Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Виявлення аномалій у поведінці користувачів
за допомогою Machine Learning
(тема)

Виконав:
студент 2 курсу, групи СШІм-20-3
Сушко А.О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник доц. Шевченко О.Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Сушку Антону Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Виявлення аномалій у поведінці користувачів за допомогою Machine Learning

затверджена наказом університету від 24 березня 20 22 р. № 414Ст
2. Термін подання студентом роботи до екзаменаційної комісії _____ _____ 20 ___ р.
3. Вихідні дані до роботи Наукові статті, технічні публікації, інформація з Інтернет-джерел щодо виявлення аномалій у поведінці користувачів за допомогою Machine Learning

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Вибір та обґрунтування алгоритмів для виявлення аномалій у поведінці користувачів за допомогою Machine Learning, його технологій та архітектури для програмної реалізації
3) Проведення виявлення аномалій у поведінці користувачів
4) Впровадження результатів та перспективи розвитку

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Рисунок 1 – Порівняння алгоритмів кластеризації; Рисунок 2 – Послідовність транзакцій; Рисунок 3 – Формат даних для аналізу користувача; Рисунок 4 – Результат пошуку аномалій серед користувачів; Рисунок 5 – Результат пошуку аномалій серед пристроїв; Рисунок 6 – Послідовність транзакцій з аномалією під номером 1673; Рисунок 7 – Приклад властивостей використаних для кластеризації

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання	28.03.2022	виконано
2	Аналіз предметної галузі	29.03.2022	виконано
3	Постановка задачі	01.04.2022	виконано
4	Аналіз методів виявлення аномалій у поведінці	06.04.2022	виконано
	користувачів за допомогою Machine Learning		
5	Виявлення характеристик аномалій у поведінці	10.04.2022	виконано
	користувачів		
6	Вибір технологій для програмної реалізації	14.04.2022	виконано
7	Програмна реалізація моделі	16.04.2022	виконано
8	Оформлення пояснювальної записки	25.04.2022	виконано
10	Надання пояснювальної записки на перевірку керівнику	01.05.2022	виконано
11	Захист кваліфікаційної роботи		

Дата видачі завдання 28 березня 20 22 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Шевченко О.Ю.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 51 с., 7 рис., 3 дод., 11 джерел.

ВИЯВЛЕННЯ АНОМАЛІЙ, ГЕОДАНІ, КЛАСТЕРИЗАЦІЯ, МАШИННЕ НАВЧАННЯ, НАВЧАННЯ БЕЗ УЧИТЕЛЯ, ПОСЛІДОВНІ ДАНІ.

Об'єкт дослідження – сучасні методи інтелектуального аналізу поведінки користувачів для пошуку аномалій.

Предмет дослідження – використання методів машинного навчання для інтелектуального аналізу поведінки користувачів.

Мета роботи – пошук оптимального алгоритму для виявлення аномалій у поведінці користувачів.

У результаті роботи був проведений аналіз технічної літератури, предметної галузі, виявлено проблемні місця та поставлено задачу розробки готового рішення. Були сформовані вимоги та досліджені вхідні дані. На основі цього було обрано оптимальні алгоритми машинного навчання для пошуку аномалій.

ABSTRACT

Explanatory note: 51 p., 7 fig., 3 ann., 10 sources.

ANOMALY DETECTION, CLUSTERING, GEODATA, MACHINE LEARNING, SEQUENT DATA, UNSUPERVISED LEARNING.

The object of research – modern methods of intellectual analysis of user behavior to find anomalies.

The subject of research is the use of machine learning methods for intellectual analysis of user behavior.

The aim of the work is to find the optimal algorithm for detecting anomalies in user behavior.

As a result of the work the analysis of technical literature, subject branch was carried out, problem places were revealed and the task of development of the ready decision was set. Requirements were formed and input data were studied. Based on this, the optimal machine learning algorithms for finding anomalies were selected.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень та термінів……...7
Вступ…….8
1 Аналіз предметної галузі……...10
1.1 Опис предметної галузі…….……10
1.2 Огляд структури задач виявлення аномалій……13
2 Постановка задачі……...16
2.1 Визначення вимог до функціоналу……..……..…..16
2.2 Програмні ресурси……..…….....18
3 Опис створеної програми……...21
3.1 Попередня обробка даних…….21
3.2 Вибір алгоритму……27
3.3 Нормалізація даних……..…….28
3.4 Пошук аномалій серед користувачів та пристроїв.……31
Висновки……38
Перелік джерел посилання……..….40
Додаток А Вихідний код програмної моделі……..42
Додаток Б Знімки екрану роботи програми……...…49
Додаток В Відомість кваліфікаційної роботи…….…51

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

Викид – у статистиці результат вимірювання, який виділяється із загальної вибірки;

ПЗ − програмне забезпечення;

ШІ − штучний інтелект – здатність інженерної системи обробляти, застосовувати та вдосконалювати здобуті знання та вміння;

AD – Anomaly Detection – виявлення аномалій – процес аналізу різноманітних даних для пошуку викидів або аномалій;

ML − машинне навчання − великий підрозділ штучного інтелекту, що вивчає методи побудови алгоритмів, здатних навчатися.

ВСТУП

У сучасному світі технології все більше беруть участь у всіх аспектах життя та мають тенденцію до швидкої адаптації до обставин. У тому числі для боротьби зі зловмисниками.

Багато сучасних компаній використовують сучасні системи управління та системи зберігання та обробки даних. У компаній, що працюють на різних ринках по всьому світі та мають багато клієнтів і користувачів, є необхідність захисту конфіденційних даних та коштів. Користувачі можуть вносити на свої рахунки кошти для проведення різних операцій і транзакцій з ними, прив'язувати банківські рахунки та картки для зручної та швидкої взаємодії.

Одна з проблем для таких систем це викрадення або сумісне ведення аккаунтів, тобто коли за допомогою одного аккаунту здійснюються операції різними людьми.

Для виявлення такого роду ситуацій можна використовувати сучасні технології замість ручної довгої праці, особливо коли об'єм даних дуже великий.

Машинне навчання – це вивчення різних даних і подальше використання їх навчання для прийняття важливих рішень. Машини повинні використовувати коди, які вже написані розробниками, з метою навчання. Розробникам завжди доведеться спочатку надавати вхідні дані, щоб моделі могли вчитися на цьому. Машинне навчання може виявляти аномалії у поведінці користувачів для виявлення порушників.

Виявлення аномалій – це процес виявлення несподіваних елементів або подій у наборах даних, які відрізняються від норми. Воно часто застосовується до немаркованих даних, що називається неконтрольованим виявленням аномалій. Виявлення аномалії має два основних припущення:

– аномалії зустрічаються в даних дуже рідко;

– їх характеристики значно відрізняються від звичайних екземплярів.

Існує два напрямки виявлення аномалій: виявлення викидів і виявлення новизни. Виявлення викидів також відоме як неконтрольоване виявлення аномалій, а виявлення новизни – як напівконтрольоване виявлення аномалій.

У контексті виявлення викидів, викиди/аномалії не можуть утворювати щільний кластер, оскільки наявні оцінювачі припускають, що викиди/аномалії розташовані в областях з низькою щільністю. Навпаки, у контексті виявлення новизни, новинки/аномалії можуть утворювати щільний кластер, якщо вони знаходяться в області низької щільності навчальних даних, що вважається нормальним у цьому контексті.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Опис предметної галузі

Виявлення аномалії – це будь-який процес, який знаходить викиди у наборі даних. Ці аномалії можуть вказувати на незвичайний мережевий трафік, виявити шахрайство або просто ідентифікувати дані для очищення перед аналізом[1].

У сучасному світі розподілених систем керування та моніторинг продуктивності системи є справою, хоча й необхідною. З сотнями чи тисячами елементів, які можна переглянути, виявлення аномалій може допомогти вказати, де відбувається помилка, покращуючи аналіз першопричин та швидко одержуючи технічну підтримку щодо проблеми. Виявлення аномалій допомагає відстежувати причину хаосу, виявляючи викиди та інформуючи відповідальні сторони, щоб діяти[4].

У IT підприємства виявлення аномалій зазвичай використовується для:

– очищення даних;

– виявлення вторгнення;

– виявлення шахрайства;

– моніторинг працездатності систем;

– виявлення подій у сенсорних мережах;

– порушення екосистеми;

– завдання виявлення аномалій.

Але навіть у цих поширених випадках використання, наведених вище, є деякі недоліки виявлення аномалій. З конференції Брема Стінвінкеля: «Системи виявлення аномалій (AD) створюються вручну експертами, які встановлюють пороги для даних, або створюються автоматично, вивчаючи доступні дані за допомогою машинного навчання»[3].

Створювати систему ручного виявлення аномалій складно. Для цього потрібне знання предметної області та вміння передбачення.

Для екосистеми, де дані змінюються з часом, це не може бути хорошим рішенням. Будівництво стіни, щоб захистити людей, працює, поки вони не знайдуть спосіб пройти через неї або обійти її. Коли система виходить з ладу, розробникам потрібно повернутися та вручну додати додаткові методи безпеки.

З огляду на інженерію хаосу, виявлення аномалій вручну є поганим рішенням, оскільки створює систему, яка не може адаптуватися (або є дорогою та несвоєчасною для адаптації).

Таким чином, машинне навчання відповідає цілям інженера для створення системи AD, яка:

– працює краще;

– адаптивний і вчасно;

– обробляє великі набори даних.

Незважаючи на ці переваги, виявлення аномалій за допомогою машинного навчання може працювати лише за певних умов.

Застосування машинного навчання для виявлення аномалій вимагає хорошого розуміння проблеми, особливо в ситуаціях з неструктурованими даними.

Структуровані дані вже мають на увазі розуміння проблемного простору. Аномальні дані може бути легко ідентифікувати, оскільки вони порушують певні правила. Якщо термометр ніколи не повинен показувати 300 градусів за Фаренгейтом, а дані показують, що він показує 300 градусів за Фаренгейтом, це аномалія. Є чіткий трешхолд, який зламано.

Виявлення шахрайства в ранніх алгоритмах аномалій могло спрацювати, оскільки дані, що містяться в них, мають значення. Дані були структурованими, тобто люди вже створили інтерпретаційні налаштування

для збору даних. Їхні дані мали значення, тому можна було створювати випадкові дерева та шукати шахрайство.

Однак зашумовані та неструктуровані дані, такі як зображення, закодовані як послідовність пікселів, або мова, закодована як послідовність символів, не мають із собою інтерпретації і роблять старі алгоритми непотрібними доки дані не стануть структурованими.

Також необхідний великий набір даних. Основоположним принципом будь-якої хорошої моделі машинного навчання є те, що для неї потрібні набори даних. Як і закон, якщо немає даних на підтвердження позову, позов не може бути задоволений в суді.

Для машинного навчання потрібні набори даних; висновки можна робити лише тоді, коли передбачення можуть бути підтверджені. Виявлення аномалій вигідно від ще більшої кількості даних, оскільки передбачається, що аномалії рідкісні.

Виявлення аномалій – це клас проблем, що набирають обертів у сфері комп’ютерної безпеки. Оскільки системи контролюються все ретельніше, а атаки стають все більш складними, традиційних систем попередження, заснованих на правилах, стає недостатньо. Таким чином, розглядаються методи виявлення аномалій на основі машинного навчання, щоб зробити системи моніторингу більш динамічними та адаптивними.

Мета полягає в тому, щоб виявити аномальну подію в інфраструктурі до того, як потенційна атака досягне своєї критичної точки. Нині системи моніторингу структуруються як «штучні імунні системи» (AIS): алгоритми, які вивчають представлення систем з огляду на історію минулих подій і подають попередження щоразу, коли щось відхиляється від базового рівня[7].

Цю формулювання виявлення аномалій можна інтерпретувати подібно до проблеми інтелекту даних «виявлення викидів». Це питання вирішується

за допомогою таких алгоритмів, як однокласні допоміжні векторні машини, кластерний аналіз або методи на основі автоматичного кодування.

1.2 Огляд структури задач виявлення аномалій

Ця робота зосереджена на виявленні аномалій у послідовностях: за допомогою потоку послідовностей навчити модель класифікувати кожну послідовність як нормальну чи аномальну. Ці обмеження зазвичай зустрічаються при обробці даних журналу дій з серверів: послідовності – це дії, які виконує кожен користувач, і мета – виявити користувачів, які мають підозрілу поведінку у їхніх діях. На порталах із дуже високим трафіком обсяг даних, який це представляє, настільки великий, що його складно структурувати і виявляти закономірності без допомоги машинного навчання. Нещодавно розроблений алгоритм Cortical Learning Algorithm (CLA) був успішно застосований для передбачення тимчасової послідовності, і вважається, що він має хороший потенціал для узагальнення на більш складні дані.

Виявлення аномалій або викидів – це пошук подій або спостережень, які рідко трапляються і можуть викликати підозру через те, що мають відмінності від тенденцій у даних.[5] Це може свідчити про деякі проблеми або рідкісні події та значення, наприклад, шахрайство з грошима, медичні проблеми, дефекти, несправності обладнання тощо. Цей зв'язок робить дуже цікавим можливість вибрати, які точки даних можна вважати аномаліями, оскільки виявлення цих подій, як правило, дуже цікаво з точки зору бізнесу.

Виявлення аномалій у поведінці користувачів це важливий механізм для бізнес-процесів у компаніях, що обробляють великий обсяг даних та мають справу з великою кількістю конфіденційних даних. Виявлення аномалій у

поведінці користувачів допомагає знаходити неявні підозрілі дії для запобігання втрат даних та коштів користувачів.

Способи навчання моделей можна поділити на два варіанти:

– навчання з вчителем;

– навчання без вчителя.

При навчання з вчителем дані позначаються «нормальні» або «аномалія».

Налаштування під наглядом є ідеальним налаштуванням. Це той випадок, коли набір даних надходить акуратно підготовлений для моделі з даних з усіма точками даних, позначеними як аномальні або нормальні. У цьому випадку всі аномальні точки відомі заздалегідь. Це означає, що є набори точок даних, які є аномальними, але не ідентифіковані як такі для навчання моделі.

Популярні алгоритми машинного навчання для структурованих даних:

– support vector machine learning;

– k-nearest neighbors (KNN);

– Bayesian networks;

– decision trees.

У навчанні без вчителя дані навчання не мають міток і складаються з «нормальних» і «аномальних» точок.

Набори даних у випадку навчання без вчителя не мають позначок класу. Немає ґрунтовної істини, від якої можна очікувати результату. Модель повинна показати модельєру, що є аномальним, а що нормальним[11].

Найбільш популярні задачі для навчання без вчителя це кластеризація, оцінка щільності та навчання репрезентації. У всіх цих випадках ми хочемо вивчити структуру наших даних і визначити потрібні нам класи або мітки, а не використовувати заздалегідь відомі[8].

У навчанні без вчителя потрібен інший набір інструментів для створення порядку в неструктурованих даних. У неструктурованих даних головна мета – створити кластери з даних, а потім знайти кілька груп, які не належать. Насправді всі алгоритми виявлення аномалій є певною формою приблизної оцінки щільності[2].

Популярні алгоритми машинного навчання для неструктурованих даних, на рисунку 1.1 наведено порівняння результатів деяких з них:

– self-organizing maps (SOM);

– K-means;

– C-means;

– one-class support vector machine;

– DBSCAN clustering.

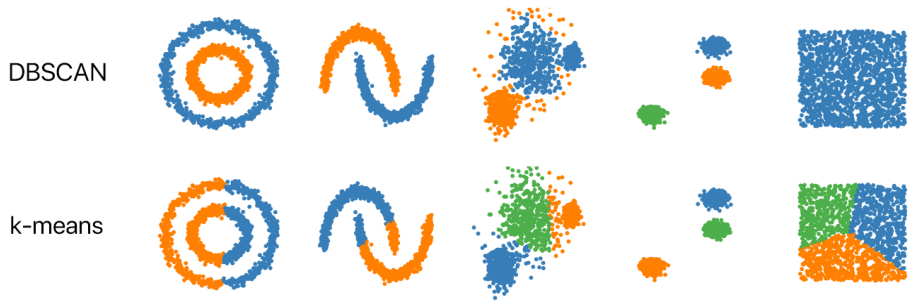


Рисунок 1.1 – Порівняння алгоритмів кластеризації

2 ПОСТАНОВКА ЗАДАЧІ

2.1 Визначення вимог до функціоналу

Для навчання алгоритму використовується навчальна вибірка даних Ця вибірка являє собою багатовимірний набір екземплярів даних. Кожний вимір у цьому наборі це властивість, яка передбачається предметною областю та потребами вирішуваної задачі.

Кластерний аналіз висуває наступні вимоги до даних:

– екземпляри не корелюють один між одним;

– має бути присутній нормальний розподіл;

– екземпляри мають бути стійкими до зовнішніх чинників;

– вибірка повинна бути однорідна.

Для визначеної задачі до використання кластерного аналізу можна сформулювати такі вимоги:

– дані повинні бути послідовними;

– кожна властивість повинна не корелювати з іншими;

– кластерний аналіз повинен виявити аномальні транзакції, що не відповідають у своїх значеннях іншим з одного набору.

Відповідно до сформульованої задачі, оптимальна вибірка повинна виглядати як набір багатовимірних послідовних даних.

Набір даних, що використовується у даній роботі це набори послідовних транзакцій користувачів, що складаються з геоданих, даних про пристрої, про час транзакції та її тип, тощо. На рисунку 2.1 зображено приклад даних.

	datetime	user_id	device_id	IP_id	latitude	longitude
0	2022-02-01 00:00:00.037000+00:00	1331	934	4732	40.632982	-74.280504
1	2022-02-01 00:00:00.211000+00:00	1137	189	4137	40.775897	-74.238230
2	2022-02-01 00:00:00.263000+00:00	1160	387	5140	39.360178	-74.419202
3	2022-02-01 00:00:00.579000+00:00	450	321	4449	40.652104	-74.237631
4	2022-02-01 00:00:00.739000+00:00	1173	837	2939	39.875614	-75.130533
5	2022-02-01 00:00:00.830000+00:00	11	1521	5146	40.720805	-74.042509
6	2022-02-01 00:00:01.082000+00:00	588	1715	2782	39.920974	-75.109249
7	2022-02-01 00:00:01.182000+00:00	576	905	6515	40.784144	-74.011237
8	2022-02-01 00:00:01.891000+00:00	116	1116	5339	40.777454	-74.018305
9	2022-02-01 00:00:02.004000+00:00	448	1272	32	40.858649	-74.139829
10	2022-02-01 00:00:02.025000+00:00	655	1258	7893	40.728382	-74.148632
11	2022-02-01 00:00:02.067000+00:00	1420	402	3646	40.724552	-74.050549
12	2022-02-01 00:00:02.086000+00:00	507	730	2846	39.642233	-75.515041
13	2022-02-01 00:00:02.815000+00:00	1425	1247	9084	39.452898	-74.491057
14	2022-02-01 00:00:02.821000+00:00	1187	637	5243	40.732744	-74.084239
15	2022-02-01 00:00:03.442000+00:00	1108	1085	432	39.902169	-75.124881
16	2022-02-01 00:00:04.055000+00:00	715	406	5020	40.789106	-74.017112
17	2022-02-01 00:00:04.119000+00:00	212	157	3381	40.669383	-74.114494
18	2022-02-01 00:00:04.220000+00:00	254	1885	2937	39.825229	-75.277465
19	2022-02-01 00:00:04.418000+00:00	152	1279	800	40.663759	-74.390510
20	2022-02-01 00:00:04.508000+00:00	296	1287	2961	40.012535	-75.002602

Рисунок 2.1 – Послідовність транзакцій

Для виявлення аномальних дій з обліковими записами або пристроями у першу чергу треба звертати увагу на послідовність геолокацій і стрибки між пристроями, користувачами і IP адресами.

2.2 Програмні ресурси

Для написання програми я використовував мову програмування Python. Python це високорівнева мова програмування загального призначення з автоматичним управлінням пам'яттю, орієнтованою на підвищення продуктивності розробника, читання коду та його якості, а також на забезпечення переносимості написаних на ній програм.

Мова програмування Python дуже добре підходить для аналізу даних у науці та у сфері бізнесу через наступні переваги:

– простота опанування мови;

– спрощена робота з обчисленням надвеликих значень;

– велика різноманітність відкритих бібліотек, що значно спрощують роботу з однотипним кодом побудови та конфігурації моделей навчання;

– велика спільнота використання та підтримки Python інфраструктури;

– регулярні оновлення вищезгаданих бібліотек.

Python спільнота налічує багато зручних інструментів для роботи з даними.

Для навчання алгоритмів потрібні зручні інструменти різноманітної обробки набору даних. Основна робота з завантаженням, обробкою та зберіганням набору даних буде виконуватися за допомогою бібліотеки Pandas. Вона надає високоефективні, прості у використанні структури даних та інструменти аналізу та призначений для швидкої та простої обробки даних, читання, агрегування та візуалізації. Pandas дозволяє завантажувати дані з CSV файлів або з бази даних SQL і створювати на їхній основі набір даних з рядками та стовпцями, що називається датафрейм. Датафрейм даних дуже схожий на таблицю Microsoft Excel або на таблицю у реляційній базі даних.

Також для обробки багатовимірних даних я використовував пакет NumPy. В ньому є в наявності багато класів та функцій для роботи з

масивами, інструментів їхньої обробки. NumPy може оброблювати масиви, дані в яких мають один тип даних Ця бібліотека полегшує математичні операції над масивами, що дозволяє підвищіти продуктивність та швидкість виконання програми.

Для написання алгоритму застосовувалося середовище Jupyter Notebook. За визначенням, Jupyter – це безкоштовний інтерактивний веб-обчислювальний ноутбук з відкритим вихідним кодом. Цей чудовий інструмент підтримує багатомовне програмування і тому став де-факто вибором для науковців з даних для практики та обміну різними кодами, швидкого створення прототипів та дослідницького аналізу.

Незважаючи на те, що таких IDE (інтегровані середовища розробки) як PyCharm, Spyder або Atom, для певної мови існує достаньо, через свою гнучкість та інтерактивність, Jupyter здобув високу популярність серед науковців з даних.

Отже, давайте розглянемо деякі переваги:

– дослідницький аналіз даних: Jupyter дозволяє користувачам переглядати результати коду в рядку без залежності від інших частин коду. У блокноті кожна клітинка коду може бути потенційно перевірена в будь-який момент. Через це, на відміну від інших стандартних IDE, таких як PyCHarm, VSCode, Jupyter допомагає у вбудованому друку виводу, що стає надзвичайно корисним для процесу дослідницького аналізу даних (EDA);

– просте кешування у вбудованій комірці: підтримувати стан виконання кожної комірки важко, але з Jupyter ця робота виконується автоматично. Jupyter кешує результати кожної запущеної клітинки – чи то код, який навчає модель ML, чи код, який завантажує гігабайти даних з віддаленого сервера;

– незалежний від мови: через своє представлення у форматі JSON, Jupyter Notebook не залежить від платформи, а також від мови. Інша причина

полягає в тому, що Jupyter можна обробляти кількома мовами та конвертувати в будь-які формати файлів, такі як Markdown, HTML, PDF та інші;

– візуалізація даних: як компонент, спільний ноутбук Jupyter підтримує візуалізації та включає відтворення деяких наборів даних, таких як графіка та діаграми, які генеруються з кодів за допомогою таких модулів, як Matplotlib, Plotly або Bokeh. Jupyter дозволяє користувачам розповідати про візуалізації, а також ділитися кодом і наборами даних, дозволяючи іншим вносити інтерактивні зміни;

– живі взаємодії з кодом: Jupyter Notebook використовує пакети «ipywidgets», які надають стандартні інтерфейси користувача для вивчення коду та інтерактивності даних. Таким чином, користувачі можуть редагувати код, а також надсилати його для повторного запуску, роблячи код Jupyter не статичним. Це дозволяє користувачам керувати джерелами введення коду та надавати зворотній зв’язок безпосередньо в браузері;

– документування зразків коду: Jupyter дозволяє користувачам легко пояснювати свої коди рядок за рядком, додаючи зворотний зв’язок на всьому шляху. Ще краще, завдяки Jupyter користувачі можуть додавати інтерактивність разом із поясненнями, при цьому код повністю функціональний.

Середовище Jupyter Notebook відіграло важливу роль у демократизації науки про дані, зробивши її більш доступною, усуваючи бар’єри входу для людей, що працюють з даними.

На додачу, поєднуючи всі вищезгадані переваги Jupyter Notebook, ключовим моментом є те, що використання Jupyter є простим способом створення історії з даними. Сьогодні Jupyter повністю трансформувався та виріс у екосистему, де він охоплює кілька альтернативних середовищ, таких як JupyterLab і Hydrogen, бібліотеки інтерактивної візуалізації та інструменти сумісні з цими середовищами.

3 ОПИС СТВОРЕНОЇ ПРОГРАМИ

3.1 Попередня обробка даних

Попередня обробка даних, компонент підготовки даних, описує будь-який тип обробки необроблених даних для підготовки їх до іншої процедури обробки даних. Традиційно це був важливий попередній крок для процесу аналізу даних. Зовсім недавно методи попередньої обробки даних були адаптовані для навчання моделей машинного навчання та моделей штучного інтелекту, а також для виконання висновків щодо них.

Попередня обробка даних перетворює дані у формат, який легше й ефективніше обробляється в аналізі даних, машинному навчанні та інших задачах науки про дані. Ці методи, як правило, використовуються на самих ранніх етапах машинного навчання та розробки AI для забезпечення точних результатів.

Існує кілька різних інструментів і методів, які використовуються для попередньої обробки даних, зокрема наступні:

– створення вибірки, яка репрезентативно відбирає підмножину з великої сукупності даних;

– трансформація, що змінює необроблені дані для отримання єдиного вхідного набору даних;

– шумозаглушення для усунення шуму із даних;

– виявлення та видалення викидів;

– імпутація, яка синтезує статистично релевантні дані для відсутніх значень;

– нормалізація, яка впорядковує дані для більш ефективного доступу;

– вилучення ознак, яке витягує відповідну підмножину ознак, яка є важливою в певному контексті.

Ці інструменти та методи можна використовувати для різних джерел даних, включаючи дані, що зберігаються у файлах або базах даних, і потокові дані.

Практично будь-який тип аналізу даних, науки про дані або розробки AI вимагає певного типу попередньої обробки даних, щоб забезпечити надійні, точні та надійні результати для корпоративних додатків[9].

Дані реального світу безладні й часто створюються, обробляються та зберігаються різними людьми, бізнес-процесами та додатками. Як наслідок, у наборі даних можуть бути відсутні окремі поля, містити помилки введення вручну або мати повторювані дані чи різні імена для опису одного й того ж. Люди часто можуть виявити та виправити ці проблеми в даних, які вони використовують у сфері бізнесу, але дані, які використовуються для навчання алгоритмів машинного навчання або глибокого навчання, мають бути автоматично оброблені.

Алгоритми машинного навчання та глибокого навчання найкраще працюють, коли дані представлені у форматі, який підкреслює відповідні аспекти, необхідні для вирішення проблеми[6]. Практика розробки функцій, яка включає обговорення даних, перетворення даних, скорочення даних, вибір функцій і масштабування функцій, допомагають реструктурувати вихідні дані у форму, яка підходить для певних типів алгоритмів. Це може значно зменшити потужність обробки та час, необхідний для навчання нового алгоритму машинного навчання чи ШІ або виконання висновку щодо нього.

Етапи попередньої обробки даних включають в себе наступні

– профілювання даних;

– очищення даних;

– скорочення даних;

– перетворення даних;

– збагачення даних;

– перевірка даних.

Профілювання даних – це процес вивчення, аналізу та перегляду даних для збору статистичних даних про їх якість. Вона починається з огляду наявних даних та їх характеристик. Дослідники даних визначають набори даних, які мають відношення до розглянутої проблеми, описують її важливі атрибути та формують гіпотезу ознак, які можуть бути релевантними для запропонованої задачі аналітики чи машинного навчання. Вони також пов’язують джерела даних з відповідними бізнес-концепціями та розглядають, які бібліотеки попередньої обробки можна використовувати.

Мета очищення даних полягає в тому, щоб знайти найпростіший спосіб виправити проблеми з якістю, наприклад, усунути погані дані, заповнити відсутні дані або іншим чином забезпечити придатність вихідних даних для розробки функцій.

Набори необроблених даних часто включають зайві дані, які виникають у результаті характеристики явищ різними способами або даних, які не мають відношення до конкретного завдання ML, AI або аналітики. скорочення або зменшення даних використовує такі методи, як аналіз основних компонентів, щоб перетворити вихідні дані в простішу форму, придатну для конкретних випадків використання.

При перетворенні даних треба прийти до рішення як різні аспекти даних повинні бути організовані, щоб мати найбільший сенс для мети. Це може включати такі речі, як структурування неструктурованих даних, поєднання важливих змінних, коли це має сенс, або визначення важливих діапазонів, на яких слід зосередитися.

На кроці збагачення даних науковці даних застосовують різні бібліотеки інженерних функцій до даних, щоб здійснити бажані перетворення. Результатом має бути набір даних, організований для досягнення

оптимального балансу між часом навчання для нової моделі та необхідним обчисленням.

На етапі перевірки даних дані розбиваються на дві групи. Перший набір використовується для навчання моделі машинного навчання або глибокого навчання. Другий набір – це дані тестування, які використовуються для оцінки точності та надійності отриманої моделі. Цей другий крок допомагає виявити будь-які проблеми в гіпотезі, яка використовується при очищенні та розробці функцій даних. Якщо науковці з даних задоволені результатами, вони можуть передати завдання попередньої обробки інженеру даних, який з'ясовує, як масштабувати його для виробництва. Якщо ні, дослідники даних можуть повернутися назад і внести зміни до того, як вони впровадили очищення даних і кроки розробки функцій.

При проведенні попередньої обробки даних визначають наступні основні етапи:

– імпорт бібліотек;

– імпорт даних;

– перевірка пропущених значень;

– перевірка на категоріальні дані;

– масштабування;

– поділ даних на набори навчання, валідації та оцінки.

Бібліотека Pandas використовується для імпорту та обробки наборів даних. В основному, набори даних доступні у форматах CSV, оскільки вони мають невеликий розмір, що робить його швидким для обробки. Отже, щоб завантажити файл CSV, використовуючи функцію read_csv бібліотеки Pandas. Як тільки набір даних завантажений, ми повинні оглянути його та знайти будь-який шум. Для цього нам потрібно створити матрицю об'єктів X та вектор спостереження Y щодо X.

Як тільки створено матрицю об'єктів, можна виявити, що в ній відсутні деякі значення. Є два методи обробки пропущених значень:

– видалення всього рядка, що містить пропущене значення, але існує ймовірність того, що буде втрачено деяку важливу інформацію. Це може бути добрим підходом, якщо розмір набору даних великий;

– якщо числовий стовпець має пропущене значення, можна оцінити це значення, взявши середнє значення, медіану, моду тощо.

Дані в наборі даних повинні бути у числовій формі, щоб виконувати обчислення на них. Оскільки моделі машинного навчання містять складні математичні обчислення, ми можемо надати їм чисельне значення. Тому важливо перетворити всі текстові значення на числові. Клас LabelEncoder () learn використовується для перетворення цих категоріальних значень у числові значення.

Значення необроблених даних дуже різняться, і це може призвести до упередженого навчання моделі або може призвести до збільшення обчислювальних витрат. Тому важливо їх нормалізувати. Масштабування об'єктів – це метод, який використовується для перенесення значення даних у більш короткий діапазон. Методи, що використовуються для масштабування об'єктів:

– масштабування (нормалізація мін-макс);

– середня нормалізація;

– стандартизація (нормалізація Z-показника);

– масштабування до довжини одиниці.

Нарешті, нам потрібно розділити наші дані на три різні набори: навчальний набір для навчання моделі, тестовий набір для перевірки точності нашої моделі та тестовий набір для перевірки продуктивності нашої моделі на загальних даних. Перед поділом набору даних важливо перетасувати набір даних, щоб уникнути зміщення. Ідеальна пропорція для поділу набору даних

– 60:20:20, тобто 60% як навчальний набір, 20% як набір для тестування та перевірки. Щоб розділити набір даних, двічі використовуйте train_test_split із sklearn.model_selection.

Набір даних, що використовується у програмі, складається з послідовності транзакцій користувачів у форматі CSV. Кожна транзакція зберігає у собі інформацію про момент часу, місцезнаходження, користувача, пристрій, IP адресу.

Дані надаються у вигляді JSON-рядків, для чого насамперед потрібно витягнути з них потрібні там дані:

– latitude;

– longitude;

– ip_address.

Також для кращої роботи алгоритму і для зберігання конфіденційності користувачів треба перетворити категоріальні рядкові дані у чисельні, наприклад IP адреса 192.168.1.1 стане 0, 192.168.0.1 стане 1, а 192.168.0.0 стане 2. Також при подальшій роботі з даними виникне необхідність розбивання стовпців с категоріальними даними на декілька с назвами ip_id_1, ip_id_2, тощо. Це необхідно для правильної роботи метрик відстані між одиницями даних. На лістингу 3.1 наведена функція попередньої обробки даних, що використовується у алгоритмі.

Лістинг 3.1 – Функція попередньої обробки даних

```
def preprocessData(data):
  # Categorize columns
    data['username'] = pd.Categorical(data['username'])
    data['user_id'] = data['username'].cat.codes
                                  data['device_uuid']        =
pd.Categorical(data['device_uuid'])
    data['device_id'] = data['device_uuid'].cat.codes
    data['IP_address'] = pd.Categorical(data['IP_address'])
    data['IP_id'] = data['IP_address'].cat.codes
    # Extracting date and time features
    data['datetime'] = pd.to_datetime(data['datetime'])
    data['date'] = data['datetime'].dt.date
    # Get the date as day's number
    min_date = data['date'].min()
    data['int_date'] = [(row['date'] - min_date).days for i,
row in data.iterrows()]
    data['time'] = data['datetime'].dt.time
    # Get the time in microseconds
    data['int_time'] = [row['time'].microsecond
                        + row['time'].second*1000000
                        + row['time'].minute*60*1000000
                        + row['time'].hour*3600*1000000
                        for i, row in data.iterrows()]
    # Get the datetime in microseconds
            data['int_datetime']   =   data['int_time']   +
86400*1000000*data['int_date']

    return data
```

3.2 Вибір алгоритму

Маючи дані у чисельному форматі і за мету пошук аномалій у таких даних, я вирішив звернутись до алгоритмів кластеризації.

Кластеризація як алгоритм навчання без учителя дає можливість визначити групи точок у багатовимірному просторі властивостей.[10] Групи з невеликою кількістю точок або поодинокі точки у такому просторі можна вважати викидами або аномаліями. Наприклад, користувач перемістився у нове місце з нового пристрою та з новою IP адресою і зробив у цьому місці лише декілька транзакцій. А всі інші транзакції згуртовані у біля постійних кластерів.

У даній роботі для кластеризації використовується алгоритм DBSCAN – це відомий алгоритм кластеризації на основі щільності, запропонований Мартіном Естером, Хансом-Петером Кригелем та іншими в 1996 році (A density-based algorithm for discovering clusters in large spatial database). Цей алгоритм може ефективно обробляти точки шуму та знаходити просторові кластери довільної форми. Порівняно з алгоритмом кластеризації k-середніх, він не вимагає введення кількості кластерів, які потрібно розділити.

DBSCAN ділить точки даних на три категорії: основні точки, граничні точки та шум; між точками є три стани: пряма досяжна щільність, досяжна щільність і щільність підключення; DBSCAN має декілька основних параметрів параметри: eps, min_samples, metric. Перший параметр відповідає за умовну віддаленість між точками у просторі для поєднання їх у кластер. Другий параметр визначає мінімальний розмір кластеру. Третій параметр це вибір метрики розрахунку відстані у просторі, наприклад евклідова або метрика Манхеттена.

3.3 Нормалізація даних

Для нормальної роботи алгоритмів, що базуються на метриках відстані між точками у багатовимірному просторі чисельних властивостей, необхідно звести всі властивості у найбільш оптимальні значення.

Нормалізація даних – це одна з операцій перетворення ознак (Feature Transformation), яка виконується за при їх генерації (Feature Engineering) на етапі підготовки даних (Data Preparation).

У випадку машинного навчання (Machine Learning), нормалізація – це процедура попередньої обробки вхідної інформації (навчальних, тестових та валідаційних вибірок, а також реальних даних), при якій значення ознак у вхідному векторі наводяться до певного заданого діапазону, наприклад, [0…1] або [-1…1]. Розглянемо поняття нормалізації, нормування та нормування.

Нормування – це коригування значень відповідно до деяких функцій перетворення, з метою зробити їх зручнішими для порівняння. Наприклад, розділивши набір вимірювань зростання людей в дюймах на 2.54, ми отримаємо значення зростання в метричній системі.

Нормування даних потрібне, коли несумісність одиниць вимірювань змінних може позначитися на результатах і рекомендується, коли підсумкові звіти можуть бути покращені, якщо виразити результати певних зрозумілих/сумісних одиницях. Наприклад, час реакції, записаний у мілісекундах, легше інтерпретувати, ніж число тактів процесора, у яких отримано дані експерименту.

Нормування – це встановлення гранично допустимих чи оптимальних нормативних значень у прикладних сферах діяльності, наприклад, нормування праці. Зазвичай, норми розробляються за результатами дослідницьких, проектних чи наукових праць, і навіть з урахуванням експертних оцінок.

Необхідність нормалізації вибірок даних обумовлена природою алгоритмів і моделей Machine Learning, що використовуються. Вихідні значення ознак можуть змінюватися в дуже великому діапазоні та відрізнятися один від одного на кілька порядків. Припустимо, датасет містить відомості про концентрацію діючої речовини, що вимірюється в десятих або сотих частках відсотків, та показники тиску в сотнях тисяч атмосфер. Або, наприклад, в одному вхідному векторі є інформація про вік і дохід клієнта.

Будучи різними за фізичним змістом, дані сильно різняться між собою за абсолютними величинами. Робота аналітичних моделей машинного навчання (нейронних мереж, карт Кохонена тощо) з такими показниками виявиться некоректною: дисбаланс між значеннями ознак може спричинити нестійкість роботи моделі, погіршити результати навчання та уповільнити процес моделювання. Зокрема, параметричні методи машинного навчання (нейронні мережі, дерева, що ростуть) зазвичай вимагають симетричного та унімодального розподілу даних. Популярний метод найближчих сусідів, що часто використовується в задачах класифікації і іноді в регресійному аналізі, також є чутливим до діапазону змін вхідних змінних.

Після нормалізації всі числові значення вхідних ознак будуть приведені до однакової області зміни – деякому вузькому діапазону. Це дозволить вивести їх разом в одній моделі Machine Learning та забезпечить коректну роботу обчислювальних алгоритмів.

Для роботи алгоритму кластеризації властивість моменту часу була нормалізована до вигляду: 12 годин дорівнюють одиниці вимірювання. Для координат я прийняв рішення нормалізувати дані, помноживши на 10 і привести мінімум до 0. У такому випадку один градус у координатах буде дорівнювати 10-11 кілометрів. Такі значення дозволяють користувачеві переміщатись у трьох вимірах: довгота, широта і час – без ймовірності стати

підозрюваним. На лістингу 3.2 наведена функція нормалізації даних зі створенної програми.

Лістинг 3.2 – Функція нормалізації даних

```
def normalizeData(data):

    # datetime normalization
      data['norm_datetime'] = data['int_datetime'] / (12 *
3600 * 1000000)

    # Coordinations normalization
     data['norm_latitude'] = MinMaxScaler(feature_range=(0,
(data['latitude'].max()

- data['latitude'].min()) * 10 + 1)) \
        .fit_transform(df[['latitude']])
     data['norm_longitude'] = MinMaxScaler(feature_range=(0,
(data['longitude'].max()

- data['longitude'].min()) * 10 + 1)) \
        .fit_transform(df[['longitude']])

    return data
```

3.4 Пошук аномалій серед користувачів та пристроїв

Для пошуку аномалій треба розбити дані по користувачам і пристроям. Кожному користувачеві було присвоєно його перелік транзакцій, кількість пристроїв і IP адрес. Ті користувачі, що мають багато пристроїв і IP адрес, у першу чергу повинні бути досліджені, бо така поведінка може свідчити про спроби користувача знайти слабкі місця у системі, підмінити дані, тощо. А

різні пристрої можуть насправді бути одним пристроєм, який намагаються програмно або апаратно змінити для шахрайства. Також такі облікові записи можуть свідчити про сумісне користування ним багатьма людьми. Наприклад у одній сім'ї або у колі друзів. Але зазвичай компанії, що надають послуги, потребують від людей створення одного особистого облікового запису.

Користувачі, що мають лише один пристрій та одну IP адресу, викликають менше підозр тому що це більш схоже на звичайного користувача і тому що підробка геоданих є складною задачею. Але це не означає, що такі користувачі завжди чисті на руку. Тому є необхідність у пошуку аномалій серед пристроїв.

З пристроями така ж ситуація, як і з користувачами. Треба розподілити транзакції для кожного пристрою, присвоїти їм кількість користувачів і IP адрес. Для пристроїв немає обмеження на цю кількість. Тому що підозрілим може бути і пристрій з одним користувачем та IP адресою, і з багатьма. Пристрій, яким належить багатьом користувачам може бути публічним персональним комп'ютером, який не може гарантувати збереження персональних і конфіденційних даних. Також такий пристрій може бути особистим, за допомогою якого один користувач створив багато облікових записів для массового отримання різних бонусів.

Перед створенням моделі треба сформувати кожний набір даних у потрібному форматі. Для кластеризації насамперед використовуються багато вимірів:

– довгота;

– широта;

– момент часу;

– належність до категоріальних даних(користувач, пристрій, IP адреса).

Останній пункт являє собою набір буліанових властивостей, кількість котрих дорівнює кількості унікальних значень категоріальних даних, що

позначають належність запису до визначеної категорії. Наприклад, якщо користувач має два пристрої і три IP адреси, то додаткових властивостей буде п’ять. Приклад такої категоризації наведено на рисунку 3.1.

	norm_datetime	norm_latitude	norm_longitude	device_277	device_547	IP_5386	IP_6696	IP_7799	IP_9409
0	0.046140	141.699426	461.399285	1	0	0	1	0	0
1	0.046934	141.699426	461.399285	1	0	0	1	0	0
2	0.046941	141.700594	461.398333	1	0	0	1	0	0
3	0.047635	141.700634	461.398313	1	0	0	1	0	0
4	0.048376	141.700634	461.398313	1	0	0	1	0	0
5	0.049117	141.700634	461.398313	1	0	0	1	0	0
6	0.049201	141.700634	461.398313	1	0	0	1	0	0
7	0.049559	141.700634	461.398313	1	0	0	1	0	0
8	0.049708	141.700634	461.398313	1	0	0	1	0	0
9	0.050298	141.700634	461.398313	1	0	0	1	0	0
10	0.050447	141.700634	461.398313	1	0	0	1	0	0
11	0.050643	141.700634	461.398313	1	0	0	1	0	0
12	0.051390	141.700634	461.398313	1	0	0	1	0	0
13	0.051957	141.700614	461.398263	1	0	0	1	0	0
14	0.052148	141.700634	461.398313	1	0	0	1	0	0
15	0.052567	141.700634	461.398313	1	0	0	1	0	0
16	0.053314	141.700634	461.398313	1	0	0	1	0	0
17	0.053363	141.700634	461.398313	1	0	0	1	0	0
18	0.054103	141.700634	461.398313	1	0	0	1	0	0
19	0.054484	141.700634	461.398313	1	0	0	1	0	0
20	0.054841	141.700634	461.398313	1	0	0	1	0	0

Рисунок 3.1 – Формат даних для аналізу користувача

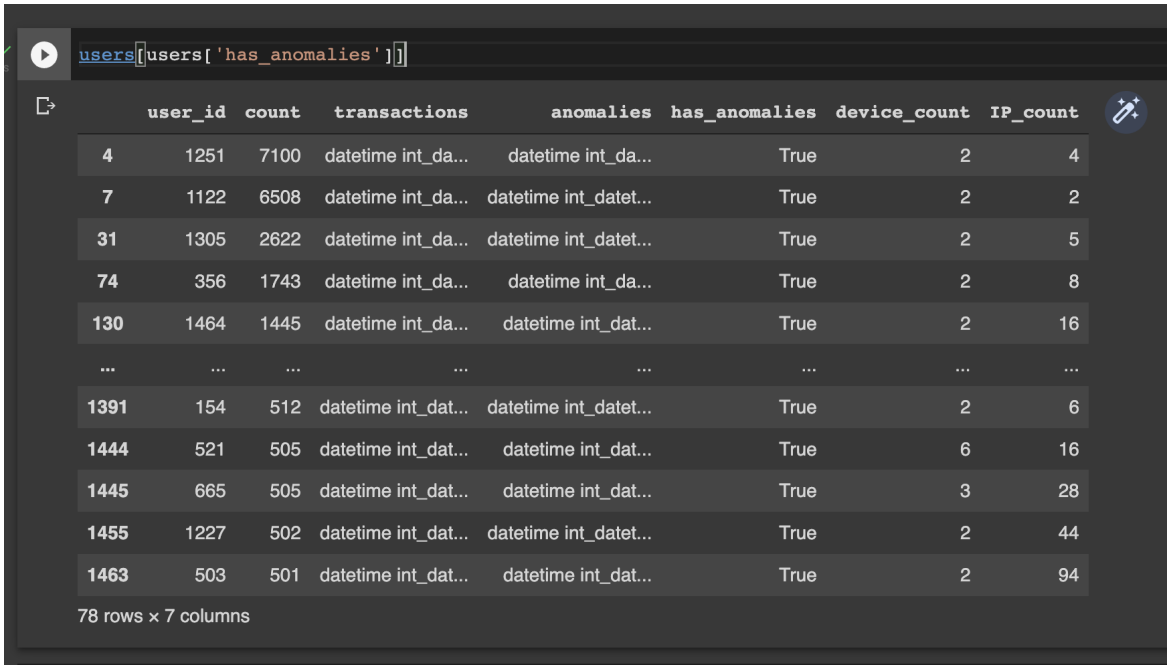
Для кластеризації був обраний алгоритм DBSCAN, що дозволяє отримати кластери складної форми, що у випадку багатовимірного простору, що має у собі координати і момент часу, дозволить відстежити пересування і не переплутати нормальні з підозрілими. На лістингу 3.3 наведена функцію для кластеризації з використаними гіперпараметрами.

Лістинг 3.3 – функція кластеризації даних

```
def clusterData(data):
    X = data.drop(aux_fields, axis=1)
    clustering_model = DBSCAN(eps=2, min_samples=2,
metric='euclidean', leaf_size=30)
    clustering_model.fit(X)
    data['cluster'] = clustering_model.labels_
return data, data[data['cluster'] == -1]
```

Результатом роботи алгоритму є класифікація користувачів і пристроїв на підозрілих та нормальних, за що відповідає поле "has_anomalies".

Результати роботи програми зображені на рисунку 3.2 та рисунку 3.3.

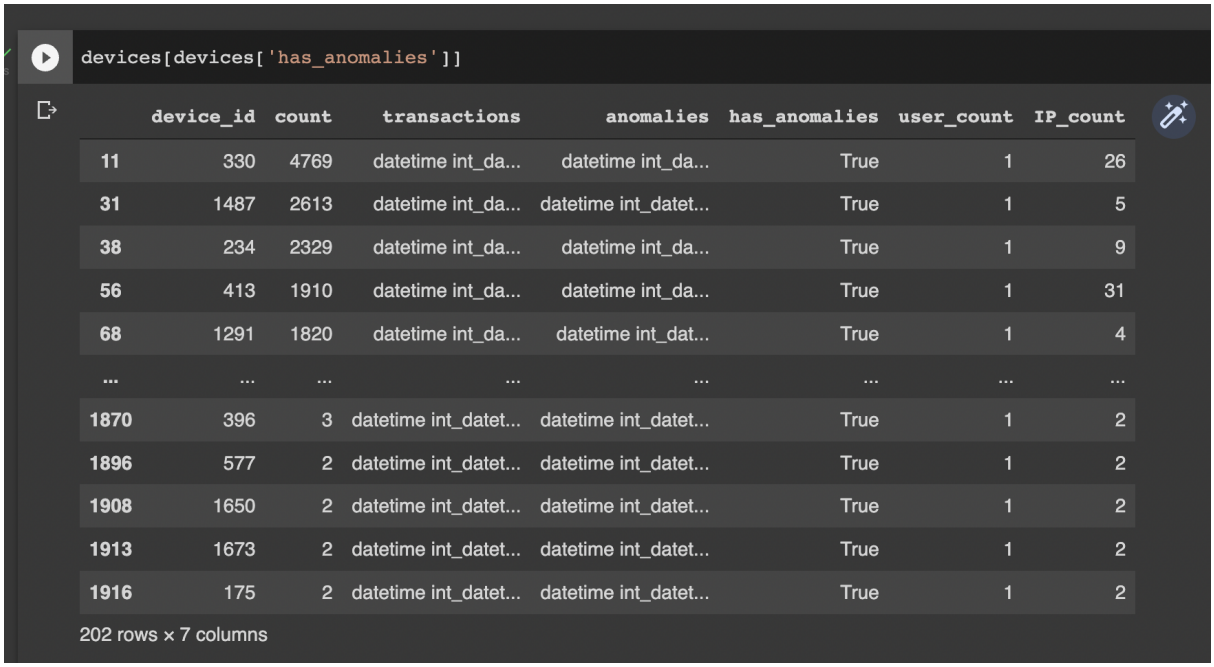


```
users[users['has_anomalies']]
```

	user_id	count	transactions	anomalies	has_anomalies	device_count	IP_count
4	1251	7100	datetime int_da...	datetime int_da...	True	2	4
7	1122	6508	datetime int_da...	datetime int_datet...	True	2	2
31	1305	2622	datetime int_da...	datetime int_datet...	True	2	5
74	356	1743	datetime int_da...	datetime int_da...	True	2	8
130	1464	1445	datetime int_da...	datetime int_dat...	True	2	16
...	...	...	...	...	...	...	...
1391	154	512	datetime int_dat...	datetime int_datet...	True	2	6
1444	521	505	datetime int_dat...	datetime int_dat...	True	6	16
1445	665	505	datetime int_dat...	datetime int_dat...	True	3	28
1455	1227	502	datetime int_dat...	datetime int_datet...	True	2	44
1463	503	501	datetime int_dat...	datetime int_dat...	True	2	94

78 rows × 7 columns

Рисунок 3.2 – Результат пошуку аномалій серед користувачів



```
devices[devices['has_anomalies']]
```

	device_id	count	transactions	anomalies	has_anomalies	user_count	IP_count
11	330	4769	datetime int_da...	datetime int_da...	True	1	26
31	1487	2613	datetime int_da...	datetime int_datet...	True	1	5
38	234	2329	datetime int_da...	datetime int_da...	True	1	9
56	413	1910	datetime int_da...	datetime int_da...	True	1	31
68	1291	1820	datetime int_da...	datetime int_dat...	True	1	4
...	...	...	...	...	...	...	...
1870	396	3	datetime int_datet...	datetime int_datet...	True	1	2
1896	577	2	datetime int_datet...	datetime int_datet...	True	1	2
1908	1650	2	datetime int_datet...	datetime int_datet...	True	1	2
1913	1673	2	datetime int_datet...	datetime int_datet...	True	1	2
1916	175	2	datetime int_datet...	datetime int_datet...	True	1	2

202 rows × 7 columns

Рисунок 3.3 – Результат пошуку аномалій серед пристроїв

Якщо розглянути ці випадки ближче, то можна помітити, що основні причини, через які вони потрапили до списку наступні:

– користувач здійснив транзакцію у незвичний час;

– користувач здійснив транзакції зі значною затримкою у часі;

– користувач здійснив транзакцію з нової геолокації;

– користувач здійснив транзакцію з нового пристрою;

– користувач здійснив транзакцію з нової IP адреси;

– користувач здійснив транзакцію з нового місця.

Коли декілька з цих причин з’являються у одній транзакції, вона має велику ймовірність попасти у список аномалій, тому що по декільком властивостям у багатовимірному просторі вона віддаляється від найближчого кластеру. На рисунку 3.4 наведений приклад послідовності транзакцій з аномалією під номером 1673.

	datetime	user_id	IP_id	device_id	latitude	longitude
1664	2022-02-02 05:54:58.592000+00:00	1251	6696	277	40.864330	-73.964765
1665	2022-02-02 05:55:47.677000+00:00	1251	6696	277	40.864330	-73.964765
1666	2022-02-02 05:56:19.611000+00:00	1251	6696	277	40.864330	-73.964765
1667	2022-02-02 05:56:51.480000+00:00	1251	6696	277	40.864330	-73.964765
1668	2022-02-02 05:57:27.184000+00:00	1251	6696	277	40.864331	-73.964766
1669	2022-02-02 05:58:16.105000+00:00	1251	6696	277	40.864331	-73.964766
1670	2022-02-02 05:58:47.996000+00:00	1251	6696	277	40.864331	-73.964766
1671	2022-02-02 05:59:20.824000+00:00	1251	6696	277	40.864331	-73.964766
1672	2022-02-02 05:59:52.969000+00:00	1251	6696	277	40.864331	-73.964766
1673	2022-02-02 06:00:03.179000+00:00	1251	9409	547	36.106859	-115.182276
1674	2022-02-02 10:56:40.828000+00:00	1251	6696	277	40.864333	-73.964765
1675	2022-02-02 10:57:13.076000+00:00	1251	6696	277	40.864333	-73.964765
1676	2022-02-02 10:57:45.165000+00:00	1251	6696	277	40.864333	-73.964765
1677	2022-02-02 10:58:17.307000+00:00	1251	6696	277	40.864333	-73.964765
1678	2022-02-02 10:58:49.257000+00:00	1251	6696	277	40.864333	-73.964765
1679	2022-02-02 10:59:21.147000+00:00	1251	6696	277	40.864333	-73.964765
1680	2022-02-02 10:59:54.977000+00:00	1251	6696	277	40.864331	-73.964766
1681	2022-02-02 11:00:26.859000+00:00	1251	6696	277	40.864331	-73.964766
1682	2022-02-02 11:00:58.857000+00:00	1251	6696	277	40.864331	-73.964766

Рисунок 3.4 – Послідовність транзакцій з аномалією під номером 1673

На рисунку 3.5 зображено набір властивостей та значень, що були передані до алгоритму кластеризації.

	norm_datetime	norm_latitude	norm_longitude	device_277	device_547	IP_5386	IP_6696	IP_7799	IP_9409	cluster
1664	2.493023	141.700583	461.398203	1	0	0	1	0	0	0
1665	2.494159	141.700583	461.398203	1	0	0	1	0	0	0
1666	2.494898	141.700583	461.398203	1	0	0	1	0	0	0
1667	2.495636	141.700583	461.398203	1	0	0	1	0	0	0
1668	2.496463	141.700594	461.398193	1	0	0	1	0	0	0
1669	2.497595	141.700594	461.398193	1	0	0	1	0	0	0
1670	2.498333	141.700594	461.398193	1	0	0	1	0	0	0
1671	2.499093	141.700594	461.398193	1	0	0	1	0	0	0
1672	2.499837	141.700594	461.398193	1	0	0	1	0	0	0
1673	2.500074	93.799227	48.358051	0	1	0	0	0	1	-1
1674	2.912056	141.700614	461.398203	1	0	0	1	0	0	0
1675	2.912803	141.700614	461.398203	1	0	0	1	0	0	0
1676	2.913545	141.700614	461.398203	1	0	0	1	0	0	0
1677	2.914290	141.700614	461.398203	1	0	0	1	0	0	0
1678	2.915029	141.700614	461.398203	1	0	0	1	0	0	0
1679	2.915767	141.700614	461.398203	1	0	0	1	0	0	0
1680	2.916550	141.700594	461.398193	1	0	0	1	0	0	0
1681	2.917288	141.700594	461.398193	1	0	0	1	0	0	0
1682	2.918029	141.700594	461.398193	1	0	0	1	0	0	0

Рисунок 3.5 – Властивості використані для кластеризації

Якщо розглядати властивості, що були використані для кластеризації, то можна помітити зміну пристрою та IP адреси на нові. Також кардинально змінено місцезнаходження, хоча різниця у часі між транзакціями незначна. Тобто це стрибок у просторі з дуже великою швидкістю.

Цей випадок можна охарактеризувати як вкрадений обліковий запис через перелічені ознаки. Наступними кроками після такого пошуку аномалій має бути складений звіт для компанії зі списками підозрілих користувачів і пристроїв, а вже зі сторони інших команд, що працюють з клієнтами, потребується застосувати рішення щодо цих підозрілих випадків.

ВИСНОВКИ

В кваліфікаційній роботі було проаналізовано предметну галузь та теоретичні джерела, що стосуються обраної теми. Були сформовані вимоги щодо кваліфікаційної роботи та досліджено методи виявлення аномалій. Був проведений збір матеріалів, які стосуються теми кваліфікаційної роботи – «Виявлення аномалій у поведінці користувачів за допомогою Machine Learning».

На основі отриманих результатів було розроблено алгоритм, який достатньо точно визначає аномалії у поведінці користувачів системи масового обслуговування за допомогою Machine Learning.

Розроблена практична реалізація була запрограмовані на мові Python в середовищі Jupyter Notebook.

Промислова розробка програмних систем вимагає великої уваги до відмови стійкості кінцевого продукту, а також швидкого реагування на відмови та збої, якщо вони все-таки трапляються. Моніторинг, звичайно ж, допомагає реагувати на відмови та збої ефективніше та швидше, але недостатньо. По-перше, дуже складно стежити за великою кількістю серверів – потрібна велика кількість людей. По-друге, потрібно добре розуміти, як влаштована програма, щоб прогнозувати її стан. Отже, потрібно багато людей, які добре розуміють розроблювані нами системи, їх показники та особливості. Припустимо, навіть якщо знайти достатньо людей, які бажають займатися цим, потрібно ще чимало часу, щоб їх навчити.

Для боротьби з проблемами такого роду можна застосовувати сучасні технології. Виявлення аномалій – це завдання ідентифікації елементів, подій або спостережень, які не відповідають очікуваному шаблону або іншим елементам набору даних. Прикладами такого завдання може бути: детекція

шахрайства, детекція відмови працездатності системи, пошук помилок у тексті тощо.

Виявлення аномалій стосується широкого кола областей, таких як система виявлення вторгнень, виявлення шахрайства, виявлення несправностей, моніторингу здоров'я, виявлення подій у мережах датчиків та виявлення порушень в екологічній сфері.

Результатом роботи алгоритмів виявлення аномалій є список підозрілих випадків, з якими у подальшому працюватимуть інші команди. Такі звіти надають компаніям можливість запобігти крадіжкам конфіденційної інформації клієнтів, їхніх коштів та порушенням законів і законодавчих актів країн та регіонів, в яких компанія веде свою діяльність.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Hawkins, Douglas M.. Identification of Outliers. Chapman and Hall London; New York, 1980.

2. Campello, R. J. G. B.; Moulavi, D.; Zimek, A.; Sander, J.. "Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection". ACM Transactions on Knowledge Discovery from Data, 2015.

3. Bram Steenwinckel. Adaptive Anomaly Detection and Root Cause Analysis by Fusing Semantics and Machine Learning. European Semantic Web Conference. 2018

4. Guide to Intrusion Detection and Prevention Systems (IDPS). Technical report, National Institute of Standards and Technology, U.S. Department of Commerce, 2007.

5. Aggarwal, Charu (2017). Outlier Analysis. Springer Publishing Company, Incorporated. ISBN 3319475770.

6. Ke Wang and Salvatore J. Stolfo. Anomalous Payload-Based Network Intrusion Detection, pages 203-222. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

7. Jeffrey O. Kephart. A biologically inspired immune system for computers. In Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, pages 130-139. MIT Press, 1994.

8. Everitt, Brian. Cluster analysis. Chichester, West Sussex, U.K: Wiley. ISBN 9780470749913. 2011.

9. John Shawe-Taylor Alex J. Smola Robert C. Williamson John Platt, Bernhard Schölkopf. Estimating the support of a high-dimensional distribution. Technical report, November 1999.

10. Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. Pattern Recogn. Lett., 24(9-10):1641-1650, June 2003.

11. Christina Warrender, Stephanie Forrest, and Barak Pearlmutter. Detecting intrusion using system calls: alternative data models. In Proceedings of the IEEE Symposium on Security and Privacy, 1999.