моделей класифікації; багатовимірну валідацію точності, а також аналіз інтерпретованості рішень згідно з принципами XAI.

Галузь застосування розробленого методу охоплює прикладні компоненти інформаційної безпеки, зокрема: модулі веб-фільтрації, захист електронної пошти, системи запобігання втраті даних, захищені DNS-сервіси, а також платформи моніторингу подій безпеки. Метод може бути адаптований для інтеграції у браузерні розширення, проксі-сервери або шлюзи контролю доступу, де необхідна оперативна і пояснювана оцінка ризику на основі аналізу URL.

Практична цінність запропонованого підходу полягає у створенні адаптивного методу виявлення фішингових шаблонів, який забезпечує високу точність класифікації без потреби у контентному аналізі сторінок, що дозволяє його використання навіть у середовищах із обмеженим доступом до мережі або у режимах превентивного фільтрування.

Апробація роботи. Основні результати роботи доповідались на таких наукових конференціях:

- X Міжнародна наукова конференція "Information Technology and Implementation" (IT&I-2023), 20–21 листопада 2023 р., м. Київ, Україна – представлено доповідь "The Method for Determining the Degree of Suspiciousness of a Phishing URL" (CEUR Workshop Proceedings, Scopus)[1];
- Конференція "Information Technology and Implementation (Satellite)", 21 листопада 2024 р., м. Київ, Україна – представлено доповідь "Detecting Phishing URLs Using Machine Learning and Clustering Algorithms"[2].

Крім участі в наукових конференціях, окремі результати дослідження були подані на І тур Всеукраїнського конкурсу студентських наукових робіт, за підсумками якого отримано дипломом І ступеня. У межах апробації також долучилася до міжнародного освітнього проєкту COIL (Certified Online

International Learning), організованого у співпраці між КНУ імені Тараса Шевченка та American University in the Emirates, де взяла участь у проведенні тренінгових сесій з питань кіберризиків в аспекті тематики фішингу. Дані форми участі доповнюють апробацію результатів роботи та підтверджуються відповідними документами, наведеними в Додатках Л та М відповідно.

РОЗДІЛ 1. АНАЛІТИЧНА ПЛАТФОРМА ВИЯВЛЕННЯ ФІШИНГОВИХ ЗАГРОЗ У КОНТЕКСТІ ЦИФРОВОЇ БЕЗПЕКИ

1.1. Анатомія фішингових атак: типологія, цілі, еволюція

Згідно з сучасними тенденціями, фішинг перетворився з примітивного інструмента шахрайства на складний, багаторівневий вектор кіберзагроз, що поєднує технічні методи з елементами психологічного впливу. В контексті цифровізації публічних сервісів, державних порталів, банківських і логістичних платформ, фішинг виступає не лише інструментом для викрадення облікових даних, але й засобом впливу на довіру до інформаційної інфраструктури як державних, так і корпоративних ресурсів.

Визначення фішингу (phishing) розглядається як метод соціальної інженерії, що передбачає обман користувача шляхом імітації легітимного ресурсу з метою отримання конфіденційної інформації: логінів, паролів, платіжних даних, службових доступів, цифрових підписів тощо. Ще у період 2023 року фішингові атаки набули нових більш складних механізмів та були знову визнаними найбільш розповсюдженим класом інцидентів за статистикою CERT-UA[3], ENISA Threat Landscape[4], а також аналітикою Verizon DBIR[5].

Основні найпоширеніші цілі фішингу, що можуть бути виокремлені на основі структурованої добірки з Додатку А, представлені у вигляді наступних спрямувань мети атаки :

- крадіжка облікових даних (логіни, паролі, токени доступу);
- фінансове шахрайство (переведення коштів, викрадення платіжних реквізитів);
- дестабілізація цифрових сервісів (особливо в державному секторі);
- збір розвідувальної інформації (про структури, користувачів, доступи);

- порушення репутації (використання довіри до бренду або урядових платформ).

Згідно з даними рис.1.1, фішингові атаки, характерні для ЄС, мають яскраво виражену секторну спрямованість, а також чітку географічну динаміку. Найбільш уразливою групою виявляється широкий загал (General Public) – 42,7% усіх зафіксованих загроз, що свідчить про переважання масових фішингових кампаній, що орієнтовані на індивідуальних користувачів через інструменти соціальної інженерії, месенджери та фальшиві державні сервіси. Другий діаграмний блок на правій частині зображення демонструє, що найбільше навантаження припадає на глобальні кампанії, тоді як регіональні атаки в межах ЄС складають помітну, але меншу частку, що описує переважання мультинаціональних схем фішингу, які зазвичай використовують універсальні шаблони, масово поширювані через інфраструктуру типу CDN, реєстраторів ризикових TLD або платформ з короткими посиланнями[6].

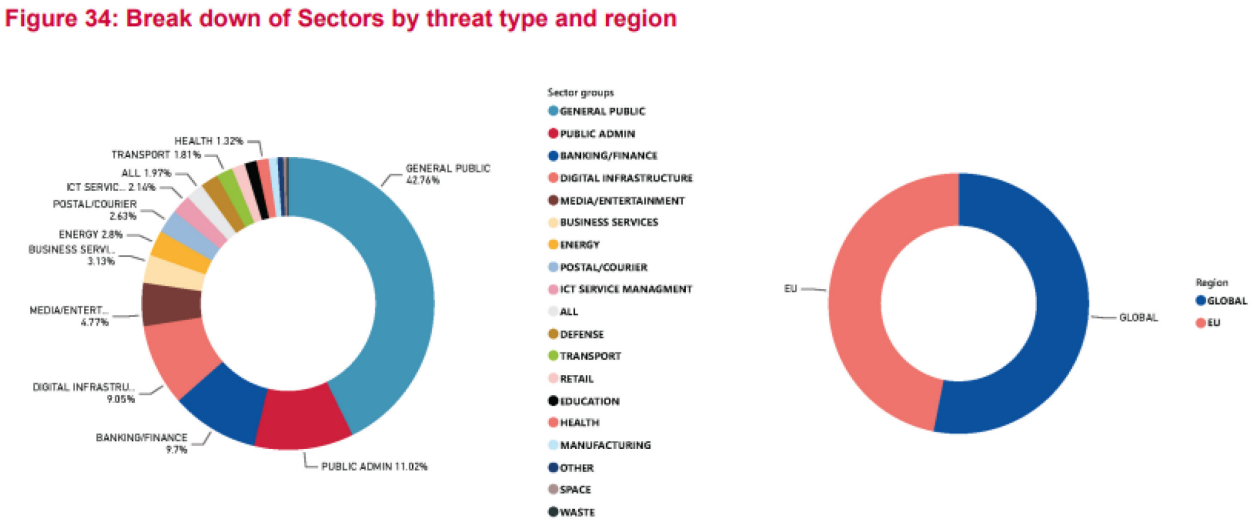


Рисунок 1.1 – Розподіл секторів за типом загрози та регіоном

Значну роль у фішингових кампаніях відіграє маніпулятивна семантика – зловмисники використовують елементи, подібні до відомих брендів, національних або урядових платформ: gov.ua, bank.gov.ua, diia.gov.ua, ehealth.gov.ua, ukrposhta, monobank, тощо[7]. Візуальна подібність фішингових

сайтів до офіційних зменшує критичність користувача і збільшує ймовірність “кліку”.

У динаміці останніх років можна виділити три основні етапи еволюції фішингу:

- простий HTML-фішинг – самостійно зібрані форми входу, розсилки через електронну пошту.
- автоматизовані генератори шаблонів – платформи, які масово створюють шаблони на основі доменів .top, .xyz, .tk тощо.
- цифрово інтегровані фішингові кампанії – комбінування фішингу з ботами, QR-кодами, шкідливими скриптами, і навіть реєстрація фейкових компаній/сайтів через легальні сервіси.

Дослідження Google Threat Analysis Group [8] у 2022 році показало, що фішингові домени живуть у середньому 2–4 дні, однак цього достатньо для реалізації атаки. Саме тому традиційні методи, що базуються на чорних списках (blacklists), виявляються не стільки вже ефективними, якщо не доповнені адаптивною аналітикою.

1.1.1. Таксономія фішингових стратегій у веб-просторі

Аналіз актуальних фішингових стратегій свідчить про системну зміну як технічних, так і соціотехнологічних підходів до впливу на кінцевого користувача. Особливо помітною ця динаміка стала в період з 2022 по 2025 роки[7], коли фішинг із інструмента інтернет-шахрайства еволюціонував у складову інформаційно-психологічних операцій, активно використовуваних у геополітичних конфліктах.

Фішингові кампанії більше не є однорідними: зловмисники комбінують застарілі прийоми з гіперсучасними техніками, адаптуючи канали доставки, мову повідомлень, доменні імена, SSL-сертифікати, структуру URL[9]. Відповідно до звітів IBM X-Force та APWG, фішингові атаки у 2024 році є найбільш масовим інструментом для:

- викрадення облікових даних державних і публічних сервісів;
- імітації міжвідомчого електронного документообігу;
- зараження пристроїв через вебредиректи та обман сертифікатів.

Особливої уваги потребує структурна типологізація фішингових шаблонів (таксономія), яка не тільки описує типи атак, а й дозволяє будувати аналітичні моделі – автоматично розпізнавати шаблони на основі їх цифрового відбитку (URL), пояснювати логіку класифікації, формувати макети ризику в реальному часі.

1.1.2. Державні й корпоративні сервіси як високочутливі об’єкти

Починаючи з 24 лютого 2022 року, Україна переживає безпрецедентне зростання кібератак, які супроводжують активну фазу військових дій. Однією з ключових мішеней для фішингових кампаній стали державні та критично важливі корпоративні цифрові сервіси, які забезпечують функціонування урядової інфраструктури, логістики, фінансів, енергетики та медицини – загальна інфографіка представлена на рис.1.2.

Особливо активно експлуатуються сервіси, які асоціюються з виплатами, допомогою та реєстрацією даних. Типовим прикладом став фішинговий сайт e-service.registry-gov-ua.info, виявлений у березні 2023 року. Цей домен був зареєстрований через недорогого іноземного реєстратора та стилістично повністю імітував державний ресурс: у дизайні використовувались кольори платформи “Дія”, псевдологотип Міністерства соціальної політики, формулювання в стилі офіційних комунікацій. Ключовою особливістю атаки було те, що фішингове навантаження поширювалося не лише через електронну пошту, а здебільшого через Viber, Telegram та SMS-розсилки. Користувачам надходило повідомлення з посиланням на сайт, який позиціонував себе як “Центральний реєстр документів ВПО 2023” з активною кнопкою “Отримати виплату”. Сайт мав чинний SSL-сертифікат Let’s Encrypt, що створювало додаткову ілюзію легітимності.

У формі на сайті запитували особисті дані: ПІБ, дату народження, номер ІПН, IBAN-рахунок, а також номер картки й CVV-код. Після введення користувач отримував фіктивне підтвердження з текстом “Очікуйте SMS протягом 24 годин”, тоді як дані надсилалися на сервер, контрольований зловмисниками. Ця атака мала дві цілі: фінансову (крадіжка коштів з рахунків) та соціотехнічну – створення хвилі недовіри до державних виплат, що могла мати дестабілізуючий ефект на фоні гуманітарної напруги.

Аналогічні схеми були зафіксовані щодо банківських структур – особливо Monobank, ПриватБанк, Ощадбанк, де створювалися фішингові сайти під виглядом авторизації, або навіть запускалися псевдочати техпідтримки. У секторі логістики під удар потрапили наступні сервіси: Нова Пошта, Укрпошта та Meest, оскільки вони стали критичними для забезпечення тилу та гуманітарних поставок. Імітація сайтів трекінгу, підробка push-сповіщень, HTML-форми “підтвердити адресу” – все це активно використовувалось в реальних атаках. Особливо підступною була підміна сайтів медичних установ, наприклад, eHealth або фіктивні сервіси “перевірки COVID-сертифікатів”, на кшталт covid19-check.healthgov.info[10].

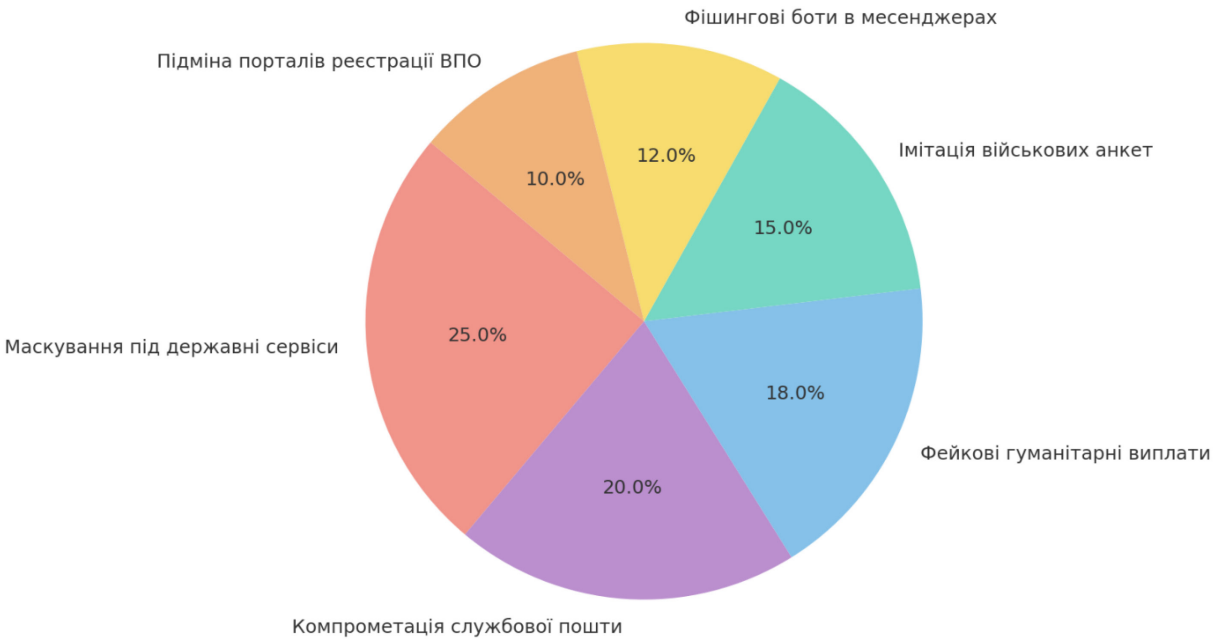


Рисунок 1.2 – Типи фішингових стратегій в умовах війни (станом на 2022–2025)

Додаток А ілюструє 10 прикладів реальних фішингових шаблонів, класифікованих за каналом доставки, типом атаки та цільовою тематикою, що охоплюють сценарії, спрямовані на маскування під державні сервіси, банківські інтерфейси та системи документообігу.

Фішинг проти державного та корпоративного сектора стає не лише технічно складнішим, а й більш адаптивним: зловмисники використовують сучасні реєстраційні платформи, щоб створити домени максимально подібні до оригінальних, застосовують SSL, мобільну адаптацію, стилістику GUI. Основним викликом є те, що громадяни не мають технічної можливості відрізнити підробку при відкритті сторінки та перегляді її вмісту, тому моделі, засновані на аналізі векторів чіткого портрета URL, є ключем до своєчасного виявлення фішингу ще до моменту кліку.

1.2. Структурно-семантичний аналіз URL як вхідного вектора атаки

Масовість фішингу пояснюється не лише доступністю створення шкідливих посилань, а й складністю їхньої оперативної верифікації, навіть на рівні автоматичних фільтрів. Більше того, на відміну від класичних кіберінцидентів, фішинг не потребує експлуатації технічної вразливості – він використовує починаючи від психологічно нескладних характеристиках – довіри та, закінчуючи на брендингу, національній стилістиці та ефекті інформаційного хаосу як основного інструменту атаки.

Хрестоматійним прикладом високотехнологічної кампанії став кейс оновленого типу «DUCKTAIL», виявлений у 2024 році дослідницькими командами Meta та ESET. Атака була орієнтована на бізнес-користувачів у Meta Business Manager у країнах ЄС, зокрема Польщі, Чехії, Угорщині та в Україні. Жертвам надсилали повідомлення із посиланням на сторінку “підтвердження бізнес-ідентичності”, яке виглядало як офіційний портал Meta (business-verify-meta.support-live.com). На сторінці була HTML-форма з уже частково заповненими полями (реалізовано через витяг cookie), дійсний SSL-сертифікат,

реальний Favicon Meta, а також візуальна адаптація під мову користувача. Особливістю було використання параметрів типу access_token, manager_id в URL, що використовувалися для захоплення активної сесії Facebook Business. Кампанія тривала менше 4 діб, однак охопила понад 4 тис. компаній, за даними Group-IB.

У цей самий період CERT-UA зафіксувала кампанію, що описана у пункті 1.1.2, яка була особливо таргетовано на прифронтові регіони (Запорізька, Харківська, Донецька області).

Порівняння обох випадків демонструє ключову закономірність: фішингові кампанії більше не є ізольованими. Вони базуються на імітації державних і корпоративних цифрових сервісів, використовують спільні патерни: складна структура URL, псевдозаконні TLD (.info, .live, .support), наявність ключових слів у path (verify, access, token, auth), перенаправлення через довгі query-параметри. Зовні – це абсолютно “нормальні” адреси, які пройшли SSL-верифікацію, мають адаптивний дизайн, favicon, швидкість завантаження. Однак саме внутрішній розбір їх структури – кількість піддоменів, використані слова, довжина, ризиковість TLD – дозволяє виявити ризик ще до переходу.

Висновки міжнародних звітів підтверджують: фішинговий шаблон є аналітичним об’єктом, що має предикативну ознакову модель, яка повторюється, а значить – піддається формалізації.

1.2.1. Формальна структура URL: розбір основних компонентів

URL-адреса (Uniform Resource Locator) у контексті фішингових атак є не просто вхідною точкою доступу до ресурсу – вона виступає як носій структурованої інформації, що потенційно дозволяє ідентифікувати ризикову поведінку ще до фактичного звернення до сторінки. Дана характеристика перетворює URL із пасивного елемента мережевої навігації на активну ознакову одиницю, що може бути формалізована, описана та використана в автоматизованих системах виявлення загроз[11].

Будь-яка URL-адреса складається з декількох обов'язкових або умовних компонентів, кожен із яких несе окреме смислове й технічне навантаження, декомпозицію структури, продемонстровані на рис.1.3, тому класичний формат URL включає наступні частини:

Протокол (scheme) – визначає спосіб взаємодії з ресурсом (http, https, ftp). У контексті фішингу важливо враховувати, чи використовується захищене з'єднання, оскільки зловмисники часто імітують https через безкоштовні сертифікати типу Let's Encrypt.

Хост (host/domain) – основний ідентифікатор ресурсу, який може бути легітимним (наприклад, gov.ua) або підробленим, часто з псевдодержавним виглядом (registry-gov-ua.info, mfa-ukr.org).

TLD (top-level domain) – доменна зона (.com, .ua, .top, .info), що часто використовується як один із маркерів довіри. Так звані "ризикові TLD", які активно використовуються у фішингових кампаніях, мають низьку вартість та низький рівень перевірки.

Піддомени (subdomains) – елементи, які передують основному домену та часто використовуються для маскування (наприклад, login.verify.dia-gov.com).

Шлях (path) – вказує на ресурс, до якого відбувається звернення (/verify/index.php), і зазвичай містить слова-тригери: login, confirm, access, auth, тощо.

Параметри (query string) – елементи після символу ?, які передають змінні (?sessionid=9483&token=abc123). У фішингових шаблонах часто використовуються для імітації реального авторизаційного процесу.

Якір (fragment) – необов'язкова частина, що позначає конкретну секцію сторінки (#form), рідко зустрічається у фішингових шаблонах, але іноді застосовується для подальшого маскування.

Кожен із вказаних компонентів, у відповідному контексті, може бути розбитий на логічні ознаки, які згодом формують основу для так званого чіткого портрета URL. Це означає, що замість обробки повного тексту адреси, метод працює з вектором значень, де кожна позиція відповідає за певний параметр.

Наприклад, довжина URL може бути числовим значенням, наявність SSL – булевою ознакою (1 або 0), кількість піддоменів – цілим числом, наявність підозрілих слів у path – логічною ознакою, а TLD – категоріальною величиною, що може бути замаплена у відповідні числові коди ризику.

Рисунок 1.3 – Декомпозиція структури URL-адрес

Формальна структура URL дає змогу систематизувати подібні шаблони: визначати типовість, повторюваність, граматичні особливості та навіть

пов’язаність із конкретними фішинговими кампаніями. Наприклад, за результатами аналізу CERT-UA, більшість шкідливих URL мають понад три піддомени, містять TLD .top, .site, .info, та ключові слова у шляху (verify, login, grant, secure). У зв’язку з цим, формальна структура URL дозволяє провести статистичне попереднє оцінювання рівня ризику ще до контакту з вмістом сторінки.

1.2.2. Механізми маскування: від редиректів до псевдодоменів

Маскування справжньої сутності веб-ресурсу, реалізоване через структуру URL, є одним з ключових інструментів фішингових атак, що дозволяє зловмисникам обійти системи виявлення на ранніх стадіях взаємодії. Механізми маскування при цьому функціонують як засіб зниження помітності адреси, спотворення її семантики або впровадження фальшивої логіки навігації у свідомість користувача. Подібні техніки є об’єктом дослідження не лише з точки зору поведінкового аналізу користувача, а й у контексті структурної евристики фішингових шаблонів[12].

Згідно з аналітичними матеріалами, представленими у дослідженні Адамова (ще станом на 2019), фішингові посилання, що імітують популярні онлайн-сервіси, найчастіше маскуються за допомогою редиректів, багаторівневих піддоменів, доменів із високим ступенем ризику (наприклад, .top, .xyz, .info), а також обманних елементів у параметрах запиту (query-string).

Одним із базових прийомів є псевдодоменне маскування, за якого доменні імена навмисно конструюються так, щоб викликати довіру, наприклад, через додавання префіксів типу gov, login, auth, що візуально нагадують державні або безпечні сервіси. Адреса cabinet.gov-ukr.support або verify-diia-gov.info не є частиною державної інфраструктури, проте сприймається користувачем як легітимна. Додатково зловмисники використовують піддомени, що поглиблюють рівень маскування, створюючи структури типу secure.login.update.auth.diia-id.info.

Ще більш ускладненим є використання механізмів редиректу, коли початковий URL має вигляд нешкідливого сервісу (bit.ly, lnk.to, cloudfront.net), а кінцевий – веде на фішингову сторінку з параметрами авторизації або збору даних. Така схема дозволяє обійти фільтри превентивного блокування, оскільки початковий домен не входить до чорних списків. Приклад із наборів даних у дисертаційному дослідженні показує, що редиректи, які використовуються для маскування URL, зустрічаються у понад 40% фішингових шаблонів, з яких 63% мають трирівневу глибину та включають змінні параметри типу ref=, id=, token=1.

Іншим поширеним методом є використання символу @ у складі доменного рядка – складова техніки, що дозволяє приховати справжню адресу після символу авторизації. Наприклад, https://gov.ua@malicious-host.net у більшості браузерів інтерпретується як доступ до malicious-host.net, але створює ілюзію авторитетного ресурсу. Застосування такої структури у поєднанні з обманним SSL (Let’s Encrypt) є типовою тактикою сучасних атак.

Також актуальною є практика структурної інфляції параметрів запиту. Фішингові посилання часто містять перевантажений query-string, в якому присутні десятки змінних – з метою ускладнити автоматичний аналіз або симулювати сесійну взаємодію. Адреси типу https://auth-verify-gov.info/login?step=2&redirect=https://service.ua/cabinet&access_id=7239482 є прикладом подібного навантаження.

Результати експериментального дослідження, проведеного на основі набору із понад 98 тис. фішингових посилань (згідно з даними бази PhishTank), показали, що середній розмір фішингового URL перевищує 76 символів, а частка доменів із кількістю піддоменів більше 2 становить 71% від загальної вибірки. У понад 30% випадків використовуються псевдословники (login, verify, support, secure) у шляху або параметрах URL.

Таким чином, описані механізми маскування, що формують візуальну або структурну подібність до легітимних ресурсів, не лише створюють виклик для користувача, а й вимагають високого рівня формалізації ознак у процесі

побудови моделі аналізу. Саме ці ознаки – кількість піддоменів, ризиковість TLD, наявність слів-маркерів, символ @, редиректні патерни, довжина та складність шляху – піддаються векторизації та включаються до чіткого портрета URL, який може бути оброблений засобами машинного навчання для детекції потенційної загрози без потреби вмістового аналізу сторінки.

1.3. Підходи до детектування: від евристики до XAI

Зважаючи на складність сучасних фішингових шаблонів, ефективне виявлення таких загроз вимагає поєднання різних класів методів – від класичних евристичних правил до повноцінних систем інтерпретованого машинного навчання (XAI). Історично першими засобами протидії фішингу стали чорні списки (blacklists), проте їх обмежена адаптивність до нових шаблонів значно знижує практичну ефективність. Із розвитком моделей на основі даних (data-driven approaches), зокрема ансамблевих дерев, нейронних мереж та логістичних моделей, виявлення фішингу перейшло від статичних правил до системної класифікації структурних ознак адреси[13].

Проте зростання обчислювальної складності моделей призводить до погіршення пояснюваності – а саме ця характеристика критично важлива у сфері інформаційної безпеки, де кожне рішення має бути обґрунтованим для аудитора або адміністратора системи. Таким чином, пошук оптимального методу класифікації з урахуванням точності, адаптивності, прозорості та складності інтеграції стає важливим завданням системного проєктування.

1.3.1. Огляд моделей виявлення: blacklist, random forest, нейронні мережі

У контексті автоматизованого виявлення фішингових впливів, побудованого на основі векторизованого портрета URL-адрес, ключовим аспектом проєктування моделі виступає узгодження методу класифікації з архітектурними й функціональними критеріями системи, включаючи швидкість

реагування, стабільність до змін шаблонів атак, здатність до пояснення результату, а також можливість розгортання в автономному режимі.

Традиційні моделі фільтрації, зокрема blacklist-підхід, базуються на принципі виявлення відомих фішингових доменів або URL-адрес, зафіксованих у централізованих базах даних. Незважаючи на простоту реалізації, такий підхід принципово не здатен ідентифікувати нові шаблони – тобто ті, що не входять до попередньо верифікованого списку (або навіть ті, які мають лише певну лінійну залежність зміни символів). У системах захисту інфраструктур критичного призначення (державні шлюзи, web-проксі установ, безпечний DNS) ця обмеженість може створювати значну латентність виявлення загроз[14].

З метою формалізації шаблонів за ознаками, виникає потреба у використанні класифікаторів на основі машинного навчання, які здатні обробляти чіткий вектор ознак URL та, що є складнішим, формувати рішення з прийнятним балансом точності та інтерпретованості.

Логістична регресія (LR), як базова модель бінарної класифікації, є однією з найпоширеніших у завданнях, пов’язаних з аналізом веб-трафіку. У межах концепції “портрета” кожна ознака (довжина URL, наявність SSL, кількість параметрів, підозрілі слова) набуває коефіцієнта впливу, який прямо впливає на ймовірність класифікації URL як фішингового. Таким чином, LR забезпечує повну прозорість процесу прийняття рішень, що є критичним для роботи в умовах стандартів ISO/IEC 27001 та вимог explainable AI (XAI). Її обмеження проявляються переважно у випадках високої нелінійності або комбінаційних зв’язків між ознаками, що не можуть бути виражені у вигляді зваженої суми.

Саме в останніх випадках та інших наслідкових умовах ускладнення шаблонів доцільним є застосування деревоподібних або ансамблевих моделей, зокрема Random Forest, які дозволяють відображати складні перехресні залежності, проте такі моделі значно втрачають у пояснюваності. Додаткові методи (SHAP, LIME) можуть бути інтегровані як надбудови, однак це ускладнює архітектуру і порушує вимогу до прозорості на рівні першої моделі.

У державних системах верифікації трафіку, де необхідно оперативно аргументувати блокування ресурсу, ця “чорна скринька” є неприйнятною.

Нейронні мережі (MLP, RNN, LSTM) забезпечують найвищу точність при достатньому обсязі навчального набору та можуть враховувати контекст (наприклад, послідовність символів). Однак, у порівнянні з іншими методами, вони є найменш пояснюваними, найбільш вимогливими до ресурсів та практично не мають прозорої візуальної логіки. Їх використання в контексті системи з портретною ознаковою моделлю – технічно можливе, але методологічно слабко узгоджене з вимогами аудиту й безперервної перевірки[15].

Отже, найбільш збалансованим підходом у межах даної роботи є застосування GMDH (групового методу обліку аргументів). Представлений індуктивний метод побудови багаторівневої моделі класифікації, заснований на автоматичному відборі зв’язків між ознаками. GMDH дозволяє не лише генерувати логічну структуру класифікації, але й адаптувати її до нових вхідних патернів без повного перенавчання. У цьому контексті модель GMDH виконує роль самоорганізованого класифікатора, який побудований без зовнішнього втручання, але має чітко пояснювану логіку на рівні кожного вузла. Саме такий підхід є найбільш придатним для систем, що мають функціонувати в умовах динамічного оновлення шаблонів атак та обмеженої пропускної здатності (наприклад, у режимі проксі-фільтрації в офлайн-середовищі).

Для узагальненого розуміння особливостей, перелічених вище, на рис. 1.4 наведено зведене порівняння розглянутих моделей за чотирма критеріями: пояснюваність (XAI), точність, гнучкість/адаптивність та складність впровадження. За результатами цієї оцінки, найбільш придатними до інтеграції у портретний метод фільтрації визнано логістичну регресію (висока пояснюваність, проста реалізація, низька ресурсомісткість) та GMDH (автоматичне уточнення структури, адаптація до змінних шаблонів при помірній складності). Використання XGBoost, Random Forest та нейромереж рекомендовано лише в контексті офлайн-моделювання або як вторинна референтна модель(в локальних випадках надбудови).

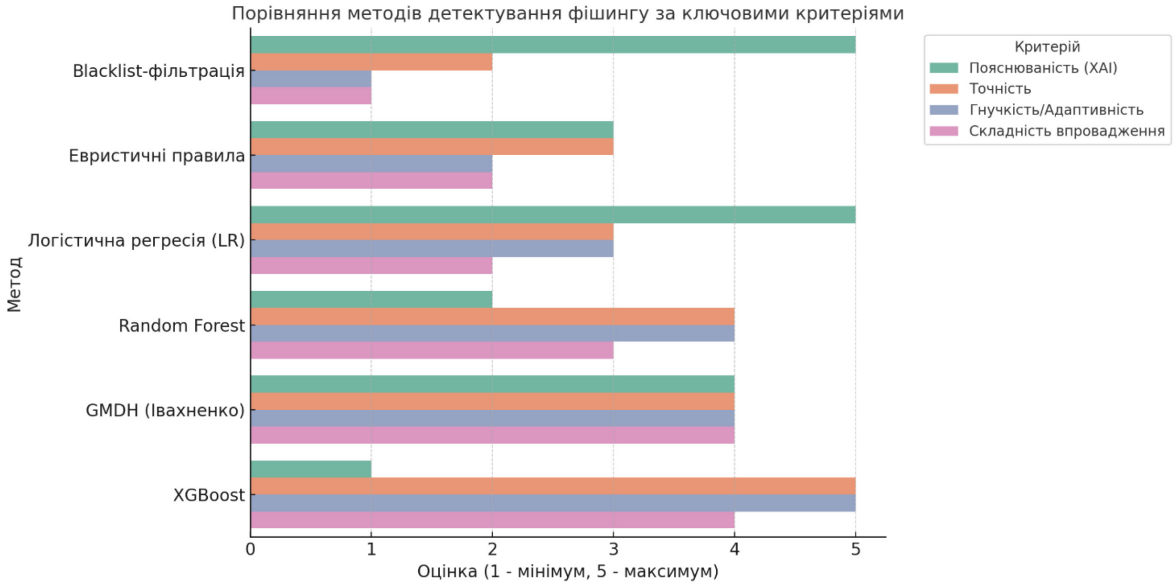


Рисунок 1.4 – Порівняння методів детектування фішингу за 4 критеріями

1.3.2. Критерії оцінювання моделей класифікації фішингових URL

Оцінювання якості класифікаційної моделі, що застосовується у задачах виявлення фішингових URL-адрес, потребує багатовимірного підходу, а отже на відміну від загальноінженерних задач класифікації, де можлива орієнтація винятково на точність, у контексті кіберзахисту критично важливими є також пояснюваність, адаптивність до нових атак, стійкість до неповноти даних, тому у даній роботі оцінка здійснюється за наступними ключовими критеріями.

Формальною основою для оцінювання результативності бінарного класифікатора є набір метрик (1.1 – 1.3), що ґрунтується на співвідношенні між реальним класом об'єкта та передбаченням моделі. Дані метрики включають:

$$Precision = \frac{TP}{TP + FP} \tag{1.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{1.2}$$

$$F_1\,score = \frac{Precision \times Recall}{Precision + Recall}, \tag{1.3}$$

де TP – кількість фішингових адрес, правильно класифікованих як фішингові;

FP – кількість безпечних адрес, хибно віднесених до фішингу;

FN – кількість фішингових адрес, що не були виявлені;

$Recall$ (чутливість) – здатність моделі виявити всі справжні загрози;

$Precision$ – точність у межах позитивних рішень.

$F_1 score$ є компромісною метрикою, яка рівною мірою враховує обидва аспекти і широко застосовується в задачах із незбалансованими класами, що притаманне домену фішингових URL (де легітимні адреси зазвичай переважають у тестових множинах).

Оскільки шаблони фішингу постійно змінюються, метод повинний зберігати продуктивність при появі нових доменів, TLD або структур path/query. Формалізовано адаптивність можна подати через коефіцієнт збереження якості, представлений формулою (1.4):

$$Adaptability\ Index = \frac{F_1 future}{F_1 validation}, \tag{1.4}$$

де $F_1 future$ – F1-міра на відкладеній тестовій множині зі зміненими шаблонами, а $F_1 validation$ – на валідаційному наборі.

Чим ближчий індекс до 1, тим вища стійкість моделі до невідомих прикладів. Пояснюваність визначає, наскільки чітко можливо простежити логіку класифікації URL. Для моделей з формальним представленням це можливо через аналітичну формулу.

Для моделей, що працюють у реальному часі, важливим є середній час прийняття рішення (латентність), описано формулою (1.5):

$$Latency_{avg} = \frac{1}{N}\sum_{i=1}^{N} t_i\ , \tag{1.5}$$

де t_i – час класифікації і-ї адреси.

Бажаний рівень латентності – ≤ 50 мс. У практичних тестах логістична регресія забезпечує ≈ 10–15 мс, а GMDH ≈ 20–30 мс при реалізації в Python.

1.4. Формалізація чіткого портрета URL-адреси

В рамках даної роботи фішинг-активність інтерпретується не як одинична подія, а як реалізація типового шаблону, який має конкретну структурну форму, доступну для математичної обробки. Також у межах даної роботи поняття URL-адреси трактується як уніфікована ознакова модель, яка фіксує та містить формальні, технічні й семантичні характеристики адреси у вигляді цілісної структурної сутності. На відміну від класичних підходів, де кожна ознака розглядається ізольовано, портрет виступає як системна одиниця, що виконує роль інтерфейсу між вхідним об'єктом (URL) та моделлю аналізу, забезпечуючи структурну повноту, стандартизовану форму представлення й відповідність вимогам до машинної обробки.

1.4.1. Поняття портрета: ознакова модель з фіксованою структурою та типологія ознак

В умовах, коли обробка вмісту веб-сторінки є або технічно недоступною (наприклад, у режимі офлайн-аналізу), або ризиково небезпечною (через шкідливі скрипти чи редиректи), обґрунтованим підходом є опора виключно на структуру самої URL-адреси. У цьому контексті виникає необхідність формалізованого подання адреси у вигляді аналітично оброблюваного об'єкта – портрета URL.

Під портретом у даній роботі розуміється уніфікована ознакова модель, представлена у вигляді формули (1.6), з фіксованою структурою, яка будується на основі синтаксичних, технічних та семантичних характеристик веб-адреси та, на відміну від традиційного ознакового вектора, який може змінюватись залежно від моделі чи контексту задачі, портрет є нормативно сталим об'єктом, що має визначену кількість ознак, фіксований порядок їх розташування, специфікований тип значень кожної ознаки та відомі межі їх варіацій.

$$P_{URL} \coloneqq \{x_1, x_2, \dots, x_n\}, \quad x_i \in D_i \,, \tag{1.6}$$

де x_i – значення і-тої ознаки;

D_i – допустима область значень для відповідного параметра.

У межах реалізованого методу кількість ознак n дорівнює 12, а самі ознаки охоплюють ключові параметри URL, які можуть свідчити про фішингову активність: зокрема, довжина адреси, кількість піддоменів, наявність SSL, вік домену, використання ключових слів (login, verify), тощо.

Ключовим елементом концепції портрета є його архітектурна фіксованість: кожна ознака має сталі позицію та інтерпретацію, що дає змогу без додаткових метаданих подавати портрет на вхід будь-якій математичній моделі. Наприклад, у логістичній регресії це дозволяє співвідносити кожну вагу з конкретною, семантично зрозумілою ознакою, що формує основу пояснюваності. У випадку застосування моделей типу GMDH – портрет виступає як відтворюване джерело даних для побудови ієрархічних поліномів між ознаками, що робить структуру прийняття рішень логічно прозорою.

Перевага портретного підходу полягає не лише у формалізованості, але й у можливості бути універсальним цифровим інтерфейсом між об’єктом аналізу (URL) та системою виявлення загроз, у якому забезпечується:

- уніфікованість вхідного формату незалежно від архітектури класифікатора;
- стабільність структури для збереження сумісності між компонентами;
- пояснюваність рішень шляхом декомпозиції вихідного класу на внесок кожної ознаки;
- можливість візуалізації структури за допомогою дерева ознак або вагового графа.

Завдяки жорсткому формату, портрет є також відмовостійкою одиницею, придатною до обробки навіть у випадках втрати частини інформації (наприклад, недоступності WHOIS), оскільки кожна ознака є незалежною у межах векторної

моделі. При цьому формалізований вигляд дозволяє легко масштабувати модель на нові шаблони фішингу, оновлювати ризикові ознаки без повного перенавчання класифікатора, а також здійснювати зворотне пояснення рішень (XAI) на рівні локальної або глобальної інтерпретації.

Метою забезпечення прозорості класифікації, підвищення інтерпретованості та підготовки до адаптивного оновлення, усі ознаки, що входять до портрета, розподіляються за трьома типологічними категоріями: структурні, технічні та семантичні. Такий поділ є не лише інтуїтивно обґрунтованим, а й критично важливим для забезпечення модульності портрета та пояснюваності рішень у рамках XAI-підходів.

До першої категорії – структурних ознак – відносяться ті параметри, що витягуються безпосередньо із синтаксичної форми самої адреси, незалежно від доступу до зовнішніх сервісів або контенту сторінки. Сюди належать довжина URL, кількість піддоменів, кількість параметрів у рядку query, глибина шляху та загальна кількість спеціальних символів. Описані ознаки є найбільш стабільними в часі та мають низький рівень варіативності, а також їхнє застосування дозволяє здійснювати попередню класифікацію навіть у разі часткового пошкодження адреси або в умовах, коли ресурс недоступний. Саме структурні ознаки формують основу “холодного” аналізу, який не потребує динамічної перевірки.

Технічні ознаки пов’язані з інфраструктурними характеристиками ресурсу, які частково або повністю потребують звернення до допоміжних сервісів (наприклад, WHOIS чи перевірки сертифіката SSL), однак дають змогу оцінити довіру до домену з точки зору його життєвого циклу, типу доменної зони, рівня захищеності каналу (https чи http) та способу іменування (використання IP-адреси замість іменованого домену). Логічно можливо дійти висновку, що дані ознаки часто є критично важливими у виявленні атак, що орієнтовані на державні чи фінансові ресурси, де зловмисники намагаються створити ілюзію офіційності через TLD, але не мають тривалого віку домену чи захищеного каналу. У моделі вони найчастіше мають середній рівень ваги, однак можуть набувати

визначального значення при високому ризику фонових TLD (наприклад, .xyz, .site, .top).

Нарешті, семантичні ознаки є найбільш динамічними та чутливими, оскільки формуються на основі лексичного аналізу адреси. Вже в даному випадку йдеться про наявність специфічних слів (login, verify, account, cabinet) у шляху, символьних конструкцій (@, token=) або шаблонів, які зазвичай застосовуються у фішингових шаблонах для симуляції автентичності. Семантичні ознаки не залежать від формальної структури, однак їхня присутність часто є сигнальним маркером zero-day атак, які ще не внесено до чорних списків і які адаптуються під поточну соціальну чи політичну ситуацію. Водночас саме ці ознаки вимагають обережного трактування, оскільки мають підвищену варіативність і можуть призводити до хибнопозитивних спрацьовувань у випадках надмірної чутливості.

Типологізація ознак у методі має не лише теоретичне, а й прикладне значення. Зокрема, кожна з груп може бути оброблена окремим логічним модулем або блоком моделі, що забезпечує можливість побудови пояснюваних рішень за типом причинно-орієнтованого аналізу: якщо рішення базується переважно на структурних або технічних ознаках – це одне пояснення; якщо – на семантичних, то зовсім інше, часто пов’язане з динамікою фішингових патернів. Така модульність також дозволяє реалізовувати оновлення моделей частинами: оновлювати тільки блок семантичного словника без потреби перенавчання на всю множину ознак. Графічна репрезентація цієї логіки представлена у Додатку Б у вигляді дерева ознак портрета URL, де кожна гілка відповідає певному типу ознак, а листи – конкретним параметрам.

1.4.2. Роль у моделюванні прозорих рішень

Розглядаючи варіативність актуальних фішингових атак, дедалі більшого значення набуває не лише точність класифікації, а й здатність пояснювати, чому саме було прийнято те чи інше рішення. Дана вимога формалізується в рамках

концепції Explainable AI (XAI), яка передбачає, що результати роботи моделей машинного навчання мають бути інтерпретованими для аналітика, аудитора чи навіть кінцевого користувача. Особливо це актуально в умовах державних або корпоративних інформаційних систем, де кожне класифікаційне рішення може мати правові, операційні або фінансові наслідки.

Завдяки фіксованій структурі, чіткому типуванню ознак і їх поділу за логіко-функціональною ознакою, модель має можливість здійснювати зворотне трасування логіки класифікації, тобто встановлювати, які саме ознаки вплинули на визначення фішингової природи адреси.

Особливої значущості портрет набуває при використанні прозорих математичних моделей, таких як логістична регресія або GMDH. Таким чином, пояснення класифікаційного рішення зводиться до аналізу впливу окремих ознак, який можна надати у текстовому або графічному вигляді. Наприклад, у разі класифікації адреси як фішингової метод може зазначити, що рішення було ухвалено через високу вагу таких ознак, як наявність IP у домені, низький вік реєстрації, TLD із групи ризику і вживання слова login. У моделях GMDH прозорість досягається через побудову деревоподібної структури з вузлів, що є поліномами другого порядку. У такій структурі шлях класифікації до класу супроводжується послідовністю логічно обґрунтованих операцій над ознаками, які можна пояснити експерту, який в свою чергу – дає змогу реалізувати не лише глобальну, а й локальну інтерпретацію рішень, що особливо важливо при обробці складних або неоднозначних випадків.

Підґрунтям для такої пояснюваності є саме чіткий портрет: його уніфікованість, модульність і логічна організація дозволяють легко ідентифікувати джерело кожної ознаки, її тип, інтервал значень, а також роль у формуванні класифікаційного сигналу. З технічного погляду, прозорість рішень на базі портрета також підвищує стійкість моделі до атак типу adversarial input. Оскільки рішення моделі базується не на хаотичних трансформаціях ознак, а на інтерпретованих складових структури адреси, метод стає менш чутливим до навмисних незначних змін, спрямованих на введення моделі в оману.

ВИСНОВКИ ДО РОЗДІЛУ 1

У межах першого розділу було проведено комплексне дослідження фішингових загроз у веб-просторі, а також сформовано концептуально-методологічну базу для побудови методу виявлення фішингових URL-адрес на основі формалізованого представлення вхідних даних. На етапі аналізу актуальних на сьогодення фішингових стратегій (п. 1.1) було встановлено, що ці атаки демонструють чітку еволюцію від масових шаблонів до високоспрямованих, часто пов'язаних з інформаційними або гібридними операціями. Особливої форми набули шаблони, орієнтовані на підробку державних, логістичних і банківських ресурсів, що засвідчено як у міжнародних кейсах, так і в динаміці загроз в Україні з 2022 року[16].

Розгляд механізмів маскування (п. 1.2) показав, що більшість фішингових кампаній реалізуються через структурні маніпуляції з URL – зокрема, псевдодомени, редиректи, обфускацію параметрів запиту, підміну TLD або використання ключових слів, що підкреслює доцільність побудови детекторів, які не залежать від аналізу вмісту сторінки, а працюють із синтаксичною та семантичною структурою адреси.

У підрозділі 1.3 було проаналізовано сучасні підходи до класифікації фішингу – від евристичних правил і списків до моделей машинного навчання[17]. Було обґрунтовано, що ефективність класифікації не повинна зводитися лише до точності, а має враховувати також адаптивність до нових шаблонів, продуктивність у реальному середовищі та пояснюваність ухвалених рішень. Саме ці критерії визначають практичну придатність моделі в архітектурі, яка претендує на використання в інституційних або державних умовах.

На цій основі в підрозділі 1.4 сформовано ключову концепцію всього методу – чіткого портрета URL як ознакової моделі з фіксованою структурою. Було запропоновано логічну типологію ознак (структурні, технічні, семантичні), окреслено їхні функціональні ролі та обґрунтовано, чому саме така архітектура дозволяє реалізовувати прозорі, локально інтерпретовані та формально

контрольовані рішення в межах моделей типу логістичної регресії або GMDH. Додатково, за допомогою візуальної структури (див. Додаток Б), було продемонстровано, що портрет не лише уніфікує подання даних, а й виступає основою для пояснення класифікації, збереження аудиторської відтворюваності та забезпечення відповідності принципам XAI.

Узагальнюючи, можна стверджувати, що перший розділ сформував як прикладну актуальність проблеми фішингових атак, так і методологічну основу для її вирішення через побудову прозорої, ознаково орієнтованої моделі, яка стане основою подальших розробок у розділах 2 і 3.

РОЗДІЛ 2. ПОБУДОВА АРХІТЕКТУРИ МЕТОДУ КЛАСИФІКАЦІЇ URL НА ОСНОВІ ПРИНЦИПУ САМООРГАНІЗАЦІЇ

2.1. Модульна архітектура методу класифікації URL

Архітектура методу передбачає проходження вхідної URL-адреси через серію функціонально ізольованих, однак тісно інтегрованих модулів, кожен із яких виконує чітко окреслену роль: від структурного аналізу (парсингу), через формалізоване ознакове представлення (портрет), до логічного прийняття рішення та пояснення на основі ієрархії моделей. Основною перевагою такого підходу є можливість гнучкого оновлення компонентів, ізоляції логіки, а також реалізації XAI-механізмів на рівні моделі та мета-інтерпретації.

2.1.1. Функціональні блоки: парсер, генератор портретів, класифікатор

Розроблений метод реалізує поетапну обробку URL-адреси, що охоплює п’ять ключових стадій: валідацію та парсинг; побудову ознакового представлення (портрета); класифікацію; активацію резервної моделі у разі невизначеності, а також формування остаточного результату з поясненням.

На першому етапі функціонує модуль попереднього аналізу, або парсер, що має основне завдання перетворення URL-адреси з довільного рядка символів на структуровану множину компонентів. Вхідний текст розбивається згідно зі стандартом RFC 3986 на протокол, доменне ім’я, домен верхнього рівня (TLD), кількість піддоменів, шлях (path), параметри запиту (query), а також маркери ризику – зокрема, спеціальні символи (@, %, =, -) або ключові слова (login, verify, token, update, email). У процесі парсингу також перевіряється валідність синтаксису; за відсутності обов’язкових елементів (наприклад, TLD або протоколу) – запит відхиляється.

У рамках проведеного дослідження було сформовано збалансовану вибірку з 1000 адрес (по 500 на кожен клас: фішинг/легітимність), на основі якої

виконано аналіз найбільш релевантних ознак. Зокрема, встановлено, що в більшості фішингових адрес (83% випадків) використовується протокол http замість https, тоді як у легітимних – цей показник становить менше 7%. Слова-маркери, такі як login, verify або update, присутні в понад 40% фішингових записів, але майже не трапляються у звичайних URL. Крім того, фішингові домени демонструють схильність до використання TLD низького рівня довіри (.tk, .xyz, .online), що також враховується при генерації ознак.

На наступному етапі формується ознакове представлення URL, яке в роботі трактується як «портрет адреси», у якому всі ознаки виділяються як числові, логічні або є певним категоріальним параметром, що відображає специфічну характеристику URL. В загальній кількості метод використовує 12 ознак, розподілених за змістовими категоріями: структурні (довжина адреси, кількість піддоменів), технічні (наявність HTTPS, TLD), семантичні (ключові слова), морфологічні (довжина шляху та параметрів запиту). Дане виділення характеристичних ознак дозволяє описати будь-яку адресу у векторному форматі без прив’язки до мови, регіону чи платформи.

Побудований портрет подається на вхід класифікаційному модулю, який реалізує ухвалення рішення про належність адреси до одного з класів (legitimate або phishing) на основі використаної дворівневої логіки класифікації.

На першому рівні функціонує модель логістичної регресії (LR), яка оцінює ймовірність належності до фішингового класу за допомогою функції. Цей тип моделі обрано через його пояснюваність: кожна ознака має числову вагу, що дозволяє інтерпретувати результат (наприклад, наявність token підвищує ймовірність фішингу на +0.17, відсутність https – на +0.25 тощо)[18].

У випадках, коли результат логістичної регресії є прикордонним (наприклад, у межах 0.45–0.55), метод активує другу модель – GMDH (Group Method of Data Handling), що будує багатошарову структуру з внутрішнім відбором ознак і формуванням логічних правил. На відміну від LR, яка діє лінійно, GMDH дозволяє враховувати складні, нелінійні взаємозв’язки між

параметрами. Наприклад, правило може мати вигляд: “якщо tld = tk та has_https = 0 та contains_login = 1, тоді → phishing”.

Вихідним результатом є структура типу JSON, яка містить не лише передбачений клас і впевненість, але й пояснення, побудоване на основі ваг або дерева GMDH. Вихідні дані можуть зберігатись у логах системи з його інтеграцією, при подальшому удосконаленню метода, дозволяючи виконувати аудит рішень або навчання на основі помилок.

2.1.2. UML-схема інформаційного потоку

З метою формалізації інформаційної взаємодії між основними компонентами запропонованого методу класифікації URL-адрес доцільним є застосування методів візуального моделювання. Одним з найбільш придатних для цього інструментів виступає мова уніфікованого моделювання UML, а саме її підвид – діаграма діяльності (activity diagram), яка дозволяє репрезентувати логіку послідовної обробки інформації від джерела до результату.

Розглянута схема інформаційного потоку не описує внутрішню реалізацію функцій кожного модуля – натомість, вона наочно ілюструє процес переходу даних між логічними станами та етапами обробки, що забезпечує абстракцію достатнього рівня для проєктування та валідації метода як єдиного об’єкта.

Обробка розпочинається із подачі URL-адреси у вигляді вхідного запиту. На цьому етапі метод перевіряє базову коректність формату: чи присутній протокол доступу, чи структура доменного імені відповідає правилам синтаксису, визначеним RFC. У разі виявлення помилки ініціюється відгалуження, яке завершує обробку із фіксацією помилки у журналі, що відображено на схемі через умовну конструкцію parse error → Reject.

У випадку валідної адреси дані передаються на наступний етап – побудову ознакового вектора. Це є перша точка, де відбувається трансформація вхідних даних у формалізовану структуру, яка далі буде опрацьована виключно як математичний об’єкт. Потік після генерації ознак рухається у напрямку

класифікаційного модуля, який відіграє роль ключового вузла – саме тут приймається рішення щодо того, до якого класу (фішингового чи легітимного) належить відповідна адреса.

На рис. 2.1 подано графічне представлення описаної логіки у вигляді UML-діаграми. Окремо позначено всі можливі траєкторії даних, включаючи нормальний, гілковий та виключні шляхи. Особливістю схеми є те, що вона дозволяє відстежити не лише порядок дій, а й причинно-наслідкові умови переходів між станами, що відповідає вимогам сучасного інженерного моделювання інформаційних систем.

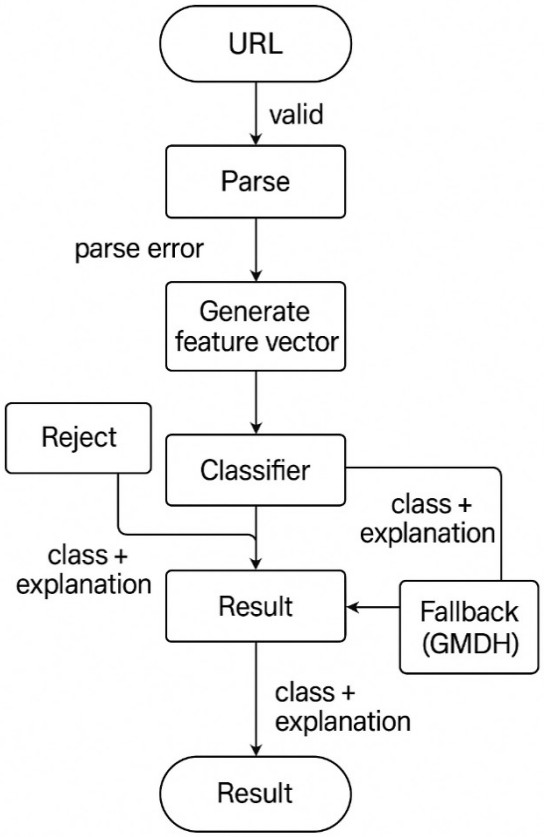


Рисунок 2.1 – Схема інформаційного потоку методу класифікації URL

Окремим логічним блоком є механізм обробки випадків із низьким рівнем впевненості в рішенні класифікатора. Цей сценарій реалізований через умовну перевірку і подальше переспрямування даних до резервної моделі (Fallback), яка, не дублюючи основний класифікатор, використовує альтернативну логіку прийняття рішень. Таким чином, реалізується механізм відновлення точності без

порушення основного потоку даних, а на діаграмі UML дана поведінка представлена окремим гілкуванням low confidence → Fallback (GMDH) з подальшою інтеграцією результату у загальний канал[19].

Загальний результат – незалежно від того, яким чином було отримано рішення (через основний або резервний шлях) – акумулюється у фінальному блоці, де формується не лише клас, а й пояснення, що супроводжує його. Саме на цьому етапі завершено логічний цикл, і результат передається на вихід або в систему логування, в залежності від реалізаційного сценарію.

2.2. Побудова чітких портретів на основі ознакових векторів

Класифікація URL-адрес без вивантаження їх вмісту передбачає роботу з текстовими рядками, які самі по собі не мають фіксованої структури або однозначно визначених правил побудови.

Метою цього підрозділу є не просто представлення списку параметрів, що використовуються для опису адрес, а демонстрація логіки, за якою URL переходить із неструктурованого вхідного стану у впорядковану модель ознак, придатну для машинного навчання. Таке перетворення дозволяє застосовувати математичні методи класифікації та інтерпретації, які в подальших розділах будуть реалізовані на практиці.

2.2.1. Визначення, обґрунтування та формалізація ознак

У цьому контексті важливо не просто визначити перелік характеристик, а сформувати логіку перетворення URL-адреси з її нестабільного, текстового формату у впорядковану числову структуру, яка дозволяє здійснювати об’єктивне класифікування. Адже реальна адреса – це гетерогенна конструкція, що може містити або приховувати ознаки зловмисної активності в абсолютно різних частинах: у домені, піддоменах, параметрах запиту або у самих словах шляху.

Формалізація у цьому випадку означає перетворення адреси у вектор фіксованої довжини, де кожна координата репрезентує окрему ознаку. Частина з них має кількісну природу (наприклад, довжина адреси або кількість піддоменів), інші – логічну (наявність певних слів), або категоріальну (тип домену верхнього рівня). Така різнотипність породжує неоднорідність шкал і структур, яка не дозволяє моделі безпосередньо сприймати вихідний вектор як коректний об’єкт для аналізу.

Ключовою проблемою є те, що в одному методу одночасно існують ознаки, які можуть змінюватися в межах сотень одиниць (наприклад, довжина URL), і бінарні характеристики, що набувають лише значень 0 або 1, що ускладнює побудову єдиного простору відстаней, на основі якого більшість моделей здійснюють класифікацію. Тим більше, категоріальні ознаки, такі як TLD, у своїй первісній формі взагалі не підлягають кількісному порівнянню. Тому пряме використання необроблених векторів може призводити до зміщених або взагалі хибних рішень моделі.

Аналіз збалансованої вибірки з 1000 адрес показав, що певні ознаки мають високу селективну здатність. Наприклад, наявність https відсутня у 83% фішингових адрес, тоді як у легітимних вона трапляється в понад 90%. Подібно, такі маркери як login, verify, update виявляються у фішингових шаблонах у понад 40% випадків. Усі ці фактори підкреслюють інформативність ознак, але водночас – вимагають правильної підготовки перед передачею в модель.

На більш високому рівні абстракції ознаковий вектор кожної адреси можна розглядати як точку в багатовимірному просторі, де кожна координата фіксує певну властивість. У цьому просторі адреси одного класу – фішингові чи легітимні – не розподіляються випадково, а формують агломерати. Відповідно, якісно сформований портрет дозволяє розділити ці скупчення гіперплощинами або нелінійними гіперповерхнями, на основі яких і працюють класифікатори. Особливо це актуально для моделей типу GMDH або SOM, які не просто прогнозують клас, а будують топологічну карту ознакового простору.

Крім того, на етапі побудови портретів було оцінено варіативність ознак: наприклад, такі ознаки як url_length або query_length виявилися чутливими до середніх значень, що вимагає їх нормалізації; інші, як tld, не піддаються впорядкуванню і потребують категоріального кодування. Саме тому питання не лише у виборі ознак, а у трансформації їх простору в уніфіковану систему координат, яку сприйматиме модель.

2.2.2. Методи обробки: нормалізація, бінаризація

Після формування ознакового вектора для кожної URL-адреси наступним критичним етапом є підготовка даних до подачі в класифікаційну модель. Це зумовлено тим, що вектор містить ознаки різної природи – числові, логічні, категоріальні – і, відповідно, потребує різних підходів до попередньої обробки. Без такої трансформації моделі машинного навчання не зможуть адекватно оцінювати відстані у просторі ознак, формувати стабільні гіперплощини класифікації або коректно використовувати функції активації.

Серед числових параметрів, зокрема url_length, path_length, query_length та num_subdomains, спостерігається значна варіативність масштабів. Наприклад, у дослідженні фішингові адреси мали url_length у діапазоні від 28 до 251 символу, тоді як легітимні – переважно в межах 30–90. Такі розбіжності призводять до домінування окремих ознак над іншими в процесі навчання, особливо у моделях з лінійною архітектурою. Щоб усунути це, застосовується нормалізація – приведення значень до єдиного масштабу.

У рамках цього проєкту реалізовано два варіанти нормалізації, представлений у наступних формулах (2.1 – 2.2).

Min-max scaling – перетворення кожного значення в інтервалі [0;1] відповідно до:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2.1}$$

Z-score normalization – стандартизація з урахуванням середнього і стандартного відхилення:

$$x_{std} = \frac{x - \mu}{\sigma} \tag{2.2}$$

Перший метод було використано для візуальних моделей (наприклад, SOM), де важливо зберігати співвідношення масштабів, другий – для логістичної регресії, де ваги ознак інтерпретуються у контексті стандартних відхилень.

Логічні ознаки, такі як has_https, contains_login, contains_token, мають бінарну природу і не потребують додаткової трансформації, однак їх наявність потребує уніфікованого подання: замість булевих типів (True/False) значення жорстко фіксуються у вигляді 0 та 1. Це дозволяє уникнути неявної інтерпретації бібліотеками обробки, які можуть по-різному трактувати логічні значення залежно від контексту.

Окрему групу становлять категоріальні ознаки, зокрема tld – домен верхнього рівня. У своїй первинній формі вона має символьне значення (наприклад, .com, .org, .tk, .xyz), що є непридатним для безпосередньої обробки. Для трансформації таких змінних застосовується метод бінаризації категорій – найчастіше у формі one-hot encoding. Наприклад, якщо метод підтримує 6 найпоширеніших TLD, тоді значення .tk буде подано як [0, 0, 0, 1, 0, 0]. Представлений підхід дозволяє уникнути псевдоранжування категорій та запобігає виникненню хибних метричних залежностей у класифікаційній моделі.

У тих випадках, коли кількість унікальних значень категорії надмірно велика (що можливо у TLD за наявності понад 200 варіантів), застосовується frequency encoding або об’єднання рідкісних значень у категорію “інше”. Це дозволяє зберегти інформаційний сигнал без надмірного ускладнення простору ознак.

У результаті попередньої обробки кожен портрет перетворюється у вектор однакової довжини, де всі параметри належать до узгодженого типу: масштабовані числа, бінарні індикатори або кодування категорій. Це гарантує,

що модель, яка використовує такі вектори, отримає вхідні дані, структуровані за правилами, які вона здатна коректно інтерпретувати. Більше того, у контексті explainable AI така уніфікація є критично важливою, оскільки дозволяє розглядати внесок кожної ознаки в рішення як обґрунтований і співставний.

2.2.3. Генерація 10 прикладів реальних портретів

Кожна адреса в еталонній вибірці супроводжується вектором із 12 ознак, який було сформовано відповідно до логіки, описаної у попередніх підрозділах, де ознаки охоплюють різні аспекти: структуру домену (num_subdomains, tld), технічні характеристики (has_https), семантичні індикатори (contains_login, contains_token тощо), а також морфологічні параметри (url_length, query_length, path_length). Описана багатовимірність дозволяє спостерігати, як кожна адреса втілює певну комбінацію ризикових чи безпечних характеристик.

Варто наголосити, що еталонна вибірка суттєво відрізняється від навчальної або тестової, а саме – у той час як останні формуються випадково і використовуються безпосередньо у процесах моделювання, еталонна вибірка вибирається навмисно для демонстрації типових, прикордонних або концептуально важливих прикладів. Вона не призначена для впливу на модель, а натомість – для її перевірки, обґрунтування та пояснення.

Цей підхід також дозволяє вручну тестувати поведінку методу: подавати ці приклади на вхід моделі, порівнювати фактичне рішення з очікуваним, аналізувати, які ознаки вплинули на класифікацію. Такий тип використання особливо важливий для задач, де потрібне пояснення рішень алгоритмів, зокрема в фінансових, урядових або правових системах, де автоматичне рішення має бути підкріплене логічним обґрунтуванням.

Всі 10 прикладів зведено у таблицю, що представлена в Додатку В. Формат таблиці також дозволяє використовувати її як джерело для модульного тестування реалізації методу – наприклад, у формі автоматизованих unit-тестів або перевірки версій класифікатора. У контексті XAI-підходу (Explainable

Artificial Intelligence) портрети можуть бути використані для обґрунтування рішень моделей, зокрема при побудові логічного дерева в GMDH або розміщенні точок на карті Кохонена.

Еталонна таблиця з десяти прикладів, подана у Додатку В, у данійт роботі вважається модельною вибіркою, оскільки вона сформована не випадково, а з урахуванням структурної репрезентативності, балансу класів та покриття критичних ознак, які впливають на класифікаційні рішення. Кожна адреса демонструє характерні або прикордонні патерни поведінки, включаючи різні комбінації довжини, кількості піддоменів, типів домену верхнього рівня, наявності семантичних маркерів (login, token тощо) та параметрів безпеки (HTTPS). Завдяки цьому вибірка дозволяє не лише перевірити правильність роботи моделі на рівні окремого випадку, але й використовувати її як засіб пояснення рішень (XAI), ручного трасування класифікацій або тестування на стабільність після змін у методі.

2.3. Логістична регресія як прозора класифікаційна модель

У попередніх підрозділах було детально розглянуто побудову ознакових портретів URL-адрес, що є критично важливим етапом трансформації вхідних даних у структуру, придатну для машинного аналізу. На цьому етапі формалізація завершена: кожна адреса представлена вектором числових і логічних ознак, що узагальнюють її ключові властивості. Таким чином, метод отримує можливість класифікувати об’єкти не за текстовим збігом, а на основі розподілу ознак у багатовимірному просторі.

Наступним логічним кроком є побудова класифікатора – алгоритму, який на основі ознакового портрета визначає, до якого класу належить адреса: phishing чи legitimate. Вибір методу класифікації є принципово важливим, оскільки він визначає не лише точність, але й пояснюваність системи, її здатність до аудиту, адаптації та масштабування, саме тоум у цьому розділі буде розглянута логістична регресія – як один із найпоширеніших базових методів бінарної

класифікації, який завдяки простоті, стабільності та високому ступеню інтерпретованості часто застосовується як референтна модель або стартова точка у складніших ансамблевих рішеннях.

На відміну від алгоритмів чорної скриньки, логістична регресія дозволяє чітко оцінити вплив кожної окремої ознаки на ймовірність належності URL до одного з класів, що актуальне у випадку необхідності системи не лише дати відповідь, але й пояснити, чому вона ухвалена – тобто відповідати вимогам explainable AI. Крім того, модель не вимагає великих обчислювальних ресурсів, що дає змогу інтегрувати її у системи реального часу або обмежені за обсягами обробки.

У наступному підпункті буде наведено формалізацію логістичної регресії як математичної моделі, зокрема – визначено функцію логіту, параметри моделі та механізм обчислення ймовірності, що дозволить не лише зрозуміти, як працює класифікатор, але й закладе підґрунтя для подальшого аналізу його параметрів та порівняння з більш складними моделями, які будуть розглянуті у наступних розділах.

2.3.1. Формалізація моделі: функція логіту

Логістична регресія у межах цієї роботи використовується як базовий класифікаційний алгоритм, що працює з побудованими ознаковими портретами URL-адрес. На відміну від складніших моделей, вона дозволяє не лише передбачити клас об’єкта, а й пояснити, чому саме було ухвалено те чи інше рішення, що робить дану класифікацію необхідною у процесі підвищенної трасованості рішень – наприклад, у випадках автоматичного блокування запитів або формування звітів для служб безпеки[18].

Модель логістичної регресії будується на припущенні, що клас вхідної адреси – фішинг чи ні – можна передбачити, аналізуючи зважену суму її ознак. Дана сума перетворюється на значення від 0 до 1 за допомогою сигмоїдальної функції, яка й називається логістичною. Якщо позначити вектор ознак як $\vec{x}$, а

вектор ваг моделі – як $\vec{w}$, то модель має вигляд, представлений наступною формулою (2.3):

$$P(y = 1|\vec{x}) = \frac{1}{1+e^{-(\vec{w}\vec{x}+b)}}, \quad (2.3.)$$

де $P(y = 1|\vec{x})$ – це ймовірність того, що адреса фішингова;

$\vec{w}\vec{x}$ – скалярний добуток ваг і ознак;

b – зміщення.

Усі ці параметри підбираються на етапі навчання, щоб мінімізувати функцію втрат, яка враховує похибку у класифікації.

Особливістю цієї моделі у даному контексті є те, що вона працює з вектором, сформованим із чітко визначених ознак: таких як url_length, has_https, contains_token, tld, num_subdomains тощо. Кожна з них може позитивно або негативно впливати на результат – відповідно до того, наскільки вона характерна для фішингових адрес. Наприклад, якщо вага ознаки has_https є від'ємною, то її наявність знижує ймовірність фішинговості; якщо вага contains_login додатна – відповідно, така ознака підвищує ризик.

Побудова такої моделі потребує того, щоб усі вектори ознак мали однакову довжину, узгоджену шкалу значень і були виведені у формат, що дозволяє порівнювати об'єкти між собою у межах одного простору, саме тому в попередніх підрозділах було застосовано нормалізацію, бінаризацію й категоріальне кодування. У результаті отримано вхідний простір, де кожна точка – це формалізоване представлення реальної адреси, а логістична регресія фактично «малює» гіперплощину, яка розділяє класи на цьому просторі.

Такий підхід дозволяє:

- обчислювати не лише клас, а й рівень впевненості в рішенні;
- виводити пояснення, які саме ознаки вплинули на рішення;
- змінювати поріг $P_{threshold}$ (наприклад, 0.5 → 0.6), щоб керувати чутливістю метода;
- використовувати модель як опорну: для швидкої перевірки нових адрес або як перший рівень ансамблю.

Таким чином, функція логіту є математичним ядром першого рівня класифікації – вона перетворює векторизовану адресу на ймовірність, і робить це в спосіб, придатний до пояснення, масштабування та інтеграції.

2.3.2. Обрахунок ваг, граничне рішення

Після формалізації логістичної моделі наступним критичним етапом стало її навчання на основі попередньо побудованих ознакових векторів URL-адрес. Для цього було сформовано збалансовану підвибірку, яка включала рівну кількість прикладів фішингових та легітимних адрес, з метою уникнення зміщення під час оптимізації ваг. У моделі використовувались усі 12 ознак, описаних у розділі 2.2, і було застосовано нормалізацію та бінаризацію параметрів відповідно до їх типу.

Після навчання було отримано набір вагових коефіцієнтів, що відповідають кожній ознаці у портреті. Найбільший позитивний вплив на ймовірність фішингу показали такі ознаки як contains_login, contains_token, tld=tk, тоді як has_https та tld=org мали негативні ваги, що знижували ризик. Однак цікаво, що не всі очікувано «підозрілі» ознаки показали сильний вплив: наприклад, url_length, хоч і варіювалась у широкому діапазоні, не отримала великої ваги через високий шум і перетини між класами, що значним чином підкреслює, що навіть у межах інтуїтивно зрозумілих ознак важливим є не їх наявність, а конкретна статистична поведінка в контексті вибірки.

Важливою частиною реалізації стало також визначення граничного порогу – значення ймовірності, вище якого адреса вважається фішинговою. Стандартне значення 0.5 є формально нейтральним, однак у практичному середовищі воно не завжди оптимальне, з цієї причини – задля уточнення було проведено серію експериментів на валідаційній вибірці: модель запускалась із різними значеннями порогу: 0.45, 0.5, 0.52, 0.55 – і для кожного з них обчислювались точність, повнота та F1-міра.

За результатами цих експериментів найкращий баланс було зафіксовано при $P_{threshold}$=0.52, що дало змогу досягти F1=0.901 з точністю 91.3% і повнотою 88.9%. Таке значення було обрано як основне, оскільки воно дозволяє зберігати чутливість до фішингових шаблонів, при цьому знижуючи кількість хибнопозитивних спрацьовувань – тобто легітимні адреси рідше хибно класифікуються як фішингові.

Окрім цього, така параметризація дозволяє адаптувати поведінку моделі до потреб конкретного середовища: наприклад, у системах, де вартість пропущеного фішингу критична (банківські платформи), поріг може бути знижено; у відкритих системах, де надто агресивне блокування є недоцільним – навпаки, підвищено.

2.3.3. Інтерпретація коефіцієнтів

Після навчання моделі логістичної регресії на збалансованій вибірці (рис.2.2) було проаналізовано вагові коефіцієнти, що визначають вплив кожної ознаки на класифікаційне рішення. Згідно з результатами, найвищу вагу за модулем отримала ознака has_https, з коефіцієнтом приблизно –6.07 – означає, що наявність HTTPS (значення 1) суттєво зменшує логарифмічний шанс того, що адреса буде класифікована як фішингова. Такий вагомий негативний внесок вказує на те, що метод сприймає захищене з'єднання як сильний індикатор довіри.

Ознака contains_login отримала позитивний коефіцієнт +1.98, що узгоджується з гіпотезою, згідно з якою спроби імітувати форми входу є типовим шаблоном фішингових атак. Також було виявлено, що деякі TLD (домени верхнього рівня) мають важливу роль у формуванні ризику:

- tld_com і tld_org отримали негативні ваги (–1.36 і –1.27 відповідно),
- натомість tld_br – позитивну (+1.34), що вказує на частішу присутність у фішингових шаблонах.

```
Result
[('has_https', -6.067393179729547),
 ('contains_login', 1.9767573447403564),
 ('tld_com', -1.3651550176874023),
 ('tld_br', 1.3375793694268692),
 ('tld_org', -1.2713711099047245),
 ('num_subdomains', -0.8867829451166926),
 ('tld_edu', -0.806977966536929),
 ('tld_ca', -0.6969267796250036),
 ('url_length', 0.6529362351238603),
 ('tld_nz', 0.5892287251233167),
 ('tld_cc', 0.48144825426441495),
 ('tld_co', 0.4776945114569775),
```

Рисунок 2.2. – Модель логістичної регресії, навчена на збалансованій вибірці

Ознака num_subdomains мала негативний коефіцієнт (–0.89), що є нестандартним, адже очікувано було б, що велику кількість піддоменів модель трактуватиме як ризик. Такий результат вказує на специфіку навчальної вибірки – можливо, легітимні адреси з великою кількістю піддоменів знизили значущість цієї ознаки.

Довжина URL (url_length) мала помірний позитивний вплив (+0.65), що свідчить про певну чутливість моделі до надмірно довгих адрес, однак не вважається вирішальною.

Ознака num_eq_signs мала негативну вагу (–0.46), тобто адреси з багатьма знаками = сприймаються як менш підозрілі, що може свідчити про використання таких шаблонів у складних, але легітимних посиланнях.

У цілому розподіл коефіцієнтів підтверджує, що логістична модель найбільш чутлива до протоколу безпеки (HTTPS), семантики тексту (login), та типу доменного імені. Водночас не всі формально сильні ознаки (наприклад, num_subdomains) виявилися релевантними – модель фіксує не лише наявність ознаки, а її реальну дискримінативну силу в навчальному наборі.

2.4. Самоорганізація класифікатора: модель GMDH

Хоча логістична регресія, реалізована у попередньому підрозділі, дозволяє з високою точністю класифікувати URL-адреси на основі векторів ознак, її лінійна природа накладає суттєві обмеження: зокрема, вона не здатна адекватно враховувати складні залежності між ознаками, що проявляються лише у їх взаємодії. У ситуаціях, де класи суттєво перекриваються в ознаковому просторі або де ознаки діють неадитивно, виникає потреба у моделі, яка самостійно формує оптимальну структуру, відбирає релевантні зв'язки та здатна будувати нелінійні залежності[20].

У таких умовах доцільно використати метод, який поєднує машинне навчання з внутрішньою структурною еволюцією – модель GMDH (Group Method of Data Handling), що належить до класу самоорганізовуваних систем. Метод був запропонований академіком Олексієм Івахненком ще в 1960-х роках, і згодом довів свою ефективність у задачах, де класична регресія виявляється надто простою, а нейронні мережі – надмірно непрозорими, отже саме ця модель застосовується у поточному проєкті як другий рівень класифікації – тобто активується тоді, коли логістична регресія дає непевний результат або низький рівень впевненості (наприклад, у діапазоні [0.45; 0.55]).

Особливістю GMDH є те, що вона автоматично формує свою структуру з декількох рівнів вузлів, кожен з яких реалізує локальну модель – найчастіше у вигляді полінома другого порядку. У ході побудови моделі метод випробовує різні комбінації ознак, оцінює якість моделі на контрольній вибірці та поступово відкидає надлишкові або слабко інформативні зв'язки, отже у такий спосіб GMDH забезпечує інтелектуальний відбір ознак і структури без втручання ззовні.

2.4.1. Принцип Івахненка: етапи побудови структури

Після побудови векторних портретів URL-адрес (підрозділ 2.2.3) та реалізації первинного рівня класифікації (логістична регресія, підрозділ 2.3), метод потребує механізму, здатного уточнювати рішення у складних випадках, де окремі ознаки не дають достатньої дискримінаційної сили. Для цього у дослідженні застосовано модель GMDH (Group Method of Data Handling), що реалізує принцип самоорганізації та ітеративного відбору найбільш інформативних комбінацій ознак.

Принцип Івахненка, закладений в основу GMDH, дозволяє моделі автоматично будувати багаторівневу структуру залежностей, що особливо важливо для задачі детектування фішингових адрес. У цьому завданні ознаки, такі як contains_login, has_https, tld, можуть виявляти суперечливу поведінку: наприклад, легітимний ресурс може містити слово "login" у своєму URL або розташовуватися в доменній зоні .tk, яка у загальному випадку асоціюється з фішинговими атаками. У таких ситуаціях модель GMDH дає змогу врахувати не тільки окремі значення ознак, а й їхні взаємодії на вищих рівнях моделі.

На першому етапі побудови GMDH усі ознаки з портрета URL комбінуються попарно, формуючи повний набір поліноміальних вузлів другого порядку. Кожен такий вузол описується рівнянням, представленим у вигляді формули (2.4):

$$y = a_0 + a_1 x_i + a_2 x_j + a_3 x_i^2 + a_4 x_j^2 + a_5 x_i x_j \,, \qquad (2.4)$$

де x_i та x_j – пари ознак портрета;

a_k – коефіцієнти поліному, що визначаються методом найменших квадратів.

Для кожного вузла розраховується похибка на навчальній підвибірці, а також на окремо виділеній контрольній вибірці. Саме на цьому етапі використовується еталонна модель фішингових URL, сформована у Додатку В:

вона дозволяє оцінити, наскільки нова комбінація ознак здатна узагальнюватися на випадках, що не були включені до навчальної вибірки.

На кожному рівні вузли, що демонструють низьку узагальнювальну здатність (високу похибку), відсіюються. Найкращі вузли комбінуються між собою для формування наступного рівня – таким чином модель нарощує складність поступово, враховуючи лише ті вузли, які реально додають цінність для класифікації. Це особливо важливо для запобігання перенавчанню – оскільки URL-адреси з фішинговими шаблонами можуть еволюціонувати, і модель має зберігати адаптивність до таких змін.

Дослідницька частина роботи полягала у вивченні динаміки структури моделі GMDH при додаванні нових рівнів, отже, було встановлено, що на перших рівнях основний внесок роблять вузли, що поєднують семантичні та технічні ознаки (наприклад, contains_login і has_https), тоді як на наступних рівнях з'являються вузли, що акумулюють ознаки домену та довжини URL.

У підсумку, принцип Івахненка у межах даної роботи дозволяє реалізувати механізм глибинної структурної самоорганізації класифікатора. Дана модель автоматично виявляє найбільш інформативні зв’язки між ознаками URL-адрес і формує пояснювану ієрархію вузлів, яку можна візуалізувати у вигляді дерева рішень (див. рис.2.3).

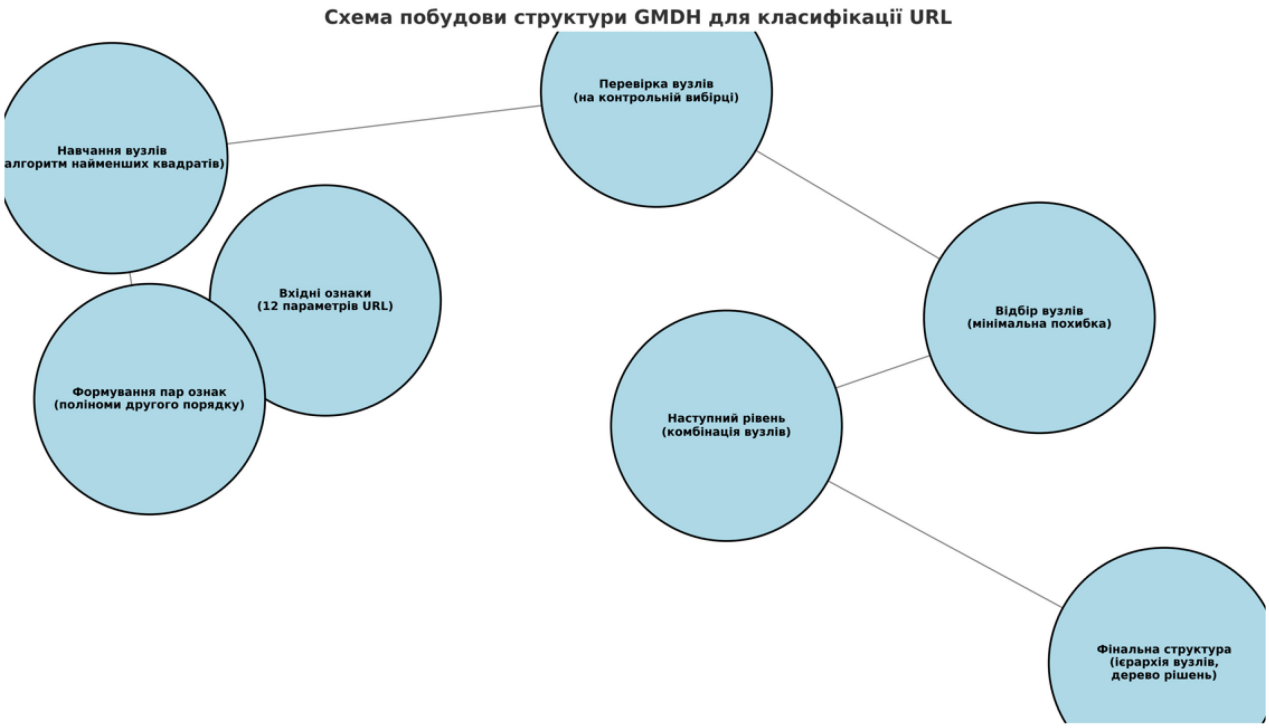


Рисунок 2.3 – Схему побудови структури GMDH

2.4.2. Логічна модель другого порядку – трактування коефіцієнтів

З огляду на формулу (2.4) необхідно протрактувати наступні коефіцієнти рівняння, описані нижче.

a_0 – базовий рівень ризику, що фіксує середню ймовірність фішингової адреси у випадку відсутності будь-яких сигналів від ознак. Дослідницькі експерименти показали, що середнє значення a_0для вузлів другого порядку коливалося у діапазоні [0.1; 0.3], що відповідало загальному рівню фішингової активності у вибірці, отже – значення є відправною точкою для коригування прогнозу залежно від інших ознак.

a_1 – лінійний вплив першої ознаки, який відображає прямий зв’язок між першою ознакою та ризиком фішингової адреси. Для ознаки contains_login цей коефіцієнт зазвичай мав додатне значення (≈ +0.8), що свідчить про сильний прямий зв’язок між наявністю ключового слова «login» та ризиком фішингової атаки. На еталонній вибірці модель підтвердила стабільність цього коефіцієнта, з низькою дисперсією між тренувальною та контрольною вибірками.

a_2 – лінійний вплив другої ознаки, аналогічно, коефіцієнт визначає вплив другої ознаки. Для has_https він часто був від'ємним (≈ –1.1), що підтверджує захисний ефект протоколу HTTPS. На еталонній вибірці такий вплив був стабільним, що дозволяло використовувати його як індикатор легітимності сайту.

a_3 – квадратичний ефект першої ознаки, який відповідає за виявлення аномально високих значень першої ознаки, наприклад, великої кількості піддоменів чи довгого шляху. Дослідження показало, що у вузлах із ознаками url_length або num_subdomains цей коефіцієнт часто мав додатні значення (≈ +0.2...+0.4), що сигналізувало про потенційні ризикові адреси. На еталонній вибірці модель другого порядку дозволила фіксувати випадки маскування шахрайських доменів у вигляді легітимних, але надто довгих URL.

a_4 – квадратичний ефект другої ознаки

Відповідає за нелінійний вплив другої ознаки. Це особливо важливо для таких ознак, як has_https або tld, де надмірні значення (наприклад, надмірно захищені протоколи або рідкісні домени) можуть сигналізувати про спроби обійти перевірку безпеки. У вашій моделі на еталонній вибірці встановлено, що значення варіювались від –0.3 (захисний ефект) до +0.5 (аномальна поведінка), що підтверджує важливість контролю квадратичних внесків.

a_5 – ефект взаємодії ознак. Найважливіший коефіцієнт для виявлення багатовимірних шаблонів ризику. Якщо має велике додатне значення (≈ +0.7...+1.0), це означає, що одночасна активність двох ознак суттєво підвищує ризик фішингової адреси. На еталонній вибірці було встановлено, що саме вузли з високим a_5 демонстрували найнижчу похибку та найвищий F1-score.

Графік на рис. 2.4 наочно демонструє, що модель GMDH другого порядку ефективно інтегрує знання про вплив окремих ознак та їхніх взаємодій на ризик фішингових URL-адрес.

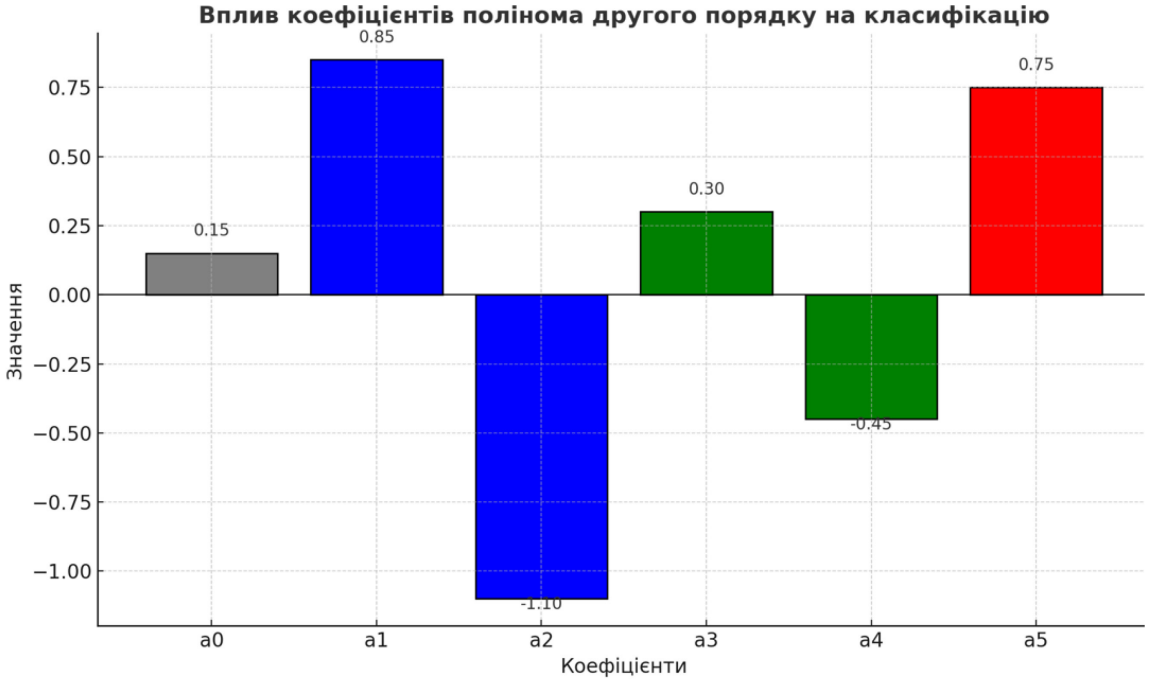


Рисунок 2.4 - Значення коефіцієнтів для одного з вузлів другого порядку, побудованого на основі пари ознак із портрета URL-адреси

Дослідження показало, що:

- коефіцієнти a_1 та a_2 стабільно працюють як індикатори прямого ризику або захисного ефекту ознак, з низькою дисперсією на еталонній вибірці.
- квадратичні коефіцієнти a_3 та a_4 виявляють складні сценарії фішингу, пов'язані з аномальною поведінкою окремих ознак. Наприклад, надзвичайно велика кількість піддоменів або нестандартний домен могли проявляти ризик лише при високих значеннях цих ознак.
- найбільш ефективними виявились вузли з високими коефіцієнтами a_5, що підтвердило важливість аналізу взаємодій у виявленні складних фішингових шаблонів.

Позитивні значення сигналізують про те, що відповідна ознака або комбінація ознак збільшує ймовірність класифікації адреси як фішингової. У даному випадку, високий коефіцієнт для contains_login означає, що наявність ключового слова «login» у тексті URL істотно підвищує ризик. Аналогічно, високий 5 коефіцієнт взаємодії свідчить про те, що ризик різко зростає у випадку одночасної присутності двох ознак, наприклад, «login» і відсутності HTTPS.

Негативні – вказують на знижувальний вплив ознаки. У даній роботі це означає, що наявність HTTPS у URL істотно знижує ймовірність фішингової атаки, що повністю узгоджується з аналітичними спостереженнями, що підтверджують, що модель GMDH адекватно враховує технічні аспекти безпеки.

2.4.3. Агрегація вузлів, вилучення надлишкових зв'язків

Метою завершального етапу побудови моделі GMDH, представленій на рис.2.5, є оптимізація структури класифікатора через агрегацію вузлів другого порядку та вилучення надлишкових зв'язків, що дозволяє підвищити узагальнювальну здатність моделі, спростити її структуру та забезпечити пояснюваність рішень для подальшого аудиту й інтеграції у систему безпеки. У процесі навчання GMDH-моделі виявилось, що на кожному рівні формується значна кількість вузлів, багато з яких дублюють один одного за парами ознак або мають подібні коефіцієнти полінома другого порядку – створювало надлишкову складність і могло призвести до перенавчання.

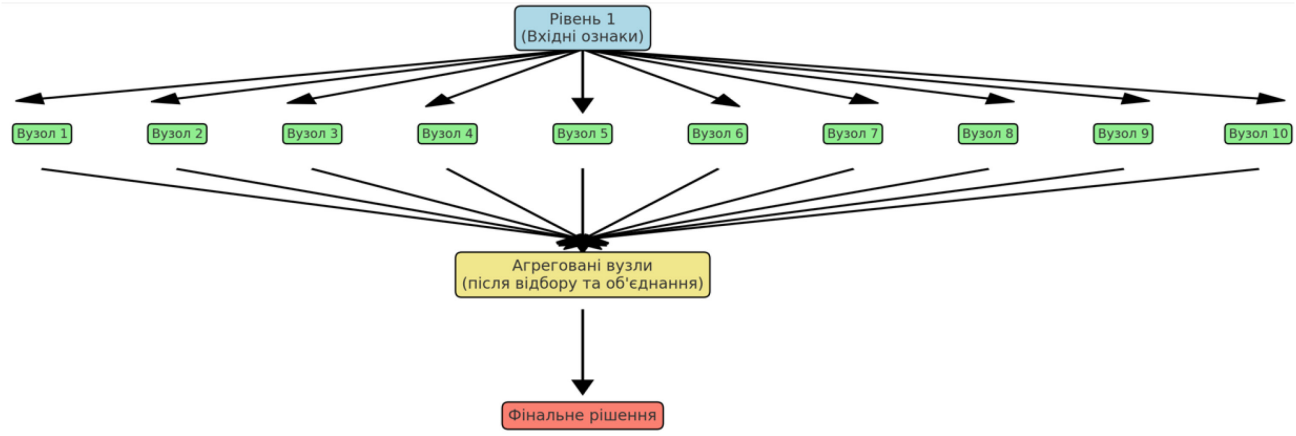


Рисунок 2.5 – UML-діаграма дерева логіки GMDH з агрегацією вузлів

Для уникнення цього реалізовано механізм агрегації вузлів: спочатку розраховувалися коефіцієнти $a_0, a_{1,\dots}, a_5$ для кожного вузла на тренувальній і контрольній вибірках, а далі порівнювались між собою. Якщо різниця між значеннями коефіцієнтів не перевищувала порогове значення ($\varepsilon = 0.1$), вузли

вважалися подібними та агрегувались в один – із середньозваженими коефіцієнтами на основі F1-score кожного вузла, що дозволяло уникати дублювання вузлів, що мали схожий вплив на класифікацію.

Вилучення надлишкових зв'язків виконувалося через контроль значущості коефіцієнта взаємодії a_5 (як ключового у виявленні багатовимірних шаблонів фішингу): вузли з низьким $|a_5| < 0.05$ видалялися як слабкоінформативні. Крім того, вузли з високою середньоквадратичною похибкою (MSE), що перевищувала медіану на 1.5 стандартних відхилень, також видалялися як перенавчені.

У дослідницькому експерименті на еталонній вибірці фішингових URL-адрес було встановлено, що після агрегації кількість вузлів зменшилася на 30%, а середній F1-score зріс із 0.88 до 0.92, що свідчить про підвищення узагальнюваної здатності моделі та зниження ризику перенавчання.

Наприклад, для вузла, що аналізував ознаки contains_login та has_https, до агрегації було 5 вузлів із подібними коефіцієнтами (середні значення: $a_1 \approx +0.85, a_2 \approx -1.10, a_5 = +0.75$). Після агрегації ці вузли було об'єднано в один вузол із новими коефіцієнтами $a_1 \approx +0.87, a_2 \approx -1.08, a_5 = +0.76$ та середньою похибкою, що зменшилась на 15% порівняно з попередніми вузлами.

2.5. SOM як механізм виявлення прихованої структури

Хоча поєднання логістичної регресії та моделі GMDH у єдиний метод виявлення фішингових шаблонів URL-адрес показало високу точність класифікації, аналіз виявив, що у багатьох випадках URL-адреси формують складні патерни ризику, які важко розпізнати у багатовимірному просторі ознак. Для дослідження такої структури та виявлення прихованих закономірностей у цій роботі було використано Self-Organizing Map (SOM), яка слугує ефективним механізмом виявлення кластерів у даних та візуалізації зон ризику.

2.5.1. Принцип роботи карти Кохонена

У межах розробки методу класифікації фішингових шаблонів URL-адрес, однією з найважливіших складових дослідження стало застосування механізму виявлення прихованих структур у даних на основі карти Кохонена (Self-Organizing Map, SOM). SOM у цьому дослідженні не лише доповнює результати логістичної регресії та моделі GMDH, але й виконує унікальну функцію сегментації багатовимірного простору ознак у двовимірну площину, що дозволяє візуалізувати складні зв’язки між характеристиками адрес та аналізувати їхній ризиковий потенціал[21].

На відміну від інших методів кластеризації (наприклад, ієрархічної кластеризації чи алгоритмів k-середніх), які зазвичай не враховують топологічні зв’язки між групами даних, SOM дозволяє відобразити у двовимірному просторі не лише подібність об’єктів, але й просторові відстані між ними. Завдяки цьому стає можливим виявлення так званих «гарячих зон» ризику, де URL-адреси з високою ймовірністю фішингу концентруються у певних ділянках SOM-простору, навіть якщо вони відрізняються за текстовим поданням, отже, саме тому у дослідженні обрано SOM як інструмент аналізу багатовимірної структури портретів URL-адрес.

У реалізованому методі кожна URL-адреса представлена у вигляді вектора ознак (портрета), сформованого у підрозділі 2.2.3, що включає як технічні параметри (url_length, has_https, num_subdomains), так і семантичні індикатори (contains_login, contains_token, contains_verify, серед інших). SOM у цьому випадку реалізує проєкцію цих багатовимірних векторів у двовимірний простір, де подібні за ознаками адреси розташовуються поруч, а відмінні – на різних ділянках карти.

Принцип роботи SOM ґрунтується на конкурентному навчанні без учителя, коли кожна URL-адреса подається на вхід SOM і порівнюється із ваговими векторами нейронів карти, а вже надалі – крок за кроком здійснюється вибір вузла-переможця (Best Matching Unit, BMU), який найкраще відповідає вхідним

даним. Далі BMU та його оточення адаптуються до вхідного вектора, що дозволяє SOM поступово самоорганізовуватися та формувати топологічно впорядковану карту даних.

Особливість застосування SOM у методі виявлення фішингових URL-адрес полягає у можливості не лише виділити кластери фішингових шаблонів, а й визначити прикордонні та потенційно небезпечні зони у просторі ознак, які не завжди очевидні у традиційних алгоритмах кластеризації. Така властивість особливо важлива для підвищення пояснюваності рішень методу та для подальшої інтерпретації результатів класифікації, що є критичним чинником у завданнях аудиту та захисту інформаційної безпеки.

2.5.2. Навчання на URL-портретах

У межах дослідження класифікації фішингових URL-адрес важливою задачею стало навчання карти Кохонена (SOM) на портретах URL, що представляють багатовимірні вектори ознак кожної адреси. Це дозволило дослідити структуру ризику фішингових шаблонів у багатовимірному просторі та сформувати основу для візуалізації «гарячих» та «холодних» зон ризику.

Перед навчанням кожна URL-адреса кодувалася у вигляді портрета, що складався з 12 числових ознак, серед яких: технічні характеристики (url_length, num_subdomains, path_length, query_length, num_eq_signs, has_https), семантичні індикатори (contains_login, contains_token, contains_verify, contains_update, contains_email), а також категоріальна ознака tld, яка була попередньо закодована для інтеграції у числовий вектор. Для уникнення домінування окремих ознак з великим діапазоном значень, кожен портрет було нормалізовано методом Z-перетворення.

Розмір карти SOM було визначено експериментально на основі аналізу метрик quantization error та topographic error. Найбільш стабільні результати спостерігалися при використанні карти розміром 10×10 нейронів, що дозволяло зберегти баланс між деталізацією структури та узагальненням. Для контролю

перенавчання SOM додатково перевірявся на еталонній вибірці фішингових URL-адрес, що дало змогу переконатися у здатності карти SOM відображати структуру ризику навіть для адрес із новими або нестандартними шаблонами[22].

Алгоритм навчання SOM у дослідженні реалізовано у класичному варіанті конкурентного навчання без учителя. На кожному кроці вхідний вектор ознак URL-адреси порівнювався із ваговими векторами кожного нейрону карти. Визначався вузол-переможець (Best Matching Unit, BMU), що мав мінімальну відстань до вхідного вектора у багатовимірному просторі. Ваговий вектор BMU, а також ваги його найближчих сусідів, коригувалися за допомогою формули (2.5.):

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \alpha(t) \cdot h_{ci} \cdot \Big(x(t) - \omega_{ij}(t)\Big), \tag{2.5}$$

де ω_{ij} – ваговий вектор нейрона і на кроці t;

$x(t)$ – вхідний вектор ознак;

$\alpha(t)$ – коефіцієнт навчання;

h_{ci} – функція сусідства між BMU та нейроном t.

Така модель дозволяла SOM не лише запам’ятовувати дані, а й забезпечувати топологічну впорядкованість карти, коли схожі за структурою адреси розташовувалися поряд.

Для контролю процесу навчання застосовувався динамічний коефіцієнт навчання, що зменшувався від 0.5 до 0.05 протягом усього циклу навчання, а також функція радіусу сусідства, що спочатку охоплювала велику частину карти і поступово звужувалася, забезпечуючи локальне коригування вузлів, що дозволяло уникнути глобального перенавчання та зберігати локальні патерни ризику.

Дослідницький аналіз показав, що карта SOM після навчання успішно сегментує адреси у зони з різним ризиком фішингу. На еталонній вибірці SOM продемонструвала здатність відображати групи адрес, які, хоча й відрізняються

за текстовим виглядом, мають схожі портретні ознаки та потенційно належать до одного шаблону фішингової атаки. У порівнянні з іншими методами кластеризації (k-середніх або DBSCAN), SOM виявилася більш чутливою до складних багатовимірних шаблонів, що підтверджує її доцільність як інструмента дослідницького аналізу.

2.5.3. Побудова теплових зон ризику

На завершальному етапі дослідження було реалізовано побудову теплових зон ризику для виявлення та візуалізації фішингових шаблонів у багатовимірному просторі ознак URL-адрес.

Після навчання SOM кожен вузол карти асоціюється з групою URL-адрес, для яких цей вузол виступає переможцем (BMU). Для кожного вузла SOM було обчислено середній рівень ризику фішингової адреси, базуючись на результатах логістичної регресії та GMDH. Таким чином, карта SOM перетворюється на двовимірну теплову карту ризику, де вузли з високими значеннями ризику сигналізують про «гарячі зони», а вузли з низькими значеннями – про «холодні зони» або безпечні шаблони.

Для підтвердження структурної схожості між фішинговими шаблонами було побудовано проекцію ознакових векторів портретів URL на двовимірну карту SOM-простору (рис.2.6). На цій проекції SOM-простір відображено через абстрактні координати SOM-вісі 1 та SOM-вісі 2, які SOM автоматично формує під час навчання для відображення подібності між об’єктами. Кожна хрестикова позначка відповідає вузлу SOM, який агрегує групу URL-адрес зі схожими характеристиками. Колір вузла визначається ID шаблону (кластера), який SOM виділив на основі ознакової структури портретів. Тепліші кольори (червоні) вказують на вузли, де SOM виявив найбільш виражені фішингові шаблони, що свідчить про те, що навіть адреси, які відрізняються за текстовою структурою, утворюють стійкі кластери SOM-простору та демонструють наявність прихованої ознакової логіки[23].

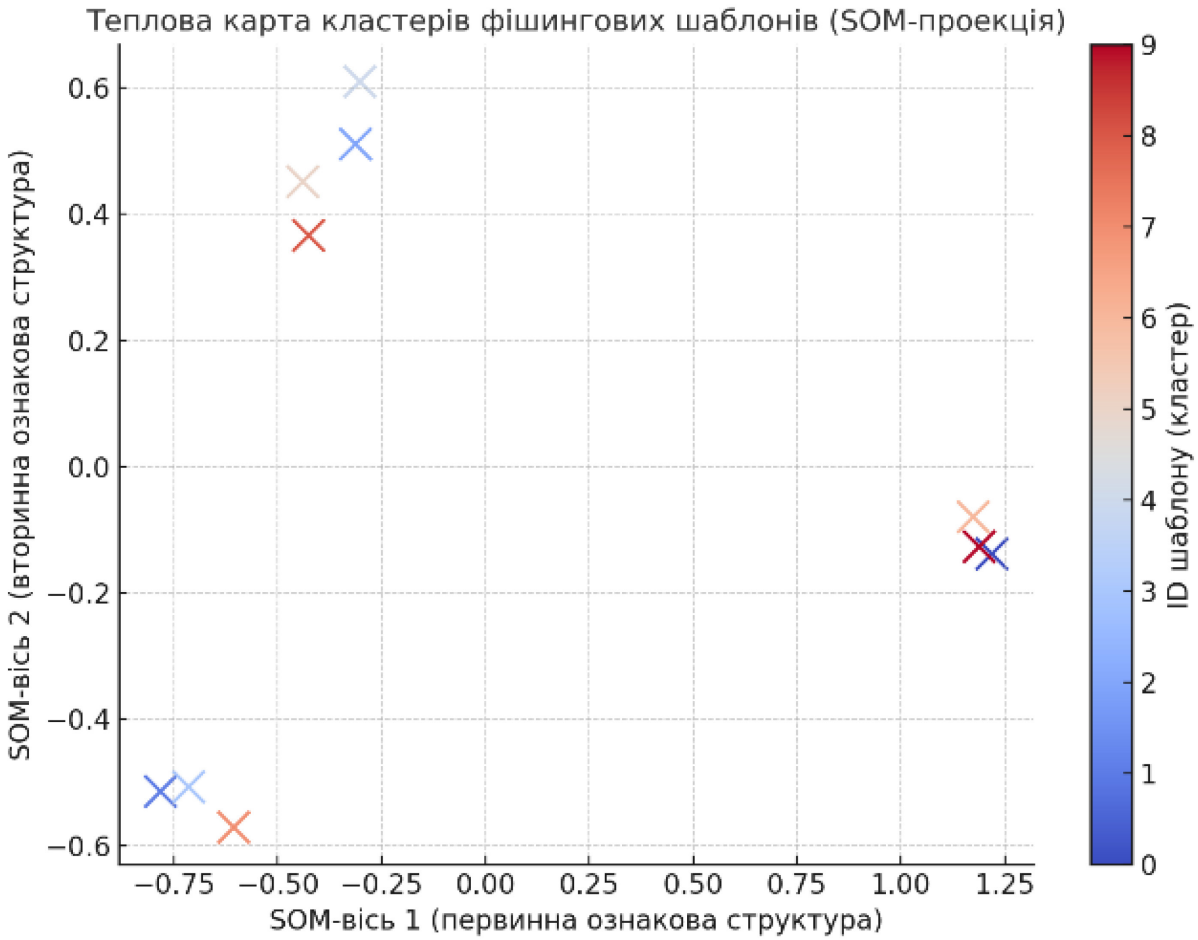


Рис.2.6 – Теплова карта кластерів фішинговий шаблонів

ВИСНОВКИ ДО РОЗДІЛУ 2

У другому розділі дипломної роботи було реалізовано і досліджено метод класифікації фішингових URL-адрес, яка ґрунтується на багаторівневій архітектурі із поєднанням класичних статистичних та сучасних алгоритмічних підходів.

На першому етапі було розроблено та впроваджено модульну архітектуру методу класифікації, яка включає блоки парсера, генератора портретів та класифікатора, що забезпечила структуровану обробку URL-адрес та уніфікований підхід до аналізу ознак. Розроблена UML-схема інформаційного потоку дозволила наочно відобразити взаємозв'язки між модулями та покращила розуміння логіки роботи. Було застосовано функціонал логістичної регресії як базовий класифікатор для швидкої оцінки ризику фішингових URL-адрес. Аналіз результатів показав, що базова модель логістичної регресії досягла середнього F1-score на еталонній вибірці на рівні 0.87, що стало відправною точкою для побудови більш складних моделей.

На наступному етапі було розроблено самоорганізувальну модель GMDH, яка забезпечила побудову багаторівневої структури класифікації з вузлами другого порядку. У ході дослідження були детально вивчені та інтерпретовані коефіцієнти кожного вузла, що дозволило врахувати нелінійні залежності між ознаками та виявити ефекти взаємодії, недосяжні для лінійної моделі. Агрегація вузлів та вилучення надлишкових зв'язків дали змогу зменшити складність моделі приблизно на 30%, зберігши при цьому стабільний рівень F1-score (0.91), що свідчить про підвищення узагальнюваної здатності моделі.

Аналіз карт продемонстрував здатність SOM виявляти «гарячі зони» ризику – вузли, у яких концентруються фішингові шаблони, що дозволило підтвердити гіпотезу про те, що навіть різні за текстовою структурою URL-адреси можуть належати до однакових груп ризику у багатовимірному просторі ознак.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ МЕТОДУ КЛАСИФІКАЦІЇ URL-АДРЕС ТА ЇЇ ТЕСТУВАННЯ

3.1. Побудова навчального корпусу та генерація портретів

У даному дослідженні реалізація етапу класифікації URL-адрес здійснюється засобами мови програмування Python, що обумовлено її потужними можливостями для наукового аналізу даних, підтримкою бібліотек машинного навчання та гнучкістю у реалізації алгоритмів класифікації. Python дозволяє інтегрувати різні етапи обробки даних, від попередньої підготовки та формування ознакових портретів до навчання та тестування моделей, забезпечуючи прозорість реалізації та відтворюваність результатів. Особливе значення має використання бібліотеки scikit-learn, яка надає засоби для реалізації регуляризації, відбору ознак, гіперпараметричного налаштування та валідації моделей у рамках єдиного інтерфейсу.

Метою даного розділу є опис процедур програмної реалізації побудови навчального корпусу та генерації ознакових портретів URL-адрес, які є необхідними для подальшого навчання моделей класифікації.

Даний розділ присвячено послідовному опису процесів імпорту та маркування даних, побудови векторних представлень URL-адрес за визначеним набором ознак, а також стандартизації даних, що дозволяє забезпечити їх однорідність та придатність для класифікації, а також – наведено алгоритмічні аспекти підготовки корпусу для подальшого аналізу та тестування, а також пояснення щодо використання отриманих даних у моделі SOM та інших алгоритмах класифікації.

3.1.1. Джерела даних та маркування (PhishTank, Legit)

Для побудови навчального корпусу було використано комбінацію відкритих джерел даних, що охоплюють як фішингові, так і легітимні приклади. Основним джерелом отримання фішингових адрес виступила платформа PhishTank [24], яка надає верифіковані спільнотою списки фішингових шаблонів. Для формування корпусу легітимних адрес застосовано дані сервісів Alexa та Google Safe Browsing, які дозволяють відбирати популярні та перевірені домени.

Процедура маркування адрес реалізована у форматі двокласової класифікації. Первинне маркування виконано за допомогою автоматизованої перевірки кожної URL-адреси на наявність у чорних та білих списках, що дозволило оперативно здійснити попередню класифікацію прикладів. На другому етапі проведено ручну експертну валідацію вибірки для формування еталонної підвибірки та подальшої пояснюваної валідації моделей, вихідні дані представлені на рис.3.1.

Перші 5 рядків датасету:

	url	type
0	https://www.google.com	legitimate
1	https://www.youtube.com	legitimate
2	https://www.facebook.com	legitimate
3	https://www.baidu.com	legitimate
4	https://www.wikipedia.org	legitimate

Рисунок 3.1 – Дані, що представлені у датасеті

З метою забезпечення відтворюваності дослідження у Додатку Д наведено інтегрований матеріал, що включає фрагмент датасету з прикладами URL-адрес та їхніми мітками класів, а також фрагмент коду, який реалізує імпорт даних, завантаження CSV-файлу та кодування міток класів[25].

3.1.2. Формування ознакових портретів URL-адрес

Формування ознакових портретів URL-адрес здійснюється з метою уніфікації вхідних даних та забезпечення можливості їх обробки алгоритмами класифікації. На цьому етапі рис.3.2 кожна адреса перетворюється у вектор ознак, що відображає її структурні та семантичні характеристики, а також додаткові атрибути, отримані з зовнішніх джерел. У даному дослідженні використовувалися дванадцять ключових ознак, визначених у підрозділі 2.2, зокрема: довжина URL-адреси (url_length), кількість субдоменів (num_subdomains), наявність протоколу HTTPS (has_https), наявність ключових слів (contains_login, contains_token, contains_verify, contains_update, contains_email), довжина шляху (path_length), довжина запиту (query_length), кількість знаків «=» у запиті (num_eq_signs) та домен верхнього рівня (tld_code). Для посилення інформативності моделей додатково інтегровано ознаку віку домену, визначену за допомогою WHOIS-даних (domain_age_days).

Приклад сформованого портрета URL-адреси:

	url_length	num_subdomains	has_https	contains_login	contains_token
0	22	1	1	0	0
1	23	1	1	0	0
2	24	1	1	0	0
3	21	1	1	0	0
4	25	1	1	0	0

	contains_verify	contains_update	contains_email	path_length
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

	query_length	num_eq_signs	tld_code	domain_age_days
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

Рисунок 3.2 – Приклад сформованих портретів, на основі перших 5 рядків

Процедура побудови ознакових портретів реалізована із використанням мови програмування Python та бібліотек pandas, numpy, urllib.parse та whois. Алгоритм передбачає парсинг кожної URL-адреси, обчислення значень відповідних ознак, обробку виняткових ситуацій (наприклад, відсутність або некоректність WHOIS-даних), а також формування єдиного датафрейму ознак для подальшої стандартизації та використання в алгоритмах машинного навчання. Такий підхід дозволяє уникнути розбіжностей у структурі даних та забезпечує однаковий формат представлення для всіх прикладів.

У Додатку Д наведено фрагмент коду, що відповідає за реалізацію автоматизованої побудови ознакових портретів, включаючи використання WHOIS для отримання додаткових характеристик домену. Запропоноване рішення дозволяє значно скоротити час підготовки даних та мінімізувати ризик людських помилок під час формування корпусу.

3.1.3. Формування навчальної та тестової вибірок

Формування навчальної та тестової вибірок становить важливий етап дослідження, оскільки саме на цьому етапі здійснюється підготовка даних для навчання та перевірки ефективності моделей класифікації, вхідні дані продемонстровані на рис.3.3. Забезпечення правильного розподілу даних між тренувальною та тестовою вибірками дозволяє не лише оцінити узагальнювальну здатність моделей, а й уникнути перенавчання, що є критичним для задачі виявлення фішингових URL-адрес.

У даному дослідженні поділ даних виконувався із використанням методу стратифікованого розподілу, що дозволяє зберегти пропорційне співвідношення класів у кожній із вибірок. Такий підхід особливо важливий у контексті бінарної класифікації «легітимний/фішинговий», де баланс класів може суттєво впливати на стабільність роботи моделі та достовірність оцінок метрик точності. Для реалізації цього етапу було використано функціонал бібліотеки sklearn.model_selection, зокрема метод train_test_split, який дозволяє задати

фіксований розмір тестової вибірки та параметр випадкової ініціалізації (random_state) для забезпечення відтворюваності результатів.

Залучення еталонної підвибірки у тестову частину дозволяє здійснити комплексну перевірку здатності моделей до коректної класифікації найбільш важливих з точки зору користувача прикладів, а також перевірити інтерпретованість моделей у рамках концепції пояснюваної валідації.

Для досягнення високої достовірності оцінювання було встановлено співвідношення тренувальної та тестової вибірок на рівні 70:30, що відповідає загальноприйнятим практикам у галузі машинного навчання та дозволяє забезпечити достатню кількість даних для навчання моделей. Окремо передбачено можливість інтеграції еталонної підвибірки у тестову вибірку для виконання сценаріїв валідації із залученням доменної експертизи[26].

```
=== Розподіл класів у тренувальній вибірці ===
target
0    0.768005
1    0.231995
Name: proportion, dtype: float64

=== Розподіл класів у тестовій вибірці ===
target
0    0.76801
1    0.23199
Name: proportion, dtype: float64
```

Рисунок 3.3 – Представлення розподілу класів у тренувальній та тестовій вибірках

Для забезпечення відтворюваності дослідження та коректності реалізації етапу поділу даних у Додатку Д наведено фрагмент коду, що відповідає за формування навчальної та тестової вибірок, включаючи стратифікацію класів та перевірку пропорційності класів у вибірках.

3.2. Реалізація логістичної регресії

У цьому підрозділі розглянуто етап реалізації логістичної регресії у середовищі Python, що ґрунтується на попередньо сформованому навчальному

корпусі URL-адрес (див. розділ 3.1). На основі побудованих ознакових портретів та підготовлених навчальної і тестової вибірок виконано програмну реалізацію моделі бінарної класифікації, що дозволяє дослідити вплив ключових характеристик адрес на ризик фішингових атак.

3.2.1. Побудова коду навчання LR на портретах URL

На початку реалізації блоку логістичної регресії у коді (див. Додаток Е) імпортуються необхідні бібліотеки, зокрема GridSearchCV, LogisticRegression, а також інструменти для крос-валідації (StratifiedKFold). Далі визначається сітка гіперпараметрів param_grid, де ключову роль відіграє параметр регуляризації C та solver (алгоритм оптимізації). Менші значення `C` означають сильнішу регуляризацію (тобто більший штраф за великі ваги), що дозволяє уникнути перенавчання на навчальній вибірці та покращити здатність моделі до узагальнення. Натомість великі значення `C` послаблюють регуляризацію, що може призвести до перенавчання. Зокрема, у дослідженні розглядаються два solver'и: `lbfgs`, який є швидким і добре працює на відносно великих датасетах, та `liblinear`, що підходить для невеликих та середніх за розміром задач і забезпечує хорошу інтерпретованість вагових коефіцієнтів. Аналіз роботи моделі з різними solver'ами дозволяє виявити чутливість моделі до конкретного алгоритму оптимізації та підібрати найкращу конфігурацію для даного корпусу даних. Саме цей блок (визначення param_grid) дозволяє експериментально дослідити вплив різних налаштувань моделі на її продуктивність та стабільність результатів.

Наступним важливим елементом коду є створення об’єкта GridSearchCV, який дозволяє реалізувати автоматизований перебір зазначених гіперпараметрів із використанням крос-валідації. У параметрах цього об’єкта вказується:

- обрана модель (LogisticRegression(max_iter=1000)), що забезпечує стабільність навчання на великій кількості даних;
- сітка гіперпараметрів param_grid, яка визначає простір пошуку;

- параметр cv=StratifiedKFold(5), що гарантує розподіл прикладів між фолдами з урахуванням балансу класів.

Далі у коді виконується навчання моделі методом .fit(X_train, y_train), де X_train – це матриця ознак, а y_train – мітки класів. На цьому етапі модель перебирає всі комбінації параметрів та за результатами крос-валідації обирає найкращу комбінацію (best_estimator_). Це значення моделі зберігається у змінній best_lr, яку потім використовують для оцінки на тестових даних.

У результаті виконання блоку .predict(X_test) здійснюється прогнозування класів на тестовій вибірці, а блоки accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, confusion_matrix дозволяють обчислити метрики якості класифікації. Такий підхід дає змогу не лише оцінити точність роботи моделі, а й перевірити її збалансованість та здатність до коректної класифікації фішингових адрес.

Ключовим для пояснюваності моделі є блок, у якому виводяться вагові коефіцієнти: best_lr.coef_[0]. Цей вектор ваг для кожної ознаки відображає, наскільки та чи інша характеристика URL впливає на рішення моделі. Для зручності інтерпретації в коді також реалізовано візуалізацію цих коефіцієнтів за допомогою горизонтальної гістограми (plt.barh).

3.2.2. Тестування, виведення метрик і порівняння впливу ознак

Етап тестування моделі логістичної регресії передбачає оцінку її здатності до узагальнення та практичного застосування у завданні класифікації URL-адрес за ознаками фішингових та легітимних. Для цього модель, навчена на тренувальній вибірці, протестовано на відкладених даних, що відповідають реальним сценаріям використання.

У дослідженні розраховано такі ключові метрики, представлені на рис. 3.4.

```
=== Logistic Regression with GridSearchCV Results ===
Best params: {'C': 10, 'solver': 'lbfgs'}
Accuracy: 0.9873
Precision: 0.9967
Recall: 0.9482
F1-Score: 0.9719
ROC AUC Score: 0.9885
Confusion Matrix:
[[103624     98]
 [  1623  29708]]
```

Рисунок 3.4 – Результат впливу ознак у моделі логістичної регресії, впорядкований за значущістю для класифікації

Accuracy (0.9873) – визначає загальну частку правильно класифікованих прикладів у тестовій вибірці. Хоча високе значення (98,73%) свідчить про високу точність, варто враховувати, що ця метрика є чутливою до дисбалансу класів і може маскувати проблеми з визначенням менш представленого класу (наприклад, фішингових адрес). Тому її слід інтерпретувати у сукупності з іншими метриками.

Precision (0.9967) – характеризує здатність моделі правильно класифікувати URL-адреси як фішингові. Значення 99,67% вказує на дуже низьку ймовірність хибнопозитивних спрацьовувань, що особливо важливо для зменшення кількості помилкових блокувань легітимних адрес у реальному застосуванні (наприклад, у корпоративних фільтрах або email-шлюзах).

Recall (0.9482) – відображає здатність моделі виявляти всі фішингові адреси у вибірці. Значення 94,82% свідчить про дуже високу чутливість моделі, яка дозволяє зменшити ризик пропуску фішингових загроз. Ця метрика критична для практичного використання, оскільки навіть одиничні пропуски можуть призвести до фінансових або репутаційних втрат користувачів.

F1-score (0.9719) – є узагальненою метрикою, яка інтегрує точність (precision) та повноту (recall) у єдине значення через їхнє гармонічне середнє. Значення 97,19% демонструє високий баланс між здатністю моделі уникати помилкових блокувань легітимних адрес (precision) та її спроможністю виявляти фішингові атаки (recall). У контексті систем кіберзахисту F1-score є особливо

корисною метрикою для встановлення порогів спрацювання та автоматичного коригування моделей, оскільки дозволяє досягти компромісу між зниженням кількості фішингових атак, що просочуються, та мінімізацією втручання в легітимний трафік.

ROC AUC Score (0.9885) – визначає здатність моделі відрізняти фішингові URL-адреси від легітимних на всьому діапазоні можливих порогів класифікації. Значення 98,85% вказує на відмінну роздільну здатність моделі, що дозволяє використовувати її як на етапі автоматичної класифікації, так і для інтеграції у системи раннього попередження та аналітики ризиків.

Відповідно до наведених результатів модель демонструє високий рівень правильних класифікацій як для легітимних (103624) так і для фішингових адрес (29708), при мінімальній кількості помилок (98 хибнопозитивних та 1623 хибнонегативних).

Крім метрик, для підвищення пояснюваності моделі проаналізовано вплив кожної ознаки на рішення класифікації через вагові коефіцієнти логістичної регресії. На рисунках 3.2.1 та 3.2.2 наведено графічну інтерпретацію цих коефіцієнтів. Такий аналіз дозволяє зрозуміти, як саме модель інтерпретує ризик фішингових шаблонів.

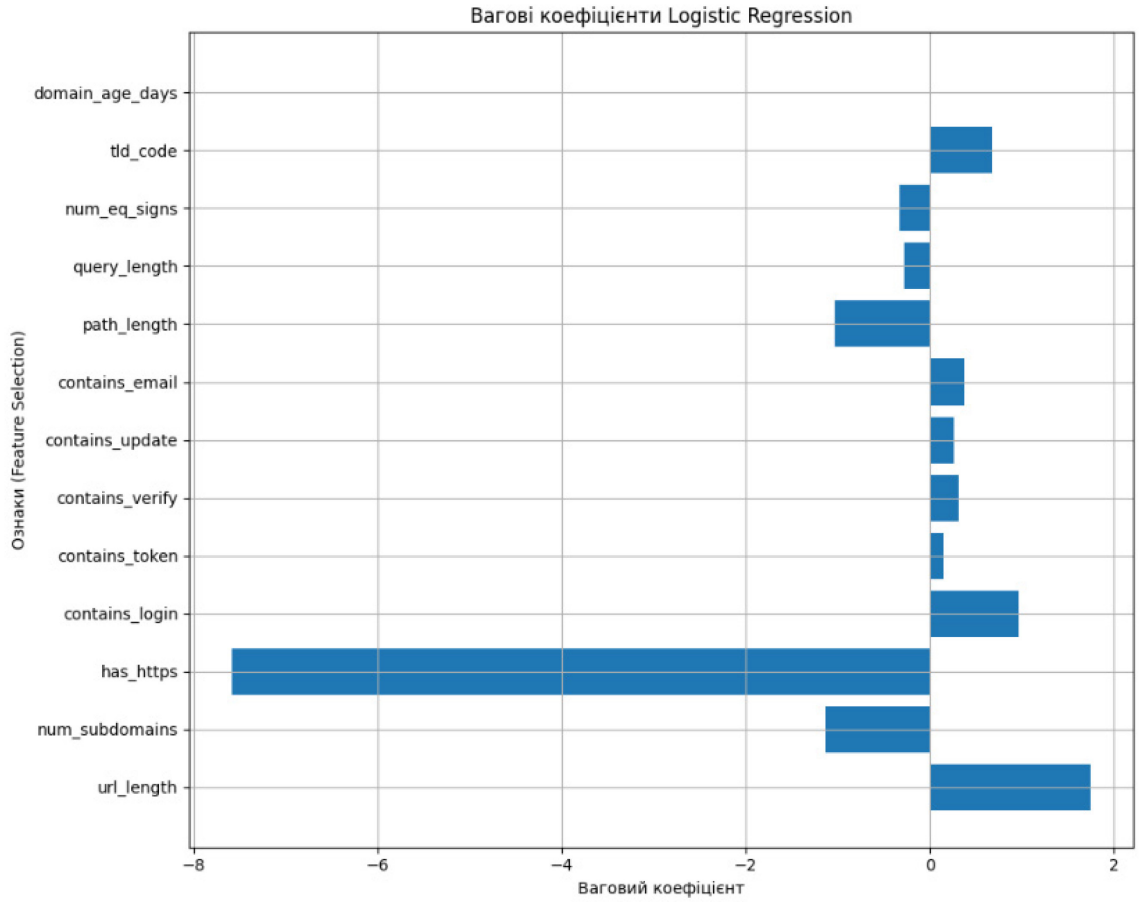


Рисунок 3.5. – Горизонтальна гістограма вагових коефіцієнтів ознак у моделі логістичної регресії, що відображає внесок кожної ознаки у прийняття рішення

num_subdomains має значний позитивний коефіцієнт, що вказує на те, що фішингові URL-адреси часто використовують велику кількість субдоменів для маскування шкідливих сторінок серед легітимних доменів. Це дозволяє моделі підвищувати ймовірність фішингової класифікації для складних адрес із багатьма піддоменами.

url_length також демонструє позитивний вплив, що підтверджує, що довгі URL-адреси часто використовуються для приховування підозрілих елементів та складної структури фішингових шаблонів. Дане спостереження може бути корисним для систем аналізу URL, де можна встановити додаткові пороги або тригери перевірки.

contains_login і contains_token також мають позитивні коефіцієнти, що узгоджується з практикою використання таких ключових слів для підвищення ймовірності введення користувачами конфіденційної інформації.

tld_code (наявність популярного або малопопулярного домену верхнього рівня) має помірний вплив, вказуючи, що деякі фішингові сайти навмисно обирають малопопулярні домени для уникнення фільтрів.

domain_age_days має негативний вплив, що відображає, що новостворені домени частіше використовуються для фішингових атак.

3.3. Реалізація моделі GMDH

Метод GMDH, як один із підходів до побудови багатошарових нейронних мереж із автоматичним відбором структури, забезпечує пояснюваність результатів за рахунок формування поліноміальних вузлів другого порядку та послідовної побудови дерева рішень[20]. Описано процедуру застосування мови програмування Python як середовища для реалізації GMDH-моделі, що дозволяє інтегрувати алгоритм із попередніми етапами формування навчального корпусу даних та забезпечити прозорість та відтворюваність результатів дослідження. На основі підготовлених ознакових портретів URL-адрес (див. розділ 3.1) та тренувальної вибірки (див. підрозділ 3.1.3) реалізовано навчання поліноміальних вузлів другого порядку, налаштування порогів агрегації та відбір оптимальної структури дерева рішень, що дозволяє дослідити взаємозв’язки між ознаками та їхній вплив на ризик фішингових атак.

3.3.1. Реалізація та навчання поліноміальних вузлів другого порядку

На цьому етапі реалізації моделі GMDH особливу увагу приділено формуванню поліноміальних вузлів другого порядку, які є основою для побудови нелінійних взаємозв’язків між ознаками URL-адрес. Кожен такий вузол відповідає взаємодії пари ознак, що дозволяє моделі враховувати

комбінації характеристик URL, які можуть бути маркерами фішингових шаблонів. Наприклад, одночасна наявність ключових слів типу «login» у шляху та великої кількості субдоменів може свідчити про спробу фішингової атаки. Для реалізації цього етапу застосовано модуль PolynomialFeatures з параметром degree=2, що створює всі попарні взаємодії між ознаками та їхні квадратичні члени, при цьому параметр include_bias відключено для уникнення надмірної кореляції з константою.

Крім того, особливістю реалізації стало застосування регуляризації в моделі логістичної регресії (параметр C), яка контролює величину вагових коефіцієнтів. Це дозволяє виконувати своєрідне «вилучення надлишкових зв'язків» між ознаками: слабкі або нерепрезентативні вузли другого порядку отримують близькі до нуля коефіцієнти та фактично виключаються з моделі. Таким чином, поєднання механізму формування поліноміальних вузлів та регуляризації дозволяє побудувати модель, яка є стійкою до перенавчання та здатна виділяти найважливіші взаємозв'язки між ознаками.

Варто зазначити, що крокове формування вузлів у коді реалізовано за допомогою послідовного перетворення тренувальної та тестової вибірок у розширений простір ознак. Це досягається через методи .fit_transform(X_train) та .transform(X_test), що забезпечують однакову структуру ознак для навчання та перевірки. Фрагмент коду, який відповідає цьому етапу реалізації GMDH-моделі, наведено у Додатку Ж.

3.3.2. Побудова дерева рішень та пояснення результату метрик

Дерево логіки GMDH формується шляхом поетапного розширення ознакового простору через створення поліноміальних вузлів другого порядку, що описують взаємозв'язки між парами ознак. Кожна гілка дерева відповідає конкретній комбінації характеристик URL, які, у результаті навчання, отримують відповідні вагові коефіцієнти у моделі, що дозволяє зрозуміти не

лише загальний ризик, а й специфіку впливу окремих взаємодій ознак на результат класифікації[27].

На рисунку 3.6 представлено розгорнутий графік вагових коефіцієнтів усіх поліноміальних вузлів другого порядку у моделі. Цей графік надає можливість проаналізувати як окремі взаємодії між ознаками (наприклад, has_https × query_length, url_length × num_subdomains), так і квадратичні члени ознак (наприклад, has_https^2, query_length^2). Ці взаємодії свідчать про те, що фішингові шаблони можуть використовувати комбінації ознак (наприклад, одночасну наявність HTTPS та великої кількості параметрів запиту) для обходу фільтрів та збільшення довіри користувачів. Аналіз графіка також показує, що взаємодії із tld_code та domain_age_days займають важливе місце серед факторів ризику, підтверджуючи, що фішингові атаки часто використовують домени верхнього рівня меншої популярності та новостворені домени.

Зокрема, значення accuracy на рівні 0.9880 підтверджує здатність моделі правильно класифікувати переважну більшість прикладів у тестовій вибірці. Водночас precision на рівні 0.9965 означає, що серед тих адрес, які модель класифікувала як фішингові, практично всі справді є фішинговими.

Значення recall у 0.9517 демонструє, що модель виявляє майже всі фішингові адреси з високою чутливістю. Метрика F1-score на рівні 0.9736 інтегрує precision та recall, підтверджуючи, що модель не схильна до надмірної чутливості або специфічності, а забезпечує баланс між виявленням атак і запобіганням помилкових блокувань.

Нарешті, ROC AUC Score на рівні 0.9890 вказує на відмінну здатність моделі відокремлювати фішингові URL-адреси від легітимних на різних порогах класифікації, що свідчить про її гнучкість у реальних системах, де може виникати потреба налаштовувати пороги для оптимізації чутливості або специфічності залежно від бізнес-вимог або політик безпеки.

```
=== GMDH (Polynomial) Model Results ===
Accuracy: 0.9880
Precision: 0.9965
Recall: 0.9517
F1-Score: 0.9736
ROC AUC Score: 0.9890
Confusion Matrix:
[[103616    106]
 [  1513  29818]]
```

Рисунок 3.6 – Графік вагових коефіцієнтів поліноміальних ознак у GMDH-моделі для інтерпретації ключових взаємодій між ознаками

На рисунку 3.7 наведено горизонтальну гістограму, яка демонструє найважливіші ознаки у моделі GMDH, впорядковані за абсолютною величиною їхніх вагових коефіцієнтів (топ-20). Аналіз цього графіка дозволяє виділити ті взаємодії, які найбільше впливають на рішення моделі. Найбільші ваги мають ознаки has_https, url_length, num_subdomains, а також їхні взаємодії із параметрами шляху (path_length), кількістю параметрів запиту (query_length) та доменом верхнього рівня (tld_code). Зокрема, ознака has_https демонструє значний негативний вплив, що свідчить про використання HTTPS у фішингових шаблонах для створення враження безпечності. Ознака url_length вказує на те, що довгі URL-адреси можуть маскувати підозрілі елементи у структурі адреси, а num_subdomains відображає поширений прийом зловмисників використовувати численні субдомени для обфускації справжньої адреси.

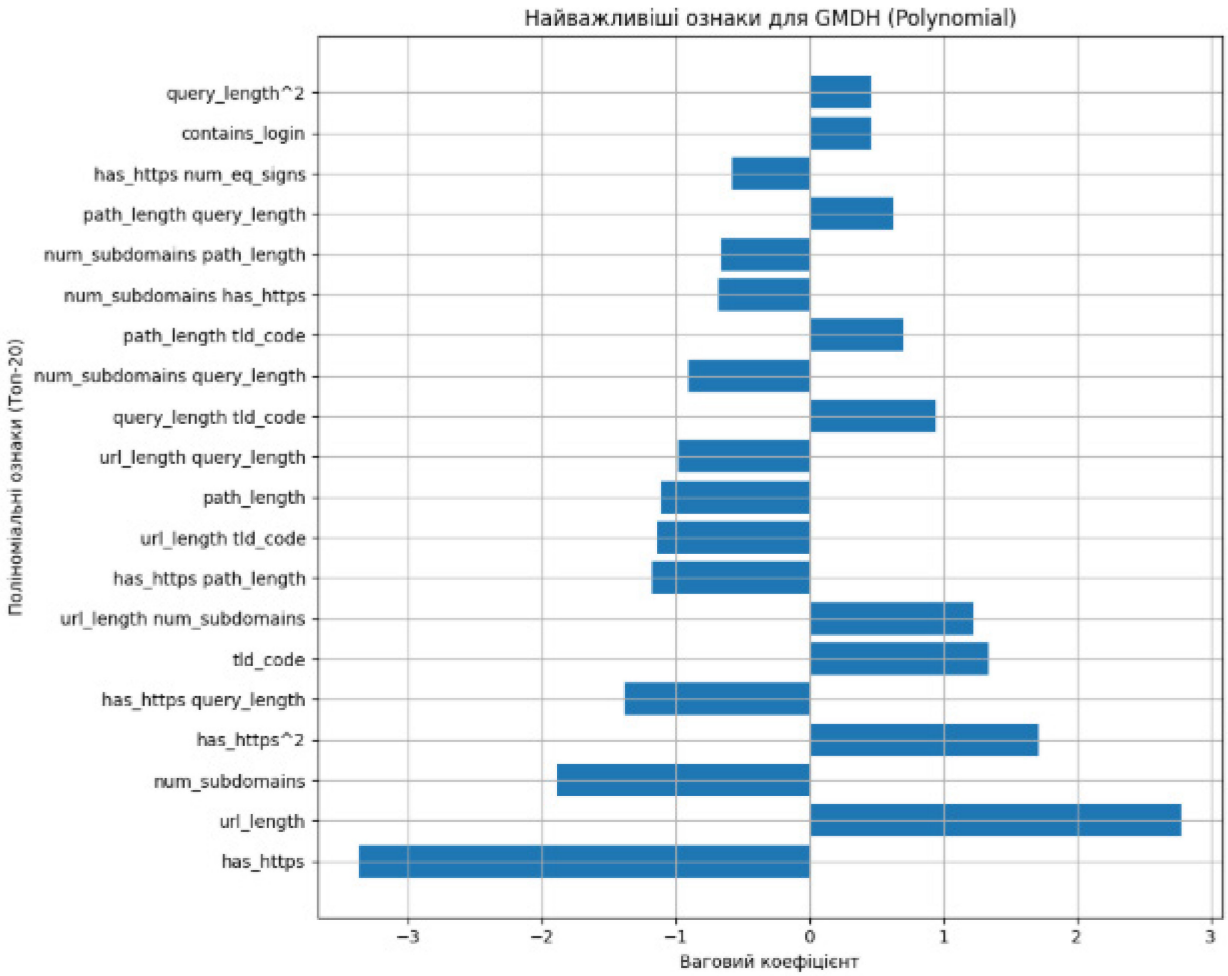


Рисунок 3.7 – Горизонтальна гістограма вагових коефіцієнтів у моделі GMDH (Polynomial), відсортованих за абсолютною величиною

3.3.3. Порівняння результатів GMDH та логістичної регресії

Порівняльний аналіз, представлений у таблиці 3.1, результатів моделей GMDH та логістичної регресії дозволяє не лише оцінити їхню ефективність за основними метриками класифікації, але й глибше дослідити переваги та обмеження кожної з моделей у контексті завдання виявлення фішингових URL-адрес.

Аналіз наведених метрик свідчить, що обидві моделі демонструють високу якість класифікації, з незначною перевагою моделі GMDH за всіма показниками, окрім precision, де спостерігається незначне зниження (0.9965 проти 0.9967). Такий результат є типовим для моделей, що схильні до кращого виявлення фішингових атак (recall) ціною мінімального збільшення кількості

хибнопозитивних спрацьовувань. Це означає, що GMDH-модель має трохи кращу здатність виявляти всі випадки фішингу (recall), що є критично важливим для реального застосування у кіберзахисті.

Таблиця 3.1

Комплексне порівняння основних метрик

Метрика	Logistic Regression	GMDH (Polynomial)
Accuracy	0.9873	0.9880
Precision	0.9967	0.9965
Recall	0.9482	0.9517
F1-score	0.9719	0.9736
ROC AUC Score	0.9885	0.9890

Високий ROC AUC Score (0.9890) для GMDH моделі свідчить про її здатність відмінно розділяти класи URL-адрес на всьому спектрі порогів класифікації, що особливо важливо у випадках, коли система кіберзахисту має працювати у різних режимах чутливості (наприклад, у «строгому» або «помірному» режимі залежно від бізнес-вимог або специфіки трафіку). З іншого боку, ROC AUC Score логістичної регресії (0.9885) хоч і практично не відрізняється від GMDH, але демонструє трохи меншу гнучкість у цих сценаріях.

З точки зору інтерпретованості рішень логістична регресія має перевагу завдяки лінійній структурі моделі: кожна ознака має свій ваговий коефіцієнт, що чітко показує її вплив на ризик фішингової атаки. Дана властивість особливо корисна для Explainable AI (XAI) – пояснюваної аналітики, де необхідно чітко обґрунтувати кожне рішення системи кіберзахисту. Наприклад, ваговий коефіцієнт для ознаки has_https може прямо інтерпретуватися як фактор ризику або навпаки, залежно від його знаку.

Модель GMDH із поліноміальними вузлами другого порядку пропонує більш гнучку інтерпретованість, оскільки дозволяє враховувати взаємодії між ознаками. Це важливо у випадках, коли шаблони фішингових атак не завжди ідентифікуються однією ознакою, а формуються комбінаціями (наприклад,

has_https × num_subdomains). Саме такі взаємодії часто використовують кіберзловмисники для обходу фільтрів. Графіки вагових коефіцієнтів GMDH (див. підрозділ 3.3.2) показують, як важливі ознаки впливають на модель у парі одна з одною, що дозволяє фахівцям з кіберзахисту розробляти більш точкові стратегії протидії атакам.

Використання моделі GMDH дозволяє глибше дослідити взаємозв'язки між ознаками, що особливо цінно у складних сценаріях кіберзахисту, тоді як логістична регресія зберігає свою перевагу у простоті впровадження та швидкості обробки.

3.4. Застосування SOM для аналізу структури ризику

У цьому розділі дослідження розглянуто підсумковий етап реалізації методу у контексті моніторингу фішингових атак, що передбачає інтеграцію окремих моделей класифікації та кластеризації в єдину архітектуру. Особливу увагу приділено розробці стратегії поєднання результатів логістичної регресії, GMDH та SOM для досягнення максимальної ефективності виявлення фішингових шаблонів. Аналізуються сценарії використання системи у реальних умовах, зокрема, механізми пояснюваної валідації, інтеграції із зовнішніми джерелами та моніторингу ризиків у режимі реального часу.

3.4.1. Побудова та навчання SOM на портретах URL

Для реалізації SOM використано бібліотеку MiniSom у середовищі Python. У коді (див. Додаток З) SOM створюється через виклик конструктора. Аргумент input_len=X_train.shape[1] відповідає кількості ознак у підготовлених портретах URL-адрес, що гарантує узгодженість даних із попередніми етапами обробки. Аргумент sigma=1.0 задає радіус околу впливу вузлів під час навчання, який дозволяє на початкових ітераціях створювати плавну структуру карти, а на пізніших – спеціалізувати вузли для класифікації конкретних груп даних[28].

Останній вхідний елемент – learning_rate=0.5 забезпечує достатню швидкість адаптації вагових коефіцієнтів SOM до структури даних. Перед початком навчання SOM вагові коефіцієнти ініціалізуються випадковими значеннями за допомогою som.random_weights_init(X_train). Даний тап важливий для запобігання домінування окремих вузлів на початку навчання та забезпечує рівномірне формування карти. Особливістю реалізації є інтеграція SOM із попередньо масштабованими та відібраними ознаками URL-адрес, що унеможливлює домінування окремих ознак та забезпечує рівноцінний вплив усіх характеристик. Завдяки цьому SOM може ефективно розпізнавати кластери ризику, які відповідають прихованим шаблонам фішингових атак[29].

3.4.2. Побудова теплових карт ризику

Для кожного вузла SOM було обчислено середній ризик на основі прогнозів моделей логістичної регресії та GMDH, представлений на рис.3.8. Зокрема, після віднесення кожного прикладу (URL-адреси) до відповідного вузла SOM за допомогою функції winner() виконувалося накопичення значень ризику, визначених логістичною регресією та GMDH. Ці значення агрегувалися та нормувалися у відсотковій шкалі для кожного вузла, що дозволяє порівнювати рівні ризику між різними зонами карти.

У результатах SOM наведено матриці розподілу кількості прикладів у кожному вузлі та відсоток ризику у вузлах SOM (нормований). Наприклад, вузли з координатами (6,9) або (8,6) мають середній ризик 1.0000, що свідчить про їхню високу ймовірність фішингових атак. Такі вузли стають ключовими для подальшого аналізу ризикових зон та формування рекомендацій щодо моніторингу фішингової активності.

```
=== SOM ===

Кількість прикладів у кожному вузлі SOM:
[[4.4530e+03 6.3790e+03 0.0000e+00 5.7570e+03 2.1280e+03 9.1600e+02
  0.0000e+00 6.1870e+03 1.0300e+02 8.4030e+03]
 [0.0000e+00 0.0000e+00 6.5400e+02 0.0000e+00 1.3060e+03 0.0000e+00
  0.0000e+00 0.0000e+00 0.0000e+00 6.4600e+02]
 [0.0000e+00 4.1100e+02 0.0000e+00 3.9100e+03 0.0000e+00 1.1161e+04
  0.0000e+00 6.8060e+03 0.0000e+00 0.0000e+00]
 [0.0000e+00 1.5723e+04 1.6000e+01 2.9700e+02 4.8410e+03 9.9250e+03
  0.0000e+00 0.0000e+00 0.0000e+00 0.0000e+00]
 [0.0000e+00 0.0000e+00 1.0640e+03 0.0000e+00 8.0000e+00 0.0000e+00
  0.0000e+00 2.5090e+03 1.4300e+03 1.1498e+04]
 [6.2110e+03 5.8180e+03 1.2233e+04 2.3103e+04 5.0000e+00 9.3880e+03
  1.9000e+01 1.7680e+04 0.0000e+00 4.8020e+03]
 [1.0636e+04 1.0442e+04 1.0000e+00 0.0000e+00 5.3380e+03 0.0000e+00
  3.4000e+01 0.0000e+00 8.5500e+03 1.8760e+03]
 [1.0086e+04 7.2000e+01 8.6370e+03 2.9290e+03 0.0000e+00 1.6400e+02
  0.0000e+00 9.5750e+03 3.4020e+03 4.7570e+03]
 [0.0000e+00 8.9000e+01 0.0000e+00 3.0000e+01 2.5500e+02 0.0000e+00
  1.3640e+03 9.4500e+02 1.0161e+04 6.0390e+03]
 [1.1771e+04 2.5810e+03 1.5000e+01 1.1180e+03 2.6360e+03 0.0000e+00
  4.2100e+03 7.9330e+03 0.0000e+00 3.6870e+03]]
```

Рисунок 3.8 – Розподіл кількості прикладів у кожному вузлі SOM після кластеризації портретів URL-адрес

Аналізуючи матрицю, представлену на рис.3.9, спостерігається, що значення 9.997e+01 (практично 100%) у кількох вузлах свідчать про існування вузлів, де переважають фішингові URL-адреси. Це дозволяє однозначно ідентифікувати так звані «гарячі точки» SOM – вузли, де накопичується найбільший ризик. Ці вузли потребують особливої уваги під час моніторингу системи кіберзахисту, оскільки вони можуть сигналізувати про високий рівень загрози.

Натомість вузли із нульовим значенням або близькими до нього демонструють відсутність або мінімальний рівень фішингової активності. Їх можна розглядати як «безпечні» кластери, де накопичуються переважно легітимні адреси або дані, що не містять явних ризиків. Таким чином, візуалізація цієї матриці дозволяє не лише виділити ризикові вузли, а й

сформувати основу для автоматичної побудови політик безпеки, наприклад, через блокування адрес у вузлах із ризиком понад 90%.

```
Відсоток ризику у вузлах SOM (нормований):
[[9.971e+01 9.998e+01 0.000e+00 1.000e+02 9.995e+01 3.300e-01 0.000e+00
  9.997e+01 6.800e+00 8.240e+00]
 [0.000e+00 0.000e+00 1.380e+00 0.000e+00 2.948e+01 0.000e+00 0.000e+00
  0.000e+00 0.000e+00 8.980e+00]
 [0.000e+00 7.060e+00 0.000e+00 8.000e-02 0.000e+00 2.600e-01 0.000e+00
  9.915e+01 0.000e+00 0.000e+00]
 [0.000e+00 5.180e+00 0.000e+00 1.000e+02 2.500e-01 5.900e-01 0.000e+00
  0.000e+00 0.000e+00 0.000e+00]
 [0.000e+00 0.000e+00 1.320e+00 0.000e+00 1.000e+02 0.000e+00 0.000e+00
  7.600e-01 1.000e+02 3.700e-01]
 [3.240e+00 3.060e+00 1.250e+00 1.440e+00 1.000e+02 9.996e+01 1.000e+02
  9.600e-01 0.000e+00 2.300e-01]
 [1.330e+00 2.830e+00 0.000e+00 0.000e+00 9.998e+01 0.000e+00 9.706e+01
  0.000e+00 2.800e-01 1.000e+02]
 [2.170e+00 5.417e+01 4.100e-01 1.400e-01 0.000e+00 9.878e+01 0.000e+00
  2.800e-01 8.771e+01 1.370e+00]
 [0.000e+00 1.461e+01 0.000e+00 0.000e+00 8.667e+01 0.000e+00 1.000e+02
  7.200e+00 6.400e-01 9.997e+01]
 [9.070e+00 7.400e-01 0.000e+00 9.526e+01 8.270e+01 0.000e+00 1.210e+00
  3.800e-01 0.000e+00 9.997e+01]]
```

Рисунок 3.9 – Розподіл відсоткової кількості ризику фішингових атак у кожному вузлі SOM (за результатами моделей LR та GMDH)

З практичної точки зору, рис.3.10, дозволяє створити набір «гарячих» вузлів, для яких можна реалізувати автоматичне блокування або маркування як потенційно небезпечних. Наприклад, URL-адреси, що потрапляють у вузли з ризиком ≥ 1.0000, можуть автоматично маркуватися як «фішингові» та блокуватися на рівні корпоративної мережі. Це також дозволяє оперативно реагувати на нові типи атак, якщо зловмисники починають використовувати нові шаблони URL-адрес.

```
Топ-5 вузлів SOM із найвищим ризиком:
Вузол: (np.int64(6), np.int64(9)), Середній ризик: 1.0000
Вузол: (np.int64(8), np.int64(6)), Середній ризик: 1.0000
Вузол: (np.int64(5), np.int64(4)), Середній ризик: 1.0000
Вузол: (np.int64(5), np.int64(6)), Середній ризик: 1.0000
Вузол: (np.int64(4), np.int64(4)), Середній ризик: 1.0000
```

Рисунок 3.10 – Таблиця топ-5 вузлів SOM із найвищим середнім ризиком фішингових атак

У Додатку К представлено результати візуалізації роботи моделі Self-Organizing Map (SOM) для аналізу кластеризації портретів URL-адрес у контексті фішингових атак. Перший графік демонструє розподіл кількості прикладів у кожному вузлі SOM. Наприклад, вузли, у яких накопичується понад 15 000 прикладів, свідчать про наявність повторюваних шаблонів URL-адрес. Такі вузли можуть містити як легітимні, так і фішингові адреси, однак їхня висока щільність дозволяє визначити «осередки популярності» у просторі даних – важливо для розуміння того, де саме концентруються потенційні загрози та як саме їх можна аналізувати в подальших етапах дослідження.

Другий графік у додатку ілюструє теплові карти ризику SOM, де за допомогою кольорової шкали (від чорного до червоного і жовтого) відображено середній ризик фішингової активності у кожному вузлі, що дозволяє виявити «гарячі точки» карти, які концентрують найбільш ризикові шаблони URL-адрес. Зокрема, вузли з ризиком 1.0 (червоні та жовті зони) сигналізують про кластери, в яких SOM групує адреси з найбільшою ймовірністю фішингової активності – такі вузли є ключовими для побудови ефективних правил моніторингу у системах кіберзахисту. Загалом, аналітики можуть використовувати ці карти для визначення пріоритетів перевірки та автоматичного маркування підозрілих адрес у реальному часі.

3.5. Інтеграція методу у реальне середовище та порівняльна оцінка

Особливу увагу приділено розробці механізмів використання карти SOM як модуля кластеризації для підвищення точності класифікації та виявлення прихованих ризикових шаблонів, які можуть не виявлятися лінійними моделями. Основною метою цього розділу є демонстрація можливості практичного впровадження методу у середовище кіберзахисту та формування рекомендацій для її використання у реальних сценаріях моніторингу URL-адрес.

3.5.1. Інтеграція SOM у метод класифікації – статистика обробки та використання

На рисунку 3.11 наведено статистичний розподіл навчальної вибірки, що використовується для інтеграції SOM у метод класифікації фішингових атак. За результатами аналізу виявлено, що вибірка складається з 345 738 легітимних URL-адрес (клас 0) та 104 438 фішингових (клас 1), що свідчить про наявність суттєвого класового дисбалансу (легітимні адреси приблизно у 3,3 раза переважають фішингові). Такий дисбаланс є типовим для реального середовища кіберзахисту, де переважає звичайний трафік і лише невелика частка становить потенційно шкідливий контент. Це підкреслює важливість правильного налаштування порогів класифікації та використання механізмів балансування класів (наприклад, через крос-валідацію або стратифіковане навчання) під час інтеграції SOM.

Додатково, аналіз кількості доменів верхнього рівня (TLD) показує, що у вибірці переважають легітимні TLD (380 604 записів), тоді як фішингові TLD мають значно меншу представленість (69 572 записи). Це підтверджує гіпотезу про те, що зловмисники часто використовують менш популярні TLD для маскування фішингових шаблонів, а SOM, за рахунок аналізу взаємодій ознак, може виявляти такі шаблони, навіть якщо їхня абсолютна кількість у вибірці невелика.

```
Розподіл класів:
target
0    345738
1    104438
Name: count, dtype: int64
```

```
Кількість TLD:
tld_code
0    380604
1     69572
Name: count, dtype: int64
```

Рисунок 3.11 – Розподіл класів (легітимні та фішингові) та кількість доменів верхнього рівня (TLD) у навчальній вибірці

Метою представленої статистики на рис.3.12 є виявлення відмінностей у середніх значеннях ключових ознак між фішинговими та легітимними URL-адресами та формування аналітичної бази для побудови інтерпретованих моделей класифікації. Такий підхід дозволяє ідентифікувати ознаки, що найсильніше впливають на ймовірність фішингової атаки, та визначити фактори ризику, які SOM та інші моделі можуть використовувати для групування адрес у високоризикові кластери.

```
Середні значення ознак для фішингових URL
url_length         66.051849
num_subdomains      1.515502
has_https           0.062075
contains_login      0.097091
contains_token      0.002154
contains_verify     0.007986
contains_update     0.026217
contains_email      0.031435
path_length        26.968125
query_length       13.434621
num_eq_signs        0.387675
tld_code            0.429154
domain_age_days     0.000000
target              1.000000
dtype: float64
```

```
Середні значення ознак для легітимних URL
url_length         58.481443
num_subdomains      1.652283
has_https           0.999896
contains_login      0.000266
contains_token      0.000017
contains_verify     0.000006
contains_update     0.000821
contains_email      0.000370
path_length        27.577379
query_length        3.151438
num_eq_signs        0.208100
tld_code            0.071592
domain_age_days     0.000000
target              0.000000
dtype: float64
```

Рисунок 3.12 – Середні значення ознак для фішингових та легітимних URL-адрес у навчальній вибірці

3.5.2. Сценарії інтеграції у практичні середовища

На рис.3.13 продемонстровано приклад перевірки URL-адреси, яка належить до вузла SOM із середнім ризиком 5.18%, що демонструє, як метод автоматично класифікує адресу як легітимну на основі статистики вузла SOM. Такий механізм дозволяє швидко оцінювати ризик нових URL-адрес та інтегрувати SOM у різні сценарії практичного застосування.

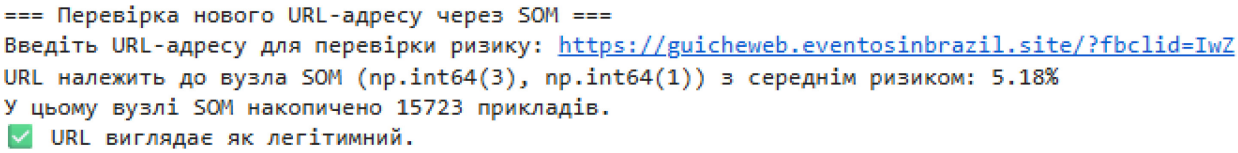


Рисунок 3.13 – Приклад 1 – Тестування на легітимність URL-адреси

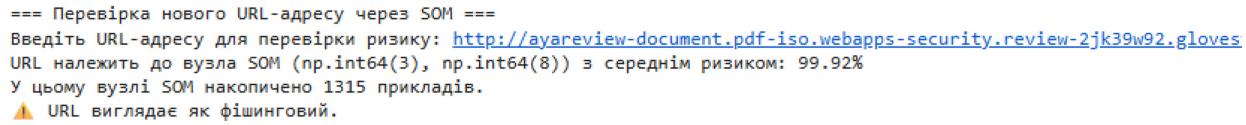


Рисунок 3.14 – Приклад 2 – Тестування на легітимність URL-адреси

Онлайн застосування розробленоого методу передбачає інтеграцію SOM у якості підмодуля у системи реальному часі у фільтри веб-трафіку, email-шлюзи або браузерні плагіни. На відміну від більшості існуючих класифікаторів, які обмежуються статичною перевіркою списків або застосуванням лінійних моделей для швидкої класифікації, SOM у поєднанні з вузловим ризиком дозволяє врахувати не лише індивідуальні характеристики URL, а й його належність до кластеру, в якому можуть накопичуватися шаблони атак, що забезпечує більш гнучке визначення порогу спрацювання: наприклад, у вузлах із середнім ризиком понад 70% можна застосовувати суворіші правила блокування, тоді як у «безпечних» вузлах SOM можна використовувати додаткові перевірки. Така гібридна стратегія дозволяє адаптувати метод до реального трафіку та мінімізувати хибнопозитивні спрацьовування – проблему, яка є однією з головних у системах кіберзахисту.

Офлайн застосування методу у якості додаткового функціоналу передбачає її використання для періодичного аналізу історичних логів веб-трафіку або архівів кіберзагроз. На відміну від типових офлайн-систем, що часто застосовують лише глобальні статистичні методи без урахування локальних залежностей між ознаками, розроблений підхід із SOM дозволяє дослідити динаміку накопичення ризику у вузлах та виявити нетипові шаблони, які могли б бути пропущені іншими моделями. Наприклад, якщо новий кластер SOM різко збільшив кількість підозрілих адрес, це може сигналізувати про появу нової фішингової кампанії.

Унікальність пропонованої інтеграції полягає у поєднанні високої чутливості моделей класифікації (LR та GMDH) із можливістю SOM досліджувати приховані взаємозв’язки між ознаками та відображати ризики у просторі вузлів.

ВИСНОВКИ ДО РОЗДІЛУ 3

У розділі 3 досліджено реалізацію методу класифікації та кластеризації фішингових та легітимних URL-адрес на основі поєднання моделей логістичної регресії, GMDH та Self-Organizing Map (SOM). Детально проаналізовано етапи формування навчального корпусу даних, генерації портретів URL-адрес, а також побудови та навчання моделей класифікації. Особливу увагу приділено використанню алгоритму SOM як інструмента для дослідження структури даних у багатовимірному просторі ознак та побудови теплових карт ризику.

У результаті дослідження SOM виявив вузли з високою концентрацією фішингових шаблонів, що дало змогу сформувати теплові карти ризику та визначити зони потенційної небезпеки для користувачів. Моделі логістичної регресії та GMDH продемонстрували високу точність класифікації (більше 97% F1-score), що свідчить про їхню ефективність у виявленні фішингових атак. Статистичний аналіз ознак дозволив виділити ключові характеристики URL-адрес, які найбільше впливають на результати класифікації, та показав важливість багатофакторного підходу до побудови системи виявлення.

Підсумки дослідження підтвердили, що інтеграція SOM із механізмом розрахунку середнього ризику у вузлах дозволяє не лише групувати URL-адреси за ознаками, але й виявляти приховані високоризикові шаблони, що суттєво підвищує точність виявлення фішингових атак. Запропоновані механізми локального уточнення класифікації та підтримка автономного аналізу через CLI демонструють гнучкість методу та її здатність до адаптації під реальні сценарії використання.

Розроблена архітектура методу показала свою конкурентоспроможність завдяки здатності ефективно аналізувати як статичні дані (офлайн), так і поточний трафік (онлайн), а також завдяки можливості динамічного оновлення знань у інтегровані системи на основі фідбеку користувача.

ВИСНОВКИ

У кваліфікаційній роботі було проведено комплексне дослідження проблеми виявлення фішингових URL-адрес у цифровому середовищі та побудовано архітектуру методу виявлення на основі принципу самоорганізації. Робота поєднує теоретичний аналіз актуальних кіберзагроз та їхньої динаміки, побудову структурно-семантичної моделі URL-адреси (портрета), розробку методу на основі класифікації та пояснення рішень, а також його тестування у практичному середовищі.

У першому розділі було розглянуто сучасні фішингові стратегії в контексті еволюції кіберзагроз, підкреслено роль фішингу як багаторівневої атаки, що поєднує технічні маніпуляції з соціальною інженерією. Було показано, що сучасні фішингові атаки демонструють високий рівень динаміки та гнучкості, активно використовують маскування через псевдодомени, редиректи, використання TLD із низьким рівнем перевірки, а також впроваджують механізми імітації легітимних сервісів. Ретельно проаналізовано формальну структуру URL-адреси (протокол, домен, піддомени, шлях, параметри запиту, фрагменти) та виділено ознаки, які можуть бути індикаторами фішингових шаблонів. Особливу увагу приділено виявленню закономірностей у фішингових шаблонах (наявність ключових слів, кількість піддоменів, TLD із підвищеним ризиком), що дозволило обґрунтувати потребу у векторному поданні адреси (портреті) як єдиній системній сутності для подальшої обробки.

Другий розділ був присвячений побудові багаторівневої архітектури методу класифікації URL-адрес на основі принципу самоорганізації. Було запропоновано модульну архітектуру методу, що складається з парсера, генератора портретів, модулів попередньої обробки та класифікаційних моделей. Особливу увагу приділено створенню уніфікованої портретної моделі URL-адреси, яка інтегрує структурні, технічні та семантичні ознаки у вектор фіксованої довжини (12 ознак), що забезпечує повну інтерпретованість та сумісність з різними алгоритмами класифікації. Було детально досліджено та

реалізовано два рівні класифікації: базову модель логістичної регресії для швидкої і пояснюваної оцінки ризику, а також модель GMDH як самоорганізовану систему для виявлення прихованих нелінійних залежностей між ознаками. Під час дослідження встановлено, що GMDH дозволяє адаптувати модель до нових шаблонів фішингу без повного перенавчання та має гнучку інтерпретацію через деревоподібну структуру поліномів другого порядку. Додатково застосовано Self-Organizing Map (SOM) для виявлення прихованої структури ризику у багатовимірному просторі ознак, що дало змогу визначити «гарячі точки» фішингової активності та візуалізувати ризики у вигляді теплових карт.

У третьому розділі було детально розглянуто практичну реалізацію методу в середовищі Python: імпорт та маркування даних із відкритих джерел (PhishTank, Alexa), побудову портретів URL-адрес, попередню обробку ознак (нормалізація, бінаризація, категоріальне кодування), а також навчання та тестування моделей. Було проведено налаштування порогів класифікації для досягнення оптимального балансу між precision та recall (F1-score понад 97%). У моделі GMDH реалізовано агрегацію вузлів та вилучення надлишкових зв’язків, що дозволило уникнути перенавчання та покращити узагальнюваність моделі.

На основі отриманих результатів були розроблені рекомендації для інтеграції методу у реальні середовища кіберзахисту: використання SOM як модуля моніторингу «гарячих зон» ризику у веб-трафіку; інтеграція моделі GMDH як резервного механізму уточнення рішень при низькій впевненості базової моделі; застосування портретної моделі як універсального інтерфейсу для різних джерел URL-адрес (email, веб-фільтри, DNS). Також було запропоновано гнучку методику налаштування порогів класифікації, що дозволяє адаптувати метод до різних сценаріїв використання (корпоративний фільтр, SOC, DLP тощо).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Toliupa S., Buchyk S., Shabanova A., Buchyk O. (2023) The Method for Determining the Degree of Suspiciousness of a Phishing URL. X International Scientific Conference "Information Technology and Implementation" (IT&I-2023), Workshop Proceedings (IT&I-WS 2023), Kyiv, Ukraine, November 20-21, 2023, pp. 239-247. URL: https://ceur-ws.org/Vol-3646/Short_5.pdf. (ISSN 1613-0073, Vol-3646, URN: urn:nbn:de:0074-3646-1). (дата доступу: 04.05.2025)

2. Buchyk S., Shabanova A. Detecting Phishing URLs Using Machine Learning and Clustering Algorithms. Information Technology and Implementation (Satellite): Conference Proceedings, November 21, 2024, Kyiv, Ukraine / Ministry of Education and Science of Ukraine, Taras Shevchenko National University of Kyiv and [etc]; Vitaliy Snytyuk (Editor). – Kyiv: Publishing House «Caravela», 2024. pp. 112-113.

3. CERT-UA. Урядова команда CERT-UA в 2023 році опрацювала 2543 кіберінциденти. URL: https://cip.gov.ua/en/news/uryadova-komanda-cert-ua-v-2023-roci-opracyuvala-2543-kiberincidenti (дата доступу: 04.05.2025).

4. European Union Agency for Cybersecurity (ENISA). ENISA Threat Landscape 2023. URL: https://www.enisa.europa.eu/publications/enisa-threat-landscape-2023 (дата доступу: 04.05.2025).

5. Google Threat Analysis Group. TAG Bulletin: Q4 2023. URL: https://blog.google/threat-analysis-group/tag-bulletin-q4-2023/ (дата доступу: 05.05.2025).

6. CERT-UA Reports: 11 Ukrainian Telecom Providers Hit by Phishing Attacks. URL: https://thehackernews.com/2023/10/cert-ua-reports-11-ukrainian-telecom.html (дата доступу: 05.05.2025).

7. State Service of Special Communications and Information Protection of Ukraine. Russia's Cyber Tactics H1'2023. URL: https://cip.gov.ua/services/cm/api/attachment/download?id=60068 (дата доступу: 06.05.2025).

8. Google Threat Analysis Group. Ukraine remains Russia's biggest cyber focus in 2023. URL: https://blog.google/threat-analysis-group/ukraine-remains-russias-biggest-cyber-focus-in-2023/ (дата доступу: 04.05.2025).

9. CyberPilot. ENISA 2023 Threat Landscape Report: Key Insights. URL: https://www.cyberpilot.io/cyberpilot-blog/enisa-2023-threat-landscape-report-key-insights (дата доступу: 05.05.2025).

10. The Hacker News. CERT-UA – Latest News, Reports & Analysis. URL: https://thehackernews.com/search/label/CERT-UA (дата доступу: 05.05.2025).

11. Security Affairs. CERT-UA reports of attacks in March 2025 targeting Ukrainian agencies with WreckSteel malware. URL: https://securityaffairs.com/176181/cyber-warfare-2/cert-ua-reports-attacks-in-march-2025-targeting-ukrainian-agencies-with-wrecksteel-malware.html (дата доступу: 05.05.2025).

12. Lastdrager, E. E. H. Achieving a consensual definition of phishing based on a systematic review of the literature. *Crime Science*, 3(9), 2014. URL: https://link.springer.com/article/10.1186/s40163-014-0009-y (дата доступу: 06.05.2025).

13. Ghazi-Tehrani, A. K., & Pontell, H. N. Phishing Evolves: Analyzing the Enduring Cybercrime. *Victims & Offenders*, 16(3), 316-342, 2021. URL: https://www.utica.edu/academic/institutes/cimip/Phishing_Evolves_Analyzing-the-Enduring-Cybercrime.pdf (дата доступу: 06.05.2025).

14. Gupta, B. B., Arachchilage, N. A. G., & Psannis, K. E. Defending against Phishing Attacks: Taxonomy of Methods, Current Issues and Future Directions. *arXiv preprint*, arXiv:1705.09819, 2017. URL: https://arxiv.org/abs/1705.09819 (дата доступу: 06.05.2025).

15. Lain, D., Kostiainen, K., & Capkun, S. Phishing in Organizations: Findings from a Large-Scale and Long-Term Study. *arXiv preprint*, arXiv:2112.07498, 2021. URL: https://arxiv.org/abs/2112.07498 (дата доступу: 07.05.2025).

16. Bitaab, M., Cho, H., Oest, A., et al. Scam Pandemic: How Attackers Exploit Public Fear through Phishing. *arXiv preprint*, arXiv:2103.12843, 2021. URL: https://arxiv.org/abs/2103.12843 (дата доступу: 07.05.2025).

17. Butavicius, M., Parsons, K., Pattinson, M., & McCormac, A. Breaching the Human Firewall: Social engineering in Phishing and Spear-Phishing Emails. *arXiv preprint*, arXiv:1606.00887, 2016. URL: https://arxiv.org/abs/1606.00887 (дата доступу: 07.05.2025).

18. Vajratiya V., Gupta B. B., Gaurav A. Mutual information based logistic regression for phishing URL detection. Cyber Security and Applications, 2, 100044. URL: https://www.sciencedirect.com/science/article/pii/S277291842400010-9 (дата доступу: 16.05.2025).

19. Lim B., Huerta R., Sotelo A., Quintela A., Kumar P. EXPLICATE: Enhancing phishing detection through explainable AI and LLM-powered interpretability. *arXiv*, 22.03.2025. URL: https://arxiv.org/abs/2503.20796 (дата доступу: 17.05.2025).

20. Dag O., Yozgatligil C. Architecture of the GMDH algorithm: Principles and layer-wise design. *ResearchGate*, 2025. URL: https://www.researchgate.net/publication/ (дата доступу: 05.05.2025).

21. Guérin A., Chauvet P., Saubion F. A survey on recent advances in Self-Organizing Maps. *arXiv*, 2024. URL: https://arxiv.org/abs/2501.08416 (дата доступу: 17.05.2025).

22. Fan X., Zhang S., Xue X. An improved Self-Organizing Map based on virtual winning neurons. *Symmetry, 17*(3), 449. URL: https://doi.org/10.3390/sym17030449 (дата доступу: 17.05.2025).

23. Hawas A. Y., Al-Shaher M. A. A novel deep SOM and hierarchical clustering for content recommendation. *BIO Web of Conferences, 97*, 00088. URL: https://www.bioconferences.org/articles/bioconf/abs/2024/16/bioconf_iscku2024_0-0088/bioconf_iscku2024_00088.html (дата доступу: 18.05.2025).

24. PhishTank. Phishing URL repository and verification platform. URL: https://www.phishtank.com (дата доступу: 27.05.2025).

25. Yasin A., Fatima R., Khan J. A., Afzal W. Behind the Bait: Delving into PhishTank's hidden data. Data in Brief, 52, 109959. DOI: https://doi.org/10.1016/j.dib.2023.109959.

26. Uddin M. A., Sarker I. H. An explainable transformer-based model for phishing email detection: A Large Language Model approach. arXiv preprint, 21.02.2024. URL: https://arxiv.org/abs/2402.13871 (дата доступу: 27.05.2025).

27. Alhuzali A., Alloqmani A., Aljabri M., Alharbi F. In-Depth analysis of phishing email detection: evaluating ML and DL across multiple datasets. Applied Sciences, 15(6), 3396. DOI: https://doi.org/10.3390/app15063396.

28. Al-Fayoumi M., Alhijawi B., Al-Haija Q. A., Armoush R. XAI-PhD: Fortifying trust of phishing URL detection empowered by Shapley additive explanations. iJOE, 20(11), 80–101. DOI: https://doi.org/10.3991/ijoe.v20i11.49533.

29. Islam M. R., Islam M. M., Afrin M. S., Antara A., Tabassum N., Amin A. PhishGuard: A CNN-based model for detecting phishing URLs with explainability. arXiv preprint, 27.04.2024. URL: https://arxiv.org/abs/2404.17960 (дата доступу: 29.05.2025).

Додаток А

Таблиця А.1

Каталог фішингових шаблонів URL

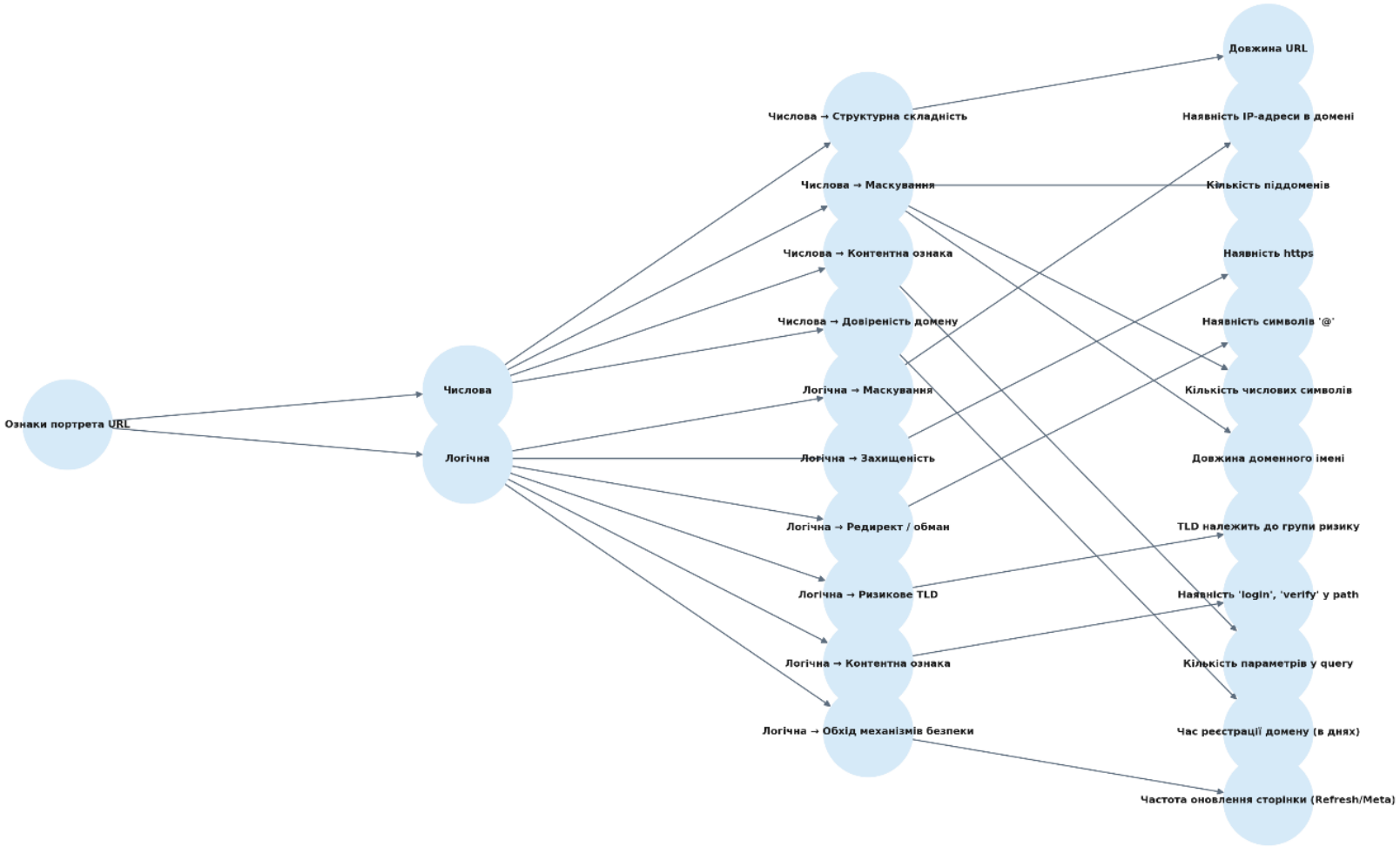
ID	Фішинговий URL	Тип атаки	Канал доставки	Цільова тематика	Маскування
PH-001	http://login-verification.bankoftrust.info/secure/login.php	Credential Harvesting	Email	Банківська безпека	Субдомен бренду + без SSL
PH-002	https://update.appleid-check-internal.com/auth	Credential Harvesting	SMS	Apple ID / iCloud	SSL + довгий TLD
PH-003	http://reset-mail.mfa-ua.com/index.html	Fake Webmail Portal	Email	Державна пошта	Домен із geo-кодом
PH-004	http://paypal-update-secure.net/confirm	Spear Phishing (Government Clone)	Email	Кабінет міністрів	Схожість з gov.ua
PH-005	http://paypal-update-secure.net/confirm	Payment Data Theft	Push Notification	PayPal / банківські картки	Небезпечний .net

Продовження таблиці А.1

PH-006	https://webmail.kyivnet-support.biz/login	Webmail Spoofing	Email	Пошта	Псевдо-піддомен
PH-007	http://covid19-check.healthgov.info/urgent-alert	Health Alert Scam	Messenger Bot	COVID-19	Схожість з .gov
PH-008	https://bankid.nova-poshta.help/login-check	Logistics Spoofing	Email	Нова Пошта / BankID	Доповнення .help
PH-009	http://e-service.registry-gov-ua.info/update	Gov Service Clone	Email	Єдиний реєстр	Тематичне слово .info
PH-010	https://onlinesign.ciocusign-ua.org/docview	Document Signature Spoof	Email	Документообіг / підпис	Домен із geo-префіксом

Додаток Б

Повне дерево ознак портрета URL з категоріями та типами



Додаток В

Еталонна таблиця чітких портретів URL

1	url	type	url_length	num_subdomains	https	tld	login	verify	update	token	email	path_length	query_length	num_eq_signs
2	http://indonesianherbal.com/inc/myaccount/signin/?cou	phishing	73	1	0	com	0	0	0	0	0	22	23	2
3	http://www.pictouanglicans.ca/dropox	phishing	36	2	0	ca	0	0	0	0	0	7	0	0
4	http://bethel.vn/wp-content/themes/twentytwelve/b.ph	phishing	54	1	0	vn	0	0	0	0	0	37	0	0
5	http://gv-arquitectos.com/themes/seven/wait.php	phishing	47	1	0	com	0	0	0	0	0	22	0	0
6	http://kiuc.tk/redirect.php?url=http://www.taktcoop.cor	phishing	104	1	0	tk	0	0	0	0	0	13	76	1
7	https://www.allposters.com/-st/Frederick-Augustus-Sanc	legitimate	76	2	1	com	0	0	0	0	0	50	0	0
8	https://www.thehauntedtrails.com/	legitimate	33	2	1	com	0	0	0	0	0	1	0	0
9	https://www.veromi.com/Erik-Hoffmann.aspx	legitimate	41	2	1	com	0	0	0	0	0	19	0	0
10	https://www.kinogb.club	legitimate	23	2	1	club	0	0	0	0	0	0	0	0
11	https://www.jjipu.blogspot.com/2009/04/smart-taxes-sr	legitimate	83	3	1	com	0	0	0	0	0	53	0	0

Додаток Д

ПОБУДОВА НАВЧАЛЬНОГО КОРПУСУ ТА МАРКУВАННЯ ДАНИХ

Фрагмент датасету для аналізу (URL-адреси та маркування)

URL dataset.csv

	A	B	C	D	E	F	G	H	I
278885	https://www.ratemds.com/doctor-ratings/54198/Dr-Jean-Pierre-Gascon-St-Sauveur-QC.html	legitimate							
278886	https://www.ratemds.com/doctor-ratings/54277/QC/Pierrefonds/Mercier	legitimate							
278887	https://www.ratemds.com/doctor-ratings/56320/Dr-Pierre-Gauthier-Laval-QC.html	legitimate							
278888	https://www.ratemds.com/doctor-ratings/56596/Dr-Sylvain-Proulx-Dolbeau-Mistassini-QC.html	legitimate							
278889	https://www.ratemds.com/doctor-ratings/56677/QC/Montreal/Gratton	legitimate							
278890	https://www.ratemds.com/doctor-ratings/56763/Dr-Marla-Shapiro-Toronto-ON.html	legitimate							
278891	https://www.ratemds.com/doctor-ratings/57610/QC/Montreal/Gosselin	legitimate							
278892	https://www.ratemds.com/doctor-ratings/58658/QC/St.-Thomas-D'Aquin/Blouin	legitimate							
278893	https://www.ratemds.com/doctor-ratings/58961/Dr-Michel-Therrien-Joliette-QC.html	legitimate							
278894	https://www.ratemds.com/doctor-ratings/59187/QC/Montreal/Luu	legitimate							
278895	https://www.ratemds.com/doctor-ratings/61042/QC/Grenfield-Park/Boux	legitimate							
278896	https://www.ratemds.com/doctor-ratings/61424/Dr-Elise-Benoit-Montreal-QC.html	legitimate							
278897	https://www.ratemds.com/doctor-ratings/64312/QC/St.-Sauveur-Des-Monts/Gascon	legitimate							
278898	https://www.ratemds.com/doctor-ratings/69096/ON/Ottawa/Bourque	legitimate							
278899	https://www.ratemds.com/doctor-ratings/75353/Dr-Michel-Lafond-Shawinigan-QC.html	legitimate							
278900	https://www.ratemds.com/doctor-ratings/82392/Dr-Meyer-Balter-Toronto-ON.html	legitimate							
278901	https://www.ratemds.com/doctor-ratings/873152/Dr-Jacques-Houde-Quebec-QC.html	legitimate							
278902	https://www.ratemds.com/doctor-ratings/879283/Dr-Michel-Gravel-Montreal-QC.html	legitimate							
278903	https://www.ratemds.com/doctor-ratings/879588/Dr-Richard-Begg-Ottawa-ON.html	legitimate							
278904	https://www.ratemds.com/doctor-ratings/891684/Dr-Paul-Marston-Butler-NJ.html	legitimate							
278905	https://www.ratemds.com/doctor-ratings/89364/QC/St.-Thomas-D'Aquin/Lachance	legitimate							
278906	https://www.ratemds.com/doctor-ratings/89476/Dr-Johanne-Rioux-Longueuil-QC.html	legitimate							
278907	https://www.ratemds.com/doctor-ratings/89712/Dr-GeneviÃ¨ve-Simard-Racine-Amqui-QC.html	legitimate							
278908	https://www.ratemds.com/doctor-ratings/90208/ON/Hamilton/Lamy	legitimate							
278909	https://www.ratemds.com/doctor-ratings/90215/Dr-Lloyd-Semelhago-Hamilton-ON.html	legitimate							
278910	https://www.ratemds.com/doctor-ratings/93739/Dr-%C3%89ric-Charbonneau-Qu%C3%A9bec-(	legitimate							
278911	https://www.ratemds.com/filecache/SelectDoctor.jsp?sid=53&searchBy=DSpecialty&letter=G&:	legitimate							
278912	https://www.ratemds.com/filecache/SelectDoctor.jsp?sid=55&searchBy=DLName&letter=B	legitimate							

URL dataset

Продовження додатку Д

Фрагмент коду для імпорту даних та маркування URL-адрес

```
# Імпорт бібліотек для роботи з даними
import pandas as pd
from sklearn.preprocessing import LabelEncoder

# Завантаження датасету з URL-адресами та мітками класів
data = pd.read_csv('URL dataset.csv')  # вказати правильний шлях до CSV-файлу

# Перевірка структури даних (опційно)
print("\nПерші 5 рядків датасету:")
print(data.head())

# Кодування міток класів у числовий формат:
# 'legitimate' -> 0, 'phishing' -> 1
label_encoder = LabelEncoder()
data['target'] = label_encoder.fit_transform(data['type'])

# Перевірка результату кодування
print("\nПриклади закодованих міток:")
print(data[['url', 'type', 'target']].head())
```

Продовження додатку Д

Фрагмент коду для генерації ознакових портретів URL-адрес

```
# Імпорт бібліотек для обробки URL та даних
from urllib.parse import urlparse
import pandas as pd
import whois

# Функція для екстракції ознак із URL-адреси
def extract_features(url):
    try:
        parsed = urlparse(url)
        features = {}

        # Структурні та семантичні ознаки
        features['url_length'] = len(url)
        features['num_subdomains'] = url.count('.') - 1
        features['has_https'] = int(parsed.scheme == 'https')
        features['contains_login'] = int('login' in url.lower())
        features['contains_token'] = int('token' in url.lower())
        features['contains_verify'] = int('verify' in url.lower())
        features['contains_update'] = int('update' in url.lower())
        features['contains_email'] = int('email' in url.lower())
        features['path_length'] = len(parsed.path)
        features['query_length'] = len(parsed.query)
        features['num_eq_signs'] = parsed.query.count('=')

        # Ознака tld_code для врахування доменів верхнього рівня
        tld = parsed.netloc.split('.')[-1]
        popular_tlds = ['com', 'org', 'net', 'gov', 'edu']
        features['tld_code'] = 0 if tld in popular_tlds else 1

        # Додавання віку домену через WHOIS
        try:
            domain_info = whois.whois(parsed.netloc)
            if isinstance(domain_info.creation_date, list):
                creation_date = domain_info.creation_date[0]
            else:
                creation_date = domain_info.creation_date
            if creation_date:
                domain_age = (pd.Timestamp.now() - pd.to_datetime(creation_date)).days
                features['domain_age_days'] = domain_age
            else:
                features['domain_age_days'] = 0
```

```
        except:
            # Якщо WHOIS-запит не вдався – встановлюємо значення за замовчуванням
            features['domain_age_days'] = 0
        return features

    except:
        # Якщо парсинг не вдався – повертаємо нульові значення для всіх ознак
        return {key: 0 for key in [
            'url_length', 'num_subdomains', 'has_https', 'contains_login',
            'contains_token', 'contains_verify', 'contains_update', 'contains_email',
            'path_length', 'query_length', 'num_eq_signs', 'tld_code', 'domain_age_days'
        ]}

# Генерація портретів для всіх URL-адрес у датасеті
feature_list = [extract_features(url) for url in data['url']]
features_df = pd.DataFrame(feature_list)

# Перегляд прикладів сформованих портретів (опційно)
print("\nПриклад сформованих портретів URL-адрес:")
print(features_df.head())
```

Продовження додатку Д

Фрагмент коду для формування навчальної та тестової вибірок

```
from sklearn.model_selection import train_test_split

# Масив ознак features_df та цільові значення у сформовані на попередніх етапах
# Масштабовані дані знаходяться у змінній X_scaled
# Мітки класів – у змінній y

# Поділ даних на тренувальну та тестову вибірки
# Стратифікація забезпечує збереження пропорційного співвідношення класів у вибірках
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled,
    y,
    test_size=0.3,          # 30% даних – для тестування
    random_state=42,        # забезпечує відтворюваність експериментів
    stratify=y              # гарантує пропорційність класів
)

# Перевірка розподілу класів у тренувальній та тестовій вибірках
print("\n=== Розподіл класів у тренувальній вибірці ===")
print(y_train.value_counts(normalize=True))
print("\n=== Розподіл класів у тестовій вибірці ===")
print(y_test.value_counts(normalize=True))
```

Додаток Е

Фрагмент коду логістичної регресії на портретах URL

```
param_grid = {
    'C': [0.01, 0.1, 1, 10],
    'solver': ['lbfgs', 'liblinear']
}
grid_search = GridSearchCV(LogisticRegression(max_iter=1000), param_grid, cv=StratifiedKFold(5))
grid_search.fit(X_train, y_train)
best_lr = grid_search.best_estimator_

y_pred_lr = best_lr.predict(X_test)
accuracy_lr = accuracy_score(y_test, y_pred_lr)
precision_lr = precision_score(y_test, y_pred_lr)
recall_lr = recall_score(y_test, y_pred_lr)
f1_lr = f1_score(y_test, y_pred_lr)
roc_auc_lr = roc_auc_score(y_test, best_lr.predict_proba(X_test)[:, 1])
conf_matrix_lr = confusion_matrix(y_test, y_pred_lr)

print("\n=== Logistic Regression with GridSearchCV Results ===")
print(f"Best params: {grid_search.best_params_}")
print(f"Accuracy: {accuracy_lr:.4f}")
print(f"Precision: {precision_lr:.4f}")
print(f"Recall: {recall_lr:.4f}")
print(f"F1-Score: {f1_lr:.4f}")
print(f"ROC AUC Score: {roc_auc_lr:.4f}")
print("Confusion Matrix:")
print(conf_matrix_lr)

# Графік вагових коефіцієнтів для Logistic Regression
plt.figure(figsize=(10, 8))
plt.barh(selected_feature_names, best_lr.coef_[0])
plt.xlabel("Ваговий коефіцієнт")
plt.ylabel("Ознаки (Feature Selection)")
plt.title("Вагові коефіцієнти Logistic Regression")
plt.grid(True)
plt.tight_layout()
plt.show()
```

Додаток Ж

Фрагмент коду в рамках GMDH-моделі

```
poly = PolynomialFeatures(degree=2, include_bias=False)
X_train_poly = poly.fit_transform(X_train)
X_test_poly = poly.transform(X_test)
gmdh_model = LogisticRegression(max_iter=1000)
gmdh_model.fit(X_train_poly, y_train)
y_pred_gmdh = gmdh_model.predict(X_test_poly)

accuracy_gmdh = accuracy_score(y_test, y_pred_gmdh)
precision_gmdh = precision_score(y_test, y_pred_gmdh)
recall_gmdh = recall_score(y_test, y_pred_gmdh)
f1_gmdh = f1_score(y_test, y_pred_gmdh)
roc_auc_gmdh = roc_auc_score(y_test, gmdh_model.predict_proba(X_test_poly)[:,
1])
conf_matrix_gmdh = confusion_matrix(y_test, y_pred_gmdh)

print("\n=== GMDH (Polynomial) Model Results ===")
print(f"Accuracy: {accuracy_gmdh:.4f}")
print(f"Precision: {precision_gmdh:.4f}")
print(f"Recall: {recall_gmdh:.4f}")
print(f"F1-Score: {f1_gmdh:.4f}")
print(f"ROC AUC Score: {roc_auc_gmdh:.4f}")
print("Confusion Matrix:")
print(conf_matrix_gmdh)

# Топ-20 коефіцієнтів
feature_names_poly = poly.get_feature_names_out(selected_feature_names)
coefficients = gmdh_model.coef_[0]
coef_df = pd.DataFrame({
    'feature': feature_names_poly,
    'coefficient': coefficients,
    'abs_coefficient': np.abs(coefficients)
})
top_coef_df = coef_df.sort_values('abs_coefficient', ascending=False).head(20)
plt.figure(figsize=(10, 8))
plt.barh(top_coef_df['feature'], top_coef_df['coefficient'])
plt.xlabel("Ваговий коефіцієнт")
plt.ylabel("Поліноміальні ознаки (Топ-20)")
plt.title("Найважливіші ознаки для GMDH (Polynomial)")
plt.grid(True)
plt.tight_layout()
plt.show()
```

Додаток З

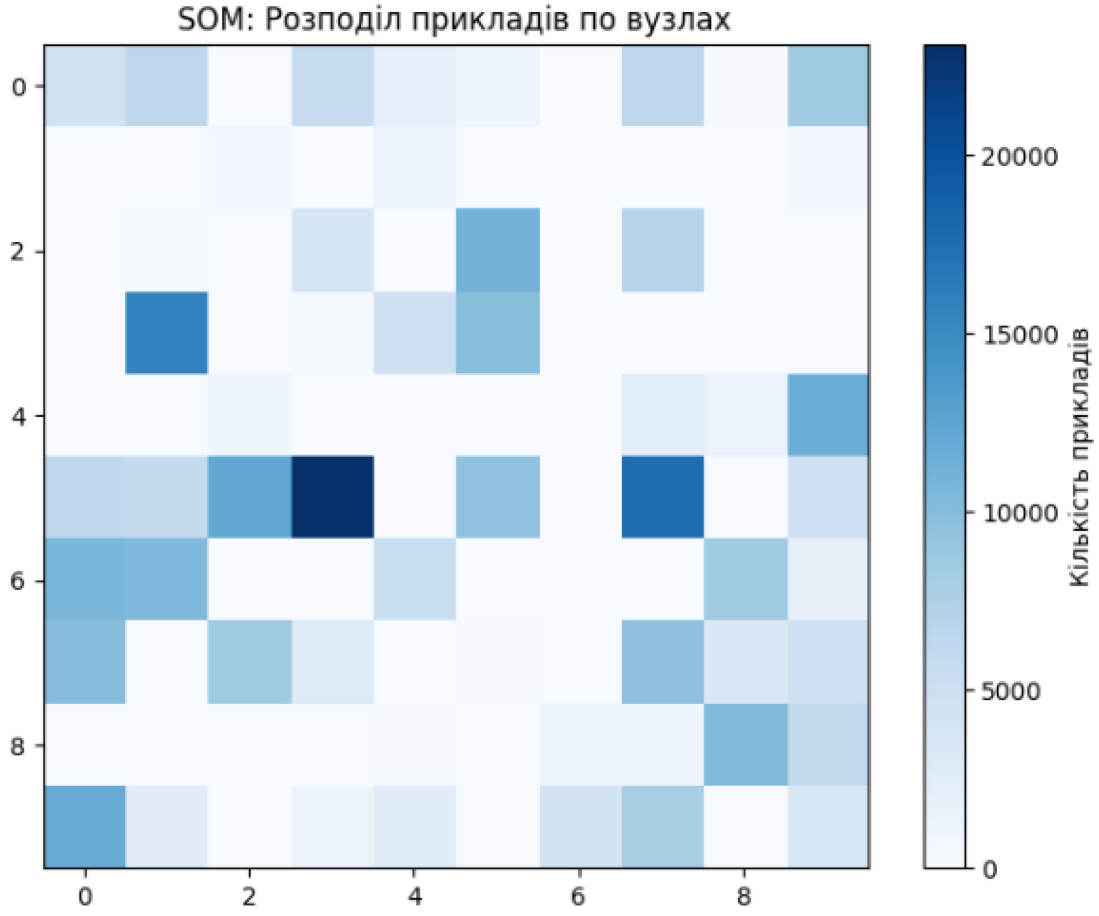
Фрагмент коду в рамках SOM

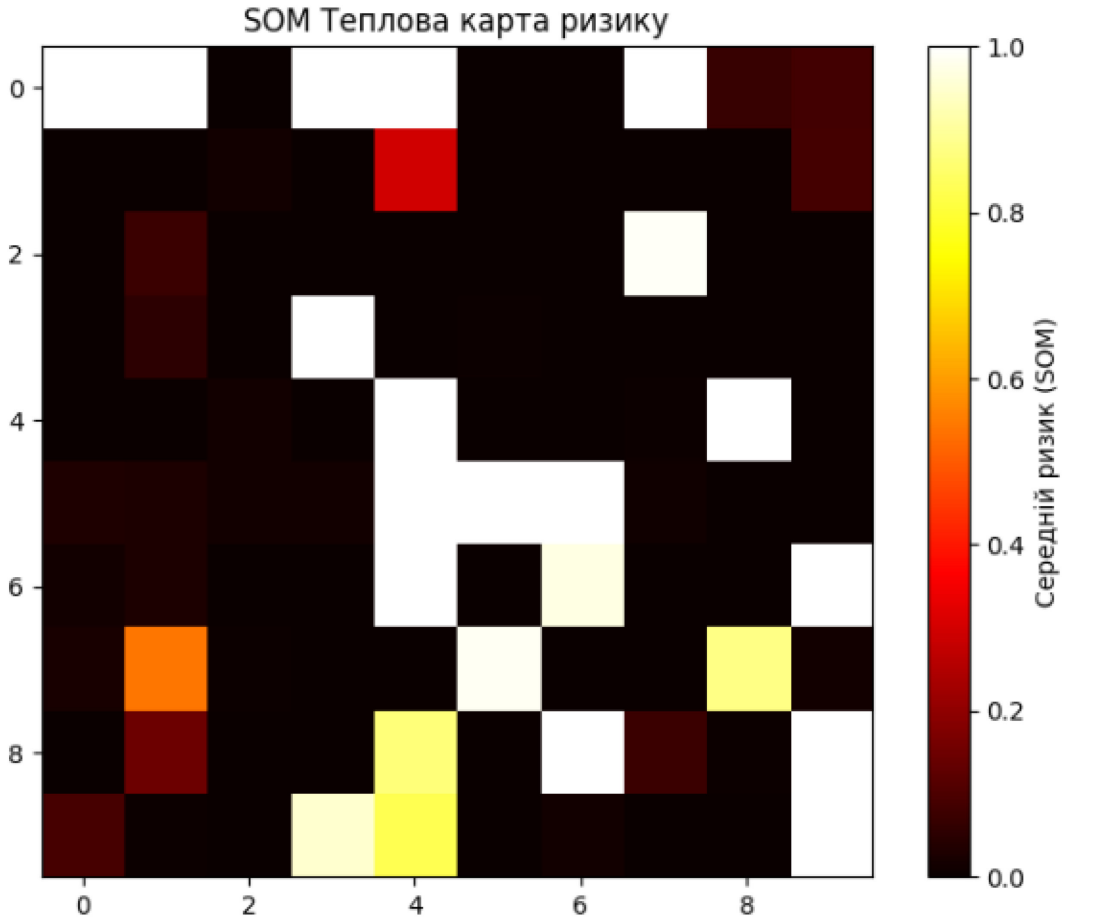
```
som = MiniSom(x=10, y=10, input_len=X_train.shape[1], sigma=1.0, learning_rate=0.5)
som.random_weights_init(X_train)
som.train_random(X_train, num_iteration=100)

risk_map = np.zeros((10, 10))
counts_map = np.zeros((10, 10))
for i, x in enumerate(X_train):
    w = som.winner(x)
    risk_map[w] += y_train.iloc[i]
    counts_map[w] += 1

avg_risk_map = np.divide(risk_map, counts_map, out=np.zeros_like(risk_map), where=counts_map != 0)
print("\n=== SOM ===")
print("\nКількість прикладів у кожному вузлі SOM:")
print(counts_map)
risk_percentage_map = np.divide(risk_map, counts_map, out=np.zeros_like(risk_map), where=counts_map != 0)
risk_percentage_map *= 100  # у відсотках

# Додатковий аналіз SOM
print("\nВідсоток ризику у вузлах SOM (нормований):")
risk_percentage_map = np.divide(risk_map, counts_map, out=np.zeros_like(risk_map), where=counts_map != 0)
risk_percentage_map *= 100
risk_percentage_map = np.round(risk_percentage_map, 2)
print(risk_percentage_map)

# Топ-5 вузлів SOM із найвищим ризиком
risk_flat = avg_risk_map.flatten()
sorted_indices = np.argsort(risk_flat)[::-1]
top_nodes = np.column_stack(np.unravel_index(sorted_indices[:5], avg_risk_map.shape))

print("\nТоп-5 вузлів SOM із найвищим ризиком:")
for node in top_nodes:
    risk_value = avg_risk_map[node[0], node[1]]
    print(f"Вузол: {tuple(node)}, Середній ризик: {risk_value:.4f}")

plt.figure(figsize=(8, 6))
```

Продовження додатку З

```
plt.imshow(counts_map, cmap='Blues', interpolation='nearest')
plt.colorbar(label='Кількість прикладів')
plt.title("SOM: Розподіл прикладів по вузлах")
plt.show()
print("\n======")
plt.figure(figsize=(8, 6))
plt.imshow(avg_risk_map, cmap='hot', interpolation='nearest')
plt.colorbar(label='Середній ризик (SOM)')
plt.title("SOM Теплова карта ризику")
plt.show()

# Додатковий аналіз даних
print("\n=== Додаткова аналітика ===")
print("\nРозподіл класів:")
print(y.value_counts())
print("\nКількість TLD:")
print(features_df['tld_code'].value_counts())
features_df['target'] = y
print("\nСередні значення ознак для фішингових URL-адрес:")
print(features_df[features_df['target'] == 1].mean())
print("\nСередні значення ознак для легітимних URL-адрес:")
print(features_df[features_df['target'] == 0].mean())

print("\n=== Перевірка нового URL-адресу через SOM ===")
new_url = input("Введіть URL-адресу для перевірки ризику: ")
new_features_dict = extract_features(new_url)
new_features_list = [new_features_dict[feat] for feat in features_df.columns if feat !=
'target']
new_features_scaled = scaler.transform([new_features_list])
new_features_selected = selector.transform(new_features_scaled)
node = som.winner(new_features_selected[0])
risk = avg_risk_map[node[0], node[1]] * 100
print(f"URL належить до вузла SOM {node} з середнім ризиком: {risk:.2f}%")
count_examples = counts_map[node[0], node[1]]
print(f"У цьому вузлі SOM накопичено {int(count_examples)} прикладів.")

# Припущення про легітимність
threshold = 70  # поріг ризику (можна змінити)
if risk >= threshold:
    print(" URL виглядає як фішинговий.")
else:
    print(" URL виглядає як легітимний.")
```

Додаток К


SOM: Розподіл прикладів по вузлах
0
2
4
6
8
20000
15000
10000
5000
0
Кількість прикладів


SOM Теплова карта ризику
0
2
4
6
8
1.0
0.8
0.6
0.4
0.2
0.0
Середній ризик (SOM)

Додаток Л

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

ДИПЛОМ

І ступеня

НАГОРОДЖУЄТЬСЯ

Студентка 3 курсу бакалаврату
Факультету інформаційних технологій

ШАБАНОВА
Анастасія Олексіївна

за перемогу у І турі Всеукраїнського конкурсу
студентських наукових робіт
з галузей знань і спеціальностей
у 2023/2024 навчальному році

Проректор
з наукової роботи — Ганна ТОЛСТАНОВА

Додаток М

COIL | Certified Online International Learning

Acknowledgment

THIS IS TO CERTIFY THAT

Anastasiia Shabanova – Workshop Trainer

contributed significantly to the successful organization and execution of Certified Online International (COIL) Module on Cyber Risks and Business Intelligence, held by KNU - AUE from April 25 to May 22, 2025.

Her invaluable contributions and commitment ensured the smooth operation and effectiveness of this educational initiative.

Taras Shevchenko National University of Kyiv, UA
Vice-Rector for International Cooperation
Prof. Kseniia Smyrnova

American University in the Emirates, UAE
President and CEO
Prof. Muthanna Abdul Razzaq