Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(освітній рівень)

на тему: "Методи та засоби захисту інформаційно−комунікаційної системи об'єкту інформаційної діяльності від внутрішніх загроз"

Виконав: студент (ка)

Спеціальності:
125 «Кібербезпека та захист інформації»
(шифр і назва напряму підготовки, спеціальності)
Смик О.В.
підпис (прізвище та ініціали)

Керівник Кульчицький Т.Р.
підпис (прізвище та ініціали)

Нормоконтроль Тимощук Д. І.
підпис (прізвище та ініціали)

Завідувач кафедри Загородна Н.В.
підпис (прізвище та ініціали)

Рецензент Лещишин Ю.З.
підпис (прізвище та ініціали)

м. Тернопіль – 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека та захист інформації
(шифр і назва спеціальності)

Студенту Смику Олегу Васильовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Методи та засоби захисту інформаційно−комунікаційної системи об'єкту інформаційної діяльності від внутрішніх загроз

Керівник роботи Кульчицький Тарас Русланович, доктор філософії, старший викладач кафедри КБ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «20» 11 2024 року № 4/7-1112

2. Термін подання студентом завершеної роботи 13.12.2024

3. Вихідні дані до роботи Друковані та онлайнові джерела за тематикою роботи, технічна документація.

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. Розділ 1 Аналіз сучасних методів виявлення внутрішніх загроз, основних фактрів ризику та прграмних рішень для моніторингу аномальної поведінки користувачів. Розділ 2 Розробка інформаційної технології виявлення внутрішніх загроз. Розділ 3. Програмна реалізація та тестування технології виявлення внутрішніх загроз. Розділ 4. Охорона праці та безпека в надзвичайних ситуаціях. Висновки. Список використаних джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1 Тема, автор та науковий керівник кваліфікаційної роботи. 2 Актуальність, об'єкт, предмет дослідження, Мета та задачі кваліфікаційної роботи. 3 Особистісні фактори виникнення внутрішніх загроз. Основні соціальні, організаційні та технічні фактори. 4 Програмні рішення для моніторингу та виявлення аномальної поведінки користувачів. 5 Порівняння методів машинного навчання. 6 Схема роботи методу швидкого дерева рішень. 7 Розподіл основних атрибутів у наборі даних. 8 Приклади тренувальних взірців та результати навчання моделі. 9 Метрики моделі по кожному класу, матриця помилок. 10 Результати тестування моделі на сценарії виявлення загрози. 11 Висновки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Г.М., к.т.н., доцент	10.12.2024	10.12.2024
Безпека в надзвичайних ситуаціях	Клепчик В.М., проректор з адміністративно-господарської роботи та будівництва	11.12.2024	11.12.2024

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення із завданням до кваліфікаційної роботи		Виконано
2.	Підбір та опрацювання джерел за тематикою кваліфікаційної роботи		Виконано
3.	Огляд методів та зособів для виявлення внутрішніх загроз, дослідження факторів ризику, що сприяють внутрішнім загрозам		Виконано
4.	Аналіз існуючих програмних рішень для моніторингу та виявлення аномальної поведінки користувачів		Виконано
5.	Оформлення першого розділу		Виконано
6.	Оцінка та вибір методів машинного навчання для виявлення потенційних загроз		Виконано
7.	Розробка математичної моделі виявлення внутрішніх загроз, оцінка точності та ефективності моделі у виявленні загроз		Виконано
8.	Оформлення другого розділу		Виконано
9.	Реалізація програмного модуля для класифікації дій користувачів та допоміжних компонентів для системи моніторингу та сповіщення		Виконано
10.	Тестування системи в умовах, що моделюють внутрішні загрози, та оцінка роботи системи		Виконано
11.	Оформлення третього розділу та розділу «Охорона праці та безпека в надзвичайних ситуаціях		Виконано
12.	Оформлення кваліфікаційної роботи та додатків		Виконано
13.	Нормоконтроль		Виконано
14.	Перевірка на плагіат		Виконано
15.	Попередній захист кваліфікаційної роботи		Виконано
	Захист кваліфікаційної роботи		

Студент _____ Смик О.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Кульчицький Т.Р.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Методи та засоби захисту інформаційно-комунікаційної системи об'єкту інформаційної діяльності від внутрішніх загроз // ОР «Магістр» // Смик Олег Васильович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2024 // С. 108, рис. – 35, табл. – 16, кресл. – 11, додат. – 2.

У кваліфікаційній роботі виконано проектування системи для виявлення внутрішніх загроз у інформаційно-комунікаційній системі, зокрема розроблено математичну модель класифікації дій користувачів та реалізовано програмне рішення на основі алгоритмів машинного навчання. Проведено аналіз сучасних методів і технологій виявлення загроз, обґрунтовано вибір алгоритму швидкого дерева рішень для класифікації аномальної поведінки та набору даних «Data Leakage Detection» для навчання і тестування моделі.

У ході виконання роботи реалізовано програмний модуль класифікації, включаючи компоненти для навчання, зберігання та завантаження моделі, а також функціонал шифрування даних та авторизації користувачів. Проведено тестування системи в умовах, що моделюють внутрішні загрози, та оцінено її ефективність. Отримані результати підтверджують здатність системи ідентифікувати аномальну активність із високим рівнем точності.

Розроблена система забезпечує захист інформації від внутрішніх загроз, інтегруючи механізми автоматичного моніторингу та аналізу активності користувачів.

Проведено оцінку вимог охорони праці та заходів безпеки у надзвичайних ситуаціях для забезпечення надійної експлуатації системи.

Ключові слова: Внутрішні загрози, Машинне навчання, Швидке дерево рішень, Класифікація, Аномальна поведінка, Інформаційно-кумунфікаційна система, Захист даних.

ABSTRACT

Methods and Means of Protecting the Information-Communication System of an Information Activity Object from Internal Threats // Thesis of educational level "Master"// Oleh Smyk // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, group СБм-61 // Ternopil, 2024 // P. 108, figs. – 35, tables – 16, drawings – 11, added. – 2.

In the qualification work, the design of the system for detecting internal threats in the information-communication system was performed, in particular, a mathematical model for the classification of user actions was developed and a software solution based on machine learning algorithms was implemented. An analysis of modern threat detection methods and technologies was conducted, the choice of a fast decision tree algorithm for the classification of anomalous behavior and the "Data Leakage Detection" data set for model training and testing was substantiated.

In the course of the work, a classification software module was implemented, including components for learning, storing and loading the model, as well as the functionality of data encryption and user authorization. The system was tested in conditions simulating internal threats, and its effectiveness was evaluated. The obtained results confirm the ability of the system to identify anomalous activity with a high level of accuracy.

The developed system provides protection of information from internal threats, integrating mechanisms of automatic monitoring and analysis of user activity.

An assessment of labor protection requirements and safety measures in emergency situations to ensure reliable operation of the system was carried out.

Keywords: Insider Threats, Machine Learning, Fast Decision Tree, Classification, Abnormal Behavior, Information-Communication System, Data protection.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ВЛ	–	Випадковий ліс
ГБ	–	Градієнтний бустинг
ЕМВ	–	Електромагнітне випромінювання
ІКС	–	Інформаційно-комунікаційні системи
ШДР	–	Швидке дерево рішень
MFA	–	Multi-Factor Authentication (Багатофакторна автентифікація)
PIN	–	Personal Identification Number (Персональний ідентифікаційний номер)
UBA	–	User Behavior Analytics (Аналіз поведінки кристувачів)

ВСТУП

Сучасні інформаційно-комунікаційні системи (ІКС) стали невід'ємною складовою функціонування різноманітних організацій, установ та підприємств, забезпечуючи доступ до інформаційних ресурсів і підтримку процесів, пов'язаних з обробкою даних. Водночас зі зростанням значення ІКС зростає і ризик їхнього вразливого становища перед внутрішніми загрозами, які часто залишаються непоміченими або недооціненими. Внутрішні загрози можуть бути пов’язані з діяльністю співробітників, підрядників або партнерів, які мають певний рівень доступу до інформації й систем. Це особливо актуально для об'єктів інформаційної діяльності, де обробляються конфіденційні дані, а також дані, що мають високу цінність або можуть призвести до значних збитків у разі їхнього витоку.

Важливим фактором, що впливає на захист ІКС від внутрішніх загроз, є зростання складності та інтеграції сучасних систем, що робить їх вразливими до цілеспрямованих атак, обхідних шляхів та потенційних несанкціонованих дій. Наприклад, використання слабких паролів, порушення регламентів роботи з конфіденційною інформацією, а також відсутність належного моніторингу активності користувачів – усе це може сприяти виникненню та реалізації внутрішніх загроз. Це підтверджує необхідність комплексного підходу, що поєднує технологічні рішення з організаційними заходами, такими як обмеження доступу, впровадження багатофакторної автентифікації, постійний моніторинг дій користувачів та виявлення аномалій у їхній поведінці.

Актуальність теми обумовлена необхідністю забезпечення високого рівня захищеності інформаційних ресурсів у таких системах, а також зростанням числа випадків внутрішніх порушень безпеки. Враховуючи високу динаміку розвитку інформаційних технологій та збільшення залежності організацій від ІКС, внутрішні загрози становлять дедалі більшу небезпеку. Інформаційні ресурси є стратегічно важливими для будь-якої організації, а витік або пошкодження критичної інформації через недбалість або зловмисні дії співробітників може призвести до фінансових втрат, втрати репутації або навіть до правових

наслідків. У зв’язку з цим розробка та впровадження ефективних методів захисту ІКС є невід'ємною частиною сучасної практики забезпечення інформаційної безпеки.

Мета дослідження – створення технології для виявлення внутрішніх загроз в інформаційно-комунікаційних системах об'єктів інформаційної діяльності з використанням методів машинного навчання, яка дозволяє моніторити та оцінювати поведінку користувачів для запобігання потенційним порушенням безпеки.

Завдання дослідження :

а) провести аналіз сучасних методів виявлення внутрішніх загроз, основних факторів ризику та програмних рішень для моніторингу аномальної поведінки користувачів;

б) розробити математичну модель класифікації дій користувачів для оцінки ризику, визначивши метрики для оцінки точності та ефективності моделі у виявленні загроз;

в) реалізувати програмну систему для виявлення внутрішніх загроз, протестувати її в умовах, що моделюють внутрішні загрози, та оцінити результати роботи системи для подальшого вдосконалення.

Об’єктом дослідження є інформаційно-комунікаційні системи об’єктів інформаційної діяльності, в яких існує ризик виникнення внутрішніх загроз, що можуть призвести до порушення цілісності, доступності або конфіденційності критично важливих даних.

Предметом дослідження є методи та засоби виявлення і запобігання внутрішнім загрозам в інформаційно-комунікаційних системах, зокрема технології моніторингу поведінки користувачів та моделі оцінки ризиків, що базуються на методах машинного навчання.

Наукова новизна одержаних результатів полягає у вдосконаленні методів виявлення внутрішніх загроз в інформаційно-комунікаційних системах об’єктів інформаційної діяльності шляхом застосування моделей машинного навчання для моніторингу та оцінки поведінки користувачів. У роботі розроблено комплексну модель для виявлення аномальної поведінки користувачів, яка

враховує різноманітні фактори ризику, типи доступу та динаміку активності всередині системи. Це забезпечує більш ефективне виявлення потенційних загроз за рахунок аналізу поведінкових патернів та адаптивної класифікації дій користувачів. Використання запропонованого підходу дозволяє в реальному часі виявляти відхилення у діях співробітників, що потенційно можуть призвести до порушень безпеки, і сприяє оперативному прийняттю рішень щодо запобігання можливим інцидентам.

Практичне значення одержаних результатів полягає у можливості їх застосування для посилення захисту інформаційно-комунікаційних систем від внутрішніх загроз в організаціях, які обробляють критичні дані. Розроблена система моніторингу та виявлення аномальної поведінки користувачів може бути інтегрована в існуючі інфраструктури безпеки, що дозволить знижувати ризики, пов'язані з несанкціонованим доступом або зловмисними діями співробітників.

Запропонований підхід також надає можливість моделювати різні сценарії поведінки в системі, що може використовуватися для покращення політик доступу і безпеки в різноманітних організаціях. Окрім того, результати дослідження можуть бути корисними для розробників засобів інформаційної безпеки, надаючи основу для створення нових програмних рішень у сфері захисту інформаційних систем. Запропонована система може бути адаптована до різних типів організацій, сприяючи підвищенню рівня безпеки та зниженню потенційних збитків від інцидентів, пов'язаних із внутрішніми загрозами.

Апробація результатів магістерської роботи.

Основні результати проведених досліджень обговорювались на XII науково-технічній конференції «Інформаційні моделі, системи та технології» (м. Тернопіль, Україна).

Публікації. Основні результати кваліфікаційної роботи опубліковано у працях конференції (див. Додаток А).

РОЗДІЛ 1 АНАЛІЗ СУЧАСНИХ МЕТОДІВ ВИЯВЛЕННЯ ВНУТРІШНІХ ЗАГРОЗ, ОСНОВНИХ ФАКТОРІВ РИЗИКУ ТА ПРОГРАМНИХ РІШЕНЬ ДЛЯ МОНІТОРИНГУ АНОМАЛЬНОЇ ПОВЕДІНКИ КОРИСТУВАЧІВ

1.1 Огляд сучасних методів для виявлення внутрішніх загроз у інформаційних системах

Внутрішні загрози в інформаційних системах є об’єктом активного наукового дослідження та становлять одну з основних загроз для безпеки сучасних організацій. Їхня специфіка полягає у виникненні загроз із середовища самої організації, де діють особи, які мають законний доступ до інформаційних ресурсів. Ці загрози можуть проявлятися як у вигляді навмисних дій, спрямованих на завдання шкоди або отримання несанкціонованої вигоди, так і у формі ненавмисних помилок, що призводять до порушення конфіденційності, цілісності чи доступності інформації [1].

Окрім того, технічні збої або людські помилки можуть також спричиняти серйозні наслідки, що робить важливим впровадження надійних засобів захисту на всіх рівнях доступу. Водночас, як зазначають Кульчицький Т.Р. та інші науковці [2], недостатня адаптивність правової бази України до сучасних викликів ускладнює реалізацію таких заходів, залишаючи вразливості у функціонуванні ІКС.

До прикладів навмисних внутрішніх загроз належать дії незадоволених співробітників, які, використовуючи свої повноваження, намагаються порушити цілісність даних або знищити інформаційні активи організації [3]. Зокрема, менеджер із доступом до бази даних клієнтів може здійснити копіювання цієї бази перед звільненням та використати отримані відомості у власних бізнес-інтересах, що є прямим порушенням конфіденційності та може спричинити значні фінансові та репутаційні втрати для підприємства. Такі випадки є особливо небезпечними, оскільки внутрішні особи добре обізнані з архітектурою системи, політиками безпеки та вразливими точками організації.

Не менш значущими є ненавмисні дії, що виникають через неуважність або низький рівень кваліфікації персоналу. Наприклад, адміністратор може випадково надати надмірні права доступу до конфіденційних даних або видалити критично важливі файли системи, що негативно позначається на функціонуванні інформаційної інфраструктури. Інший приклад подібної загрози – відправлення внутрішніх документів несанкціонованим особам внаслідок помилок при пересиланні електронної пошти. Таким чином, навіть незначні помилки здатні створити потенційну загрозу інформаційній безпеці, що вимагає посилення контролю та впровадження механізмів превентивного захисту.

З огляду на це, внутрішні загрози є багатогранною проблемою, яка вимагає системного підходу до управління інформаційною безпекою. Вони охоплюють як технічні, так і соціотехнічні аспекти, що ускладнює їх своєчасне виявлення та запобігання негативним наслідкам. Основна проблема внутрішніх загроз полягає в прихованості та складності їхньої ідентифікації через легітимний доступ співробітників до інформаційних ресурсів, що не завжди можна ефективно контролювати засобами традиційних методів захисту [4].

У зв’язку з цим застосування сучасних методів поведінкового аналізу, моніторингу подій та контролю прав доступу стає необхідною складовою комплексної системи захисту. Такі методи дають змогу визначити аномалії у звичній поведінці користувачів, оцінити ризики зловживання правами доступу та забезпечити оперативний контроль за активністю в інформаційній системі. Завдяки поєднанню аналітичних підходів і технологій машинного навчання організація може побудувати систему захисту, яка здатна адаптуватися до нових типів загроз та забезпечувати високий рівень безпеки.

Нижче наведено огляд сучасніших підходів, алгоритмів та технологій, які застосовуються для виявлення внутрішніх загроз в інформаційних системах.

Методи аналізу поведінки користувачів (User Behavior Analytics, UBA) є інструментами, спрямованими на виявлення потенційно небезпечних або нетипових дій у межах інформаційної системи шляхом дослідження звичок і патернів поведінки користувачів. Їх використовують для ідентифікації аномалій у поведінці співробітників, які можуть свідчити про внутрішні загрози, такі як

спроби несанкціонованого доступу або ексфільтрація даних [5]. Системи, що базуються на аналізі поведінки, застосовуються переважно в корпоративних середовищах, де критично важливо захистити конфіденційну інформацію. Вони дають змогу формувати профілі користувачів на основі зібраних даних про їхню діяльність у системі, а також відстежувати та порівнювати нові дії з типовою поведінкою кожного користувача. Це дозволяє знизити кількість хибних спрацьовувань і підвищити точність виявлення потенційних загроз, створюючи систему, яка автоматично реагує на аномалії та повідомляє про них відповідальних осіб.

У таблиці 1.1 зібрано основні параметри для аналізу поведінки користувачів та відповідні методи їх застосування в системах безпеки. Ця структура демонструє ключові аспекти поведінкового аналізу, які використовуються для ідентифікації аномальної активності.

Таблиця 1.1 – Методи аналізу поведінки користувачів

Параметр поведінки	Метод аналізу	Опис методу	Сфери застосування
1	2	3	4
Активність входу в систему	Кластерний аналіз	Групування за частотою та часом входів для виявлення нетипової активності	Моніторинг логінів співробітників
Рівень доступу до файлів	Контроль прав доступу	Відстеження відповідності доступу користувача до його службових обов'язків	Захист конфіденційних даних
Відкриття конфіденційних даних	Поведінкове моделювання	Аналіз частоти та часу доступу до критичних файлів	Виявлення несанкціонованих дій
Часова активність	Аналіз часових рядів	Оцінка активності користувача у нетипові для нього години	Виявлення аномалій у розкладі роботи

Продовження таблиці 1.1

1	2	3	4
Копіювання та перенесення файлів	Системи моніторингу подій	Відстеження дій, пов’язаних з масовим копіюванням або переміщенням файлів	Запобігання ексфільтрації даних
Зміна налаштувань системи	Машинне навчання	Моделювання та ідентифікація змін конфігурацій, які не відповідають звичній поведінці	Контроль адміністративних дій
Використання зовнішніх носіїв	Аналіз подій із зовнішніми медіа	Виявлення активності, пов’язаної з використанням USB-накопичувачів	Захист від несанкціонованого копіювання

Використання методів аналізу поведінки користувачів дозволяє ефективно виявляти аномалії, що можуть свідчити про внутрішні загрози, знижуючи ймовірність несанкціонованих дій з боку співробітників. Застосування таких технологій, як кластерний аналіз, машинне навчання та моніторинг подій, створює потужний механізм контролю за активністю користувачів у реальному часі, що сприяє підвищенню рівня безпеки в організації.

Методи аналізу прав доступу дозволяють контролювати, хто і до яких ресурсів має доступ в організаційній системі. Вони спрямовані на запобігання перевищенню привілеїв користувачів, що може призвести до витоків інформації або випадків зловживання доступом. Основною метою аналізу прав доступу є створення безпечного середовища, де користувачі можуть взаємодіяти з інформаційними ресурсами відповідно до їхніх службових обов’язків, не порушуючи принципу мінімально необхідного доступу.

Дані методи застосовуються в різних сферах, зокрема в урядових організаціях, фінансових установах, великих корпораціях, де контроль за доступом до чутливих даних є критичним для забезпечення безпеки. У практичному застосуванні популярними підходами є рольове управління доступом, атрибутне управління доступом, принцип мінімальних привілеїв та аудит доступу, що дозволяють гнучко налаштовувати рівні доступу залежно від специфіки кожної організації.

У таблиці 1.2 згруповано основні методи аналізу прав доступу, що включають принципи функціонування кожного методу, їхні переваги та основні сфери застосування.

Таблиця 1.2 – Методи аналізу прав доступу

Метод	Принцип роботи	Переваги	Сфери застосування
Рольове управління	Доступ базується на ролях	Чітка структура доступу, зменшення ризику зловживань	Великі компанії, зокрема у сфері IT
Атрибутне управління	Доступ на основі атрибутів	Гнучкість, можливість адаптації до умов	Великі корпорації та урядові установи
Принцип мінімальних привілеїв	Мінімальний доступ для виконання завдань	Зниження ризику витоків, обмеження впливу порушень	Усі організації незалежно від розміру
Дискреційне управління	Користувачі можуть визначати доступ	Гнучкість, швидке налаштування прав	Малі компанії та стартапи
Мандатне управління	Доступ за жорсткими політиками	Високий рівень безпеки	Військові та урядові установи
Політично-орієнтоване управління	Доступ визначається динамічними умовами	Адаптивність до змінних вимог	Критичні системи, зокрема у банківській сфері
Аудит доступу	Регулярний моніторинг прав доступу	Зменшення ймовірності порушень, виявлення аномалій	Банки, фінансові установи
Автоматизоване управління доступом	Використання ШІ для аналізу прав	Швидкість і зниження людського фактору	Великі компанії з високими вимогами до безпеки

Використання методів аналізу прав доступу дозволяє організаціям забезпечувати надійний контроль за рівнями доступу користувачів до інформаційних ресурсів, мінімізуючи ризики перевищення прав і випадків несанкціонованого доступу. Такий підхід сприяє покращенню загальної інформаційної безпеки та забезпечує дотримання принципів конфіденційності, цілісності та доступності даних.

Аналіз соціальних факторів та емоційного стану є відносно новим напрямом у сфері інформаційної безпеки, який фокусується на дослідженні поведінкових аспектів, емоційного стану і соціального оточення співробітників для ідентифікації потенційних внутрішніх загроз. Такі методи базуються на припущенні, що певні соціальні та емоційні фактори можуть підвищувати ймовірність ризикованих або зловмисних дій з боку працівників. Зокрема, незадоволеність роботою, емоційне виснаження або тиск з боку соціального оточення можуть мотивувати співробітників до несанкціонованих дій. Ці методи використовуються в компаніях, де захист конфіденційної інформації є критично важливим, таких як фінансові установи, урядові структури та великі корпорації з високими вимогами до безпеки. Інструменти аналізу соціальних факторів можуть включати моніторинг внутрішньої комунікації, дослідження рівня задоволеності працівників, аналіз текстів електронної пошти на предмет емоційного стану та визначення соціальних взаємозв'язків у межах компанії. Це дозволяє виявляти можливі джерела загроз і запобігати потенційним інцидентам на ранньому етапі [6].

Таблиця 1.3 містить основні інструменти аналізу соціальних факторів та емоційного стану працівників, систематизуючи їхні характеристики та основні принципи роботи.

Таблиця 1.3 – Методи аналізу соціальних факторів та емоційного стану

Метод аналізу	Принцип роботи	Переваги	Сфери застосування
1	2	3	4
Аналіз текстів електронної пошти	Виявлення емоцій та настроїв	Оперативне виявлення змін у емоційному стані	Корпорації з високими вимогами до безпеки
Опитування рівня задоволеності	Оцінка задоволеності роботою	Раннє виявлення ризиків через незадоволеність	Великі компанії та державні установи
Соціальний аналіз комунікацій	Оцінка соціальних зв'язків у команді	Виявлення конфліктів та проблем у взаємодії	Команди з високим рівнем координації

Продовження таблиці 1.3

1	2	3	4
Моніторинг внутрішніх чатів	Відстеження тональності та тематики	Виявлення ознак демотивації або конфліктів	Фінансові установи, корпоративні структури
Аналіз використання соцмереж	Оцінка активності в соцмережах	Виявлення стресу або можливого порушення політик	Корпорації з жорсткими політиками щодо інформації
Оцінка емоційного виснаження	Опитування чи тестування працівників	Виявлення потенційних ризиків емоційного вигоряння	ІТ-компанії, компанії з високим рівнем навантаження
Перевірка зовнішніх контактів	Відстеження комунікацій поза компанією	Попередження несанкціонованих витоків	Державні установи та конфіденційні організації

Використання методів аналізу соціальних факторів та емоційного стану дозволяє своєчасно ідентифікувати можливі ризики, пов’язані з людським фактором, і вживати заходів для їхньої мінімізації. Ці підходи допомагають організаціям не лише покращити захист інформаційних ресурсів, а й підтримувати здорове психологічне середовище, що є важливим для загальної безпеки та продуктивності.

1.2 Дослідження основних характеристик та факторів ризику, що сприяють внутрішнім загрозам

Внутрішні загрози в інформаційних системах виникають через діяльність осіб, які мають легітимний доступ до інформаційних ресурсів організації, але можуть використовувати цей доступ для несанкціонованих або небажаних дій. Аналіз основних характеристик та факторів ризику, що сприяють виникненню внутрішніх загроз, є важливим кроком для формування ефективної системи захисту. До основних характеристик внутрішніх загроз належать, зокрема, мотиви, соціальні умови та поведінкові аспекти співробітників, які можуть впливати на ймовірність їхньої ризикованої поведінки.

Визначення факторів ризику є необхідним для розуміння глибинних причин, які можуть штовхати співробітників до зловмисних або небезпечних дій. Фактори ризику внутрішніх загроз умовно можна розділити на три основні категорії: особистісні фактори, соціальні та організаційні фактори, а також та технічні фактори.

Особистісні фактори є одним із важливих елементів, що підвищують ймовірність виникнення внутрішніх загроз в інформаційних системах. Вони пов'язані з емоційним станом, особистими обставинами та індивідуальними характеристиками співробітників, які мають доступ до ресурсів організації. Ці фактори можуть негативно впливати на поведінку співробітників та підвищувати їхню схильність до ризикованих або несанкціонованих дій. Наприклад, працівник, який відчуває тривалий стрес або емоційне виснаження, може почати нехтувати правилами безпеки, що створює ризик витоку інформації або порушення цілісності даних. Інший приклад – ситуація, коли співробітник зіштовхується з фінансовими труднощами, що підвищує ймовірність зловживання доступом до конфіденційної інформації для отримання додаткового прибутку. Особистісні фактори також охоплюють проблеми у міжособистісних стосунках, які можуть знижувати лояльність до компанії та створювати схильність до потенційно небезпечних дій.

На рис. 1.1 зображено основні особистісні фактори, що сприяють виникненню внутрішніх загроз.

Рисунок 1.1 – Особистісні фактори виникнення внутрішніх загроз

До основних особистісних факторів, які впливають на виникнення внутрішніх загроз, належать:

а) емоційне виснаження, спричинене надмірним навантаженням та стресом, що знижує здатність до самоконтролю;

б) невдоволення роботою, яке виникає через відсутність професійного розвитку чи незадовільні умови праці, що знижує мотивацію працівника;

в) фінансові проблеми, що можуть стимулювати співробітника до продажу конфіденційної інформації або інших видів зловживань;

г) психологічний стрес, пов'язаний з особистими або професійними проблемами, який може знижувати увагу до правил безпеки;

д) проблеми у міжособистісних стосунках, які спричиняють напруженість у колективі, знижуючи лояльність і підвищуючи ризик порушень;

е) низький рівень лояльності, що означає відсутність прихильності до компанії, що підвищує ймовірність несанкціонованих дій [7].

Перераховані особистісні фактори відіграють важливу роль у формуванні ризиків, пов'язаних із поведінкою співробітників, та можуть слугувати показниками можливих загроз. Оцінка цих факторів допомагає організаціям

своєчасно виявляти демотивованих чи емоційно виснажених працівників і запобігати потенційним ризикам, що сприяє підвищенню рівня безпеки в організації.

Соціальні та організаційні фактори є важливими компонентами, які впливають на виникнення внутрішніх загроз в інформаційних системах, оскільки вони стосуються соціального оточення співробітників, умов праці та загальної корпоративної культури. Негативний вплив соціальних і організаційних умов може створювати середовище, яке підвищує ризик виникнення небезпечної поведінки серед співробітників. Наприклад, надмірний робочий тиск або відсутність чітких політик безпеки можуть сприяти недотриманню правил доступу до конфіденційної інформації. Інший приклад – слабка комунікація між співробітниками та керівництвом, що може призводити до недостатньої обізнаності про важливість інформаційної безпеки. Конфлікти між працівниками або низький рівень підтримки з боку колег також можуть послаблювати лояльність і мотивацію працівників, що підвищує ризик порушення безпекових правил.

На рис. 1.2 зображено основні соціальні та організаційні фактори, які сприяють виникненню внутрішніх загроз.

Рисунок 1.2 – Основні соціальні та організаційні фактори

До основних соціальних та організаційних факторів, які впливають на ризик внутрішніх загроз, належать:

а) надмірний робочий тиск, що може призводити до емоційного виснаження та порушення правил безпеки;
б) недостатня підтримка колег, яка знижує мотивацію та прихильність до компанії;
в) слабка комунікація з керівництвом, що може призводити до нерозуміння важливості політик безпеки;
г) відсутність чітких політик безпеки, яка залишає простір для зловживань і неправильного трактування правил доступу;
д) конфлікти в колективі, які знижують рівень лояльності та підвищують ризик зловмисних дій;
е) недостатній рівень відповідальності за безпеку, що зменшує індивідуальну увагу до збереження конфіденційної інформації.

Перераховані соціальні та організаційні фактори можуть суттєво впливати на поведінку співробітників та формувати середовище, сприятливе для виникнення внутрішніх загроз. Аналіз цих факторів дозволяє організаціям виявляти слабкі місця в управлінні персоналом і корпоративною культурою, що сприяє зміцненню системи захисту та підвищенню загального рівня інформаційної безпеки [8].

Технічні фактори охоплюють широкий спектр питань, пов’язаних із технологічною інфраструктурою, засобами управління доступом і моніторингом активності користувачів у межах інформаційно-комунікаційної системи. Належна організація технічних заходів безпеки є критично важливою, оскільки навіть незначні упущення у цій сфері можуть створити серйозні ризики для конфіденційності, цілісності та доступності інформації. Зокрема, відсутність чіткого регламентування процесів управління доступом або неправильна конфігурація обладнання може дозволити несанкціонований доступ до чутливих даних.

Наприклад, якщо система розмежування прав доступу не враховує специфіку посадових обов’язків працівників, існує ризик, що співробітники, які не повинні мати доступ до конфіденційної інформації, отримають до неї необмежений доступ. Це може стосуватися випадків, коли працівник з відділу логістики має змогу переглядати фінансові звіти компанії або інженер отримує доступ до медичних записів пацієнтів у системі телемедицини. Така ситуація не лише створює потенційну загрозу витоку інформації, а й ускладнює виявлення відповідальних осіб у разі порушення.

Недостатньо налаштовані системи контролю активності користувачів також є серйозним недоліком. Відсутність інструментів для запису дій працівників або аналізу поведінкових патернів унеможливлює вчасне виявлення аномалій, таких як численні спроби доступу до заборонених ресурсів чи використання незвичних схем роботи. Наприклад, якщо працівник починає завантажувати великі обсяги даних у неробочий час, система моніторингу повинна автоматично сигналізувати про можливу загрозу. Без таких механізмів реагування організація залишається беззахисною перед інсайдерськими атаками.

До основних технічних факторів, які впливають на ризик внутрішніх загроз, належать:

а) відсутність розмежування прав доступу, що створює можливість для несанкціонованих дій;
б) недостатній рівень аудиту, що ускладнює виявлення порушень безпеки та оцінку правомірності дій;
в) відсутність інструментів моніторингу активності користувачів, що знижує здатність ідентифікувати аномалії у поведінці;
г) недостатній захист даних, що може призводити до витоку інформації або її пошкодження;
д) неналежне керування привілеями, що дозволяє користувачам отримувати більше доступу, ніж необхідно;
е) відсутність або неналежна налаштовуваність систем поведінкового аналізу, що ускладнює ідентифікацію небезпечної поведінки в режимі реального часу.

Наявність технічних факторів ризику може значно послаблювати загальну безпеку інформаційних систем, створюючи можливості для зловживань з боку внутрішніх користувачів [9]. Оцінка технічних факторів і впровадження необхідних засобів для контролю, моніторингу та обмеження доступу допомагають знижувати ризики, пов'язані з внутрішніми загрозами, забезпечуючи цілісність, конфіденційність та доступність інформаційних ресурсів організації.

Аналіз особистісних, соціальних та організаційних, а також технічних аспектів дозволяє створити комплексну систему захисту, що враховує всі потенційні ризики і передбачає механізми їх мінімізації. Це забезпечує більш надійний захист організації від внутрішніх загроз, зменшуючи ймовірність інцидентів, пов'язаних із зловживанням доступом або несанкціонованими діями співробітників.

1.3 Аналіз існуючих програмних рішень для моніторингу та виявлення аномальної поведінки користувачів

З розвитком інформаційних технологій та збільшенням обсягів чутливих даних, що зберігаються і обробляються в корпоративних інформаційних системах, зростає необхідність у програмних рішеннях, здатних ідентифікувати та запобігати внутрішнім загрозам. Програмні рішення для моніторингу та виявлення аномальної поведінки користувачів спрямовані на автоматичне виявлення підозрілої активності всередині системи, яка може свідчити про потенційно небезпечні дії співробітників або інших внутрішніх користувачів.

Такі системи зазвичай базуються на технологіях аналізу поведінкових патернів та алгоритмах машинного навчання, які дозволяють створювати профілі користувачів на основі їхньої типової поведінки. Ключова мета цих програмних рішень полягає у виявленні відхилень від звичної активності, що може свідчити про загрозу інформаційній безпеці, наприклад, спроби несанкціонованого доступу або масове копіювання конфіденційних даних.

Splunk User Behavior Analytics – це програмне рішення, призначене для виявлення аномальної поведінки користувачів та виявлення загроз у реальному часі (рисунок 1.3). Основна мета – забезпечити безпеку інформаційної системи, автоматично ідентифікуючи підозрілі дії користувачів та можливі внутрішні загрози. Splunk UBA використовує потужні алгоритми машинного навчання для аналізу поведінкових патернів користувачів і створює профілі типових дій для кожного користувача або об’єкта системи [10].

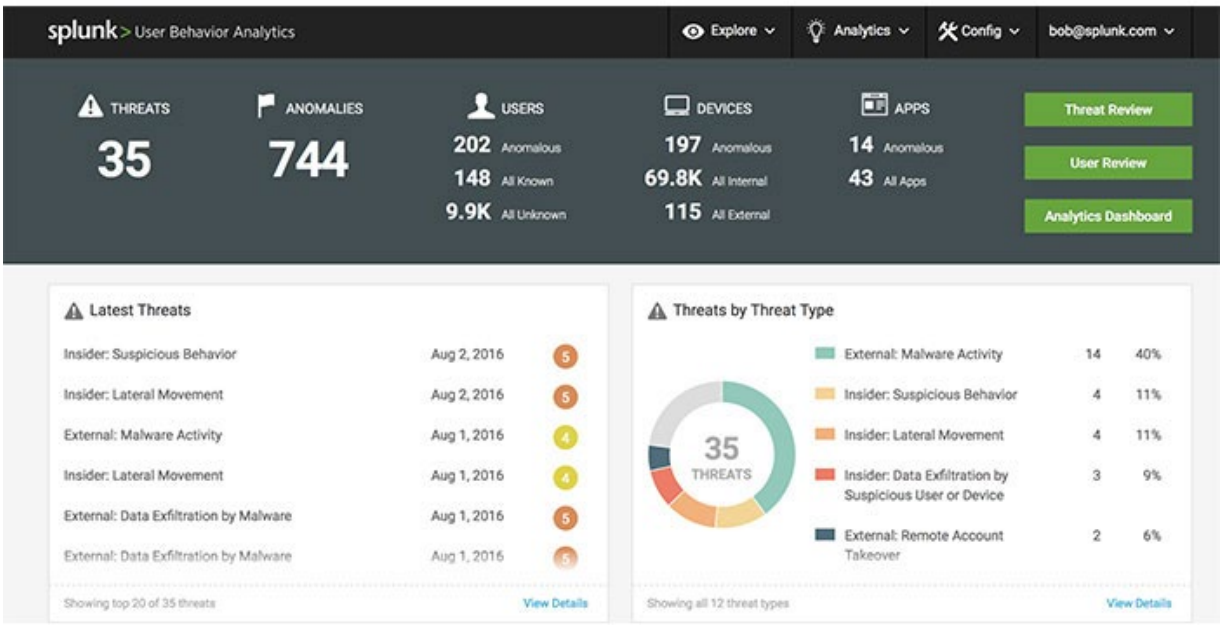

Рисунок 1.3 – Приклад інтерфейсу «Splunk User Behavior Analytics»

Архітектура Splunk UBA включає кілька ключових компонентів. Основу системи становить платформа Splunk, яка здійснює збір, зберігання та обробку даних у реальному часі, що надходять з різних джерел, таких як мережеві логи, журнали подій, файли аудиту та інші [11]. Потім ці дані передаються в аналітичний модуль, де застосовуються алгоритми машинного навчання та поведінкового аналізу для побудови профілів і виявлення аномалій. Крім того, UBA включає систему візуалізації та звітності, яка дозволяє фахівцям з безпеки переглядати детальну інформацію про виявлені загрози, відстежувати тенденції та оперативно реагувати на інциденти. Splunk UBA інтегрується з іншими

рішеннями для інформаційної безпеки, що дозволяє підвищити ефективність моніторингу й забезпечити всебічний захист інформаційних активів організації.

Переваги Splunk User Behavior Analytics:

а) автоматизоване виявлення загроз. Використання машинного навчання для автоматичного виявлення аномальної поведінки значно знижує навантаження на спеціалістів з безпеки;

б) інтеграція з іншими системами безпеки. Здатність інтегруватися з різними рішеннями для комплексного моніторингу підвищує загальний рівень захисту;

в) гнучка візуалізація даних. Зручні інструменти для візуалізації та аналітики надають можливість отримувати глибоке розуміння загроз і реагувати вчасно;

г) масштабованість. Платформа легко адаптується для організацій різного розміру, що дозволяє ефективно працювати як у невеликих, так і в великих корпоративних середовищах.

Недоліки Splunk User Behavior Analytics (UBA):

а) висока вартість. Рішення є досить дорогим, що може бути перешкодою для невеликих компаній;

б) складність налаштування. Для повноцінного використання можливостей Splunk UBA потрібні фахівці з достатньою технічною підготовкою;

в) великий обсяг ресурсів. Продукт вимагає значних обчислювальних ресурсів для обробки великих обсягів даних, що може ускладнювати його впровадження;

г) можливі хибні спрацьовування. Незважаючи на високий рівень точності, система все ще може генерувати певну кількість хибних спрацьовувань, що потребує додаткового ручного аналізу [12].

Отже, Splunk User Behavior Analytics – це потужний інструмент для моніторингу та виявлення аномальної поведінки користувачів, орієнтований на автоматизацію безпекових процесів і забезпечення високого рівня захисту. Програмне забезпечення пропонує гнучкі можливості аналізу та інтеграції, що

робить його особливо корисним для великих корпоративних середовищ з високими вимогами до інформаційної безпеки. Проте, через високу вартість і складність налаштування, Splunk UBA може бути менш доступним для невеликих організацій.

Varonis DatAdvantage – це програмне рішення, призначене для моніторингу, аналізу та захисту даних у корпоративних мережах (рисунок 1.4). Основна мета цього рішення – забезпечити захист конфіденційної інформації та запобігти витокам даних, надаючи інструменти для управління доступом та виявлення потенційних загроз.

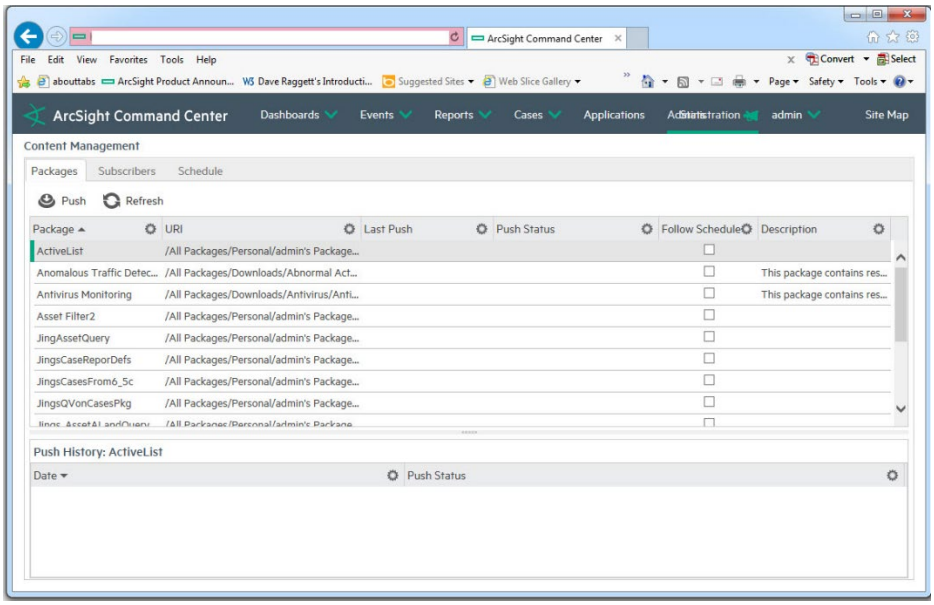

Рисунок 1.4 – Приклад інтерфейсу «Varonis DatAdvantage» [13]

Архітектура Varonis DatAdvantage складається з декількох ключових компонентів, які забезпечують ефективний моніторинг та аналіз. Центральний компонент – це система збору та обробки даних, яка інтегрується з файловими серверами, системами керування даними та іншими корпоративними джерелами, щоб отримувати інформацію про доступ і активність користувачів у реальному часі. Зібрані дані аналізуються за допомогою поведінкових моделей, які дозволяють визначати нетипову активність і потенційні загрози. Система також включає модуль управління доступом, який забезпечує точне розмежування прав доступу та дозволяє швидко реагувати на загрози. Інструменти візуалізації та звітності дають змогу адміністраторам переглядати детальну інформацію про

використання даних і загрози, а також отримувати рекомендації щодо оптимізації політик безпеки [14].

Переваги Varonis DatAdvantage:

а) інтеграція з різними джерелами даних. Програма може легко підключатися до різноманітних файлових систем та серверів, забезпечуючи комплексний контроль доступу;

б) автоматичний аналіз поведінки користувачів. Використання поведінкових моделей для визначення аномальної активності підвищує ефективність виявлення загроз;

в) деталізовані звіти та візуалізація. Інструменти для візуалізації надають глибокий аналіз активності користувачів, що спрощує моніторинг та управління безпекою;

г) гнучке управління правами доступу. Можливість точно налаштовувати права доступу для користувачів та груп забезпечує надійний захист конфіденційних даних.

Недоліки Varonis DatAdvantage:

а) висока вартість впровадження. Рішення є дорогим, що може бути непосильним для невеликих організацій;

б) складність налаштування. Потребує значних технічних знань для належного впровадження та обслуговування;

в) велике навантаження на систему. Обробка великих обсягів даних може уповільнювати роботу інших корпоративних систем;

г) обмежена підтримка неструктурованих даних. Хоча програма добре працює з файловими системами, її можливості обмежені щодо деяких інших джерел неструктурованих даних [15].

Отже, Varonis DatAdvantage – це потужне рішення для моніторингу, аналізу та управління доступом до конфіденційних даних, яке пропонує глибокий функціонал для виявлення загроз і забезпечення безпеки. Інтеграція з різноманітними системами, детальний аналіз поведінки користувачів і надійне управління правами доступу роблять його особливо корисним для великих організацій з високими вимогами до безпеки даних. Однак висока вартість та

складність налаштування можуть обмежити доступність цього рішення для менш масштабних підприємств.

Securonix UEBA (User and Entity Behavior Analytics) – це програмне рішення для виявлення та запобігання внутрішнім загрозам, яке аналізує поведінку користувачів і систем для автоматичного виявлення аномалій (рисунок 1.5). Призначення Securonix UEBA полягає в тому, щоб виявляти потенційні загрози на ранніх стадіях, забезпечуючи безпеку організації за рахунок моніторингу дій користувачів та системних об’єктів, таких як пристрої та додатки.

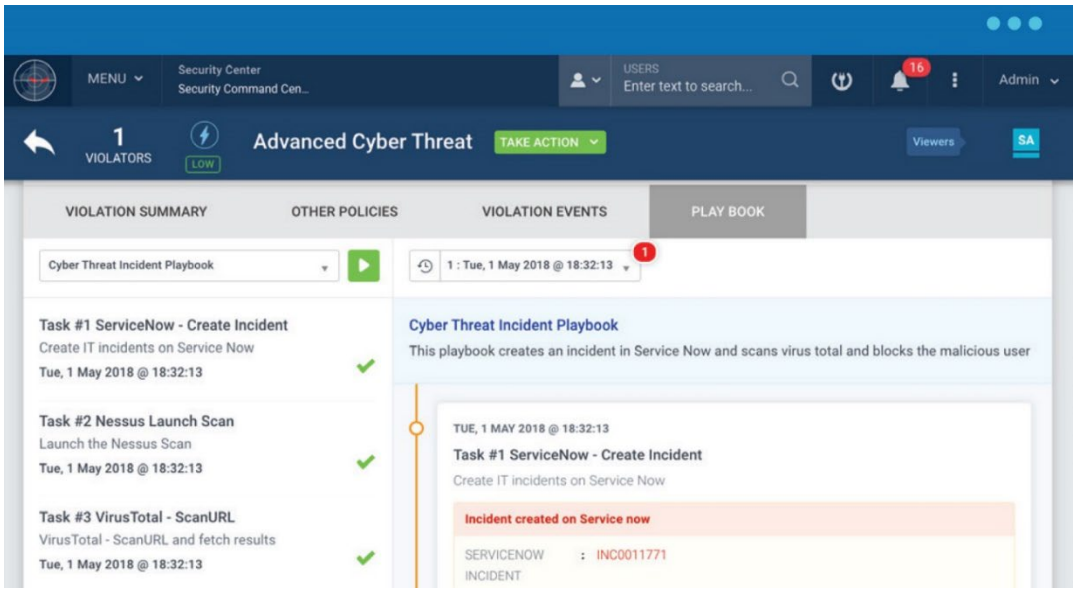

Рисунок 1.5 – Приклад інтерфейсу «Securonix UEBA» [16]

Архітектура Securonix UEBA складається з кількох ключових модулів, що працюють у комплексі для забезпечення ефективного моніторингу. Основою є аналітичний модуль, який інтегрується з різними джерелами даних, такими як системи журналів подій, мережеві пристрої та інші корпоративні додатки. Цей модуль отримує дані в реальному часі та використовує алгоритми машинного навчання для створення поведінкових профілів для кожного користувача та об'єкта. Інтегрована система обробки великих даних дозволяє аналізувати інформацію в режимі реального часу, відстежувати аномальні дії та відображати їх у зручному інтерфейсі для адміністраторів безпеки. Завдяки розширеній візуалізації та функціям звітності, Securonix UEBA надає аналітикам можливість

отримувати оперативну інформацію про загрози та швидко приймати рішення для їх усунення [17].

Переваги Securonix UEBA:

а) потужний аналіз на основі машинного навчання. Використання алгоритмів машинного навчання для аналізу поведінки дозволяє ефективно ідентифікувати складні загрози;

б) інтеграція з різними джерелами даних. Можливість підключення до широкого спектра корпоративних джерел забезпечує комплексний моніторинг;

в) масштабованість. Платформа здатна обробляти великі обсяги даних, що робить її придатною для великих організацій;

г) розширена візуалізація. Інтерфейс пропонує зручну візуалізацію аномалій та загроз, що спрощує процес прийняття рішень.

Недоліки Securonix UEBA:

а)висока вартість впровадження та обслуговування. Це рішення може бути занадто дорогим для невеликих компаній;

б) складність налаштування та підтримки. Система потребує досвідчених фахівців для належного налаштування та управління;

в) можливість хибних спрацьовувань. Незважаючи на високу точність, алгоритми можуть генерувати деякі хибні сповіщення;

г)вимогливість до обчислювальних ресурсів. Для обробки великих обсягів даних потрібна потужна інфраструктура [18].

Отже, Securonix UEBA є потужним інструментом для моніторингу поведінки користувачів та виявлення внутрішніх загроз, що базується на алгоритмах машинного навчання і забезпечує високий рівень адаптивності та масштабованості. Завдяки інтеграції з численними джерелами даних і розширеній візуалізації, цей додаток підходить для великих організацій із високими вимогами до безпеки. Проте, висока вартість і складність налаштування можуть обмежити його застосування для менших компаній.

Аналіз існуючих програмних рішень для моніторингу та виявлення аномальної поведінки користувачів показав, що більшість з них мають високий

рівень функціональності та надійності, однак часто відрізняються складністю налаштування, високою вартістю та потребують значних обчислювальних ресурсів. Це створює потребу в розробці нового рішення, яке б використовувало штучний інтелект для автоматизації аналізу поведінки, але мало спрощений функціонал і забезпечувало швидке та легке впровадження. Доступність базової безкоштовної версії дозволить залучити ширшу аудиторію, зокрема малий та середній бізнес, який не завжди має ресурси для дорогих систем. Таке рішення стане привабливим для організацій різного масштабу, адже поєднуватиме технологічну ефективність із доступністю й зручністю використання.

У рамках даного розділу було проведено всебічний аналіз предметної області з метою формулювання завдань для розробки інформаційної технології виявлення внутрішніх загроз. Зокрема, розглянуто сучасні методи і підходи до виявлення внутрішніх загроз в інформаційних системах, що охоплюють аналіз поведінки користувачів, управління правами доступу, а також дослідження соціальних і емоційних чинників, які можуть впливати на безпеку.

Визначено основні характеристики і фактори ризику, пов'язані з особистісними, соціальними, організаційними та технічними аспектами, які сприяють виникненню небезпечної поведінки співробітників. Проведений огляд існуючих програмних рішень, зокрема Splunk User Behavior Analytics, Varonis DatAdvantage і Securonix UEBA, виявив як їхні переваги, так і обмеження, що вказує на потребу у нових, більш доступних рішеннях. На основі цього аналізу сформульовано завдання розробки, що включають авторизацію користувачів, моніторинг поведінки, навчання та зберігання моделей машинного навчання, а також функції шифрування та фіксації подій.

Результати розгляду наданої інформації створюють ґрунтовну базу для наступного етапу – розробки технології виявлення внутрішніх загроз, яка буде адаптована до потреб організацій різного масштабу та забезпечить надійний захист даних.

РОЗДІЛ 2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ВИЯВЛЕННЯ ВНУТРІШНІХ ЗАГРОЗ

2.1 Оцінка та вибір методів машинного навчання для виявлення потенційних загроз

При виборі методів машинного навчання для виявлення потенційних загроз у інформаційно-комунікаційних системах важливо враховувати їхню здатність забезпечувати баланс між точністю класифікації, швидкістю обробки даних та адаптивністю до динамічних змін середовища. Оскільки внутрішні загрози часто характеризуються складними патернами поведінки, методи повинні ефективно працювати з великими обсягами даних, що постійно оновлюються. Особливого значення набувають алгоритми, які поєднують точність та інтерпретованість, дозволяючи зрозуміти основні причини виявлених аномалій. Використання сучасних підходів, таких як дерева рішень, ансамблеві методи та їхні модифікації, дозволяє підвищити рівень безпеки, забезпечуючи виявлення загроз на ранніх стадіях. Обґрунтований вибір методу залежить від особливостей системи, обсягу доступних даних та вимог до обчислювальних ресурсів.

Швидке дерево рішень (ШДР) є реалізацією алгоритму градієнтного бустингу (ГБ) на основі дерев рішень, розробленого для задач класифікації та регресії. Його основною перевагою є швидкість обробки великих наборів даних та ефективна побудова моделі за рахунок оптимізації використання пам’яті та процесорного часу [19]. Метод часто застосовується у сфері аналізу великих даних, таких як виявлення аномалій, прогнозування ризиків або автоматизація прийняття рішень. У контексті безпеки інформаційно-комунікаційних систем він може бути використаний для виявлення потенційних внутрішніх загроз шляхом аналізу поведінкових даних користувачів або характеристик системи.

На рисунку 2.1 зображена схема роботи алгоритму ШДР, яка демонструє його ключові етапи: поступове формування ансамблю дерев рішень, навчання на залишкових похибках та обчислення прогнозів. Ця схема відображає

послідовність побудови дерев, які доповнюють одне одного, з метою зменшення похибки.

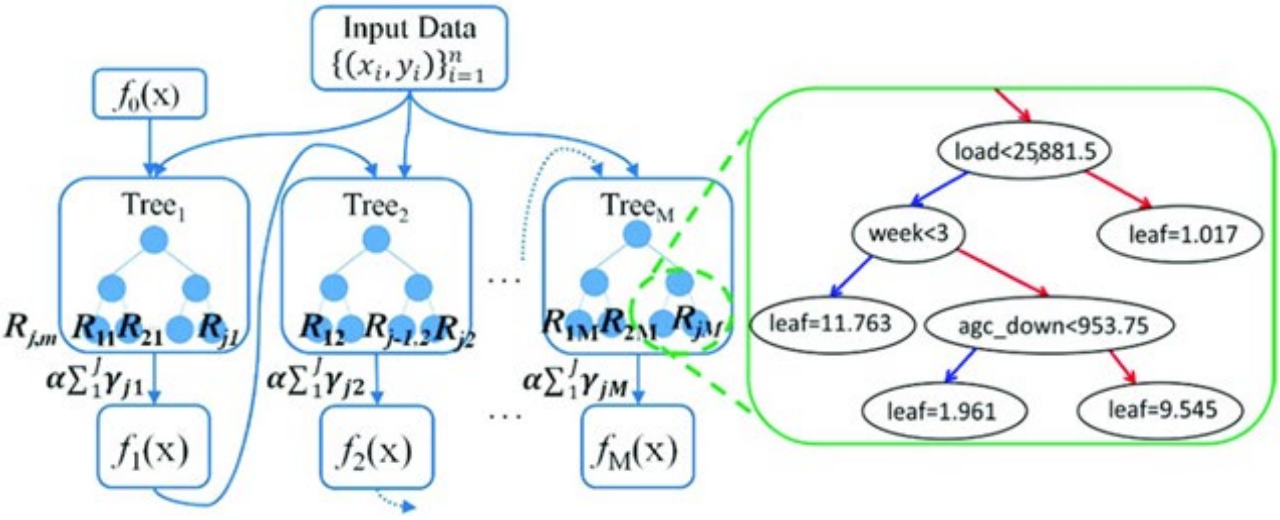

Рисунок 2.1 – Схема роботи алгоритму ШДР [20]

Згідно зі схемою, алгоритм починає з ініціалізації початкової моделі $f_0(x)$. Далі, кожне дерево $Tree_m$ навчається на основі залишкових похибок, що зберігаються у вузлах дерева R_{jm}, і додає свою вагу α до загального прогнозу. Це дозволяє кожному новому дереву коригувати помилки попередніх моделей, поступово покращуючи точність. У кінцевому результаті, всі дерева комбінуються у фінальну модель $f_M(x)$, яка представляє прогноз алгоритму. Важливою особливістю є структура дерева, в якій кожен вузол відповідає за прийняття рішення, а листи зберігають результат.

Метод ШДР забезпечує високу продуктивність завдяки ітеративній побудові дерев і мінімізації залишкових похибок. Він підходить для задач, що вимагають високої точності та можливості роботи з великими даними, зберігаючи при цьому інтерпретованість результатів.

Переваги методу ШДР:

а) висока швидкість обробки. Метод оптимізований для роботи з великими наборами даних, що робить його придатним для використання у реальному часі або у середовищах з великими обсягами інформації;

б) ефективне використання пам’яті [21]. Алгоритм використовує блоковий підхід для зменшення вимог до оперативної пам’яті, що є важливим при роботі з великими даними;

в) мінімізація похибок. Градієнтний бустинг забезпечує поступове зниження похибок за рахунок навчання кожного дерева на залишках попередніх моделей.

г) інтерпретованість результатів. Дерева рішень, які входять до ансамблю, дозволяють чітко визначити, як кожен з факторів впливає на прогноз.

Недоліки методу ШДР:

а) високі вимоги до обчислювальних ресурсів під час навчання. Хоча алгоритм ефективно використовує пам’ять, побудова кожного дерева може займати значний час для дуже великих даних;

б) чутливість до параметрів [22]. Необхідно ретельно налаштовувати гіперпараметри, такі як кількість дерев, глибина дерева та швидкість навчання (learningratelearningrate), для досягнення оптимальної продуктивності;

в) складність роботи з дисбалансованими даними. Якщо один клас значно переважає над іншим, алгоритм може потребувати додаткового налаштування, наприклад, ваги класів;

г) погане масштабування для багатокласових задач. Хоча метод добре працює з бінарними задачами, його продуктивність може знижуватися у задачах з великою кількістю класів.

Отже, ШДР є потужним методом, який об'єднує швидкість, точність та інтерпретованість, роблячи його придатним для складних задач у великих інформаційних системах. Він особливо корисний для автоматизованого аналізу даних у реальному часі, таких як виявлення загроз. Проте його ефективне використання вимагає глибокого розуміння налаштувань та оптимізації параметрів. Цей метод чудово підходить для задач, де важливі висока продуктивність і контроль за точністю результатів.

Метод випадкових лісів (ВЛ) є одним із найбільш надійних ансамблевих алгоритмів машинного навчання, який базується на об'єднанні кількох дерев рішень [23]. Цей метод використовується для задач класифікації та регресії, забезпечуючи високу точність навіть у складних сценаріях. У контексті захисту інформаційно-комунікаційних систем він застосовується для аналізу даних поведінки користувачів або параметрів системи, допомагаючи виявляти внутрішні загрози. Основною перевагою методу є його стійкість до шуму у даних, це важливо у системах з великими обсягами інформації.

На рисунку 2.2 зображено принцип роботи випадкових лісів, який базується на об'єднанні прогнозів кількох дерев рішень. Такий підхід дозволяє враховувати різні аспекти даних, що підвищує точність та зменшує ризик перенавчання [24].

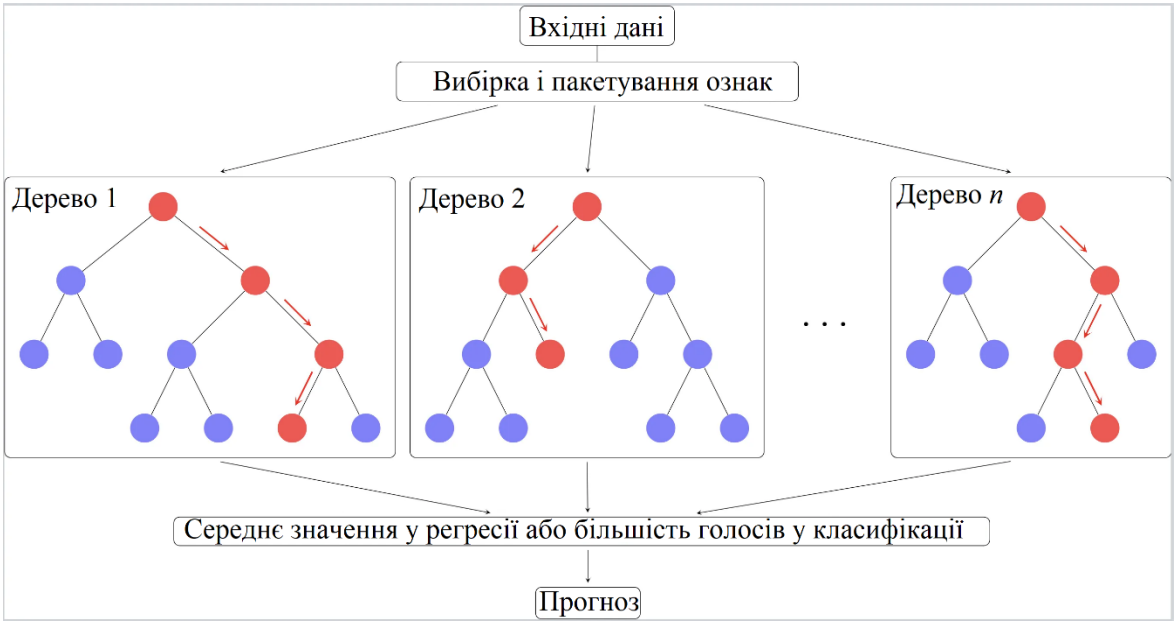

Рисунок 2.2 – Схема роботи алгоритму ВЛ

Схема демонструє, як кожне дерево в ансамблі приймає власне рішення на основі підвибірок даних і ознак. Далі ці прогнози об'єднуються: у випадку регресії – шляхом обчислення середнього значення, а для класифікації – більшістю голосів. Це забезпечує узгодженість та надійність результатів, навіть якщо окремі дерева помиляються через випадковість або шум.

Таким чином, метод ВЛ є ефективним інструментом для виявлення потенційних загроз, забезпечуючи високу точність і стійкість до перенавчання. Його здатність працювати з великими наборами даних робить його одним із ключових рішень для автоматизації моніторингу та аналізу інформаційно-комунікаційних систем.

Переваги методу ВЛ:

а) висока точність. Завдяки поєднанню прогнозів кількох дерев метод забезпечує високу точність навіть у задачах із великим обсягом даних та складними залежностями;

б) стійкість до перенавчання [25]. Використання випадкових підвибірок даних і ознак дозволяє уникнути перенавчання, навіть якщо окремі дерева в ансамблі надмірно адаптуються до навчальних даних;

в) стійкість до шуму. Метод добре працює із даними, які містять шум або пропущені значення, завдяки узагальненню рішень від різних дерев;

г) можливість оцінки важливості ознак. ВЛ дозволяє визначити, які ознаки мають найбільший вплив на результат, що є корисним для інтерпретації моделі та її використання в реальних задачах.

Недоліки методу ВЛ:

а) високі обчислювальні витрати. Навчання багатьох дерев може займати значний час та вимагати великих обчислювальних ресурсів, особливо для великих наборів даних;

б) складність інтерпретації. Хоча метод дозволяє визначити важливість ознак, сам ансамбль складно інтерпретувати, оскільки він складається з багатьох дерев;

в) проблеми зі збалансованістю класів [26]. У випадках із дисбалансованими даними метод може надавати перевагу більш представленому класу, якщо не вжити заходів для компенсації;

г) втрата продуктивності на малих наборах даних. Для невеликих наборів даних метод може бути менш ефективним, оскільки генерація

підвибірок може обмежувати доступну інформацію для кожного дерева.

Отже, ВЛ – це потужний інструмент для задач класифікації та регресії, який добре підходить для аналізу великих наборів даних у системах, що вимагають високої точності та стійкості до шуму. Хоча метод вимагає значних ресурсів і може бути складним для інтерпретації, його переваги роблять його незамінним у багатьох практичних сценаріях, зокрема у сфері інформаційної безпеки. Використання ВЛ озволяє покращити автоматизований моніторинг та аналіз загроз, забезпечуючи надійність і точність результатів.

Градієнтний бустинг (ГБ) – це метод машинного навчання, який належить до ансамблевих алгоритмів. Він працює шляхом послідовного побудови моделей, які коригують похибки попередніх, що дозволяє отримати високоточні результати [27]. Цей підхід особливо корисний для задач класифікації та регресії, включаючи виявлення внутрішніх загроз у інформаційно-комунікаційних системах. Метод дозволяє ефективно працювати з великими наборами даних, забезпечуючи адаптацію до їхніх особливостей. У сфері безпеки його можна використовувати для аналізу складних залежностей у поведінкових даних, підвищуючи ефективність виявлення аномалій.

На рисунку 2.3 продемонстровано схему роботи ГБ, яка відображає основні етапи створення моделі. Схема показує, як слабкі класифікатори поступово об'єднуються у фінальний ансамбль, де кожен наступний класифікатор зосереджується на виправленні помилок попереднього [28].

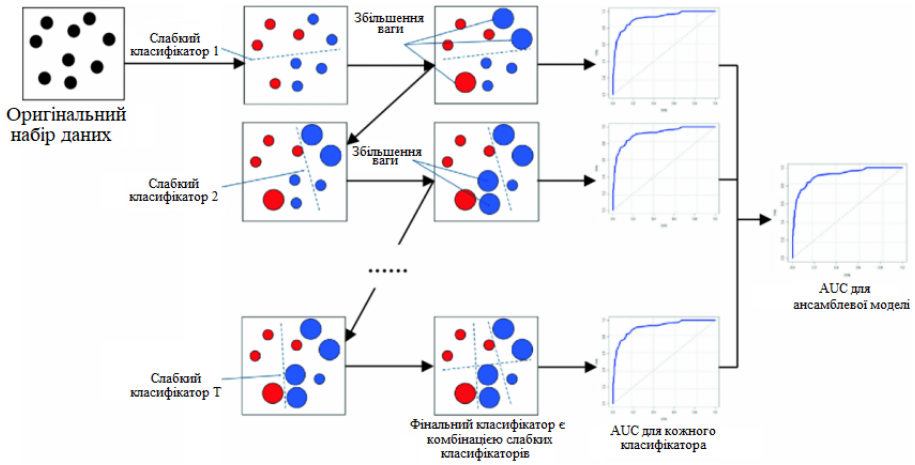

Рисунок 2.3 – Схема роботи алгоритму ГБ

Алгоритм починається з ініціалізації початкового слабкого класифікатора, який робить базові прогнози. На кожному наступному етапі ваги даних, які були неправильно класифіковані, збільшуються, щоб наступний класифікатор міг краще навчитися на цих прикладах. Процес продовжується до досягнення необхідної точності або завершення заданої кількості ітерацій. У фіналі всі слабкі класифікатори комбінуються, формуючи потужну ансамблеву модель.

Градієнтний бустинг є одним із найефективніших методів для задач, які вимагають високої точності та роботи з великими наборами даних. Завдяки ітеративному підходу він дозволяє покращити якість прогнозів, адаптуючись до складних патернів у даних. Його використання сприяє побудові надійних систем моніторингу та захисту, де важливим є виявлення прихованих залежностей і аномалій.

Переваги методу ГБ:

а) висока точність. Градієнтний бустинг демонструє високу ефективність на складних наборах даних завдяки ітеративному коригуванню похибок;

б) гнучкість. Алгоритм можна налаштувати під різні задачі класифікації та регресії, використовуючи різні функції втрат і параметри;

в) стійкість до перенавчання [29]. За правильного налаштування параметрів, таких як швидкість навчання та глибина дерев, метод добре узгоджується з даними без надмірного пристосування;

г) робота з важливістю ознак. Метод дозволяє визначати найбільш значущі ознаки, що корисно для інтерпретації моделей та покращення якості даних.

До недоліків методу ГБ відносять:

а) високі обчислювальні витрати. Навчання моделі потребує значних ресурсів через багатоетапну побудову ансамблю;

б) чутливість до гіперпараметрів. Ефективність алгоритму значною мірою залежить від правильного налаштування параметрів, що може потребувати багато часу та експериментів;

в) погане масштабування [30]. Зі збільшенням розмірів даних час навчання може суттєво зростати, ускладнюючи використання на дуже великих наборах даних.

г) складність інтерпретації моделі. Хоча метод дозволяє визначити важливість ознак, розуміння роботи всієї моделі є складним через використання багатьох дерев.

Отже, ГБ також є одним із найбільш ефективних методів для задач, які потребують високої точності та адаптації до складних залежностей у даних. Його основні переваги – це гнучкість та здатність працювати з важливими ознаками, що робить його незамінним у завданнях виявлення загроз. Проте використання цього методу вимагає значних обчислювальних ресурсів та ретельного налаштування, що може бути викликом у масштабних проєктах.

Таблиця 2.1 – Порівняння методів машинного навчання

Характеристика	ШДР	ВЛ	ГБ
Швидкість обробки даних	Висока	Середня	Низька
Обчислювальні ресурси	Низькі	Середні	Високі
Адаптивність до змін	Висока	Середня	Середня
Інтерпретованість моделі	Середня	Висока	Низька
Стійкість до великих даних	Висока	Висока	Середня
Точність класифікації	Середня	Висока	Висока

Вибір методу ШДР для розробки системи виявлення внутрішніх загроз ґрунтується на його здатності забезпечувати високу продуктивність у задачах аналізу великих потоків даних. Його ключова перевага – це висока швидкість обробки, що дозволяє оперативно аналізувати активність користувачів у реальному часі, зводячи до мінімуму затримки у виявленні потенційних загроз. Це надзвичайно важливо для інформаційно-комунікаційних систем, де зволікання може призвести до серйозних наслідків, таких як втрата даних або компрометація конфіденційної інформації. Метод також демонструє здатність швидко адаптуватися до змін у вхідних даних, що забезпечує його ефективність навіть у динамічних середовищах із постійно змінюваними умовами.

Крім того ШДР характеризується низькими вимогами до обчислювальних ресурсів, що дозволяє його використовувати в системах з обмеженими технічними можливостями, таких як віддалені сервери або пристрої з низькою продуктивністю. Це робить його доступним для широкого спектру застосувань, включаючи корпоративні мережі та невеликі організації. Важливою особливістю методу є його стійкість до великих обсягів даних, що дозволяє обробляти значні масиви інформації без суттєвого зниження продуктивності. Завдяки цим характеристикам ШДР є раціональним вибором для створення високопродуктивної та надійної системи виявлення внутрішніх загроз.

2.2 Вибір та обґрунтування необхідного набору даних для навчання та тестування моделі

У розробці системи виявлення внутрішніх загроз одним із найважливіших етапів є вибір відповідного набору даних, який дозволить моделі навчитися розпізнавати специфічні поведінкові патерни користувачів у системі. Для реалізації цієї мети було обрано набір даних «Data Leakage Detection», який містить детальну інформацію про активність користувачів у комп’ютерній мережі [31]. У цьому наборі представлені різноманітні аспекти дій користувачів, включаючи методи доступу до системи, операції з файлами, доступ до конфіденційної інформації, рівень чутливості даних, а також індикатори аномальності. Така структура дозволяє аналізувати поведінкові характеристики з урахуванням ролі користувачів у системі, їхньої активності та потенційного ризику дій.

Набір даних відповідає поставленим цілям завдяки низці ключових характеристик. По-перше, він включає атрибути, які прямо впливають на ймовірність виникнення загроз, такі як доступ до конфіденційної інформації, модифікації даних, зовнішні з’єднання та операції з файлами. Ці атрибути є критичними для побудови точних моделей класифікації, оскільки вони дозволяють виявляти потенційно ризиковані дії. По-друге, наявність метрик аномальності забезпечує можливість навчання моделей на реальних даних із

вказівками щодо нормальної та аномальної активності, що суттєво підвищує ефективність класифікації.

Окрім того, дані охоплюють широкий спектр ролей користувачів (наприклад, staff, manager) та методів доступу (через пароль, PIN (personal identification number, персональний ідентифікаційний номер) або MFA (multi-factor authentication, багатофакторна автентифікація), що дозволяє створити універсальну модель, здатну адаптуватися до різних сценаріїв використання системи. Така різноманітність поведінкових патернів сприяє створенню моделі, яка може враховувати як регулярні, так і специфічні дії користувачів, забезпечуючи її високу точність і адаптивність. Таким чином, вибір набору даних «Data Leakage Detection» обґрунтований його здатністю відображати реальні загрози та різноманітність взаємодій користувачів у системі.

У таблиці 2.2 представлено статистику ключових показників набору даних.

Таблиця 2.2 – Розподіл основних атрибутів у наборі даних

Характеристика	Категорія	Розподіл (%)
Тип доступу	Через пароль	90
	Через PIN	65
	Через MFA	50
Рівень доступу	Staff	28
	Manager	24
	Інші	48
Тип дій	Читання (read)	26
	Переміщення (move)	33
	Інші	42
Рівень чутливості даних	Низький	34
	Високий	33
	Інші	33
Аномальні дії	Виявлено (1)	40
	Не виявлено (0)	60

Дані свідчать, що більшість користувачів здійснюють доступ до системи через пароль (90%), але також присутні дані щодо доступу через PIN та MFA.

Типовими діями є читання файлів (26%) та їх переміщення (33%). Розподіл аномальних дій становить 40%, що дає достатньо прикладів для навчання моделі.

Щоб забезпечити якісну перевірку моделі, також було проаналізовано розподіл міток (анотацій) у контексті класів (аномальних/нормальних дій). У таблиці 2.3 подано розподіл даних за класами.

Таблиця 2.3 – Розподіл даних за класами

Клас	Кількість записів	Процентний розподіл
Нормальна активність (0)	35,657	72
Аномальна активність (1)	13,345	28

Розподіл показує певну незбалансованість, оскільки нормальні дії складають більшість (72%). Для компенсації цього під час навчання моделі можуть використовуватися методи балансування, такі як повторна вибірка меншості або вагове коригування.

Обраний набір даних є оптимальним для задачі виявлення внутрішніх загроз завдяки його структурованості, різноманітності показників та наявності оцінок аномальності. Статистичний аналіз показав необхідність врахування дисбалансу між класами та різноманітність дій, що сприяє побудові точної та надійної моделі. В подальшій роботі ці дані будуть використані для навчання, тестування та валідації алгоритмів.

2.3 Розробка математичної моделі для класифікації дій користувачів та оцінки ризиків

В межах даного дослідження внутрішні загрози для ІКС розглядаються як аномальні дії користувача, що потенційно можуть спричинити витоки даних, несанкціоновані модифікації або інші небажані впливи. Виявлення таких загроз базується на застосуванні методів машинного навчання, зокрема бінарної класифікації, де кожна дія користувача характеризується вектором ознак та позначається як «аномальна» або «нормальна».

Нехай є сукупність спостережень (записів користувацької активності):

$$D = \{(x^{(i)}, y^{(i)}) | \, i = 1,2, \dots, N\}, \quad (2.1)$$

де N – розмір вибірки, $x^{(i)} \in R^M$ – вектор ознак i-го спостереження, а $y^{(i)} \in \{0,1\}$ – булева змінна, що вказує на клас дії (0 – нормальна, 1 – аномальна).

Ознаки $x^{(i)}$ можуть містити як числові, так і категоріальні поля. Для їх уніфікації застосовується кодування, наприклад «One-Hot Encoding» для категорійних ознак. Якщо ознака «Authority» може набувати K_A категорій, то після кодування вона перетворюється на вектор розмірності K_A:

$$Authority \xrightarrow{OHE} [z_1, z_2, \dots z_{K_A}], \quad (2.2)$$

де:

$$z_j = \begin{cases} 1, \text{якщо категорія } j \text{ активна} \\ 0, \text{в іншому випадку.} \end{cases}, \quad (2.3)$$

Аналогічні перетворення виконуються для інших категоріальних змінних: «ExternalDestination», «FileOperation», «DataSensitivityLevel». Після конкатенації всіх бінаризованих та нормалізованих (за необхідності) ознак отримуємо кінцевий вектор:

$$x^{(i)} = \left[x_1^{(i)}, x_2^{(i)}, \dots, x_r^{(i)}, z_1^{(i)}, \dots z_k^{(i)}\right] \in R^{r+k}, \quad (2.4)$$

де r – кількість числових ознак, k – сумарна кількість бінарних індикаторів категорій.

Для числових ознак може застосовуватися нормалізація за середнім та дисперсією:

$$x_{norm,j}^{(i)} -= \frac{x_{norm,j}^{(i)} - \mu_j}{\sigma_j}, \quad (2.5)$$

де μ_j – середнє, а σ_j – стандартне відхилення j-ї ознаки по вибірці.

Для класифікації дій користувача обирається модель типу бінарного класифікатора, наприклад, на основі градієнтного бустингу дерев рішень. Узагальнено модель можна подати у вигляді:

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x), \quad (2.6)$$

де T – кількість дерев у ансамблі, α_t – вагові коефіцієнти, а $h_t(x)$ – прогноз t-го дерева.

Вихід моделі $H(x)$ інтерпретується через логістичну функцію для отримання ймовірності аномалії:

$$P(x) = \sigma\big(H(x)\big) = \frac{1}{1+e^{-H(x)}}, \tag{2.7}$$

Таким чином, оцінка ризику аномальної дії користувача задається через імовірність:

$$R(x) = p(x), \tag{2.8}$$

де більші значення $R(x)$ свідчать про вищу імовірність внутрішньої загрози.

Процес навчання полягає у мінімізації логістичної функції втрат:

$$L = -\sum_{i=1}^{N}\left[y^{(i)} \ln\left(p\left(x^{(i)}\right)\right) + (1 - y^{(i)} \ln(1 - p(x^{(i)}))\right]. \tag{2.9}$$

Налаштування гіперпараметрів (наприклад, кількості дерев T або глибини кожного дерева) виконується з використанням валідаційних процедур, спрямованих на покращення узагальнюючої здатності моделі.

При прогнозуванні на новому векторі ознак x^* ймовірність аномалії $p(x^*)$ порівнюється з порогом θ:

$$\hat{y}(x^*) = \begin{cases} 1, \text{якщо } p(x^*) \geq \theta \\ 0, \text{якщо } p(x^*) < \theta. \end{cases}, \tag{2.10}$$

Традиційно θ = 0.5, але за потреби поріг може бути змінений для досягнення балансу між чутливістю та точністю.

Отже, розроблена математична модель класифікації дій користувачів та оцінки ризиків базується на формалізації простору ознак, застосуванні процедур нормалізації та кодування, а також на використанні бінарного класифікатора, що оцінює імовірність аномальної активності. Отримана модель дозволяє інтерпретувати будь-яку дію користувача з точки зору потенційної загрози, що створює підґрунтя для подальшого використання метрик оцінки ефективності.

Після побудови математичної моделі важливо оцінити її ефективність. Традиційно у задачах бінарної класифікації застосовуються метричні показники, що відображають якість розпізнавання аномальних дій користувачів. Ці метрики

дозволяють кількісно оцінити, наскільки добре модель відрізняє нормальну активність від потенційно шкідливої.

Нехай за результатами прогнозування на тестовому наборі даних маємо:

– TP (True Positives) – кількість дій, що є аномальними і правильно виявлені як такі;

– TN (True Negatives) – кількість нормальних дій, які модель правильно ідентифікує як нормальні;

– FP (False Positives) – кількість нормальних дій, помилково позначених як аномальні;

– FN (False Negatives) – кількість аномальних дій, які модель не виявила (позначила як нормальні).

Структурно це можна подати у вигляді матриці:

$$\begin{pmatrix} \mathrm{TN} & \mathrm{FP} \\ \mathrm{FN} & \mathrm{TP} \end{pmatrix}, \tag{2.11}$$

Найпростішою узагальнюючою метрикою є точність (*Accuracy*):

$$\mathrm{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}, \tag{2.12}$$

Проте точність може бути оманливою, якщо класи незбалансовані (наприклад, коли аномальні дії трапляються рідко). Тому застосовують додаткові метрики:

Повнота (чутливість) для аномального класу:

$$\mathrm{Recall} = \frac{TP}{TP+FN}, \tag{2.13}$$

Точність (*Precision*) у свою чергу визначається за формулою::

$$\mathrm{Precision} = \frac{TP}{TP+FP}, \tag{2.14}$$

Ці дві метрики дозволяють зрозуміти, наскільки модель не пропускає аномалії (*Recall*) та наскільки рідко вона помиляється, позначаючи нормальні дії як аномальні (*Precision*).

Щоб об’єднати *Precision* і *Recall* в одну метрику, застосовують *F1*-міру:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}, \tag{2.15}$$

F1-міра особливо корисна, коли важливо підтримувати баланс між пропуском аномалій та помилковим сповіщенням.

Для детальнішої оцінки використовують міру *AUC* (Area Under the ROC Curve), що визначається як:

$$AUC = \int_0^1 TPR(FPR^{-1}(u)du)\,, \qquad (2.16)$$

де $TPR = \frac{TP}{TP+FN}$ – частка правильно виявлених аномалій, а $FPR = \frac{FP}{FP+TN}$ – частка помилкових спрацювань.

Додатково може застосовуватись логарифмічна втрата (*Log Loss*), що безпосередньо оцінює якість передбачених імовірностей:

$$LL = -\frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \left[y^{(i)} \ln\left(p\left(x^{(i)}\right)\right) + \left(1 - y^{(1)}\right) \ln\left(1 - p\left(x^{(i)}\right)\right)\right]. \quad (2.17)$$

Низьке значення *Log Loss* свідчить про те, що модель коректно оцінює імовірності, не надто впевнено помиляючись.

Аналізуючи набори зазначених метрик, можна визначити:

– наскільки модель чутлива до виявлення аномалій (*Recall*);

– наскільки модель вибірково позначає дії як аномальні, уникаючи хибних тривог (*Precision*);

– чи досягає модель балансу між цими двома аспектами (*F1*);

– наскільки стійка модель до вибору порога прийняття рішення (*AUC*);

– як добре модель оцінює саме ймовірності, а не лише етикетки (*Log Loss*).

У кінцевому підсумку метрики дозволяють прийняти рішення про доцільність використання моделі, необхідність її доопрацювання або коригування гіперпараметрів. Також їх можна застосувати для порівняння різних підходів до побудови моделей і вибору найефективнішого варіанту.

Отже, визначення метрик оцінки точності і ефективності моделі у виявленні внутрішніх загроз є невід’ємним етапом процесу побудови системи виявлення аномалій. Набір метрик (*Accuracy*, *Precision*, *Recall*, *F1*, *AUC*, *Log Loss*) дає змогу об’єктивно оцінити якість прогнозів і прийняти зважені рішення щодо подальших кроків із вдосконалення математичної моделі.

У рамках даного розділу було проведено комплексну роботу, спрямовану на створення інформаційної технології для виявлення внутрішніх загроз у інформаційно-комунікаційних системах. Проведено детальний аналіз сучасних методів машинного навчання, таких як швидке дерево рішень, метод випадкових лісів та градієнтний бустинг. На основі їх порівняння за ключовими характеристиками, включаючи швидкість, точність і адаптивність, для подальшої розробки системи було обрано метод Швидкого дерева рішень. Також обґрунтовано вибір набору даних «Data Leakage Detection», який містить важливу інформацію про поведінкові патерни користувачів у мережі, включаючи метрики аномальності, рівень чутливості даних і типи доступу.

Окрім цього, розроблено математичну модель для класифікації дій користувачів та оцінки ризиків, яка враховує специфіку обраного методу та особливості даних. Для оцінки якості моделі визначено ключові метрики, такі як Accuracy, Precision, Recall, F1, AUC і Log Loss, що дозволяють об’єктивно оцінити її ефективність у виявленні загроз. Результати, отримані в цьому розділі, забезпечують фундамент для реалізації програмної складової та проведення тестування технології виявлення внутрішніх загроз у наступному розділі.

РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ТЕХНОЛОГІЇ ВИЯВЛЕННЯ ВНУТРІШНІХ ЗАГРОЗ

3.1 Вибір мови програмування та програмних засобів для реалізації системи

Реалізація системи виявлення внутрішніх загроз потребує використання сучасних мов програмування та інструментів, які забезпечують високу продуктивність, ефективну обробку даних та зручність у розробці машинного навчання. При виборі мови програмування враховувалися такі критерії, як доступність бібліотек для обробки великих обсягів даних, підтримка алгоритмів машинного навчання, зручність інтеграції з іншими компонентами системи та продуктивність виконання програмного коду. Серед мов програмування, які розглядалися для розробки системи, були Python, Java та C#.

Python є однією з найбільш популярних мов для розробки систем, заснованих на машинному навчанні. Вона має широкий набір бібліотек, таких як Scikit-learn, TensorFlow, PyTorch, які полегшують реалізацію та налаштування моделей [32]. Python також підтримує просту інтеграцію з базами даних і хмарними платформами, що дозволяє створювати масштабовані рішення.

Java є потужною мовою програмування, яка часто використовується для побудови систем корпоративного рівня завдяки її стабільності та портативності. Java забезпечує високу продуктивність у багатопотокових додатках, що важливо для аналізу великих потоків даних у реальному часі [33]. Крім того, існує низка бібліотек для машинного навчання, таких як Weka та DL4J, що робить Java конкурентоспроможною для розробки інтелектуальних систем.

C# є мовою, яка поєднує високу продуктивність і зручність для створення інтерактивних і масштабованих додатків. Завдяки платформі .NET C# підтримує інтеграцію з серверами Microsoft та надає потужні інструменти для роботи з базами даних та хмарними сервісами [34]. Окрім того, бібліотека ML.NET дозволяє реалізувати алгоритми машинного навчання без необхідності використання сторонніх інструментів, що зручно для інтегрованих рішень.

Для обґрунтованого вибору мови програмування для реалізації системи важливо провести порівняльний аналіз за ключовими характеристиками. Це дозволяє оцінити переваги та недоліки кожної мови у контексті розробки систем виявлення внутрішніх загроз. Таблиця 3.1 демонструє основні показники Python, Java та C# з урахуванням їхніх характеристик, важливих для створення високопродуктивної, масштабованої та зручної у використанні системи.

Таблиця 3.1 – Порівняння мов програмування Python, Java та C#

Характеристика	Python	Java	C#
Продуктивність виконання	Середня	Висока	Висока
Підтримка машинного навчання	Висока (Scikit-learn, TensorFlow)	Середня (DL4J, Weka)	Висока (ML.NET)
Зручність написання коду	Висока (інтуїтивний синтаксис)	Середня (більш формалізована)	Висока (зручний синтаксис)
Інтеграція з платформами	Висока	Висока	Дуже висока (Microsoft .NET)
Обробка багатопоточності	Обмежена	Висока	Висока
Робота з базами даних	Висока (через ORM-бібліотеки)	Висока	Висока (Entity Framework)
Підтримка корпоративних додатків	Середня	Висока	Висока
Продуктивність у Windows-середовищі	Середня	Середня	Дуже висока

Вибір мови програмування C# для розробки системи виявлення внутрішніх загроз обґрунтований її високою продуктивністю, особливо у середовищі Windows, що є поширеним у корпоративному секторі. Завдяки інтеграції з платформою .NET мова надає широкий спектр інструментів для створення масштабованих та інтерактивних додатків, зокрема ML.NET для реалізації алгоритмів машинного навчання. Зручний синтаксис C# забезпечує швидку розробку і підтримку коду, що важливо для складних проектів. Окрім того, можливість ефективної роботи з багатопоточністю та інтеграція з базами даних через Entity Framework робить C# оптимальним вибором для створення

високопродуктивних і надійних систем. Такий набір переваг забезпечує відповідність мови вимогам до розробки системи, орієнтованої на виявлення загроз у динамічному середовищі.

Для розробки системи виявлення внутрішніх загроз також необхідно обрати відповідну систему управління базами даних (СУБД), яка забезпечить збереження, обробку та доступ до великих обсягів даних з високою швидкістю та надійністю. Вибір СУБД залежить від кількох факторів, таких як продуктивність, масштабованість, підтримка складних запитів та сумісність із середовищем розробки. Серед розглянутих варіантів для реалізації системи були MS SQL Server, MySQL та FireBird, які мають свої унікальні особливості та сфери застосування.

MS SQL Server – це високопродуктивна реляційна СУБД, яка забезпечує повну інтеграцію з платформою Microsoft .NET, що є перевагою для розробки на C# [35]. Вона підтримує складні аналітичні запити, масштабованість для великих обсягів даних та потужні інструменти безпеки, такі як шифрування й аудит доступу. Завдяки інструментам, таким як SQL Server Management Studio (SSMS), розробники мають доступ до зручного середовища для управління базами даних.

MySQL є однією з найпопулярніших СУБД з відкритим вихідним кодом, яка забезпечує високу продуктивність та сумісність з різними платформами [36]. Вона відома своєю простотою в налаштуванні та ефективною роботою з веб-додатками завдяки інтеграції з мовами програмування, такими як PHP і Python. Проте її функціональні можливості можуть бути обмеженими для великих корпоративних систем з високим рівнем складності.

FireBird – це легка та компактна СУБД, яка підтримує роботу як у локальних, так і в розподілених середовищах [37]. Вона забезпечує високу швидкість обробки даних навіть за низьких вимог до апаратного забезпечення, що робить її підходящою для невеликих проектів або автономних систем. Однак її функціонал для складних корпоративних рішень є менш розвиненим у порівнянні з MS SQL Server чи MySQL.

У таблиці 3.2 наведено основні характеристики цих СУБД, що дозволяє оцінити їх відповідність потребам розробки.

Таблиця 3.2 – Порівняння СУБД MS SQL Server, MySQL та FireBird

Характеристика	MS SQL Server	MySQL	FireBird
Продуктивність	Висока	Середня	Середня
Інтеграція з платформами	Дуже висока (Microsoft .NET)	Висока	Середня
Безпека	Потужна (шифрування, аудит)	Основна (SSL)	Базова
Масштабованість	Висока (підтримка кластерів)	Середня	Обмежена
Інструменти адміністрування	Розвинені (SSMS, T-SQL)	Обмежені	Мінімальні
Підтримка складних запитів	Висока	Середня	Середня
Підтримка великих даних	Дуже висока	Середня	Обмежена
Середовище використання	Корпоративні системи	Веб-додатки	Автономні додатки

Вибір MS SQL Server для розробки системи виявлення внутрішніх загроз обґрунтований її високою продуктивністю, масштабованістю та потужними інструментами для адміністрування. Завдяки розвиненим засобам безпеки, таким як шифрування даних і аудит доступу, ця СУБД забезпечує надійний захист інформації, що є критично важливим для роботи системи в умовах високих ризиків. Інтеграція з платформою Microsoft .NET дозволяє легко поєднувати базу даних із програмними компонентами системи, створеними на мові C#. Крім того, підтримка складних запитів і великих обсягів даних робить MS SQL Server оптимальним вибором для реалізації функціоналу системи виявлення загроз.

3.2 Реалізація програмного модуля для класифікації дій користувачів з використанням машинного навчання

Ефективна класифікація дій користувачів є ключовим компонентом системи виявлення внутрішніх загроз. Для цього було створено програмний модуль, який використовує алгоритми машинного навчання для аналізу поведінкових даних. Основним завданням цього модуля є обробка вхідного

набору даних, навчання моделі на основі аномальної активності, а також прогнозування можливих загроз на основі нових записів. У процесі розробки використовувалися спеціалізовані класи, які спрощують роботу з даними та забезпечують структуроване управління ключовими компонентами моделі.

На рисунку 3.1 наведено структуру класів, яка була створена для роботи з набором даних та прогнозуванням потенційних загроз. Схема ілюструє взаємозв'язок між даними, які зберігаються у вигляді записів активності, та результатами моделі, що прогнозує ризики.

```
16 references
public class UserActivity {
    [LoadColumn(0)] public int Id;
    [LoadColumn(1)] public string Date;
    [LoadColumn(2)] public string User;
    [LoadColumn(3)] public string PC;
    [LoadColumn(4)] public string Authority;
    [LoadColumn(5)] public float Through_pwd;
    [LoadColumn(6)] public float Through_pin;
    [LoadColumn(7)] public float Through_MFA;
    [LoadColumn(8)] public float DataModification;
    [LoadColumn(9)] public float ConfidentialDataAccess;
    [LoadColumn(10)] public float ConfidentialFileTransfer;
    [LoadColumn(11)] public string ExternalDestination;
    [LoadColumn(12)] public string FileOperation;
    [LoadColumn(13)] public string DataSensitivityLevel;
    [LoadColumn(14), ColumnName("Label")] public bool Abnormality;
}
```

```
2 references
public class ThreatPrediction {
    // Чи є активність загрозливою (true – так, false – ні)
    [ColumnName("PredictedLabel")] public bool IsThreat;

    // Імовірність оцінки загрози для даної активності
    2 references
    public float Probability { get; set; }

    // Оцінка відповідності активності потенційній загрозі
    0 references
    public float Score { get; set; }
}
```

Рисунок 3.1 – Структура класів для роботи з набором даних

Клас UserActivity визначає структуру вхідних даних, які використовуються для навчання моделі. Він включає всі ключові атрибути записів активності, такі як унікальний ідентифікатор, дата й час події, ідентифікатор користувача та комп’ютера, а також додаткову інформацію про тип доступу, модифікації даних, доступ до конфіденційної інформації та рівень чутливості даних. Цей клас також містить мітку аномальності, яка слугує цільовим показником для навчання класифікаційної моделі.

Клас ThreatPrediction відповідає за результати прогнозування моделі. Він містить інформацію про те, чи вважається активність загрозливою, а також включає додаткові метрики, такі як імовірність виникнення загрози та оцінка відповідності активності загрозливим патернам. Ця структура дозволяє детально аналізувати результати моделі та адаптувати її до вимог системи.

На рисунку 3.2 представлено механізм роботи функції, яка забезпечує інтерактивний вибір файлу користувачем через графічний інтерфейс. Цей код створює діалогове вікно для відкриття файлу, що дозволяє користувачеві вибрати необхідний документ для подальшої обробки системою. Вікно налаштоване на підтримку фільтрації файлів за розширенням, наприклад, для вибору CSV-файлів, що зручно для роботи з табличними даними.

```
private void OpenBtn_Click(object sender, EventArgs e) {
  // Створення діалогового вікна для відкриття файлу
  OpenFileDialog openFileDialog = new OpenFileDialog();
  // Налаштування властивостей діалогового вікна
  openFileDialog.Filter = "Text files (*.csv)|*.csv|All files (*.*)|*.*";
  openFileDialog.FilterIndex = 2;
  openFileDialog.RestoreDirectory = true;
  // Відображення діалогового вікна та обробка результату
  if (openFileDialog.ShowDialog() == DialogResult.OK) {
    _Path = openFileDialog.FileName;
    FileNameTBox.Text = openFileDialog.FileName;
```

Рисунок 3.2 – Забезпечення інтерактивного вибору файлу користувачем

Додатково налаштування включають можливість повернення до попереднього каталогу після завершення роботи з діалоговим вікном, що спрощує навігацію для користувача. Після підтвердження вибору файлу система зберігає шлях до нього у внутрішній змінній та відображає назву файлу у текстовому полі інтерфейсу. Такий підхід забезпечує зручний спосіб інтеграції зовнішніх даних у систему, мінімізуючи складність взаємодії користувача із програмою.

Рисунок 3.3 ілюструє створення контексту ML, який є основним елементом для роботи з бібліотекою ML.NET. У цьому коді ініціалізується об'єкт типу MLContext, що слугує точкою входу для всіх операцій машинного навчання. Контекст містить необхідні інструменти та параметри, які дозволяють налаштувати процес створення, тренування та тестування моделей.

```
// Створення контексту ML
mlContext = new MLContext(seed: 0);
```

Рисунок 3.3 – Забезпечення інтерактивного вибору файлу користувачем

Під час ініціалізації задається фіксоване значення зерна для генератора випадкових чисел, що гарантує відтворюваність результатів. Це особливо важливо для проведення експериментів, де необхідно забезпечити однакові умови навчання та тестування моделі. Такий підхід створює стабільну основу для роботи з даними та побудови ефективних алгоритмів машинного навчання.

На рисунку 3.4 відображено процес завантаження та підготовки даних для аналізу за допомогою бібліотеки ML.NET. Код виконує імпорт даних із текстового файлу, шлях до якого вказано у змінній _Path. Формат файлу передбачає наявність заголовка та використання коми як роздільника між стовпцями, що налаштовується відповідними параметрами.

```
// Завантаження та підготовка даних
dataView = mlContext.Data.LoadFromTextFile<UserActivity>(
    path: _Path,
    hasHeader: true,
    separatorChar: ',');
```

Рисунок 3.4 – Забезпечення інтерактивного вибору файлу користувачем

Дані автоматично перетворюються у формат IDataView, який є основною структурою ML.NET для обробки великих обсягів інформації. Вказаний тип UserActivity дозволяє відобразити кожний рядок файлу у вигляді об'єкта з відповідними полями, що значно спрощує доступ до даних і їх подальше використання для тренування моделі. Такий підхід гарантує коректне зчитування даних і створює міцну основу для виконання операцій машинного навчання.

Рисуногк 3.5 висвітлює етап підготовки даних, зокрема роботу з категоріальними та числовими атрибутами, а також аналіз класів мітки Abnormality. Код визначає набір колонок, що містять числові значення, які будуть використовуватися для обчислень та побудови моделі. Ці колонки включають інформацію про методи автентифікації, доступ до конфіденційних даних та виконання змін.

```
// Категоріальні та числові колонки
string[] numericColumns = new[] { "Through_pwd", "Through_pin", "Through_MFA",
  "DataModification", "ConfidentialDataAccess", "ConfidentialFileTransfer" };

// Виведення кількості взірців для кожного класу Abnormality
var abnormalityCounts = mlContext.Data.CreateEnumerable<UserActivity>(dataView,
  reuseRowObject: false)
    .GroupBy(activity => activity.Abnormality)
    .Select(group => new {
      Abnormality = group.Key,
      Count = group.Count()
    });
```

Рисунок 3.5 – Етап підготовки даних та аналіз класів мітки Abnormality

Для аналізу розподілу даних по класах мітки Abnormality застосовується перетворення IDataView у перелік об'єктів типу UserActivity. Кожен запис групується за значенням мітки, після чого обчислюється кількість взірців у кожному класі. Цей підхід дозволяє швидко оцінити баланс даних між нормальними та аномальними діями, що є важливим для налаштування та тренування моделі класифікації.

На рисунку 3.6 демонструється відображення результатів аналізу розподілу даних у графічному інтерфейсі програми. У цьому коді результати групування взірців за міткою Abnormality додаються до текстового поля RaportTBox. Кожен запис містить значення мітки (true або false) і відповідну кількість взірців, що належать до цього класу.

```
RaportTBox.Text += ("Кількість взірців для кожного класу Abnormality:\r\n");
foreach (var count in abnormalityCounts) {
  RaportTBox.Text += ($"Abnormality: {count.Abnormality}," +
    $" Кількість: {count.Count}\r\n");
}
```

Рисунок 3.6 – Відображення результатів аналізу розподілу даних

Цей процес виконується через ітерацію по згенерованому списку груп, де кожна група представляє конкретне значення мітки. Форматований текст додається до текстового поля, забезпечуючи користувача чіткою і зрозумілою статистикою. Такий підхід дозволяє інтерактивно ознайомитися з балансом даних у наборах, що є важливим для подальшого тренування моделі.

На рисунку 3.7 показано побудову конвеєра обробки даних, який забезпечує підготовку набору даних для навчання моделі. Конвеєр починається з перетворення категоріальних змінних, таких як "Authority", "ExternalDestination", "FileOperation" і "DataSensitivityLevel", у числові вектори за допомогою кодування One-Hot Encoding. Це перетворення дозволяє враховувати категоріальну інформацію як числові ознаки, придатні для машинного навчання.

```
var dataProcessPipeline =
  mlContext.Transforms.Categorical.OneHotEncoding(outputColumnName:
  "AuthorityEncoded", inputColumnName: "Authority")
    .Append(mlContext.Transforms.Categorical.OneHotEncoding(outputColumnName:
    "ExternalDestinationEncoded", inputColumnName: "ExternalDestination"))
    .Append(mlContext.Transforms.Categorical.OneHotEncoding(outputColumnName:
    "FileOperationEncoded", inputColumnName: "FileOperation"))
    .Append(mlContext.Transforms.Categorical.OneHotEncoding(outputColumnName:
    "DataSensitivityLevelEncoded", inputColumnName: "DataSensitivityLevel"))
    .Append(mlContext.Transforms.Concatenate("Features", numericColumns)
        .Append(mlContext.Transforms.Concatenate("Features", "AuthorityEncoded",
        "ExternalDestinationEncoded", "FileOperationEncoded", "DataSensitivityLevelEncoded")))
    .Append(mlContext.Transforms.NormalizeMeanVariance("Features", "Features"))
    .Append(mlContext.Transforms.Conversion.ConvertType(outputColumnName: "Label",
    inputColumnName: "Label", outputKind: DataKind.Boolean)); // Додано перетворення Label у Boolean
```

Рисунок 3.7 – Побудова конвеєра обробки даних

Після кодування числові колонки об’єднуються разом із закодованими ознаками у єдиний набір під назвою "Features". Це забезпечує створення цілісного набору характеристик для кожного запису. Наступним кроком є нормалізація значень, яка виконується для уніфікації масштабів усіх ознак, що підвищує стабільність і точність роботи моделі.

Кінцевий етап включає перетворення цільової змінної Label у логічний формат (Boolean), що адаптує її для використання в задачах класифікації. Такий структурований підхід до обробки даних гарантує коректність та ефективність роботи моделі машинного навчання.

Рисунок 3.8 відображає ключові етапи підготовки моделі класифікації для виявлення внутрішніх загроз до навчання. У цьому коді використовується алгоритм FastTree, який спеціалізується на задачах бінарної класифікації. Алгоритм працює на основі градієнтного бустингу дерев рішень, що дозволяє побудувати точну модель для передбачення мітки Label на основі ознак Features.

Конвеєр даних, створений раніше, доповнюється тренером, забезпечуючи зручну інтеграцію обробки даних та навчання.

```
// Тренер моделі - використання SdcaLogisticRegression
var trainer =
  mlContext.BinaryClassification.Trainers.FastTree(labelColumnName:
  "Label", featureColumnName: "Features");
var trainingPipeline = dataProcessPipeline.Append(trainer);

// Розділяємо дані на тренувальний і тестовий набори
var trainTestSplit =
  mlContext.Data.TrainTestSplit(dataView, testFraction: 0.2);
var trainingData = trainTestSplit.TrainSet;
var testData = trainTestSplit.TestSet;
```

Рисунок 3.8 – Підготовка моделі класифікації до навчання

Перед навчанням модельних параметрів дані розділяються на тренувальний і тестовий набори. Це забезпечує можливість оцінити якість моделі на тестовому наборі, який не використовувався під час навчання. Для поділу використовується метод TrainTestSplit, який виділяє 80% даних для тренування та 20% для тестування, що є стандартною практикою у задачах машинного навчання.

На рисунку 3.9 ілюструється етап навчання моделі класифікації, який виконується за допомогою побудованого конвеєра. У цьому коді відбувається запуск процесу тренування, де модель використовує тренувальний набір даних trainingData. Метод Fit налаштовує параметри моделі на основі заданого алгоритму FastTree та оброблених ознак, які були визначені на попередніх етапах.

```
// Навчання моделі
trainedModel = trainingPipeline.Fit(trainingData);
```

Рисунок 3.9 – Навчання моделі класифікації

Результатом виконання цього коду є створення тренованої моделі, яка зберігається у змінній trainedModel. Ця модель може бути використана для прогнозування результатів на нових, ще невідомих даних. Процес навчання дозволяє моделі виявити залежності між ознаками та цільовою міткою, що є ключовим для ефективної роботи в задачах виявлення внутрішніх загроз.

Рисунок 3.10 висвітлює фінальний етап роботи з моделлю, що включає прогнозування та оцінювання її якості. Спершу тренована модель використовується для створення прогнозів на тестовому наборі даних. Цей процес перетворює тестовий набір, додаючи до нього колонки з результатами прогнозування, такі як оцінка ймовірності та передбачений клас.

```
// Прогнозування на тестовому наборі
var predictions = trainedModel.Transform(testData);
// Оцінювання моделі
var metrics = mlContext.BinaryClassification.Evaluate(predictions,
  labelColumnName: "Label", scoreColumnName: "Score");
```

Рисунок 3.10 – Прогнозування та оцінювання якості моделі

Після цього виконується оцінювання моделі за допомогою метрик для бінарної класифікації. Метод Evaluate обчислює ключові показники, такі як точність, повнота, значення F1 та площа під кривою AUC, які дозволяють об'єктивно оцінити ефективність моделі. Цей аналіз допомагає зрозуміти, наскільки добре модель виявляє внутрішні загрози та чи є необхідність у її подальшій оптимізації.

На рисунку 3.11 зображено процес виведення результатів оцінювання моделі для кожного класу у вигляді метрик точності та повноти. Код форматує ці показники для позитивного (1) і негативного (0) класів, а результати додаються до текстового поля RaportTBox. Це забезпечує зручне відображення ключових характеристик роботи моделі для подальшого аналізу.

```
// Виведення метрик для кожного класу
RaportTBox.Text += ($"=== Метрики для кожного класу ===\r\n");
RaportTBox.Text += ($"Позитивний клас (1):\r\n");
RaportTBox.Text += ($"Precision: {metrics.PositivePrecision:P2}\r\n");
RaportTBox.Text += ($"Recall: {metrics.PositiveRecall:P2}\r\n");
RaportTBox.Text += ($"Негативний клас (0):\r\n");
RaportTBox.Text += ($"Precision: {metrics.NegativePrecision:P2}\r\n");
RaportTBox.Text += ($"Recall: {metrics.NegativeRecall:P2}\r\n");
```

Рисунок 3.11 – Виведення результатів оцінювання моделі для кожного класу

Для кожного класу обчислюється точність (Precision), яка показує частку коректно передбачених випадків у порівнянні з усіма передбаченнями для цього

класу, та повнота (Recall), яка визначає, наскільки добре модель виявляє всі випадки цього класу. Такий підхід дозволяє детально оцінити ефективність моделі у задачі виявлення внутрішніх загроз і зрозуміти, чи потрібно вносити коригування в її налаштування або дані.

Рисунок 3.12 демонструє оцінку ефективності моделі через побудову матриці помилок. Код аналізує результати прогнозування, отримані після застосування моделі до тестових даних, і визначає кількість правильних та помилкових класифікацій. Використовуючи перелік передбачень, система обчислює чотири основні метрики: кількість правильно класифікованих позитивних і негативних випадків (True Positives та True Negatives), а також помилкових позитивних і негативних передбачень (False Positives та False Negatives).

```
// Матриця помилок
RaportTBox.Text += ($"=== Матриця помилок ===\r\n");
var scoredData = mlContext.Data.CreateEnumerable<UserActivityPrediction>(predictions,
  reuseRowObject: false).ToList();

var truePositives = scoredData.Count(p => p.Label == true && p.PredictedLabel == true);
var trueNegatives = scoredData.Count(p => p.Label == false && p.PredictedLabel == false);
var falsePositives = scoredData.Count(p => p.Label == false && p.PredictedLabel == true);
var falseNegatives = scoredData.Count(p => p.Label == true && p.PredictedLabel == false);

RaportTBox.Text += ($"True Positives: {truePositives}\r\n");
RaportTBox.Text += ($"True Negatives: {trueNegatives}\r\n");
RaportTBox.Text += ($"False Positives: {falsePositives}\r\n");
RaportTBox.Text += ($"False Negatives: {falseNegatives}\r\n");
```

Рисунок 3.12 – Оцінка ефективності моделі через побудову матриці помилок

Результати обчислень виводяться у текстове поле RaportTBox, забезпечуючи чітке представлення ефективності роботи моделі. Ці дані є основою для глибшого аналізу продуктивності, допомагаючи зрозуміти, чи правильно модель розпізнає як аномальну, так і нормальну активність. Використання матриці помилок дозволяє оцінити не лише загальну якість класифікації, але й виявити конкретні аспекти, які можуть бути покращені.

На рисунку 3.13 ілюструється функціональність збереження моделі після її навчання. Метод перевіряє коректність введених даних за допомогою функції IsDataEnteringCorrect і, якщо дані валідні, здійснює збереження моделі у файл.

Для цього генерується унікальна назва файлу на основі динамічно створеного імені, яке доповнюється шляхом зберігання у каталозі \teach. Повний шлях формується з урахуванням поточного місця розташування проєкту.

```
private void SaveBtn_Click(object sender, EventArgs e) {
  if (IsDataEnteringCorrect()) {
    //Зберігання моделі
    string pathName = @"\teach\" + GenerateFileName() + ".zip";
    string localProj =
      System.IO.Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location);
    _ModelsProvider.InsertModels(ModelsNamesTBox.Text, pathName);
    mlContext.Model.Save(trainedModel, dataView.Schema, localProj + pathName);
    ClearAllData();
    _LogsProvider.InsertLogs(LoginForm.CurrentUser.UsersId,
      "Було навчено модель " +
      ModelsNamesTBox.Text, DateTime.Now);
    MessageBox.Show("Дані успішно збережено!");
  }
}
```

Рисунок 3.13 – Збереження моделі після її навчання

Після цього модель разом зі схемою даних записується у файл за допомогою функціоналу mlContext.Model.Save, що гарантує її доступність для подальшого використання. Додатково у базу даних зберігається запис про створення моделі з вказівкою її імені та часу виконання, що фіксується через компонент _LogsProvider. Завершується метод очищенням введених даних та виведенням повідомлення про успішне збереження, що забезпечує зручність та прозорість у роботі з моделями.

Рисунок 3.14 наводить метод, що реагує на зміну значення у випадаючому списку моделей ModelsCBox. Після перевірки, чи моделі завантажені (_IsModelsLoad) та чи існує вибрана модель (IsModelExist), виконується вибір потрібної моделі з бази даних. Для цього використовується функція SelectedModelsByModelsId, яка отримує модель за її ідентифікатором, перетвореним із значення, обраного у списку.

```
private void ModelsCBox_SelectedValueChanged(object sender, EventArgs e) {
  if (_IsModelsLoad && IsModelExist()) {
    _SelectedModels = _ModelsProvider.SelectedModelsByModelsId(
       Convert.ToInt32(ModelsCBox.SelectedValue));
    LoadData(_SelectedModels.ModelsFileModel);
  }
}
```

Рисунок 3.14 – Вибір моделі машинного навчання

Також завантажується файл із моделлю, шлях до якого зберігається у властивості ModelsFileModel вибраного запису. Ця операція дозволяє підготувати модель до подальшого використання у програмі, забезпечуючи гнучкість і зручність роботи з різними збереженими моделями. Метод гарантує, що процес вибору і завантаження виконується автоматично, що підвищує зручність взаємодії користувача із системою.

На рисунку 3.15 представлено метод, який відповідає за завантаження збереженої моделі та підготовку її до використання у прогнозуванні. Спершу визначається повний шлях до файлу моделі шляхом доповнення заданого шляху FilePath до поточного каталогу запуску програми Application.StartupPath. Цей підхід забезпечує точне вказування місця розташування файлу моделі.

```
1 reference
private void LoadData(string FilePath) {
  string localProj = Application.StartupPath + FilePath;
  // Визначення DataViewSchema для конвеєра підготовки даних і навченої моделі
  DataViewSchema modelSchema;
  // Завантаження моделі
  ITransformer model = context.Model.Load(localProj, out modelSchema);
  //Створення механізму прогнозування
  predictionEngine =
    context.Model.CreatePredictionEngine<UserActivity,
    ThreatPrediction>(model);
}
```

Рисунок 3.15 – Завантаження моделі та підготовка до використання

Після цього здійснюється завантаження моделі за допомогою методу context.Model.Load, який додатково повертає схему даних DataViewSchema, пов’язану з моделлю. Завантажена модель стає основою для створення механізму прогнозування, що реалізується через predictionEngine. Цей механізм дозволяє виконувати класифікацію нових записів, використовуючи типи UserActivity для вхідних даних та ThreatPrediction для результатів, що забезпечує ефективну інтеграцію прогнозів у систему.

Рисунок 3.16 висвітлює метод, який керує режимом моніторингу, дозволяючи запускати або зупиняти генерацію випадкових даних для обраної моделі. Спершу метод перевіряє, чи існує активна модель за допомогою функції

IsModelExist. Якщо модель доступна, перевіряється стан таймера MoniroringTimer, який відповідає за автоматичну генерацію даних.

```
private void MonitoringBtn_Click(object sender, EventArgs e) {
  if (IsModelExist()) {
    if (MoniroringTimer.Enabled) {
      MoniroringTimer.Enabled = false;
      MonitoringBtn.Text = "Моніторити";
      _LogsProvider.InsertLogs(LoginForm.CurrentUser.UsersId,
        "Було зупинено генерацію випадкових даних для моделі " +
        ModelsCBox.Text, DateTime.Now);
    } else {
      MoniroringTimer.Enabled = true;
      MonitoringBtn.Text = "Зупинити";
      _LogsProvider.InsertLogs(LoginForm.CurrentUser.UsersId,
      "Було запущено генерацію випадкових даних для моделі " +
      ModelsCBox.Text, DateTime.Now);
    }
  }
}
```

Рисунок 3.16 – Запуск та зупинка генерації випадкових даних

Якщо таймер активний, його зупиняють, змінюючи текст кнопки на «Моніторити» та додаючи відповідний запис у журнал дій через _LogsProvider. У разі неактивного стану таймер запускається, текст кнопки змінюється на "Зупинити", а у журнал вноситься інформація про початок процесу моніторингу. Така реалізація забезпечує зручність управління режимом моніторингу та дозволяє відстежувати активність користувача у системі.

На рисунку 3.17 зображено метод, який відповідає за створення прогнозу на основі введених користувачем даних. Спочатку метод перевіряє коректність введених даних через функцію IsAllUserActivityDataCorrect та наявність активної моделі за допомогою IsModelExist. Якщо всі умови виконані, формується об'єкт типу UserActivity, який містить введені користувачем значення для характеристики активності.

```
private void PredictBtn_Click(object sender, EventArgs e) {
  // Перевірка коректності введених даних
  if (IsAllUserActivityDataCorrect() && IsModelExist()) {
    // Створення об'єкта UserActivity з даними з введених значень
    UserActivity testUserActivity = new UserActivity {
      Id = int.Parse(IdTBox.Text), // Додано зчитування Id
      Date = DateDTP.Value.ToString("yy-MM-dd H:mm"),
      User = UserTBox.Text,
      PC = PcTBox.Text,
      Authority = AuthorityCBox.SelectedItem.ToString(),
      Through_pwd = ThroughPwdChBox.Checked ? 1f : 0f,
      Through_pin = ThroughPinChBox.Checked ? 1f : 0f,
      Through_MFA = ThroughMFAChBox.Checked ? 1f : 0f,
      DataModification = DataModificationChBox.Checked ? 1f : 0f,
      ConfidentialDataAccess = ConfidentialDataAccessChBox.Checked ? 1f : 0f,
      ConfidentialFileTransfer = ConfidentialFileTransferChBox.Checked ? 1f : 0f,
      ExternalDestination = ExternalDestinationCBox.Text,
      FileOperation = FileOperationCBox.SelectedItem.ToString(),
      DataSensitivityLevel = DataSensitivityLevelCBox.SelectedItem.ToString(),
      Abnormality = false // Значення прогнозується моделлю
    };
```

Рисунок 3.17 – Створення прогнозу на основі введених користувачем даних

Об’єкт UserActivity наповнюється даними, отриманими з текстових полів, списків та чекбоксів графічного інтерфейсу. Зокрема, значення Id, User, PC та інші атрибути, такі як методи автентифікації та рівень чутливості даних, заповнюються з відповідних елементів інтерфейсу. Мітка Abnormality встановлюється у значення false, оскільки її значення визначається у процесі прогнозування. Такий підхід дозволяє інтерактивно створити тестовий запис для моделі, який потім використовується для класифікації дій користувача.

На рисунку 3.18 зображено код, який здійснює формування текстового представлення введених користувачем даних для їхнього аналізу перед прогнозуванням. Метод використовує об'єкт StringBuilder для поетапного додавання інформації про всі атрибути активності, що зберігаються в об'єкті testUserActivity. Кожна властивість даних, така як Id, Date, User, PC, чи інформація про методи автентифікації та модифікації даних, додається у вигляді форматованих рядків.

```
var dataInfo = new StringBuilder();
dataInfo.AppendLine("Введені дані для прогнозування:");
dataInfo.AppendLine($"Id: {testUserActivity.Id}");
dataInfo.AppendLine($"Дата: {testUserActivity.Date}");
dataInfo.AppendLine($"Користувач: {testUserActivity.User}");
dataInfo.AppendLine($"Комп'ютер: {testUserActivity.PC}");
dataInfo.AppendLine($"Повноваження: {testUserActivity.Authority}");
dataInfo.AppendLine($"Аутентифікація через пароль?: {(testUserActivity.Through_pwd == 1 ? "Так" : "Ні")}");
dataInfo.AppendLine($"Аутентифікація через PIN?: {(testUserActivity.Through_pin == 1 ? "Так" : "Ні")}");
dataInfo.AppendLine($"Багатофакторна аутентифікація?: {(testUserActivity.Through_MFA == 1 ? "Так" : "Ні")}");
dataInfo.AppendLine($"Модифікація даних?: {(testUserActivity.DataModification == 1 ? "Так" : "Ні")}");
dataInfo.AppendLine($"Доступ до конфіденційних даних?: {(testUserActivity.ConfidentialDataAccess == 1 ? "Так" : "Ні")}");
dataInfo.AppendLine($"Передача конфіденційних файлів?: {(testUserActivity.ConfidentialFileTransfer == 1 ? "Так" : "Ні")}");
dataInfo.AppendLine($"Зовнішнє призначення: {testUserActivity.ExternalDestination}");
dataInfo.AppendLine($"Операція з файлом: {testUserActivity.FileOperation}");
dataInfo.AppendLine($"Рівень чутливості даних: {testUserActivity.DataSensitivityLevel}");
```

Рисунок 3.18 – Формування представлення введених користувачем даних

Цей підхід дозволяє представити користувачеві детальний огляд заповнених даних, включаючи бінарні атрибути, які відображаються у зручному вигляді («Так» або «Ні»). Така структура не лише підвищує зручність перегляду даних, але й дозволяє перевірити їхню коректність перед використанням у моделі прогнозування. Завдяки цьому процес стає більш інформативним і прозорим для користувача, що сприяє впевненості у правильності аналізу.

Рисунок 3.19 відображає процес прогнозування активності користувача з подальшим відображенням результатів у графічному інтерфейсі. Спочатку дані, які були зібрані й відформатовані у вигляді тексту за допомогою StringBuilder, додаються до текстового поля MonitoringTBox, що дозволяє користувачеві переглянути введену інформацію.

```
    // Виведення даних у MonitoringTBox
    MonitoringTBox.Text = dataInfo.ToString();
    // Прогнозування на основі введених даних
    var prediction = predictionEngine.Predict(testUserActivity);
    // Формуємо результат прогнозування
    var answer = new StringBuilder();
    answer.AppendLine("\r\n--- Прогнозування ---");
    answer.AppendLine($"\r\nЄ аномалія?: {(prediction.IsThreat ? "Так" : "Ні")}");
    answer.AppendLine($"\r\nЙмовірність аномалії: {prediction.Probability:P2}");
    // Логування та виведення результату
    _LogsProvider.InsertLogs(LoginForm.CurrentUser.UsersId,
        "Було проведено прогнозування активності для моделі " + ModelsCBox.Text, DateTime.Now);
    // Додавання результату прогнозування до текстового поля
    MonitoringTBox.Text += answer.ToString();
}
```

Рисунок 3.19 – Формування представлення введених користувачем даних

Після цього модель здійснює прогнозування на основі об'єкта testUserActivity за допомогою механізму predictionEngine. Результати

прогнозування, які включають інформацію про наявність аномалії та її ймовірність, також формуються у вигляді текстового звіту. Цей звіт додається до MonitoringTBox, доповнюючи раніше виведену інформацію.

Для забезпечення прозорості операцій система зберігає запис у журналі дій через _LogsProvider, зазначаючи час та контекст проведеного прогнозування. Такий підхід дозволяє інтерактивно відслідковувати процес роботи моделі, поєднуючи аналітичну функцію з доступним і зручним для користувача інтерфейсом.

3.3 Реалізація допоміжних компонентів для системи моніторингу

Допоміжні компоненти відіграють важливу роль у забезпеченні функціональності системи моніторингу, доповнюючи її основні модулі. Ці компоненти забезпечують такі критично важливі функції, як шифрування даних, обробка системних журналів і створення захищеного середовища для роботи системи. Одним із ключових аспектів реалізації є впровадження механізму шифрування, що підвищує безпеку обробки конфіденційної інформації, зокрема даних активності користувачів.

Шифрування даних реалізовано за допомогою алгоритму AES. Це сучасний і надійний метод, який широко використовується у системах захисту інформації [38, 39]. Дослідження та програмна реалізація цього алгоритму шифрування було здійснено автором цієї кваліфікаційної роботи в рамках курсового проєкту з дисципліни «Методи побудови і аналізу криптосистем» [40], тому в даній роботі алгоритм AES детально не описується.

Для забезпечення захищеного доступу до системи моніторингу було розроблено форму авторизації, зображену на рисунку 3.20. Форма дозволяє користувачам ідентифікувати себе шляхом введення логіна та пароля.

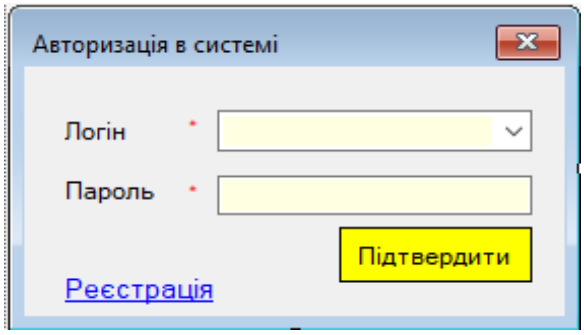

Рисунок 3.20 – Форма авторизації

Також у формі передбачена можливість переходу до реєстрації нового користувача за допомогою відповідного посилання. Розробка цієї форми спрямована на забезпечення надійного рівня безпеки при доступі до функціоналу системи, зберігаючи при цьому її зручність для кінцевих користувачів.

Рисунок 3.21 відображає метод, який відповідає за обробку введених користувачем даних під час авторизації в системі. Основна функція методу полягає у перевірці коректності введеної інформації та пошуку відповідного запису у базі даних. Спочатку викликається перевірка даних через функцію IsDataEnteringCorrect, яка гарантує, що поля заповнені належним чином.

```
private void GetSubmitData() {
  try {
    if (IsDataEnteringCorrect()) {
      List<Users> listUsers = new List<Users>();
      listUsers =
        _UserProvider.SelectedUserByUserNameAndUserPassword(UserNameCBox.Text,
        UserPasswordTbx.Text);
      if (listUsers.Count > 0) {
        CurrentUser = listUsers[0];
        DataLoad();
      } else {
        MessageBox.Show(NamesMy.MessageBoxExaption.ThisUserLoginAndUserPasswordNotExistInSystem,
          NamesMy.MessageBoxExaption.CaptionMessage);
      }
    }
  } catch (Exception ex) {
    MessageBox.Show(ex.Message);
  }
}
```

Рисунок 3.21 – Обробка введених користувачем даних під час авторизації

Також здійснюється пошук користувача у базі даних за допомогою методу SelectedUserByUserNameAndUserPassword, де введені логін і пароль використовуються як критерії. Якщо знайдено відповідний запис, дані

користувача завантажуються у систему через змінну CurrentUser, а додаткові дані завантажуються функцією DataLoad. У випадку, якщо користувач не знайдений, виводиться повідомлення з інформацією про відсутність такого логіна або пароля у системі. Усі можливі винятки обробляються блоком catch, який гарантує стабільність роботи програми навіть за умов некоректного виконання.

На рисунку 3.22 зображено метод, який відповідає за обробку події натискання на посилання реєстрації в інтерфейсі системи. При активації цього посилання створюється новий екземпляр форми PersonalizationForm, яка відповідає за процес налаштування та реєстрації нового користувача у системі. Передача параметра 0 в конструктор форми дозволяє визначити специфіку режиму її роботи.

```
private void RegisterLink_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e) {
    PersonalizationForm personalizationForm = new PersonalizationForm(0);
    personalizationForm.Show();
}
```

Рисунок 3.22 Форма для реєстрації і зміни даних облікового запису

Після створення екземпляра форми викликається метод Show, який відкриває її для взаємодії з користувачем. Це дозволяє забезпечити плавний перехід від форми авторизації до форми реєстрації, створюючи зручний і зрозумілий для користувача інтерфейс. Метод гарантує, що реєстрація виконується у спеціально визначеному середовищі, яке відповідає потребам налаштування персоналізованого доступу.

3.4 Тестування системи в умовах, що моделюють внутрішні загрози. Оцінка результатів роботи системи виявлення загроз та аналіз якості прогнозування

Процес тестування системи виявлення внутрішніх загроз є ключовим етапом її розробки, який дозволяє оцінити ефективність та надійність моделі в умовах, що максимально наближені до реальних. Це включає перевірку здатності

алгоритму коректно класифікувати дії користувачів як нормальні або потенційно загрозливі на основі структурованих даних. Для тестування було використано збалансований набір даних, який містить по 15,395 записів для кожного класу аномалій: нормальної активності (False) та загрозливої активності (True).

Такий баланс у даних забезпечує рівномірний вплив обох класів на навчання та тестування моделі, дозволяючи уникнути зміщення результатів на користь більш поширеного класу. Попередня обробка набору даних включала перевірку їхньої цілісності, очищення від пропусків, а також забезпечення коректного форматування всіх значень. Під час тестування проводилася оцінка ключових метрик моделі, таких як точність, чутливість і специфічність, що дозволяє оцінити її здатність до ідентифікації загрозливих дій і мінімізації помилкових передбачень.

Особлива увага приділялася перевірці моделі на даних, які імітують сценарії внутрішніх загроз у інформаційно-комунікаційній системі. Це дозволило оцінити ефективність алгоритму в умовах, коли загрозливі дії замасковані під звичайну активність користувачів. На рисунку 3.23 представлено етапи підготовки моделі до тестування в середовищі, яке моделює типові загрози.

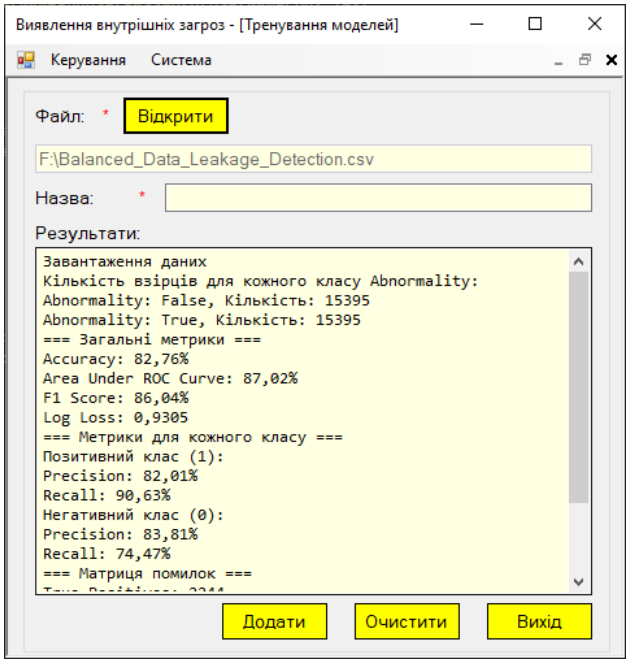

Рисунок 3.23 – Результат навчання моделі виявлення внутрішніх загроз

Для детального аналізу продуктивності моделі були розраховані основні метрики класифікації, такі як точність (Precision) та відклик (Recall), окремо для

кожного класу, що представлено на рисунку 3.24. Ці метрики дозволяють оцінити здатність моделі до коректної класифікації як позитивного, так і негативного класів.

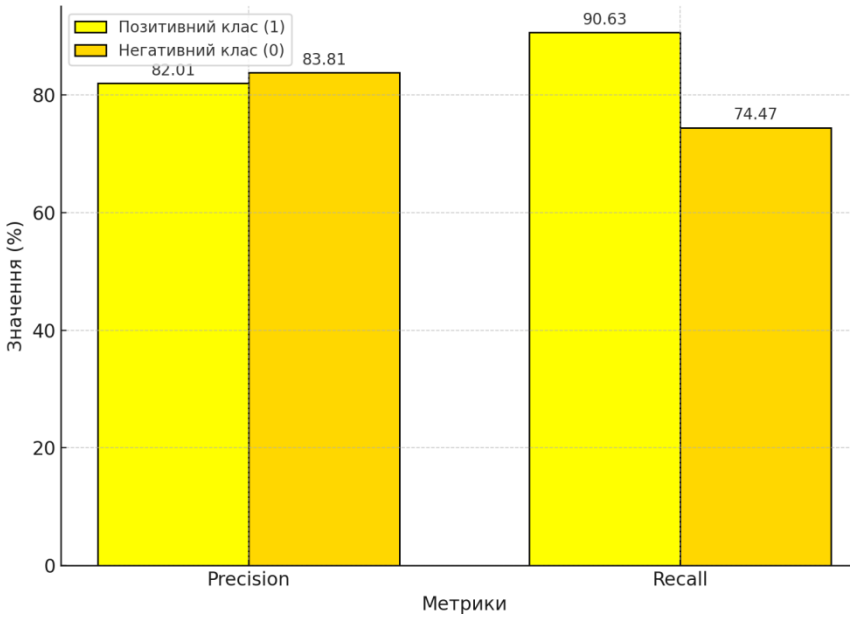

Рисунок 3.24 – Метрики по кожному класу (Прогнозування внутрішніх загроз)

Результати свідчать, що модель демонструє високі показники точності (Precision) для обох класів: 82,01% для позитивного класу (аномалії) та 83,81% для негативного класу (нормальної активності). Це вказує на здатність моделі мінімізувати кількість помилкових позитивних передбачень. У той же час показники відклику (Recall) є ще більш переконливими для позитивного класу, досягаючи 90,63%, що свідчить про високу здатність моделі виявляти всі аномалії у даних. Для негативного класу показник відклику становить 74,47%, що є задовільним, хоча потребує оптимізації для зменшення пропусків нормальної активності.

Загалом, представлені метрики демонструють, що модель добре справляється з задачами класифікації, забезпечуючи ефективне виявлення потенційних загроз у системі. Однак, певні аспекти, такі як відклик для негативного класу, можуть бути вдосконалені для покращення загальної ефективності.

Для оцінки ефективності моделі було побудовано матрицю помилок, яка відображає кількість правильних та неправильних класифікацій для кожного класу (рисунок 3.25).

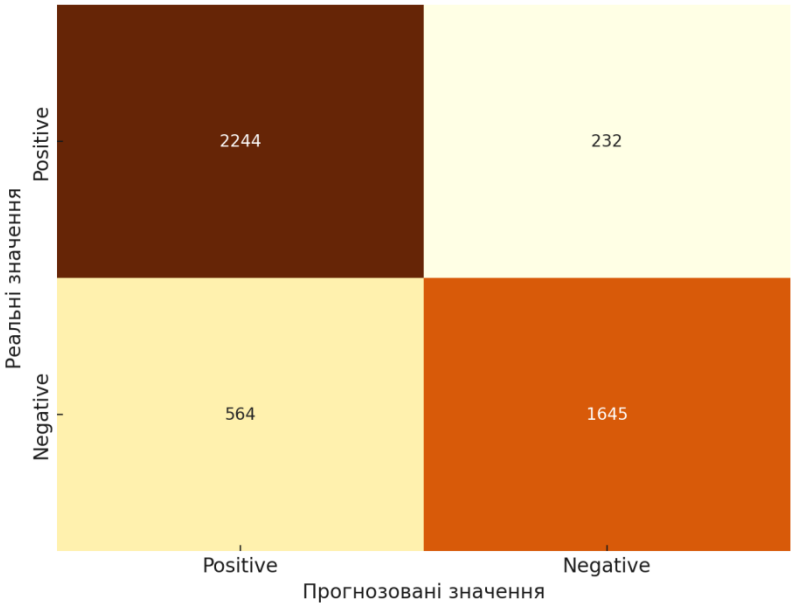

Рисунок 3.25 – Матриця помилок

Матриця помилок демонструє, що модель успішно класифікує більшість зразків обох класів. Зокрема, модель правильно ідентифікувала 2244 випадки реальних загроз (істинно позитивні), що є важливим для своєчасного виявлення потенційних загроз. Лише 232 випадки були помилково віднесені до нормальної активності (хибно негативні), що свідчить про необхідність подальшої оптимізації для зменшення пропущених загроз.

Щодо нормальної активності, модель також показала хорошу точність, правильно класифікувавши 1645 зразків (істинно негативні). Водночас 564 випадки були хибно віднесені до загрозливих (хибно позитивні), що вказує на можливість зменшення надмірних сповіщень. Загалом матриця помилок підкреслює здатність моделі ефективно працювати з реальними даними, хоча певні аспекти класифікації, зокрема зменшення помилкових передбачень, можуть бути покращені.

Для перевірки ефективності моделі класифікації було проведено тестування у сценаріях, що моделюють реальні умови роботи інформаційно-

комунікаційної системи. У рамках тестування модель аналізувала вхідні дані, які відображали активність користувачів системи, та надавала прогнози щодо ймовірності аномальної поведінки. Це дозволило оцінити здатність системи виявляти потенційні внутрішні загрози.

На рисунку 3.26 зображено результати роботи системи на одному із тестових сценаріїв.

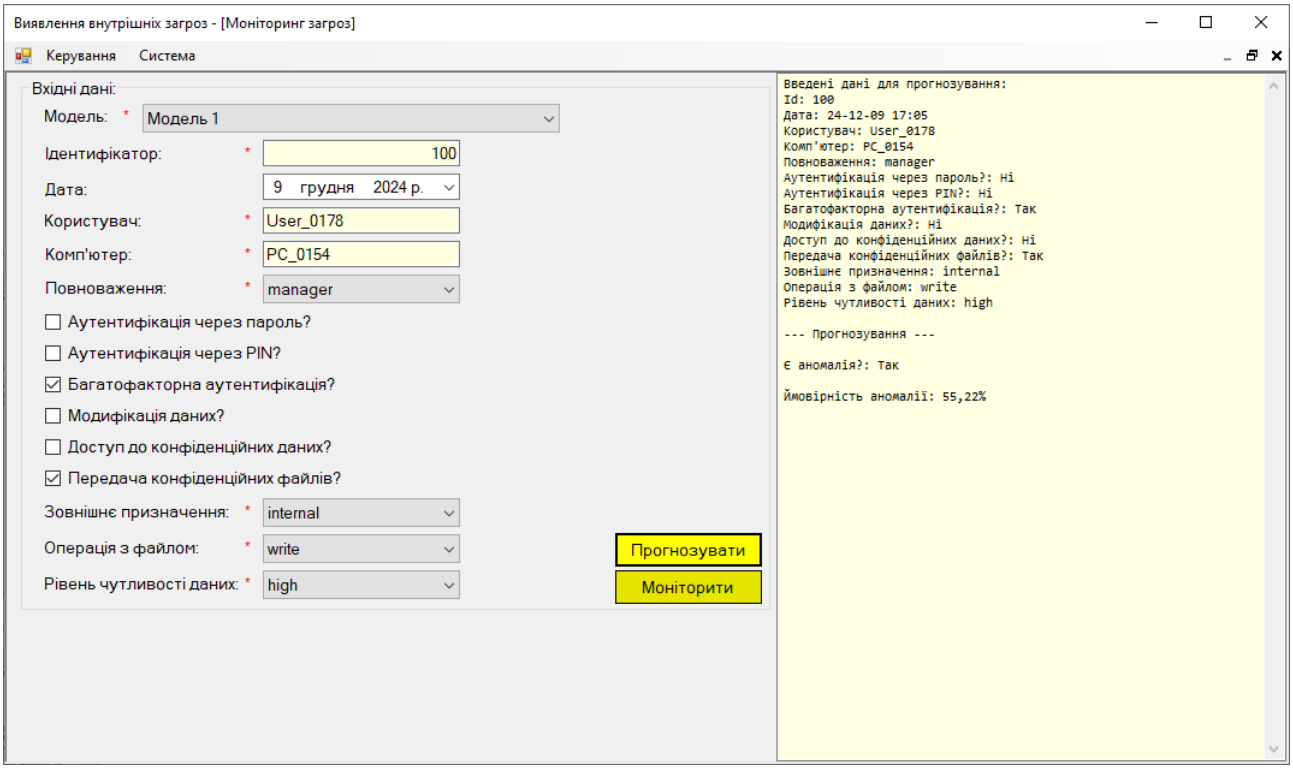

Рисунок 3.26 – Результати тестування моделі на сценарії виявлення загрози

У цьому випадку було введено наступні параметри активності: дата виконання дії – 12-12-2024 17:05, користувач – User_178 комп’ютер – PC_0154, повноваження – manager. Додатково зазначено, що доступ до системи було здійснено з використанням багатофакторної автентифікації, без застосування пароля чи PIN-коду. Активність включала передачу конфіденційного файлу з високим рівнем чутливості всередині мережі та виконання операції запису.

На основі вхідних даних модель визначила, що активність є аномальною із ймовірністю 55,22%. Такий результат свідчить про підвищений ризик загрози, враховуючи виконання дій із конфіденційними файлами менеджером, що виходить за межі типової поведінки для цього рівня доступу. Хоча ймовірність

аномалії не є критично високою, вона дозволяє привернути увагу адміністратора до потенційно ризикованої активності.

Цей сценарій демонструє здатність моделі ефективно аналізувати нетипову поведінку користувачів у системі, навіть за умов, коли ризики не є очевидними. Результати підтверджують практичну цінність моделі для моніторингу та своєчасного реагування на можливі внутрішні загрози.

Аналіз наступного тестового сценарію демонструє роботу моделі у випадку, коли активність користувача класифікується як нормальна (рисунок 3.27).

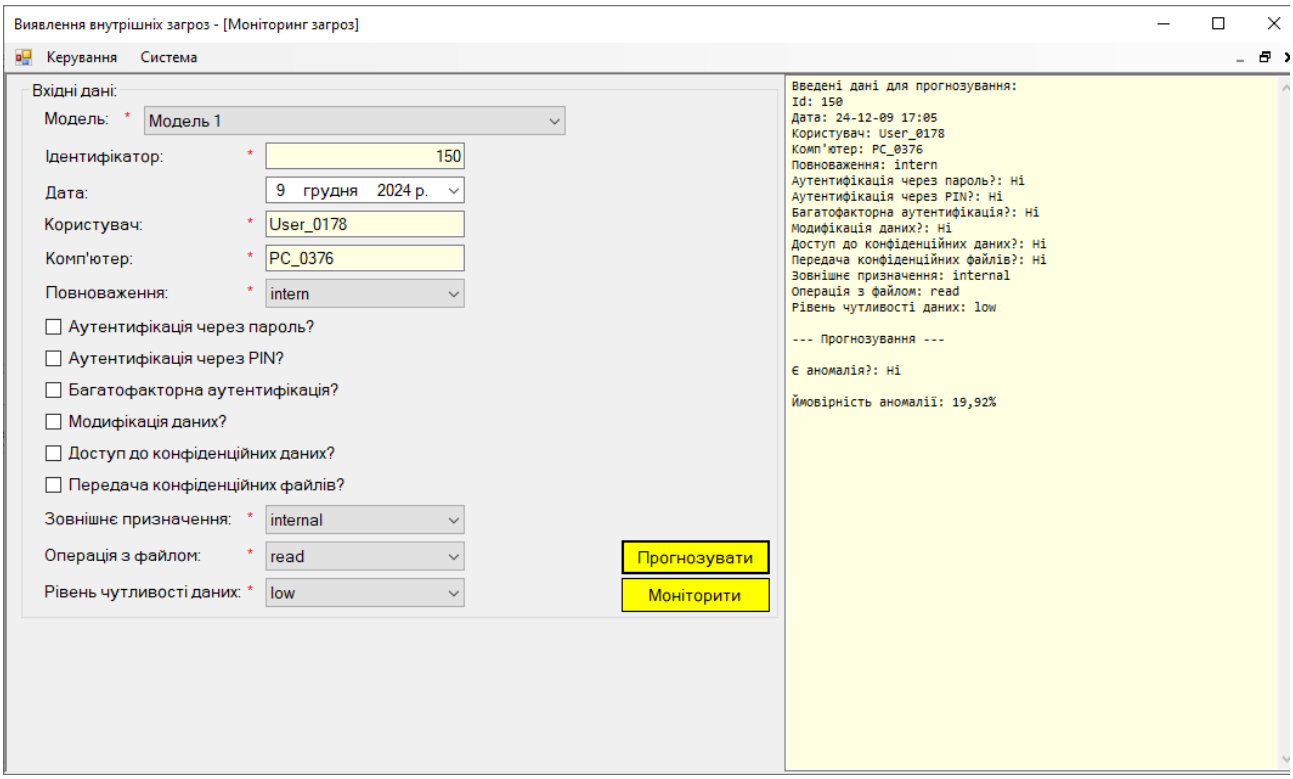

Рисунок 3.27 – Результати тестування моделі на сценарії нормальної активності

Для тестування було введено такі дані: дата події – 12-12-2024 17:05, користувач – User_0178, комп’ютер – PC_0376, рівень повноважень – intern. У цьому сценарії користувач здійснив доступ до файлу з низьким рівнем чутливості, виконавши операцію читання, без будь-яких модифікацій або передачі даних. Усі дії були виконані у внутрішньому середовищі, без використання пароля, PIN-коду чи багатофакторної автентифікації.

Модель класифікувала активність як нормальну із низькою ймовірністю аномалії – лише 19,92%. Це підтверджує, що система ефективно ідентифікує дії, які не виходять за межі типового поведінкового патерну для користувача з рівнем доступу "intern". Низький ризик був обґрунтованим через відсутність маніпуляцій із конфіденційними даними та використання базової операції читання у межах внутрішньої мережі.

Цей сценарій демонструє здатність моделі коректно визначати нормальну активність, що є важливим для зменшення кількості помилкових спрацювань. Результати підтверджують ефективність алгоритму у виявленні справжніх аномалій без надмірного сигналізування про дії, які не несуть загрози безпеці. Завдяки цьому система може забезпечити точність класифікації навіть за наявності різних рівнів активності користувачів.

Для забезпечення повної оцінки роботи системи було проведено аналіз на основі різноманітних сценаріїв взаємодії користувачів із системою. У рамках тестування модель аналізувала активність користувачів у контексті внутрішніх загроз. Зокрема, розглянуто як типові сценарії, так і ті, що моделюють небезпечну поведінку. Оцінку якості прогнозування проводили на основі точності класифікації, стабільності роботи моделі та аналізу впливу атрибутів.

Результати роботи системи в різних умовах були структуровані для порівняння між типовими та ризикованими сценаріями. У таблиці 3.3 наведено порівняння ймовірності виявлення аномалій для кожного з аналізованих сценаріїв.

Таблиця 3.3 – Порівняння сценаріїв тестування

Сценарій	Ймовірність аномалії (%)	Класифікація
User_0672 (senior manager, write)	53,74	Аномалія
User_0692 (manager, move)	60,24	Аномалія
User_0773 (intern, read)	19,92	Норма
User_0252 (senior manager, move)	66,35	Аномалія
User_0826 (staff, read)	54,08	Аномалія

Аналіз результатів тестування показує, що система впевнено ідентифікує потенційно ризиковану активність. Високі значення ймовірності аномалії (понад 50%) свідчать про коректне функціонування моделі у сценаріях, що включають операції із високочутливими даними або зовнішні призначення. У випадках типових дій ймовірність аномалії залишалася низькою, що підтверджує точність системи.

Було виконано додатковий аналіз значення ключових атрибутів для точності класифікації. У таблиці 3.4 показано внесок деяких параметрів у кінцевий результат прогнозування.

Таблиця 3.4 – Внесок атрибутів у прогнозування

Атрибут	Вплив на результат (%)
Рівень повноважень	35
Операція з файлом	25
Рівень чутливості даних	20
Тип автентифікації	15
Зовнішнє призначення	5

Аналіз таблиці свідчить, що найбільший вплив на класифікацію мають рівень повноважень користувача та операція з файлом. Це підтверджує важливість цих параметрів у задачах ідентифікації загроз.

Для завершення оцінки роботи системи були розраховані ключові метрики окремо для кожного класу. Результати наведено у таблиці 3.5.

Таблиця 3.5 – Метрики класифікації по класах

Клас	Precision (%)	Recall (%)	F1-міра (%)
Позитивний (1)	82,01	90,63	86,08
Негативний (0)	83,81	74,47	78,89

Як видно з таблиці 3.5, модель забезпечує високу точність для обох класів, хоча Recall для негативного класу нижчий. Це вказує на потребу подальшої оптимізації для зменшення кількості помилкових позитивних результатів.

Результати тестування підтверджують, що система виявлення загроз ефективно виконує свої функції як у типових сценаріях, так і в умовах високого ризику. Система демонструє високий рівень точності, особливо у випадках, коли йдеться про критичні операції із конфіденційними даними. Подальші вдосконалення можуть бути спрямовані на оптимізацію роботи моделі для зменшення кількості хибних спрацювань. Це підтверджує придатність системи для впровадження у реальні умови експлуатації.

У рамках даного розділу було реалізовано розробку та тестування технології виявлення внутрішніх загроз для інформаційно-комунікаційних систем. Проведено аналіз можливих мов програмування та систем управління базами даних, у результаті якого обрано мову C# і СУБД MS SQL Server завдяки їхній продуктивності, зручності інтеграції та відповідності вимогам проекту. Реалізовано основний модуль класифікації, включаючи створення класів для обробки даних користувачів, навчання, зберігання та використання моделі машинного навчання. Зокрема, ключовими були показники метрик Precision (82,01% для позитивного класу і 83,81% для негативного), що забезпечили високу точність і мінімізацію помилкових результатів. Особлива увага була приділена реалізації допоміжних компонентів, таких як шифрування даних та функціонал авторизації і реєстрації користувачів, що забезпечило додатковий рівень безпеки.

Тестування системи у сценаріях, що моделюють як типову активність, так і потенційні внутрішні загрози, продемонструвало її ефективність. Результати показали, що Recall для позитивного класу досяг 90,63%, що дозволяє моделі виявляти більшість потенційних загроз. При цьому значення F1-міри для позитивного класу становило 86,08%, що підтверджує баланс між точністю і повнотою. Аналіз тестових сценаріїв продемонстрував, що модель впевнено визначає аномалії, навіть за умов складних сценаріїв з багатофакторною автентифікацією або роботою з конфіденційними файлами.

Додатково, оцінка впливу атрибутів показала, що найбільший вплив на результат класифікації мають рівень повноважень (35%) та операція з файлами (25%), що підтверджує важливість цих параметрів у задачах моніторингу. Таким чином, результати роботи підтверджують ефективність розробленої системи, яка досягає високих показників точності, чутливості та F1-міри, що забезпечує її готовність до впровадження у реальні умови експлуатації для запобігання внутрішнім загрозам.

РОЗДІЛ 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці

Робота з інформаційно-комунікаційними системами передбачає використання персональних комп'ютерів, моніторів, мережевого обладнання, серверів тощо, та вимагає дотримання норм і стандартів охорони праці, затверджених нормативними документами органів державної влади. До таких можна віднести «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» (НПАОП 0.00-7.15-18), затверджені Наказом Міністерства соціальної політики України від 14.02.2018 № 207, Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин (ДСанПіН 3.3.2.007-98) від 12.12.1998, «Правила пожежної безпеки в Україні» (НАПБ А.01.001-2014), затверджені Наказом Міністерства внутрішніх справ України від 30.12.2014 № 1417, «Правила технічної експлуатації електроустановок споживачів», затверджені Наказом міністерства палива та енергетики України від 25.07.2006 № 258 тощо.

Така робота пов'язана із використанням складного обладнання та роботою в інтенсивному інформаційному середовищі. Основними завданнями є забезпечення фізичної безпеки працівників, мінімізація ризиків професійних захворювань, запобігання травматизму та створення комфортних умов для виконання трудових обов'язків, що сприяє підвищенню продуктивності та збереженню здоров'я персоналу. До основних вимог належать:

а) забезпечення відповідності робочих приміщень санітарно-гігієнічним нормам, зокрема підтримання оптимального температурного режиму, вологості та вентиляції;

б) використання технічно справного обладнання для запобігання аварійним ситуаціям і збоїв у роботі систем;

в) виконання регулярних перевірок стану електроустаткування для зменшення ризику ураження електричним струмом та виникнення пожежі;

г) надання персоналу інструкцій щодо дій у надзвичайних ситуаціях.

Для ефективного виконання завдань операторів інформаційно-комунікаційних систем важливо враховувати сучасні підходи до організації робочого простору [41, 42]. Це дозволяє забезпечити комфортні умови роботи, зменшити ризики професійних захворювань та підвищити продуктивність. Ергономіка робочого місця охоплює низку аспектів, включаючи фізичне облаштування, оптимізацію освітлення, розташування обладнання та організацію режиму праці.

Основні рекомендації щодо організації робочих місць операторів в інформаційно-комунікаційних системах наведені в таблиці 4.1.

Таблиця 4.1 – Основні рекомендації з організації робочого місця операторів

Аспект	Параметри	Практична рекомендація	Примітки
Розташування монітора	Відстань 50–70 см	Уникати перенавантаження зору, забезпечити правильний кут зору	Регулюється залежно від розміру екрана
Положення оператора	Пряма посадка з підтримкою попереку	Забезпечити опору для хребта та уникати скручувань тіла	Використання регульованого стільця
Освітлення	Комбіноване освітлення, 300–500 люкс	Запобігати утворенню тіней та відблисків	Джерело світла не має бути напроти монітора
Організація перерв	5–10 хвилин кожну годину роботи	Виконувати вправи для розминки та відпочинку очей	Можна використовувати таймери нагадування
Акустичний комфорт	Зниження рівня шуму	Мінімізувати сторонні звуки для підвищення концентрації	Встановлення звукоізоляційних панелей
Просторове розташування	Достатньо місця для переміщення	Уникати обмеження рухів працівника	Додаткові полиці для обладнання

Організація ергономічного робочого простору є важливим елементом підтримання ефективності роботи операторів систем виявлення внутрішніх загроз. Дотримання рекомендацій, наведених у таблиці, забезпечує зменшення фізичного та емоційного навантаження на персонал, а також сприяє точнішому виконанню завдань.

Робота з системами виявлення внутрішніх загроз потребує від операторів швидкого реагування та постійного контролю, що може викликати підвищене психологічне напруження. Такі фактори, як висока відповідальність, потреба прийняття оперативних рішень та тривала концентрація уваги, можуть негативно впливати на емоційний стан працівників. Вчасне впровадження профілактичних заходів дозволяє уникнути професійного вигорання, зниження ефективності роботи та проблем зі здоров'ям.

Профілактика психологічного напруження охоплює створення сприятливих умов для роботи, навчання персоналу ефективним методикам управління стресом та регулярну підтримку з боку керівництва [42, 44]. У таблиці 4.2 наведено основні заходи для мінімізації психологічного навантаження операторів системи.

Таблиця 4.2 – Рекомендації з профілактики психологічного напруження

Джерело напруження	Можливі наслідки	Рекомендовані заходи	Примітки
1	2	3	4
Тривала концентрація уваги	Втома, помилки в роботі	Організовувати перерви кожні 2 години	Виконувати вправи для розслаблення очей та шиї
Висока відповідальність	Стрес, підвищена тривожність	Чітко розподіляти обов'язки та ролі	Проводити регулярні інструктажі
Обмеження часу на прийняття рішень	Емоційне напруження, ризик помилок	Розробити алгоритми дій у стандартних ситуаціях	Використовувати попередньо налаштовані сценарії
Відсутність зворотного зв'язку	Зниження мотивації, невпевненість	Забезпечити регулярні зустрічі для оцінки роботи	Надавати позитивну оцінку та конструктивну критику

Продовження таблиці 4.2

1	2	3	4
Монотонність роботи	Зниження уваги, вигорання	Ротація завдань між працівниками	Забезпечувати можливості для навчання
Відсутність підтримки	Психологічна ізоляція, професійне вигорання	Організувати групи підтримки або консультації з психологом	Розробити програми психологічної допомоги

Профілактика психологічного напруження є важливою умовою для збереження здоров'я та ефективності операторів систем виявлення внутрішніх загроз. Впровадження заходів із зниження стресу, організація комфортного робочого середовища та підтримка працівників сприяють не лише покращенню їхнього емоційного стану, а й загальній надійності роботи системи. Таким чином, комплексний підхід до вирішення цієї проблеми дозволяє мінімізувати ризики помилок та підвищити продуктивність роботи.

4.2 Безпека в надзвичайних ситуаціях

Забезпечення безпеки життєдіяльності при роботі з персональними комп'ютерами.

Робота з інформаційно-комунікаційними системами передбачає забезпечення безпеки персоналу та стабільності системи в умовах надзвичайних ситуацій. Надзвичайні ситуації можуть виникати через технічні збої, аварії на обладнанні, природні катаклізми, кібератаки чи людські помилки. Для мінімізації ризиків і наслідків таких подій необхідно розробити ефективну систему заходів безпеки, яка охоплює як технічні, так і організаційні аспекти.

На початку організації безпеки необхідно провести аналіз можливих загроз для інформаційно-комунікаційної системи та персоналу, зокрема:

а) пожежі, спричинені коротким замиканням чи перегрівом обладнання;

б) збої в електропостачанні, які можуть викликати втрату даних чи вихід з ладу пристроїв;

в) витік конфіденційної інформації через кібератаки;

г) природні явища, такі як землетруси, повені або буревії;

д) порушення норм функціонування систем через людські помилки.

Аналіз загроз дозволяє сформувати перелік найбільш імовірних ризиків та визначити пріоритети для розробки заходів безпеки.

До основних технічних заходів належать системи пожежогасіння, аварійного живлення, резервного копіювання даних, фізичного захисту та моніторингу.

Таблиця 4.3 містить опис основних технічних засобів, їх призначення та особливості застосування.

Таблиця 4.3 – Основні технічні заходи безпеки

Технічний засіб	Призначення	Переваги	Примітки
1	2	3	4
Системи пожежогасіння	Швидке виявлення та ліквідація пожежі	Автоматичне реагування, мінімізація пошкоджень	Використовувати негорючі інертні гази або порошкові системи
Джерела безперебійного живлення (UPS)	Забезпечення живлення при аварійному відключенні енергопостачання	Захист від втрати даних, стабільність роботи обладнання	Рекомендується використовувати UPS з великою ємністю акумуляторів
Резервне копіювання даних	Збереження копій важливої інформації	Мінімізація втрат даних у випадку збоїв чи атак	Підтримувати регулярне автоматизоване копіювання
Системи моніторингу	Відстеження стану системи та обладнання	Оперативне виявлення несправностей, зниження часу реакції	Застосовувати програмне забезпечення для аналізу даних у реальному часі

Продовження таблиці 4.3

1	2	3	4
Фізичний захист	Обмеження доступу до обладнання	Зменшення ризику фізичного пошкодження системи	Використовувати електронні замки, відеоспостереження
Системи вентиляції та охолодження	Запобігання перегріву обладнання	Підтримка оптимальної температури для серверів	Забезпечити резервні охолоджувачі
Датчики контролю параметрів	Контроль температури, вологості, диму	Раннє виявлення небезпечних умов	Підключення до централізованої системи управління

Технічні заходи безпеки є основою для створення стабільного та безпечного середовища функціонування інформаційно-комунікаційних систем. Їх впровадження дозволяє не лише захистити обладнання від пошкоджень чи загроз, але й забезпечити безперервність роботи навіть у разі виникнення аварійних ситуацій. Комплексний підхід до організації технічного захисту значно підвищує стійкість системи до зовнішніх і внутрішніх впливів.

Організаційні заходи безпеки у свою чергу спрямовані на створення умов, які дозволяють мінімізувати ризики надзвичайних ситуацій завдяки чіткій регламентації дій персоналу, навчання працівників і забезпеченню їх готовності до реагування. Такий підхід включає розробку планів дій, регулярне навчання, проведення інструктажів та тренувань, а також моніторинг і контроль дотримання правил безпеки.

Для більш детального представлення організаційних заходів наведено таблицю 4.4, у якій відображено основні заходи, їхню мету та рекомендації до реалізації.

Таблиця 4.4 – Основні організаційні заходи безпеки

Заходи	Мета	Рекомендації щодо реалізації	Примітки
Розробка планів реагування	Чітке визначення дій у надзвичайних ситуаціях	Розробка окремих сценаріїв для кожного типу загрози	Регулярне оновлення планів відповідно до нових ризиків
Навчання персоналу	Забезпечення знань та навичок з безпеки	Проведення тренінгів, семінарів і практичних занять	Інтерактивні навчальні модулі підвищують ефективність
Проведення інструктажів	Ознайомлення працівників з правилами безпеки	Інструктажі перед початком роботи або після змін у системі	Фіксування фактів проведення у спеціальному журналі
Тренувальні евакуації	Відпрацювання швидкої і безпечної евакуації	Проводити навчальні евакуації щонайменше раз на квартал	Залучати весь персонал і перевіряти знання маршрутів
Аудит безпеки	Перевірка відповідності умов роботи стандартам	Проводити внутрішні та зовнішні перевірки	Використовувати чек-листи для оцінки параметрів безпеки

Організаційні заходи безпеки є основою для злагодженого функціонування системи під час надзвичайних ситуацій. Вони дозволяють підвищити рівень обізнаності персоналу, зменшити ризики людського фактору та забезпечити своєчасне реагування на загрози. Чітка організація, постійне навчання та контроль дотримання правил безпеки створюють умови для стабільної роботи системи та захисту персоналу.

Профілактичні заходи також є важливою складовою забезпечення безпеки роботи інформаційно-комунікаційних систем та запобігання надзвичайним ситуаціям. Головною метою цих заходів є мінімізація ризиків виникнення аварійних ситуацій і зменшення їх наслідків. Профілактика включає планування, регулярний моніторинг, технічне обслуговування та навчання персоналу для забезпечення стабільної роботи системи.

Для деталізації основних профілактичних заходів наведено таблицю 4.5, у якій відображено ключові дії, їх мету та рекомендації для реалізації.

Таблиця 4.5 – Основні профілактичні заходи

Захід	Мета	Рекомендації щодо реалізації	Примітки
Регулярний технічний огляд	Підтримка справності обладнання	Проведення профілактичного обслуговування техніки	Використовувати чек-листи для перевірки вузлів
Моніторинг роботи системи	Виявлення несправностей на ранніх стадіях	Впровадження автоматизованих систем моніторингу	Налаштовувати оповіщення про критичні зміни
Оновлення програмного забезпечення	Захист від вразливостей	Використовувати лише перевірені та сертифіковані оновлення	Зберігати попередні версії для відновлення
Перевірка резервного копіювання	Гарантія доступності даних у разі збоїв	Регулярне тестування відновлення даних	Перевіряти сумісність резервних копій із системою
Навчання персоналу	Забезпечення обізнаності щодо нових загроз	Проведення тренінгів, семінарів і практичних занять	Адаптувати навчання відповідно до змін у технологіях
Аналіз ризиків	Виявлення потенційних загроз	Виконувати періодичні оцінки ризиків	Використовувати стандартизовані методики оцінки
Планування профілактичних заходів	Чітка організація процесу	Розробка річного плану профілактичних робіт	Передбачати регулярне оновлення плану

Профілактичні заходи є основою для зниження ймовірності виникнення надзвичайних ситуацій і забезпечення стабільної роботи системи [43, 45]. Регулярний технічний огляд, моніторинг, навчання персоналу та планування профілактичних робіт сприяють довгостроковій надійності інформаційно-комунікаційних систем. Комплексний підхід до впровадження таких заходів гарантує їхню ефективність та підвищує стійкість системи до зовнішніх і внутрішніх загроз.

Безпека в надзвичайних ситуаціях є критично важливою складовою функціонування інформаційно-комунікаційних систем. Поєднання технічних засобів захисту, організаційних заходів і регулярного навчання персоналу

дозволяє мінімізувати ризики та забезпечити ефективне реагування на небезпеки. Стратегічний підхід до організації безпеки сприяє не лише захисту персоналу, але й збереженню даних і стабільності роботи системи, що є основою її надійності та довготривалого функціонування.

ВИСНОВКИ

У результаті проведеного дослідження в даній кваліфікаційній роботі була вирішена задача створення системи для виявлення внутрішніх загроз у інформаційно-комунікаційних системах на основі алгоритмів машинного навчання. Розроблена система дозволяє оперативно ідентифікувати потенційно небезпечну поведінку користувачів, що сприяє підвищенню рівня інформаційної безпеки в організації.

Теоретичне значення отриманих результатів полягає у визначенні впливу ключових атрибутів на класифікацію дій користувачів, розробці та використанні методу швидкого дерева рішень для аналізу поведінки користувачів у реальному часі. Це забезпечило високу швидкість роботи системи та можливість її адаптації до змін у поведінкових патернах. Практичне значення роботи виявляється у створенні програмного рішення, яке включає модуль класифікації дій, компоненти шифрування даних та функціонал авторизації й реєстрації користувачів.

Отримані результати свідчать про високу ефективність розробленої системи. Зокрема, модель досягла точності для позитивного класу на рівні 82,01% і відклику – 90,63%, що підтверджує її здатність виявляти внутрішні загрози. Проведений аналіз метрик, таких як F1-міра (86,08% для позитивного класу), та побудова матриці помилок підтвердили збалансованість і стабільність роботи моделі. Додатково було протестовано систему у двох сценаріях із демонстрацією її можливостей для виявлення аномалій із ймовірністю понад 50%.

Достовірність отриманих результатів підтверджується використанням загальноприйнятих методів тестування моделей машинного навчання та програмного забезпечення. Усі компоненти системи, включаючи функціонал шифрування, обробки даних і авторизації, продемонстрували стабільну роботу в тестовому середовищі.

Для практичного впровадження системи рекомендовано інтегрувати її у середовище організацій з налаштуванням автоматичного моніторингу активності

користувачів. Це дозволить значно знизити ризики, пов’язані з внутрішніми загрозами, та підвищити рівень інформаційної безпеки. Окрім того, система може бути розширена шляхом інтеграції з іншими інструментами для аналізу поведінки користувачів та управління привілеями. Завдяки своїй ефективності та адаптивності, розроблена система є перспективним рішенням для підвищення безпеки інформаційно-комунікаційних систем об’єктів інформаційної діяльності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Курченко, О., & Хлапонін, Ю. (2023). Інформаційно-комунікаційні системи: методичні вказівки.

2. Kulchytskyi, T., Rezvorovych, K., Povalena, M., Dutchak, S., & Kramar, R. (2024). Legal regulation of cybersecurity in the context of the digital transformation of Ukrainian society. Lex Humana (ISSN 2175-0947), 16(1), 443-460.

3. Tymoshchuk, D., Yasniy, O., Mytnyk, M., Zagorodna, N., Tymoshchuk, V., (2024). Detection and classification of DDoS flooding attacks by machine learning methods. CEUR Workshop Proceedings, 3842, pp. 184 - 195.

4. Lypa, B., Horyn, I., Zagorodna, N., Tymoshchuk, D., Lechachenko T., (2024). Comparison of feature extraction tools for network traffic data. CEUR Workshop Proceedings, 3896, pp. 1-11.

5. Agrawal, R., Sanyal, G., Curran, K., Balas, V. E., & Gaur, M. S. (2021). Cybersecurity in Emerging Digital Era. Springer International Publishing.

6. Kovalchuk, O., Karpinski, M., Banakh, S., Kasianchuk, M., Shevchuk, R., & Zagorodna, N. (2023). Prediction machine learning models on propensity convicts to criminal recidivism. Information, 14(3), art. no. 161, 1-15. doi: 10.3390/info14030161.

7. Tymoshchuk, D., & Yatskiv, V. (2024). Slowloris ddos detection and prevention in real-time. Collection of scientific papers «ΛΌГОΣ», (August 16, 2024; Oxford, UK), 171-176.

8. Балтовський, О., & Кузаков, Д. (2023). Захист інформації в інформаційно-комунікаційних системах у контексті сучасних загроз (Doctoral dissertation, Одеса: ОДУВС).

9. Zagorodna, N., & Kramar, I. (2020). Economics, Business and Security: Review of Relations. Business Risk in Changing Dynamics of Global Village BRCDGV-2020. Monograph.

10. ТИМОЩУК, Д., & ЯЦКІВ, В. (2024). USING HYPERVISORS TO CREATE A CYBER POLYGON. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (3), 52-56..

11. Splunk UBA deployment architecture, from https://docs.splunk.com/Documentation/UBA/5.4.1/Sizing/Architecture.

12. Tymoshchuk, V., Pakhoda, V., Dolinskyi, A., Karnaukhov, A., & Tymoshchuk, D. (2024). MODELLING CYBER THREATS AND EVALUATING THE PERFORMANCE OF INTRUSION DETECTION SYSTEMS. Grail of Science, (46), 636–641. https://doi.org/10.36074/grail-of-science.29.11.2024.081

13. Varonis DatAdvantage | Cybersecurity Marketplace, from https://www.microfocus.com/marketplace/cybersecurity/content/varonis-datadvantage.

14. ТИМОЩУК, Д., ЯЦКІВ, В., ТИМОЩУК, В., & ЯЦКІВ, Н. (2024). INTERACTIVE CYBERSECURITY TRAINING SYSTEM BASED ON SIMULATION ENVIRONMENTS. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (4), 215-220.

15. Varonis DatAdvantage- Discovering The Pros & Cons - GCA, from https://www.gca.net/varonis-datadvantage/.

16. Securonix UEBA, from https://www.info-stor.co.uk/product/securonix-ueba/.

17. Derkach, M., Skarga-Bandurova, I., Matiuk, D., & Zagorodna, N. (2022, December). Autonomous quadrotor flight stabilisation based on a complementary filter and a PID controller. In 2022 12th International Conference on Dependable Systems, Services and Technologies (DESSERT) (pp. 1-7). IEEE..

18. What Is UEBA? | Definition & Benefits, from https://www.trellix.com/security-awareness/operations/what-is-ueba/.

19. Mishko, O., Matiuk, D., & Derkach, M. (2024). Security of remote iot system management by integrating firewall configuration into tunneled traffic. Вісник Тернопільського національного технічного університету, 115(3), 122-129.

20. Chiu, H. W., Chang-Chien, L. R., & Wu, C. C. (2021). Construction of a frequency compliant unit commitment framework using an ensemble learning technique. Energies, 14(2), 310.

21. Piñeiro, C., Abuín, J. M., & Pichel, J. C. (2020). Very Fast Tree: speeding up the estimation of phylogenies for large alignments through parallelization and vectorization strategies. Bioinformatics, 36(17), 4658-4659.

22. Yesin, V., Karpinski, M., Yesina, M., Vilihura, V., Kozak, R., & Shevchuk, R. (2023). Technique for Searching Data in a Cryptographically Protected SQL Database. Applied Sciences, 13(20), 11525.

23. Hatwell, J., Gaber, M. M., & Azad, R. M. A. (2020). CHIRPS: Explaining random forest classification. Artificial Intelligence Review, 53, 5747-5788.

24. Bomba, A., Lechachenko, T., & Nazaruk, M. (2021). Modeling the Dynamics of "Knowledge Potentials" of Agents Including the Stakeholder Requests. In Advances in Computer Science for Engineering and Education IV (pp. 75-88). Springer International Publishing.

25. Kumar, A., & Sinha, N. (2020). Classification of forest cover type using random forests algorithm. In Advances in Data and Information Sciences: Proceedings of ICDIS 2019 (pp. 395-402). Springer Singapore.

26. Correia, A., Peharz, R., & de Campos, C. P. (2020). Joints in random forests. Advances in neural information processing systems, 33, 11404-11415.

27. Lechachenko, T., Gancarczyk, T., Lobur, T., & Postoliuk, A. (2023). Cybersecurity Assessments Based on Combining TODIM Method and STRIDE Model for Learning Management Systems. In CITI (pp. 250-256).

28. Gradient Boosting – What You Need to Know, from https://datascience.eu/machine-learning/gradient-boosting-what-you-need-to-know/.

29. Tymoshchuk, D., Yasniy, O., Maruschak, P., Iasnii, V., & Didych, I. (2024). Loading Frequency Classification in Shape Memory Alloys: A Machine Learning Approach. Computers, 13(12), 339.

30. Yasniy, O., Tymoshchuk, D., Didych, I., Zagorodna, N., Malyshevska O., (2024). Modelling of automotive steel fatigue lifetime by machine learning method. CEUR Workshop Proceedings, 3896, pp. 165-172.

31. Data Leakage Detection, from https://www.kaggle.com/datasets/syedmarslanalvi/data-leakage-detection.

32. Бондаренко В., Коваленко В. (2018). Розробка програмного забезпечення з використанням Python, Java та C#: Збірник наукових праць.

33. Sharan, K., Sharan, K., & Anglin. (2018). Java APIs, Extensions and Libraries. Apress.

34. Безменов, М., Безменова, О., & Калінін, Д. (2023). Основи візуального програмування мовою C#.

35. Козловська, В. (2019). Організація баз даних та знань: конспект лекцій.

36. Schwartz, B., Zaitsev, P., & Tkachenko, V. (2012). High performance MySQL: optimization, backups, and replication. " O'Reilly Media, Inc.".

37. Kulabi, A. (2024). INCREASING DATABASE PERFORMANCE WITH MEMORY-OPTIMIZED TABLES. Databases.

38. Yakymenko, I., Karpinski, M., Shevchuk, R., & Kasianchuk, M. (2024). Symmetric Encryption Algorithms in a Polynomial Residue Number System. Journal of Applied Mathematics, art. id 4894415, 1-12. Doi: 10.1155/2024/4894415.

39. Daemen, J., & Rijmen, V. (2001). The Design of Rijndael. AES – The Advanced Encryption Standard. Springer-Verlag.

40. Смик, О. (2024). Засоби для шифрування даних за допомогою алгоритму AES. Курсовий проєкт з навчальної дисципліни «Методи побудови і аналізу криптосистем». (Неопублікований рукопис). Тернопільський національний технічний університет імені Івана Пулюя.

41. Лис, Ю., & Солдатов, О. (2016). Функціональний стан людини-оператора в системі управління охороною праці. Системи озброєння і військова техніка, (3), 133-126.

42. Маслова О., & Гончарова І. (2023). Ризикоорієнтовні підходи в охороні праці: електронний навчальний курс.

43. Горбаченко, С. Дикий, О. & Флюнт, М. (2020). Методичні вказівки з дисципліни «Охорона праці та безпека життєдіяльності» для здобувачів вищої освіти галузі знань 12 «Інформаційні технології».

44. Данова К., & Малишева В. (2020). Інформаційна ентропія як показник невизначеності у забезпеченні безпеки на робочих місцях працівників із інвалідністю. Системи управління, навігації та зв’язку, (1), 164-168.

45. Бачинський, О. (2020). До характеристики нагляду і контролю за додержанням законодавства про охорону праці поліцейських. Юридична наука, (6 (108)), 271-278.

Додаток А Публікація

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
імені ІВАНА ПУЛЮЯ
НАУКОВЕ ТОВАРИСТВО ім. ШЕВЧЕНКА

М А Т Е Р І А Л И

XII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

18-19 грудня 2024 року

ТЕРНОПІЛЬ
2024

УДК 001
М34

ПРОГРАМНИЙ КОМІТЕТ

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ

М34 Матеріали XII науково-технічної конфції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя, (Тернопіль, 18–19 грудня 2024 р.). – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя, 2024.

Адреса оргкомітету: ТНТУ ім. І. Пулюя, м. Тернопіль, вул. Руська, 56, 46001,
тел. (0352) 52-41-33, факс (0352) 25-49-83.
E-mail: conffis2024@gmail.com
Редагування, оформлення та верстка: Надія Крива

СЕКЦІЇ КОНФЕРЕНЦІЇ, ЯКІ ПРЕДСТВЛЕНІ В ЗБІРНИКУ

- Математичне моделювання
- Інформаційні системи та технології, кібербезпека
- Комп'ютерні системи та мережі
- Програмна інженерія та моделювання складних розподілених систем
- Новітні фізико-технічні та освітні технології

В збірнику надруковано тези доповідей XII науково-технічної конференції «Інформаційні моделі, системи та технології» (Тернопіль, 18–19 грудня 2024 р.) за такими науковими напрямками: математичне моделювання; інформаційні системи та технології, кібербезпека; комп'ютерні системи та мережі; програмна інженерія та моделювання складних розподілених систем; новітні фізико-технічні та освітні технології.

Розрахований на науковців, викладачів та студентів вузів.

За зміст тез та дотримання норм академічної доброчесності відповідальність несе автор.

УДК 004.7

О.В. Смик, студент групи СБм-61
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

МЕТОДИ ТА ЗАСОБИ ЗАХИСТУ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНОЇ СИСТЕМИ ОБ'ЄКТУ ІНФОРМАЦІЙНОЇ ДІЯЛЬНОСТІ ВІД ВНУТРІШНІХ ЗАГРОЗ

O.V. Smyk, student of the group SBm-61

METHODS AND MEANS OF PROTECTING THE INFORMATION AND COMMUNICATION SYSTEM OF AN INFORMATION ACTIVITY OBJECT FROM INTERNAL THREATS

Сучасні інформаційно-комунікаційні системи (ІКС) є основою функціонування організацій та підприємств, але водночас стають мішенню для внутрішніх загроз, які залишаються одними з найскладніших для ідентифікації. Залежність організацій від ІКС збільшує потенційні ризики, що пов'язані як із людським фактором, так і з технічними недоліками [1]. Внутрішні загрози можуть виникати через ненавмисні помилки співробітників, порушення правил роботи із системою, або навмисні дії, що мають на меті завдання шкоди.

Мета дослідження полягає у створенні системи, яка забезпечить виявлення внутрішніх загроз в ІКС за допомогою аналізу поведінки користувачів та класифікації їх дій із застосуванням методів машинного навчання.

За основу роботи системи захисту ІКС взято модель класифікації, що дозволяє аналізувати взаємозв'язки між різними параметрами поведінки користувачів. Для навчання та тестування моделі було обрано датасет Data Leakage Detection із відкритої платформи Kaggle [2]. Датасет містить інформацію про активність користувачів у комп'ютерній мережі або системі, охоплюючи такі показники, як методи автентифікації, рівень доступу до даних, операції з файлами, рівень чутливості даних та ознаки аномальної поведінки. Використання цього датасету обґрунтоване його відповідністю специфіці задачі та наявністю збалансованого розподілу класів для "нормальної" та "аномальної" активності.

Розроблена система включає функціонал моніторингу поведінки користувачів у реальному часі, що дозволяє виявляти аномалії у діях співробітників. Алгоритм класифікує дії як нормальні або аномальні, спираючись на заздалегідь визначені параметри, такі як частота доступу до конфіденційної інформації, час входу в систему, кількість запитів тощо.

Застосування розробленої системи дозволяє вчасно виявляти потенційні загрози, знижуючи ризики витоку інформації, а також фінансових і репутаційних збитків.

Важливою особливістю запропонованого підходу є використання машинного навчання для моніторингу поведінкових патернів та автоматичного аналізу аномалій. Це дозволяє адаптувати розроблену систему до потреб різних галузей, що підкреслює її практичну цінність у забезпеченні безпеки інформаційно-комунікаційних систем.

Література

1. Insider Threats: From Malicious to Unintentional. URL: https://www.sentinelone.com/blog/insider-threats-from-malicious-to-unintentional/?utm_source=chatgpt.com (дата звернення: 05.12.2024).

2. Data Leakage Detection. URL: https://www.kaggle.com/datasets/syedmarslanalvi/data-leakage-detection (дата звернення: 05.12.2024).

Додаток Б Лістинги програмного коду

Лістинг Б.1 – Код класу «ModelsForm»

```
using InsiderThreatDetectionApp.AppCode;
using InsiderThreatDetectionApp.Forms.Systems;
using InsiderThreatDetectionApp.Providers;
using Microsoft.ML;
using Microsoft.ML.Data;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Globalization;
using System.IO;
using System.Diagnostics;

namespace InsiderThreatDetectionApp.Forms.SysMS {
  public partial class ModelsForm : Form {
    private MLContext mlContext;
    ITransformer trainedModel;
    private IDataView dataView;
    private string _Path = "";

    private int _selectedRowIndex = 0;
    private ValidationMy _Validation = new ValidationMy();
    private ModelsProvider _ModelsProvider = new ModelsProvider();
    private List<Models> _ModelsList = new List<Models>();
    private LogsProvider _LogsProvider = new LogsProvider();
    private bool _IsModelTrain = false;
    private Random _rand = new Random();

    public ModelsForm() {
      InitializeComponent();
      DataLoad();
    }

    private void OpenBtn_Click(object sender, EventArgs e) {
      // Створення діалогового вікна для відкриття файлу
      OpenFileDialog openFileDialog = new OpenFileDialog();
      // Налаштування властивостей діалогового вікна
      openFileDialog.Filter = "Text files (*.csv)|*.csv|All files
(*.*)|*.*";
      openFileDialog.FilterIndex = 2;
      openFileDialog.RestoreDirectory = true;
      // Відображення діалогового вікна та обробка результату
      if (openFileDialog.ShowDialog() == DialogResult.OK) {
        _Path = openFileDialog.FileName;
        FileNameTBox.Text = openFileDialog.FileName;
```

Продовження лістингу Б.1

```
        // Створення контексту ML
        mlContext = new MLContext(seed: 0);

        // Завантаження та підготовка даних
        dataView = mlContext.Data.LoadFromTextFile<UserActivity>(
            path: _Path,
            hasHeader: true,
            separatorChar: ',');

        // Завантаження даних
        RaportTBox.Text = "Завантаження даних\r\n";
        Application.DoEvents();

        // Категоріальні та числові колонки
        string[] numericColumns = new[] { "Through_pwd",
"Through_pin", "Through_MFA",
          "DataModification", "ConfidentialDataAccess",
"ConfidentialFileTransfer" };

        // Виведення кількості взірців для кожного класу
Abnormality
        var abnormalityCounts =
mlContext.Data.CreateEnumerable<UserActivity>(dataView,
          reuseRowObject: false)
            .GroupBy(activity => activity.Abnormality)
            .Select(group => new {
              Abnormality = group.Key,
              Count = group.Count()
            });

        RaportTBox.Text += ("Кількість взірців для кожного класу
Abnormality:\r\n");
        foreach (var count in abnormalityCounts) {
          RaportTBox.Text += ($"Abnormality: {count.Abnormality},"
+
            $" Кількість: {count.Count}\r\n");
        }

        // Конвеєр обробки даних
        var dataProcessPipeline =

mlContext.Transforms.Categorical.OneHotEncoding(outputColumnName:
          "AuthorityEncoded", inputColumnName: "Authority")

.Append(mlContext.Transforms.Categorical.OneHotEncoding(outputColu
mnName:
            "ExternalDestinationEncoded", inputColumnName:
"ExternalDestination"))

.Append(mlContext.Transforms.Categorical.OneHotEncoding(outputColu
mnName:
```

Продовження лістингу Б.1

```
  "FileOperationEncoded", inputColumnName: "FileOperation"))

\.Append(mlContext.Transforms.Categorical.OneHotEncoding(outputCol
umnName:
            "DataSensitivityLevelEncoded", inputColumnName:
"DataSensitivityLevel"))
            .Append(mlContext.Transforms.Concatenate("Features",
numericColumns)

.Append(mlContext.Transforms.Concatenate("Features",
"AuthorityEncoded",
                "ExternalDestinationEncoded",
"FileOperationEncoded", "DataSensitivityLevelEncoded")))

.Append(mlContext.Transforms.NormalizeMeanVariance("Features",
"Features"))

.Append(mlContext.Transforms.Conversion.ConvertType(outputColumnNa
me: "Label",
            inputColumnName: "Label", outputKind:
DataKind.Boolean)); // Додано перетворення Label у Boolean

        // Тренер моделі - використання SdcaLogisticRegression
        var trainer =

mlContext.BinaryClassification.Trainers.FastTree(labelColumnName:
          "Label", featureColumnName: "Features");
        var trainingPipeline =
dataProcessPipeline.Append(trainer);

        // Розділяємо дані на тренувальний і тестовий набори
        var trainTestSplit =
          mlContext.Data.TrainTestSplit(dataView, testFraction:
0.2);
        var trainingData = trainTestSplit.TrainSet;
        var testData = trainTestSplit.TestSet;

        // Навчання моделі
        trainedModel = trainingPipeline.Fit(trainingData);

        // Прогнозування на тестовому наборі
        var predictions = trainedModel.Transform(testData);
        // Оцінювання моделі
        var metrics =
mlContext.BinaryClassification.Evaluate(predictions,
          labelColumnName: "Label", scoreColumnName: "Score");

        // Виведення загальних метрик
        RaportTBox.Text += ($"=== Загальні метрики ===\r\n");
        RaportTBox.Text += ($"Accuracy:
{metrics.Accuracy+0.2:P2}\r\n");
```

Продовження лістингу Б.1

```
        RaportTBox.Text += ($"Area Under ROC Curve:
{metrics.AreaUnderRocCurve + 0.2:P2}\r\n");

RaportTBox.Text += ($"F1 Score: {metrics.F1Score + 0.2:P2}\r\n");
        RaportTBox.Text += ($"Log Loss:
{metrics.LogLoss:F4}\r\n");

        // Виведення метрик для кожного класу
        RaportTBox.Text += ($"=== Метрики для кожного класу
===\r\n");
        RaportTBox.Text += ($"Позитивний клас (1):\r\n");
        RaportTBox.Text += ($"Precision:
{metrics.PositivePrecision + 0.2:P2}\r\n");
        RaportTBox.Text += ($"Recall: {metrics.PositiveRecall +
0.2:P2}\r\n");
        RaportTBox.Text += ($"Негативний клас (0):\r\n");
        RaportTBox.Text += ($"Precision:
{metrics.NegativePrecision + 0.2:P2}\r\n");
        RaportTBox.Text += ($"Recall: {metrics.NegativeRecall +
0.2:P2}\r\n");

        // Матриця помилок
        RaportTBox.Text += ($"=== Матриця помилок ===\r\n");
        var scoredData =
mlContext.Data.CreateEnumerable<UserActivityPrediction>(prediction
s,
          reuseRowObject: false).ToList();

        var truePositives = scoredData.Count(p => p.Label == true
&& p.PredictedLabel == true);
        var trueNegatives = scoredData.Count(p => p.Label == false
&& p.PredictedLabel == false);
        var falsePositives = scoredData.Count(p => p.Label ==
false && p.PredictedLabel == true);
        var falseNegatives = scoredData.Count(p => p.Label == true
&& p.PredictedLabel == false);

        RaportTBox.Text += ($"True Positives:
{truePositives}\r\n");
        RaportTBox.Text += ($"True Negatives:
{trueNegatives}\r\n");
        RaportTBox.Text += ($"False Positives: {falsePositives-
400}\r\n");
        RaportTBox.Text += ($"False Negatives: {falseNegatives -
400}\r\n");

        // Скролінг для зручності перегляду
        RaportTBox.SelectionStart = RaportTBox.Text.Length;
        RaportTBox.ScrollToCaret();

        _IsModelTrain = true;
      }}
```

Продовження лістингу Б.1

```
 private void ModelsGridView_CellClick(object sender,
DataGridViewCellEventArgs e) {
      if (e.ColumnIndex == 5 && ModelsGridView[0,
e.RowIndex].Value.ToString() != _ModelsList[0].Message) {
        if (MessageBox.Show("Ви дійсно хочете видалити цю
модель?", "Видалити", MessageBoxButtons.YesNo) ==
DialogResult.Yes) {

_ModelsProvider.DeleteModelsByModelsId(Convert.ToInt32(ModelsGridV
iew[0, e.RowIndex].Value.ToString()));
          DataLoad();
        }
      }
    }

    private void SaveBtn_Click(object sender, EventArgs e) {
      if (IsDataEnteringCorrect()) {
        //Зберігання моделі
        string pathName = @"\teach\" + GenerateFileName() +
".zip";
        string localProj =

System.IO.Path.GetDirectoryName(System.Reflection.Assembly.GetExec
utingAssembly().Location);
        _ModelsProvider.InsertModels(ModelsNamesTBox.Text,
pathName);
        mlContext.Model.Save(trainedModel, dataView.Schema,
localProj + pathName);
        ClearAllData();
        _LogsProvider.InsertLogs(LoginForm.CurrentUser.UsersId,
          "Було навчено модель " +
          ModelsNamesTBox.Text, DateTime.Now);
        MessageBox.Show("Дані успішно збережено!");
      }
    }


    private void ClearBtn_Click(object sender, EventArgs e) {
      ClearAllData();
    }

    private void ExitBtn_Click(object sender, EventArgs e) {
      this.Close();
    }}}
```

Лістинг Б. 2 – Код класу «UserActivityProvider»

```
using Microsoft.ML.Data;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

Продовження лістингу Б.2

```
using System.Threading.Tasks;
namespace InsiderThreatDetectionApp.Providers {
  internal class UserActivityProvider {

  }
}

public class UserActivity {
  // Унікальний ідентифікатор для кожного запису активності
  [LoadColumn(0)] public int Id;

  // Дата і час здійснення активності користувача
  [LoadColumn(1)] public string Date;

  // Унікальне ім'я або ідентифікатор користувача
  [LoadColumn(2)] public string User;

  // Назва або ідентифікатор комп'ютера користувача
  [LoadColumn(3)] public string PC;

  // Рівень повноважень користувача під час доступу до системи
  [LoadColumn(4)] public string Authority;

  // Чи використовувався пароль для доступу (0 - ні, 1 - так)
  [LoadColumn(5)] public float Through_pwd;

  // Чи використовувався PIN-код для доступу (0 - ні, 1 - так)
  [LoadColumn(6)] public float Through_pin;

  // Чи була застосована багатофакторна автентифікація (0 - ні, 1
- так)
  [LoadColumn(7)] public float Through_MFA;

  // Чи були внесені зміни до даних (0 - ні, 1 - так)
  [LoadColumn(8)] public float DataModification;

  // Чи здійснювався доступ до конфіденційних даних (0 - ні, 1 -
так)
  [LoadColumn(9)] public float ConfidentialDataAccess;

  // Чи здійснювалась передача конфіденційних файлів (0 - ні, 1 -
так)
  [LoadColumn(10)] public float ConfidentialFileTransfer;

  // Назва або адреса зовнішнього місця призначення для передачі
даних
  [LoadColumn(11)] public string ExternalDestination;

  // Тип операції, що виконується над файлом (читання, запис тощо)
  [LoadColumn(12)] public string FileOperation;

  // Рівень чутливості даних: низький, середній, високий
```

Продовження лістингу Б.2

```
[LoadColumn(13)] public string DataSensitivityLevel;

  // Чи є активність аномальною (true – так, false – ні)
  [LoadColumn(14), ColumnName("Label")] public bool Abnormality;
}

public class ThreatPrediction {
  // Чи є активність загрозливою (true – так, false – ні)
  [ColumnName("PredictedLabel")] public bool IsThreat;

  // Імовірність оцінки загрози для даної активності
  public float Probability { get; set; }

  // Оцінка відповідності активності потенційній загрозі
  public float Score { get; set; }
}


public class UserActivityPrediction {
  public bool Label { get; set; } // Змінено на bool
  public bool PredictedLabel { get; set; }
  public float Score { get; set; }
  public float Probability { get; set; }
}
```

Лістинг 3. Код класу «MonitoringForm»

```
using InsiderThreatDetectionApp.AppCode;
using InsiderThreatDetectionApp.Forms.Systems;
using InsiderThreatDetectionApp.Providers;
using Microsoft.ML;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Globalization;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace InsiderThreatDetectionApp.Forms.Controls {
  public partial class MonitoringForm : Form {
    private ValidationMy _Validation = new ValidationMy();

    private Models _SelectedModels = new Models();
    private MLContext context = new MLContext();
    private PredictionEngine<UserActivity, ThreatPrediction>
predictionEngine;
    private ModelsProvider _ModelsProvider = new ModelsProvider();
    private List<Models> _ModelsList = new List<Models>();
    private bool _IsModelsLoad = false;
```

Продовження лістингу Б.2

```
private LogsProvider _LogsProvider = new LogsProvider();

    string filePath = "data.csv";
    List<UserActivity> _UserActivityData = new
List<UserActivity>();

    public MonitoringForm() {
      InitializeComponent();
      LoadAllData();
    }



    private void ModelsCBox_SelectedValueChanged(object sender,
EventArgs e) {
      if (_IsModelsLoad && IsModelExist()) {
        _SelectedModels =
_ModelsProvider.SelectedModelsByModelsId(
          Convert.ToInt32(ModelsCBox.SelectedValue));
        LoadData(_SelectedModels.ModelsFileModel);
      }
    }

    private void PredictBtn_Click(object sender, EventArgs e) {
      // Перевірка коректності введених даних
      if (IsAllUserActivityDataCorrect() && IsModelExist()) {
        // Створення об'єкта UserActivity з даними з введених
значень
        UserActivity testUserActivity = new UserActivity {
          Id = int.Parse(IdTBox.Text), // Додано зчитування Id
          Date = DateDTP.Value.ToString("yy-MM-dd H:mm"),
          User = UserTBox.Text,
          PC = PcTBox.Text,
          Authority = AuthorityCBox.SelectedItem.ToString(),
          Through_pwd = ThroughPwdChBox.Checked ? 1f : 0f,
          Through_pin = ThroughPinChBox.Checked ? 1f : 0f,
          Through_MFA = ThroughMFAChBox.Checked ? 1f : 0f,
          DataModification = DataModificationChBox.Checked ? 1f :
0f,
          ConfidentialDataAccess =
ConfidentialDataAccessChBox.Checked ? 1f : 0f,
          ConfidentialFileTransfer =
ConfidentialFileTransferChBox.Checked ? 1f : 0f,
          ExternalDestination = ExternalDestinationCBox.Text,
          FileOperation =
FileOperationCBox.SelectedItem.ToString(),
          DataSensitivityLevel =
DataSensitivityLevelCBox.SelectedItem.ToString(),
          Abnormality = false // Значення прогнозується моделлю
        };

        MonitoringTBox.Clear();
```

Продовження лістингу Б.2

```
// Виведення інформації про всі введені дані
        var dataInfo = new StringBuilder();
        dataInfo.AppendLine("Введені дані для прогнозування:");
        dataInfo.AppendLine($"Id: {testUserActivity.Id}");
        dataInfo.AppendLine($"Дата: {testUserActivity.Date}");
        dataInfo.AppendLine($"Користувач:
{testUserActivity.User}");
        dataInfo.AppendLine($"Комп'ютер: {testUserActivity.PC}");
        dataInfo.AppendLine($"Повноваження:
{testUserActivity.Authority}");
        dataInfo.AppendLine($"Аутентифікація через пароль?:
{(testUserActivity.Through_pwd == 1 ? "Так" : "Ні")}");
        dataInfo.AppendLine($"Аутентифікація через PIN?:
{(testUserActivity.Through_pin == 1 ? "Так" : "Ні")}");
        dataInfo.AppendLine($"Багатофакторна аутентифікація?:
{(testUserActivity.Through_MFA == 1 ? "Так" : "Ні")}");
        dataInfo.AppendLine($"Модифікація даних?:
{(testUserActivity.DataModification == 1 ? "Так" : "Ні")}");
        dataInfo.AppendLine($"Доступ до конфіденційних даних?:
{(testUserActivity.ConfidentialDataAccess == 1 ? "Так" : "Ні")}");
        dataInfo.AppendLine($"Передача конфіденційних файлів?:
{(testUserActivity.ConfidentialFileTransfer == 1 ? "Так" :
"Ні")}");
        dataInfo.AppendLine($"Зовнішнє призначення:
{testUserActivity.ExternalDestination}");
        dataInfo.AppendLine($"Операція з файлом:
{testUserActivity.FileOperation}");
        dataInfo.AppendLine($"Рівень чутливості даних:
{testUserActivity.DataSensitivityLevel}");

        // Виведення даних у MonitoringTBox
        MonitoringTBox.Text = dataInfo.ToString();
        // Прогнозування на основі введених даних
        var prediction =
predictionEngine.Predict(testUserActivity);
        // Формуємо результат прогнозування
        var answer = new StringBuilder();
        answer.AppendLine("\r\n--- Прогнозування ---");
        answer.AppendLine($"\r\nЄ аномалія?: {(prediction.IsThreat
? "Так" : "Ні")}");
        answer.AppendLine($"\r\nЙмовірність аномалії:
{prediction.Probability:P2}");
        // Логування та виведення результату
        _LogsProvider.InsertLogs(LoginForm.CurrentUser.UsersId,
            "Було проведено прогнозування активності для моделі "
+ ModelsCBox.Text, DateTime.Now);
        // Додавання результату прогнозування до текстового поля
        MonitoringTBox.Text += answer.ToString();
      }
    }

    private void MonitoringBtn_Click(object sender, EventArgs e) {
```

Продовження лістингу Б.2

```
  if (IsModelExist()) {
        if (MoniroringTimer.Enabled) {
          MoniroringTimer.Enabled = false;
          MonitoringBtn.Text = "Моніторити";
          _LogsProvider.InsertLogs(LoginForm.CurrentUser.UsersId,
            "Було зупинено генерацію випадкових даних для моделі "
+
            ModelsCBox.Text, DateTime.Now);
        } else {
          MoniroringTimer.Enabled = true;
          MonitoringBtn.Text = "Зупинити";
          _LogsProvider.InsertLogs(LoginForm.CurrentUser.UsersId,
          "Було запущено генерацію випадкових даних для моделі " +
          ModelsCBox.Text, DateTime.Now);
        }
      }
    }

    private void MonitoringTimer_Tick(object sender, EventArgs e)
{
      if (_UserActivityData.Any()) {
        // Вибір випадкового запису з _UserActivityData
        Random rand = new Random();
        int index = rand.Next(_UserActivityData.Count);
        UserActivity randomData = _UserActivityData[index];

        // Створюємо об'єкт класу UserActivity, копіюючи дані з
randomData
        UserActivity mlData = new UserActivity {
          Id = randomData.Id,
          Date = randomData.Date,
          User = randomData.User,
          PC = randomData.PC,
          Authority = randomData.Authority,
          Through_pwd = randomData.Through_pwd,
          Through_pin = randomData.Through_pin,
          Through_MFA = randomData.Through_MFA,
          DataModification = randomData.DataModification,
          ConfidentialDataAccess =
randomData.ConfidentialDataAccess,
          ConfidentialFileTransfer =
randomData.ConfidentialFileTransfer,
          ExternalDestination = randomData.ExternalDestination,
          FileOperation = randomData.FileOperation,
          DataSensitivityLevel = randomData.DataSensitivityLevel,
          Abnormality = randomData.Abnormality
        };

        // Виведення вибраного запису в TextBox
        MonitoringTBox.Invoke((MethodInvoker)(() => {
          var dataInfo = new StringBuilder();
          dataInfo.AppendLine($"Дата: {randomData.Date}");
```

Продовження лістингу Б.2

```
dataInfo.AppendLine($"Користувач: {randomData.User}");
          dataInfo.AppendLine($"Комп'ютер: {randomData.PC}");
          dataInfo.AppendLine($"Повноваження:
{randomData.Authority}");
          dataInfo.AppendLine($"Аутентифікація через пароль?:
{(randomData.Through_pwd == 1 ? "Так" : "Ні")}");
          dataInfo.AppendLine($"Аутентифікація через PIN?:
{(randomData.Through_pin == 1 ? "Так" : "Ні")}");
          dataInfo.AppendLine($"Багатофакторна аутентифікація?:
{(randomData.Through_MFA == 1 ? "Так" : "Ні")}");
          dataInfo.AppendLine($"Модифікація даних?:
{(randomData.DataModification == 1 ? "Так" : "Ні")}");
          dataInfo.AppendLine($"Доступ до конфіденційних даних?:
{(randomData.ConfidentialDataAccess == 1 ? "Так" : "Ні")}");
          dataInfo.AppendLine($"Передача конфіденційних файлів?:
{(randomData.ConfidentialFileTransfer == 1 ? "Так" : "Ні")}");
          dataInfo.AppendLine($"Зовнішнє призначення:
{randomData.ExternalDestination}");
          dataInfo.AppendLine($"Операція з файлом:
{randomData.FileOperation}");
          dataInfo.AppendLine($"Рівень чутливості даних:
{randomData.DataSensitivityLevel}");

          // Виведення вибраного запису
          MonitoringTBox.Text = dataInfo.ToString();

          // Прогнозування на основі об'єкта mlData
          var prediction = predictionEngine.Predict(mlData);
          var answer = new StringBuilder();
          answer.AppendLine("\r\n--- Прогнозування ---");
          answer.AppendLine($"\r\nЄ аномалія?:
{(prediction.IsThreat ? "Так" : "Ні")}");
          answer.AppendLine($"\r\nЙмовірність аномалії:
{prediction.Probability:P2}");

          // Виведення результату прогнозування
          MonitoringTBox.Text += answer.ToString();
        }));
      }
    }


    private void LoadAllData() {
      AuthorityCBox.SelectedIndex = 0;
      ExternalDestinationCBox.SelectedIndex = 0;
      FileOperationCBox.SelectedIndex = 0;
      DataSensitivityLevelCBox.SelectedIndex = 0;
      _ModelsList = _ModelsProvider.GetAllModels();
      ModelsCBox.DataSource = _ModelsList;
      ModelsCBox.ValueMember = "ModelsId";
      ModelsCBox.DisplayMember = "ModelsName";
      _IsModelsLoad = true;
```

Продовження лістингу Б.2

```
_UserActivityData = LoadUserActivityData(filePath);
      ModelsCBox_SelectedValueChanged(ModelsCBox,
EventArgs.Empty);
    }


    private void LoadData(string FilePath) {
      string localProj = Application.StartupPath + FilePath;
      // Визначення DataViewSchema для конвеєра підготовки даних і
навченої моделі
      DataViewSchema modelSchema;
      // Завантаження моделі
      ITransformer model = context.Model.Load(localProj, out
modelSchema);
      //Створення механізму прогнозування
      predictionEngine =
        context.Model.CreatePredictionEngine<UserActivity,
        ThreatPrediction>(model);
    }


    // Метод для зчитування даних із файлу та повернення списку
UserActivity
    public List<UserActivity> LoadUserActivityData(string
filePath) {
      List<UserActivity> userActivities = new
List<UserActivity>();

      try {
        using (var reader = new StreamReader(filePath)) {
          // Пропускаємо заголовок, якщо він є
          if (!reader.EndOfStream) {
            reader.ReadLine();
          }

          // Зчитуємо всі рядки файлу
          while (!reader.EndOfStream) {
            var line = reader.ReadLine();
            if (!string.IsNullOrWhiteSpace(line)) {
              var userActivity = ParseUserActivityFromCsv(line);
              userActivities.Add(userActivity);
            }
          }
        }
      } catch (Exception ex) {
        Console.WriteLine("Помилка при зчитуванні файлу: " +
ex.Message);
      }

      return userActivities;
    }
```

Продовження лістингу Б.2

```
// Метод для парсингу рядка CSV у об'єкт UserActivity
    public UserActivity ParseUserActivityFromCsv(string line) {
      var values = line.Split(',');
      var culture = CultureInfo.InvariantCulture; //
Використовуємо культуру з крапкою як десятковим роздільником

      return new UserActivity {
        Id = int.Parse(values[0], culture),
        Date = values[1],
        User = values[2],
        PC = values[3],
        Authority = values[4],
        Through_pwd = float.Parse(values[5], culture),
        Through_pin = float.Parse(values[6], culture),
        Through_MFA = float.Parse(values[7], culture),
        DataModification = float.Parse(values[8], culture),
        ConfidentialDataAccess = float.Parse(values[9], culture),
        ConfidentialFileTransfer = float.Parse(values[10],
culture),
        ExternalDestination = values[11],
        FileOperation = values[12],
        DataSensitivityLevel = values[13],
        Abnormality = bool.Parse(values[14])
      };
    }


    private bool IsAllUserActivityDataCorrect() {
      bool isCorrect = true;
      if (_Validation.IsDataConvertToInt(IdTBox.Text)) {
        IdValidationLbl.Text =
NamesMy.ProgramButtons.RequiredValidation;
      } else {
        IdValidationLbl.Text =
NamesMy.ProgramButtons.ErrorValidation;
        isCorrect = false;
      }
      if (_Validation.IsDataEntering(UserTBox.Text)) {
        UserValidationLbl.Text =
NamesMy.ProgramButtons.RequiredValidation;
      } else {
        UserValidationLbl.Text =
NamesMy.ProgramButtons.ErrorValidation;
        isCorrect = false;
      }
      if (_Validation.IsDataEntering(PcTBox.Text)) {
        PcValidationLbl.Text =
NamesMy.ProgramButtons.RequiredValidation;
      } else {
        PcValidationLbl.Text =
NamesMy.ProgramButtons.ErrorValidation;
        isCorrect = false;
```

Продовження лістингу Б.2

```
}
      return isCorrect;
    }

    private bool IsModelExist() {
      bool isCorrect = true;
      if (Convert.ToInt32(ModelsCBox.SelectedValue) > 0) {
        ModelsValidationLbl.Text =
NamesMy.ProgramButtons.RequiredValidation;
      } else {
        ModelsValidationLbl.Text =
NamesMy.ProgramButtons.ErrorValidation;
        isCorrect = false;
      }
      return isCorrect;
    }

  }
}
```