НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

До захисту допущено
Завідувач кафедри

_____ **Дмитро ЛАНДЕ**
(підпис)

«_____»_____2023 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системи, технології та математичні методи кібербезпеки»
спеціальності 125 «Кібербезпека»

на тему: Визначення аномалій в логах RDP автентифікацій

Виконав (-ла): здобувач вищої освіти **IV** курсу, групи ФБ-93
(шифр групи)

Колесник Андрій Миколайович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник Доцент каф. ІБ, к. е. н., Ткач Володимир Миколайович _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент Доцент каф. ММЗІ к.ф.-м.н., Южакова Ганна Олексіївна _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Здобувач вищої освіти _____
(підпис)

Київ – 2023 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма «Системи, технології та математичні методи кібербезпеки»

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ Дмитро ЛАНДЕ
(підпис)

«___»_____2023 р.

ЗАВДАННЯ
на дипломну роботу здобувачу вищої освіти

Колеснику Андрію Миколайовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Визначення аномалій в логах RDP автентифікацій, керівник роботи Ткач Володимир Миколайович, к. е. н., доцент.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «26» травня 2023 р. №2028-с

2. Термін подання здобувачем вищої освіти роботи «9» червня 2023 р.

3. Вихідні дані до роботи підмножина входів в систему з аномальною поведінкою

4. Зміст роботи дослідження та порівняння методів машинного навчання для проблеми пошуку аномалій для логів системи автентифікації

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): презентація

6. Дата видачі завдання 1 лютого 2023 року.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка
1	Розробка теми дослідження, формування проблематики та постановка задач	1.02.23-11.02.23	Виконано
2.	Вивчення літератури за темою дослідження	12.02.23-27.02.23	Виконано
3.	Формування задачі на дипломну роботу	28.02.23-1.03.23	Виконано
4.	Теоретична підготовка до розробки програмного застосунку	2.03.23-20.03.23	Виконано
5.	Розробка програмного застосунку	21.03.23-21.04.23	Виконано
6.	Аналіз результатів	22.04.23-1.05.23	Виконано
7.	Робота над розділом 1. RDP автентифікація, загрози, аномалії	1.05.23-12.05.23	Виконано
8.	Робота над розділом 2. Методи пошуку аномалій в машинному	13.05.23-22.05.23	Виконано
9.	Робота над розділом 3. Побудова системи пошуку аномалій	23.05.23-5.06.23	Виконано
10.	Створення презентаційного матеріалу	5.06.23 – 9.06.23	Виконано
11.	Захист дипломної роботи	19.06.23-23.06.23	

Здобувач вищої освіти _____ Андрій КОЛЕСНИК
(підпис) (**Власне ім'я, ПРІЗВИЩЕ**)

Керівник роботи _____ Володимир ТКАЧ
(підпис) (**Власне ім'я, ПРІЗВИЩЕ**)

РЕФЕРАТ

Обсяг роботи 96 сторінки, 18 ілюстрації, 3 таблиць, 1 додаток, 15 джерело літератури.

Об’єктом дослідження є записи аудиту входів системи автентифікації.

Предметом дослідження є методи машинного навчання для виявлення аномалій в системах автентифікації.

Метою роботи було дослідити і проаналізувати методи машинного навчання для проблеми виявлення аномалій, а також розробити модель машинного навчання для пошуку аномалій в логах входів до системи.

Було проведено дослідження цих методів, аналіз даних, що отримуються з логів, і побудовано декілька моделей для пошуку аномалій в записах входів. В результаті дослідження ми отримали набір даних з аномальних входів, які містять ідентифікатори атак.

Було розроблено програмний продукт на мові Python, який виконує аналіз даних та побудову моделей.

Ключові слова: Автентифікація, Логи, Машинне навчання, Пошук аномалій, Навчання без вчителя.

ABSTRACT

The volume of work is 96 pages, 18 illustrations, 3 tables, 1 appendix, 15 literary sources.

The research object is authentication system input audit records.

The research subject is machine learning methods for anomaly detection in authentication systems.

The aim of the study was to investigate and analyze machine learning methods for the anomaly detection problem, as well as develop a machine learning model for anomaly detection in system login logs.

Research was conducted on these methods, data analysis obtained from the logs, and several models were built for anomaly detection in login records. As a result of the research, we obtained a dataset of anomalous logins that included attack identifiers.

A software product was developed in Python that performs data analysis and model building.

Keywords: Authentication, Logs, Machine Learning, Anomaly Detection, Unsupervised Learning.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

RDP - Remote Desktop Protocol -протокол віддаленого робочого стола

ІК - Індикатор компрометації

ІА - індикатор атаки

ІР - індикатор ризику

SVM - Support Vector Machine - Метод опорних векторів

DBSCAN - density-based spatial clustering of applications with noise - просторове кластеризування додатків із шумом на основі щільності

LOF - Local outlier factor - Локальний рівень викиду

PCA - Principal Component Analysis - Метод головних компонент

t-SNE - t-Distributed Stochastic Neighbor Embedding - T-розподілене вкладення стохастичної близькості

Gower distance/metric - відстань Говера

Jaccard index/metric/distance - метрика Жакарда

IF- Isolation forest

ВСТУП

Системи автентифікації в сучасному світі займають важливе місце у забезпеченні безпеки, особливо в онлайн-сервісах та інтернет-банкінгу. Ми маємо з ними справу кожен день, коли запускаємо комп'ютер, виконуємо вхід у облікові записи в соціальних мережах, банкінгах і інших системах. Ці системи забезпечують контроль доступу до різних ресурсів та інформації. Однак, збільшення кількості інформації та користувачів, що мають доступ до цих ресурсів, призвело до збільшення ризиків безпеки та вразливості систем автентифікації. А отже важливим аспектом кібербезпеки є забезпечення безпеки в системах автентифікації, оскільки за допомогою цих системи здійснюється контроль доступу до конфіденційної інформації та ресурсів.

Для забезпечення безпеки необхідно вживати перних методів які мають за ціль унеможливити атаки на ці системи, це можуть бути вимоги до паролів, системи, двофакторна аутентифікацію, захист від брутфорсу, використовування шифрування. Всі ці кроки можуть бути виконані окремо або в комбінації для забезпечення більш високого рівня безпеки системи автентифікації. Але слід зауважити, що безпека системи автентифікації є постійним процесом і вона повинна бути оновлювана та модернізована з часом для запобігання новим загрозам та вразливостям.

Але також для забезпечення безпеки можуть бути використанні інші методи. При роботі з логами для підвищення рівня безпеки в системі можуть бути використані різні техніки по пошуку нетипових, незвичних та дивних входів в систему, які ми можемо назвати аномаліями. Аномалії в таких системах може свідчити про те, що вхід відбувається з іншого місця або якимось новим користувачем, що є не дуже критичним на перший погляд, але такі входи можуть свідчити про викрадання паролів та отримання інших даних або спроб зламу акаунтів.

В той же час сфера машинного навчання все більше зростає за рахунок збільшення обчислювальної потужності, що може дати нам досить широкий спектр

рішень для проблеми пошуку аномалій. Окрім цього алгоритми машинного навчання здатні знаходити досить глибокі зв’язки між показниками системи, що дасть нам змогу побудувати надійну і прогресивну систему безпеки.

Актуальність роботи полягає в тому, що системи автентифікації є частиною нашого повсякденного життя і їх використання стрімко зростає, разом з ним і атаки, а також ризики для великих систем і компаній. У зв'язку з цим, актуальним є підвищення рівня безпеки систем автентифікації шляхом використання нових методів захисту. Також досить важливим є використання машинного навчання як рішення для цієї проблеми, адже це досить прогресивний інструмент для вирішення і дослідження різних проблем в сфері безпеки інформаційних систем.

Метою дослідження є вивчення та аналіз існуючих методів пошуку аномалій в логах системах автентифікації за допомогою машинного навчання, їх можливостей для підвищення рівня безпеки.

Об'єктом дослідження є записи аудиту входів системи автентифікації.

Предметом дослідження є методи машинного навчання для виявлення аномалій в системах автентифікації.

Методи дослідження включають аналіз наукової літератури, аналіз існуючих методів машинного навчання, проведення експериментальних досліджень.

Науковою новизною одержаних результатів є виявлення нових можливостей та шляхів підвищення рівня безпеки систем автентифікації.

Практичне значення одержаних результатів полягає у можливості використання розроблених рекомендацій та методів для підвищення рівня безпеки систем автентифікації.

Отже, наше дослідження присвячено вивченню проблеми забезпечення безпеки в системах автентифікації та методів пошуку аномалій, що дозволяють ефективно виявляти потенційні загрози та підвищувати рівень безпеки систем за допомогою методів машинного навчання.

1 RDP АВТЕНТИФІКАЦІЯ, ЗАГРОЗИ, АНОМАЛІЇ

Під час розгляду проблеми аномальних зразків в логах системах автентифікації нам необхідний набір даних для дослідження і пошуку нетипових зразків. В такому наборі необхідні записи входів і інформацію про користувачів, систему, хост, час входу та інші дані. В даній роботі ми використовуємо набір даних сформований з логів, які були отримані з системи автентифікації за допомогою протоколу RDP, тому давайте розглянемо сам протокол, принцип дії і чому ця проблема ще більш важлива в контексті цього протоколу.

RDP (англ. Remote Desktop Protocol, протокол віддаленого робочого стола) — протокол прикладного рівня, що використовується для підключення користувача до віддаленого робочого стола за допомогою сервісу термінальних з'єднань. Технологія з'явилася у 90-х роках і досі активно використовується. За допомогою RDP можна підключатися до віддаленого комп'ютера і працювати з ним, ніби ви фізично знаходитесь перед ним. Це дозволяє працівникам працювати зі своїх домашніх або віддалених місць на робочих станціях, а також системним адміністраторам керувати серверами.

Принцип роботи RDP виходить з протоколу TCP. Поєднання клієнт-сервер відбувається на транспортному рівні. Локальний комп'ютер (клієнт) встановлює з'єднання з віддаленим комп'ютером (сервером) за допомогою RDP-клієнта. Клієнт надсилає свої облікові дані (ім'я користувача і пароль) на сервер для перевірки. Якщо облікові дані вірні, сервер надає дозвіл на доступ. Далі відбувається створення сеансу і клієнт отримує графічний вміст (екран, вікна, інтерфейс) від сервера та відображає його на своєму екрані. Зміни, зроблені на клієнтському боці, передаються на сервер, який оновлює відповідний вміст на своєму екрані.

Протокол широко використовується в системах які потребують віддаленого адміністрування, що включає керувати серверами, мережевими пристроями і комп'ютерами. Це дозволяє виконувати налагодження, налаштування і управління ресурсами без необхідності фізичного присутності біля кожного пристрою. Великі організації можуть використовувати RDP для надання співробітникам доступу до

віддалених робочих столів або віртуальних машин. Це дозволяє співробітникам працювати з будь-якого місця, забезпечуючи доступ до робочих додатків і даних. Загалом, зрозуміло чому RDP популярний серед користувачів і як він може допомогти виконувати широкий спектр завдань на ПК, навіть не перебуваючи поруч з ним. Враховуючи поточну ситуацію та зростаючий попит на роботу з дому, можна сказати, що це справжня порятунок.

Однак ми не можемо ігнорувати ризики використання інструментів RDP, оскільки вони також можуть сприяти небажаному доступу, злому, крадіжці даних. Проблема полягає в тому, що зловмисники можуть намагатись отримати дані для входу (логін і пароль) до системи RDP і використовувати їх для несанкціонованого доступу до комп'ютера або сервера. При отримані даних для входи може відбуватись великий виток персональний та/або корпоративних даних, отримання несанкціонованого доступу, крадіжка даних, пошкодження або знищення даних та інші зловмисні дії. Якщо зловмиснику вдасться отримати доступ до облікового запису з правами адміністратора, він матиме повний контроль над системою і може завдати значної або навіть фатальної шкоди інформаційній системі.

Саме тому наша задача полягає в виявлянні нетипових входів в систему, щоб попереджувати адміністратора безпеки та зменшувати шанси проведення таких атак, що підвищить рівень безпеки в системі.

1.1 Нетипові входження. Аномалії

Процес пошуку і аналізу аномалій у даних особливо важливим і необхідним. Ці зразки потрібні для періодичного аналізу і огляду даних, адже це ще однин пункт в забезпеченні безпеки для систем які використовують автентифікацію, аналіз трафіка, тощо. Особливо корисно використовувати цей процес коли аналіз даних може убезпечити від фатальних наслідків, наприклад виявлення шахрайства, медична діагностика, контроль якості виробництва, біржова торгівля або виявлення

мережевого вторгнення. В попередній розділах ми могли зустріти такі слова як аномалії, нетипові зразки, викиди та інші синоніми, які і в наступних розділах будуть часто використовуватись. Але давайте розберемось з чим ми маємо справу.

У контексті даних, аномалії - це значення або шаблони, які суттєво відрізняються від очікуваної і звичайної поведінки даних. Вони можуть бути результатом виняткових подій, несподіваних змін у системі або помилок в даних. Такими подіями може бути зміною користувачем IP адреси, робочого часу, робочою станції або комп'ютеру. Зміна всіх цих показників може бути виправдана та пояснена користувачем, але для системи важливо бачити ці зміни і звертати на них увагу. Адже кожен з цих входів може виявитись здійсненим зловмисником та понести за собою дуже багато негативних наслідків.

В той же час викиди (outliers) - це значення або спостереження, яке відрізняється від інших даних в наборі та може бути результатом помилок вимірювання, аномальних подій або випадкових відхилень. Для прикладу при дослідження числових значень в наших даних ми завжди звертаємо увагу на розподіл. Якщо ми бачимо якесь значення яке знаходиться явно поза розподілом ми можемо визначити його як викид.

Поняття викидів і аномалій дуже близькі, та часто їх навіть взаємо заміняють, але ми в нашому дослідженні будемо розглядати і використовувати поняття аномалій.

1.2 Проблема аномалій в системах автентифікації

Проблема аномалій в системах автентифікації на основі аналізу логів входів є важливим аспектом забезпечення безпеки інформації та конфіденційності даних. Аномалії, можуть свідчити про незвичайні або підозрілі входи користувачів, які в свою чергу можуть вказувати на можливі атаки на систему. Недостатня обробка аномалій в логах входу RDP автентифікації може призвести до витоку

конфіденційної інформації та несанкціонованого доступу. Зловмисники, які отримають неправомірний доступ до системи, можуть збирати цінну інформацію, таку як паролі, облікові записи користувачів або інші конфіденційні дані. Це може призвести до серйозних наслідків, таких як порушення безпеки даних, фінансові втрати або компрометація бізнес-секретів та крадіжки даних.

Несанкціонований доступ може послужити входовою точкою для розповсюдження шкідливого програмного забезпечення. Якщо зловмисники отримають доступ до систем, вони можуть встановити шкідливе програмне забезпечення, таке як троянські коні, шпигунське ПЗ або шифрувальні програми-вимагачі. Це може бути виконано як на одному комп’ютері, так і розповсюдити по декільком комп’ютерам або навіть по всій системі. Наслідками цього є пошкодження або втрати даних, порушення конфіденційності користувачів або недоступності системи.

Аналіз і пошук дивних входжень можуть бути ознаками багатьох інших атак, наприклад брудфорсу. Брутфорс атаки є одними з найпоширеніших атак, пов'язаних з автентифікацією. Зловмисники використовують автоматизовані засоби для спроби послідовно вгадати правильний пароль для облікового запису RDP. Ці атаки можуть бути успішними, якщо обліковий запис користувача має слабкий пароль або якщо не застосовані заходи обмеження спроб аутентифікації.

Тим не менш маючи гарно побудований алгоритм аналізу значень які оброблює аудит може, як і значно підвищити рівень безпеки в системі, так і пришвидшити виявлення та реагування на такі атаки відповідно. Наприклад, у випадку викрадання даних входу ми можемо побачити що користувач зайшов у неробочий(або нетиповий для самого себе) час, з іншого комп’ютеру, або навіть з іншої країни. Ми не можемо стверджувати, що хтось заволодів даними для входу, але це вже виглядає підозрілим та необхідно розглянути детальніше цей вхід, поцікавитись в користувача, у випадку чого заморозити обліковий запис, це дасть нам змогу знайти і знерухомити порушника до того як безпека системи буде порушена. В той же час атаки брудфорсу дуже помітні при аналізі таки входів, адже

ми бачимо на них дуже багато невдалих входів для конкретного користувача, тому й можна завдати якихось дій, до того як атака стане успішною.

1.3 Індикатор компрометації, атаки та ризику в контексті логів системи автентифікації

У сучасних умовах частота та складність кібератак продовжують зростати, тому організації використовують більш прогресивний підхід до кібербезпеки. Індикатори компрометації (ІК), індикатори атаки (ІА) та індикатори ризику(ІР) є сімейством методів, які допомагають організаціям виявляти потенційні загрози та захищатися від них. Ці методи є важливими для реагування на інциденти та отримання інформації про загрози, проте вони мають різні цілі. Нижче ми розглянемо відмінності між ІК, ІА та ІР, їх важливість в контексті виявлення аномальних зразків в логах входів та як організації можуть використовувати їх для поліпшення своєї кібербезпеки.

ІК (індикатори компрометації) - це артефакти або докази, що вказують на те, що система була порушена або компрометована. ІК можуть бути виявлені в журналах, мережевому трафіку, пам'яті системи та інших джерелах. Приклади включають IP-адреси, доменні імена, хеші файлів та моделі поведінки.

ІК використовуються для виявлення відомої шкідливої діяльності, такої як віруси, фішинг та розшифрування файлів. Вони також використовуються для виявлення загальних методів атак, таких як атаки методом перебору та SQL-ін'єкції. Ці індикатори зазвичай виявляються дослідниками з кібербезпеки та розповсюджуються серед кібербезпекової спільноти, щоб допомогти виявляти та зменшувати загрози.

ІК в контексті аномальних входжень можуть бути проявленні в такі елементах:

- Несанкціонований доступ до облікових записів: Виявлення незвичайної активності, такої як входження під обліковими записами, які не мали раніше доступу до системи або надто часті спроби авторизації під тим самим обліковим записом, можуть бути показниками компрометації.
- Зміна автентифікаційних параметрів: Виявлення незвичайних змін в налаштуваннях автентифікації, таких як зміна паролів, методів аутентифікації або додавання нових облікових записів без відповідного авторизованого дозволу, може свідчити про компрометацію.
- Використання недійсних або скомпрометованих сертифікатів: Виявлення використання сертифікатів, які не є дійсними або були відкликані, може вказувати на можливу компрометацію системи автентифікації.

В свою чергу індикатори атаки (ІА) - це ознаки того, що атака вже відбувається або що атака вже сталася. ІА є більш прогресивними від ІК та можуть виявляти потенційні загрози до того, як вони завдають шкоди. ІА базуються на тактиках, методах і процедурах, що використовуються зловмисниками для отримання доступу до системи.

В контексті логів аномальних входжень ІА можуть включати:

- Неуспішні спроби аутентифікації: Послідовні неуспішні спроби входу під одним або декількома обліковими записами можуть свідчити про спробу перебору пароля або використання інших атак на автентифікацію.
- Незвичайні зміни в поведінці: Виявлення незвичайних змін у звичному поведінці користувачів, таких як незвичайна послідовність дій під час автентифікації або зміна звичайних параметрів, може вказувати на атаку або недоброзичливу діяльність.
- Аномальний обсяг запитів: Виявлення великої кількості запитів на автентифікацію з одного або кількох джерел може свідчити про спробу перевантаження системи або атаку на підсистему автентифікації.

Індикатор ризику вказує на потенційну загрозу безпеці інформаційної системи або мережі. Індикатор ризику виокремлюється з певних змін або активності, які можуть свідчити про незвичайну поведінку або потенційні проблеми безпеки.

В логах аномальних входжень в систему автентифікації можуть бути різноманітні індикатори ризику, що вказують на можливі атаки або компрометацію системи. Деякі приклади таких індикаторів ризику включають:

- Невдалі або незвичайні спроби входу: Записи про багаторазові невдачі автентифікації, спроби введення неправильних паролів або незвичайні спроби доступу можуть свідчити про можливу атаку або недобропорядну діяльність.
- Незвичайний обсяг або патерни активності: Великі обсяги аутентифікаційних запитів, незвичайні патерни використання ресурсів або незвичайні комбінації дій можуть бути ознаками небезпеки або спроби атаки.

Виявлення ІК, ІА та ІР в логах аномальних входжень дозволяє організаціям швидко розпізнати потенційні загрози та прийняти необхідні заходи безпеки. Ці індикатори допомагають покращити виявлення, реагування та запобігання атакам на систему автентифікації та забезпечити безпеку організації. Ці індикатори допомагають покращити виявлення вразливостей, розслідування інцидентів та зміцнення кібербезпеки організації. Окрім того, що основною нашою ідеєю є використання машинного навчання для підвищення рівня безпеки в системах автентифікації, це може бути використано в контексті ІК, ІА та ІР. Ми вважаємо, що логи можуть бути корисними в ідентифікації індикатор ризику, а також навіть виявляти деякі ідентифікатори атаки. Це все дасть змогу підняти рівень безпеки системи і зменшити швидкість реакції на інциденти.

1.4 Машинне навчання як рішення

Як вже було сказано, маючи дуже великі багато загроз для систем автентифікації, часто приходять до досить широкого спекту рішень які мають фокус на побудові безпечної інформаційної системи. Але на жаль, традиційні методи забезпечення безпеки, такі як логування, фільтрація трафіку та інші, не завжди можуть ефективно виявляти такі аномалії, оскільки зловмисники постійно шукають нові способи обійти ці заходи захисту.

У зв'язку з цим, важливо розробляти нові методи пошуку аномалій в системах автентифікації, що дозволяють виявляти підозрілу поведінку та забезпечувати більш високий рівень безпеки. Одним з таких методів є машинне навчання, яке дозволяє аналізувати великі обсяги даних та виявляти незвичайні патерни в поведінці користувачів. Наприклад, модель машинного навчання може бути навчена розпізнавати незвичайну частоту залогінень в певному часовому проміжку, що може свідчити про спроби зламати пароль або вкрасти обліковий запис. Пошук аномалій дозволяє виявити незвичайні або підозрілі дії, які можуть бути зв'язані зі зловживанням прав доступу або порушенням правил користування системою.

Машинне навчання - це галузь штучного інтелекту, яка застосовує статистичні прийоми для надання комп'ютерам здатності «навчатися» з даних, без того, щоби бути програмованими явно. Машинне навчання досліджує алгоритми та моделі, які можуть навчатися і покращуватися самостійно на основі даних. Простіше, машинне навчання - це процес навчання моделей прогнозувати або приймати рішення на основі вхідних даних.

У залежності від задачі, машинне навчання може використовувати різні типи моделей, такі як нейронні мережі, дерева рішень, метод опорних векторів та інші. Ці моделі навчаються на великому наборі даних, що дозволяє їм виявляти певні закономірності та залежності у даних, які потім можуть бути використані для прогнозів, класифікації чи інших задач машинного навчання.

Наприклад, машинне навчання може бути використане для прогнозування цін на нерухомість на основі історичних даних, визначення емоцій на основі текстових повідомлень, розпізнавання облич на фотографіях або визначення шаблонів у поведінці користувачів для персоналізованої рекомендації товарів.

Отже, дипломна робота присвячена вивченню проблеми забезпечення безпеки в системах автентифікації та методів пошуку аномалій в логах таких систем, що дозволяють ефективно виявляти потенційні загрози та підвищувати рівень безпеки систем за допомогою методів машинного навчання.

1.5 Аналіз існуючих досліджень

Під час огляду можливих рішень також було розглянуті дослідження на схожі теми, такими темам в основному були пошук аномалії в логах систем автентифікації, це все були різні системи, які мали за ціль захиститись від різних типів атак.

Багато наукових досліджень, які мають за мету вирішення проблеми аномалій в логах входів, використовують навчання з учителем, тобто на розміщених даних. Вони мають зразки, які вже явно ідентифіковані як аномальні і нормальні. Ці зразки використовуються для того, щоб навчити модель класифікувати нові входження і ідентифікувати аномальні зразки. Для цього використовуються багато різних алгоритмів класифікації, наприклад, Random Forest, Decision Tree, Support Vector Machine (SVM) та інші алгоритми класифікації, які ми описували в попередніх пунктах. Також досить популярним рішенням у дослідженнях виявились моделі навчання без вчителя, такі як Isolation Forest, методи кластеризації, методи щільності (DBSCAN, LOF). Нижче ми розглянемо декілька досліджень, щоб зрозуміти і оцінити, які рішення використовуються для цієї проблеми.

Наприклад, у дослідженні [1] розглядається можливість передбачення облікових записів, які ймовірніше будуть генерувати підозрілу активність у майбутньому, що може свідчити про те, що вони стали жертвами масштабної соціально-інженерної атаки. Для цього пропонується система попередження, яка використовує навчання з вчителем для ідентифікації облікових записів, поведінкові патерни яких вказують на їх схожість з іншими законними обліковими записами, які у минулому відзначалися як підозрілі. Система використовує алгоритм Random Forest і була реалізована і протестована на реальних даних, що охоплюють сотні мільйонів користувачів, і продемонструвала свою ефективність.

Дослідження [2] розглядає проблему виявлення аномальної поведінки користувачів на основі даних про їх авторизацію в системі. Зокрема, у роботі досліджується проблема виявлення атак на основі викрадення або перехоплення логін-даних користувачів. Автори статті пропонують метод, який використовує аналіз залежностей між користувачами та їхніми логін-даними для виявлення аномальної поведінки. Алгоритм опорних векторів (SVM) використовується для виявлення аномалій у залежностях між користувачами та їх даними автентифікації. Але слід зазначити, що порівняння велось з моделями, які шукають аномальні входи через правила або за допомогою статистичних методів.

Також існує інше дослідження, яке фокусується на навчанні без вчителя. Наприклад, дослідження [3] присвячене виявленню аномальної поведінки користувачів у системах з використанням розширеного алгоритму ізоляційного лісу. У цій роботі представлено виявлення аномальної поведінки користувачів завдяки розширеній версії алгоритму "Isolation Forest". Метод є швидким і масштабованим і не потребує наявності прикладів аномалій у тренувальному наборі даних. Алгоритм застосовується до корпоративного набору даних. Експериментальні результати показують, що система може виділити аномальні екземпляри від базової моделі користувача, використовуючи одну або комбіновані ознаки. У розширеному ізоляційному лісі використовуються додаткові техніки, які дозволяють обробляти категоріальні дані. Алгоритм потребує лише того, щоб для

кожної категоріальної ознаки значення мали порядок. Порядок може бути довільним. Потім кожне значення відображається на числове значення на основі його порядку.

Дослідження [4], зосереджене на виявленні аномалій в аутентифікаційних процесах (VPN). Дослідження використовує алгоритм EM (Expectation-maximization) для кластеризації подій автентифікації на основі їх характеристик часу та місця з метою виявлення моделей поведінки, які відхиляються від норми. Кластеризація групує схожі з'єднання за їх характеристиками. Далі використовуються ймовірнісні розподіли, щоб виявити з'єднання, які не підходять до груп і є потенційно аномальними. Ці аномалії можуть потребувати додаткового дослідження з боку аналітика з безпеки. Застосовується підхід зі значеннями z-оцінок, які вказують, наскільки сильно кожне з'єднання відрізняється від типового з'єднання в своїй групі. За допомогою цих оцінок аналітик може вибрати потенційно підозрілі з'єднання для подальшого аналізу. Це дослідження використовує комбінований підхід, де машинне навчання комбінується з виключно статистичною ознакою.

Також дуже близькою до нашого дослідження є [5] яке розглядає багато технік машинного навчання для виявлення бокового руху (Lateral movement, LM) в мережі за допомогою протоколу віддаленого робочого столу (RDP). Дана робота розглядає дещо конкретну задачу, але слід зазначити, що для нашого більш загального пошуку аномалій в логах, розгляд і аналіз результатів цього дослідження є дуже корисним. Отже, автор використовує алгоритми машинного навчання для аналізу трафіку RDP і виявлення шаблонів та аномалій, які вказують на LM. У статті оцінюється запропонований підхід, використовуючи набір даних трафіку RDP, зібраного з реальної мережі. Результати показують, що цей підхід є ефективним у виявленні бокового руху з високим рівнем виявлення та низьким рівнем помилкових позитивних результатів. Набір даних включає як звичайні, так і шкідливі сеанси RDP.

У роботі в основному використовувалися типові класифікатори для навчання з учителем, такі як Logistic Regression, Decision Tree, Gaussian-NB, Random Forest, а також ансамблеві методи класифікації, наприклад LogitBoost. Також були спроби комбінувати LogitBoost з методами навчання без вчителя, але ці комбінації показали гірші результати, ніж LogitBoost сам по собі.

Проте, використання нерозмічених даних у таких типах класифікаторів стає проблемою, оскільки ми не заздалегіть знаємо, які дані є аномальними, а які ні. Крім того, слід зазначити, що набір даних складається з декількох різних наборів і має фокус на конкретну атаку. Тому система може не давати точних результатів при використанні на реальних даних.

Саме з огляду на всі ці роботи було прийнято рішення більш детально розглянути проблему пошуку аномалій в логах входів за допомогою допомогою машинного навчання з фокусом на навчання без учителя. Адже сфера машинного навчання активно зростає останнім часом і дослідження в цій сфері можуть відкрити багато нових рішень для цієї проблеми.

Висновки до розділу 1

В першому розділі дипломної роботи ми розглянули об’єкт нашого дослідження – логи входів RDP автентифікації. Були наведені теоретичні відомості про протокол RDP, а також наведені особливості загроз для цього протоколу. Основною небезпекою є заволодіння віддаленого несанкціонованого доступу до системи, це може привести до ряду фатальних для системи загроз і атак. Наприклад отримання доступу до файлів, шифрування даних, підмінна даних та багато інших. Основною тезою нашого дослідження є те, що ми можемо використовувати методи машинного навчання для пошуку дивних, нетипових входів для того потенційно зупинити атаки і компрометацію системи. Це можливо за допомогою спеціальних методів машинного навчання які будуть описані та проаналізовані в наступному

розділу нашої роботи. Було проаналізовано в чому саме полягає проблема аномальних входів, що це означає і про що можуть свідчити дані входи, також нашою ідеєю є те, що ці входи можуть представляти індикатори атаки і ризиків. Окрім цього, були проаналізовані існуючі рішення, а також методи які використовувались в минулому для цієї проблеми.

2 МЕТОДИ ПОШУКУ АНОМАЛІЙ В МАШИННОМУ НАВЧАННІ

Машинне навчання - це інструмент, який можна використовувати для дуже різних задач. Сьогодні ми бачимо застосування машинного навчання в багатьох сферах нашого життя: ідентифікація образів на картинках, прогнозування бізнес-процесів, перетворення аудіо повідомлень в текст, навіть ідентифікація ракових пухлин в медицині. Для всіх цих задач використовуються багато різних і специфічних алгоритмів, підбір яких залежить від конкретної задачі, формату даних і обчислювальних потужностей. Наприклад, глибинне навчання (Deep Learning), як правило, потребує великого обсягу даних для навчання моделі, проте після цього нейронна мережа може розуміти складні зв'язки між параметрами і показувати вражаючі результати. У той же час, для деяких бізнес-задач, наприклад, прогнозування кількості покупців, можуть бути достатніми моделі на основі дерев рішень або інші моделі, які вимагають менших обсягів даних, ніж нейронні мережі. Однією з основних задач, що вирішуються за допомогою машинного навчання, є пошук аномалій. Ця задача має важливе застосування в системах операцій банків, при дослідженні різних показників важкої техніки (наприклад, двигунів для літаків), а також при аналізі різних медичних даних (попередження інфаркту, виявлення ракових пухлин) та інших областях. Ми розуміємо, що це дуже важлива задача, яка може не лише уникнути збитків компанії, а й врятувати життя людей. Але що саме це за задача і як вона формулюється?

Зазвичай пошук аномалій розуміється як ідентифікація рідкісних елементів, подій або спостережень, які значно відрізняються від більшості даних і не відповідають чітко визначеному поняттю нормальної поведінки. Такі приклади можуть викликати підозри, що вони згенеровані іншим механізмом, або виглядати несумісними з рештою цього набору даних. Спочатку аномалії шукали як відхилення або відсутність в даних, щоб допомогти статистичному аналізу, наприклад, для обчислення середнього значення або стандартного відхилення. Вони також видалялися для поліпшення прогнозів моделей, таких як лінійна

регресія, і, останнім часом, їх видалення сприяє ефективності алгоритмів машинного навчання. Однак, у багатьох застосуваннях самі аномалії є об'єктом інтересу і є найбажанішими спостереженнями в усьому наборі даних, які необхідно виявити і відокремити від шуму або незначних відхилень.

Існують три основні категорії методів виявлення аномалій.

1. Методи виявлення аномалій без вчителя (неконтрольовані алгоритми) визначають аномалії на непозначеному наборі даних, виходячи з припущення, що більшість зразків у цьому набору є нормальними, і шукаючи зразки, що виглядають якнайменше відповідними решті набору даних.
2. Методи контрольованого виявлення аномалій вимагають набору даних, що позначено як «нормальні» або «аномальні», та включають навчання класифікатора (ключовою відмінністю від інших задач класифікації є притаманно незбалансований характер виявлення викидів).
3. Методи напів-контрольованого виявлення аномалій створюють модель, що представляє нормальну поведінку, виходячи із заданого нормального навчального набору даних, і потім перевіряють правдоподібність того, що тестовий екземпляр було породжено вивченою моделлю.

Далі давайте більш детально розглянемо методи для кожної категорії виявлення аномалій і подивимось на основні алгоритмі які використовує кожен підхід.

2.1 Методи виявлення аномалій з вчителем

Методи виявлення аномалій з вчителем(або контрольоване виявлення аномалій) – це методи і алгоритми машинного навчання базуються на навчанні з

учителем. Як вже було сказано ці алгоритми використовуються навчання на розмічених наборах даних які мають мітки нормальний зразок і аномальний. Фактично, ця задача пошуку аномалій перетворюється на задачу бінарної класифікації. Класифікація в свою чергу це задача яка полягає в тому, що необхідно на основі ознак об’єкту прийняти рішення до якого з класів відноситься такий об’єкт. Єдина особливість полягає в тому, що зазвичай кількість аномалій значно менше аніж кількість нормальних зразків, такі класи називається незбалансованими. Саме тому нам необхідно отримати повне і точне охоплення аномальних зразків.

Отже, після навчанні моделі на таких наборах, її можна використовувати для прогнозування мітки для нових екземплярів, а екземпляри, які будуть визначені аномальними, можна позначити як потенційні аномалії, для подальшого дослідження або реакції системи на такі зразки. Для таких методів можна використовувати дуже багато різних алгоритмів навчання які здатні виконувати функцію класифікації, наприклад почнемо з лінійної регресії.

2.1.1 Логістична регресія

Логістична регресія є одним із методів машинного навчання, використовуваним для бінарної класифікації, коли необхідно визначити, до якого класу належить певний об'єкт на основі набору вхідних ознак. Вона також може бути розширена для виконання мультикласової класифікації.

Основна ідея логістичної регресії полягає в тому, що вона моделює ймовірність того, що певний об'єкт належить до певного класу, використовуючи логістичну функцію, також відома як сигмоїда(рис. 1.1). Логістична функція приймає будь-яке дійсне число і "стискає" його в діапазоні від 0 до 1. Саме її інтерпретують як ймовірність.

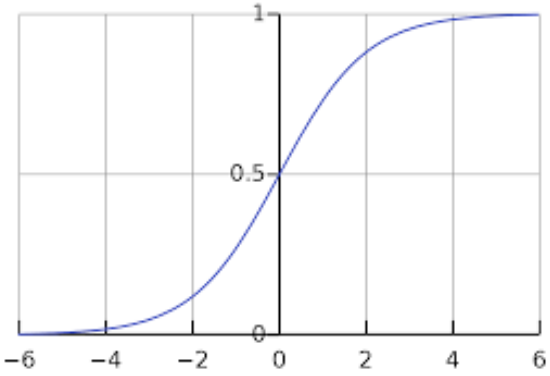

Рисунок 2.1 - Сігмоїда

Перш за все для логістичній регресії обчислюється лінійна комбінація вхідних ознак, яку можна сформулювати як:

$$z(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n. \quad (2.1)$$

де $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$ - параметри моделі, а $x_1, x_2, \ldots, x_n$ - вхідні ознаки об'єкта.

При отриманні лінійної комбінації з ми використовуємо сігмоїду для того, щоб перетворити це значення у проміжок [0,1]. Отже сігмоїду за допомогою формули можна описати як:

$$f(x) = \frac{1}{(1 + e^z)} \quad (2.2)$$

де z – це результат лінійної комбінації.

Значення в цьому проміжку показують ймовірність того, що деякий об'єкт є певним класом. Наприклад входження X є аномальним класом з ймовірністю 0.65. Зазвичай ми вважаємо, що об’єкт є певним класом якщо значення ймовірності більше аніж 0.5.

Далі задача навчання логістичної регресії полягає у підборі оптимальних значень коефіцієнтів b0, b1, b2, ..., bn, які найкраще апроксимують навчальні дані. Це може бути досягнуто за допомогою методу максимальної правдоподібності або інших оптимізаційних алгоритмів, наприклад, градієнтного спуску. Методи підбору параметрів використовуються для налаштування моделей з метою знаходження оптимальних значень параметрів, які найкраще відповідають

навчальним даним або досягають найкращої продуктивності моделі. Це допомагає підвищити точність прогнозування та здатність моделі до узагальнення на нові дані.

Наступним, ми розглянемо сімейство алгоритмів які зв’язані з деревами, для початку візьмемо Decision Tree.

2.1.2 Decision tree

Decision tree (дерево рішень) є моделлю машинного навчання, яка використовується для класифікації та регресії. Вона представляє собою деревоподібну структуру, де кожний вузол відповідає певному тесту на вхідних даних, а кожне ребро вузла вказує на можливі результати цього тесту. Вузли, що не мають дочірніх вузлів, називаються листками і містять мітку класу або прогнозовані значення. Наприклад ми маємо задачу прогнозування того які люди зможуть вижити при потопленні лайнеру, на основі деякого набору даних. Нижче ви можете побачити досить просте дерево, яке є прикладом використанням дерев для такої задачі(рис 1.2).

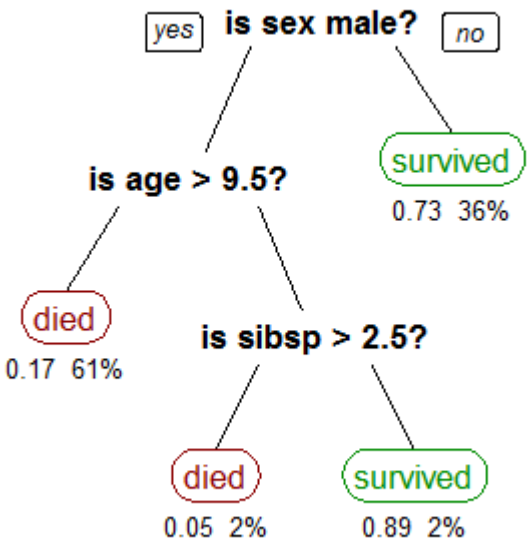

Рисунок 2.2 – Дерево рішень

Побудова дерева рішень відбувається на основі навчальних даних, де модель шукає оптимальні тести (змінні та їх значення), які найкраще розділяють дані на

різні класи або прогнозують вихідне значення. Цей процес включає рекурсивну розбиття набору даних на підмножини, засновані на різних характеристиках та їх значеннях, з метою максимізації інформаційного приросту або зменшення функції помилки. Після побудови дерева рішень воно може бути використано для класифікації нових даних шляхом пройдення по шляху від кореня до листка, враховуючи значення вхідних характеристик і тестів на кожному вузлі. У випадку регресії, значення виходу буде прогнозуватися на основі значень вхідних характеристик у вузлах досягнення листків. Окрім дерево рішень є основою для створення більш потужних алгоритмів машинного навчання, таких як ансамблеві або навіть бустингу (boosting). Основна перевага дерева рішень полягає в його простоті та інтерпретованості. Воно в змозі моделювати складні залежності між характеристиками і вихідними значеннями, враховуючи важливість окремих характеристик. Однак, дерева рішень мають свої обмеження, зокрема схильність до перенавчання та низьку стійкість до варіацій в даних. Так як, великі дерева схильні призводити до перенавчання, тому існують методи підтримки обмеження розміру дерева, такі як обрізка (pruning). Крім того, дерева рішень можуть бути чутливі до невеликих змін в навчальних даних, що може призводити до недостатньої стійкості моделі.

Як ми сказали вище, дерева є основою і компонентом багатьох ансамблевих і бустингових алгоритмів. ансамблевим є алгоритм, який використовує декілька моделей одного типу (наприклад, дерев рішень) і об'єднує їх прогнози для отримання кінцевого прогнозу. Тобто використовують кілька дерев для отримання кращої прогнозної ефективності, ніж можна було б отримати від будь-якого із складових алгоритмів навчання окремо. Саме тому давайте розглянемо досить потужний і алгоритм який часто використовують - Random Forest.

2.1.3 Random Forest

Random Forest (або випадковий ліс) є ансамблевим методом машинного навчання, який використовується для класифікації та регресії. Він поєднує декілька

дерев рішень (decision trees), що навчаються на випадкових підмножинах навчальних даних та використовуються для отримання прогнозів.

Основна ідея Random Forest полягає у використанні ансамблю дерев рішень для покращення точності та стійкості прогнозів. Процес побудови Random Forest може бути узагальнений такими кроками:

- Вибір випадкової підвибірки з навчальних даних: З кожної навчальної вибірки вибирається підмножина даних випадковим чином (з повторенням або без повторення).
- Побудова дерев рішень: Для кожної випадкової підвибірки будується окреме дерево рішень, застосовуючи процес побудови дерева рішень, як описано раніше. Процес розбиття вузлів може використовувати випадковий підвибір характеристик замість усіх характеристик.
- Прогнозування: Коли Random Forest побудовано, прогнози отримуються шляхом агрегації прогнозів кожного дерева. Для класифікації може використовуватись голосування більшості(рис 1.3), а для регресії - середнє значення.

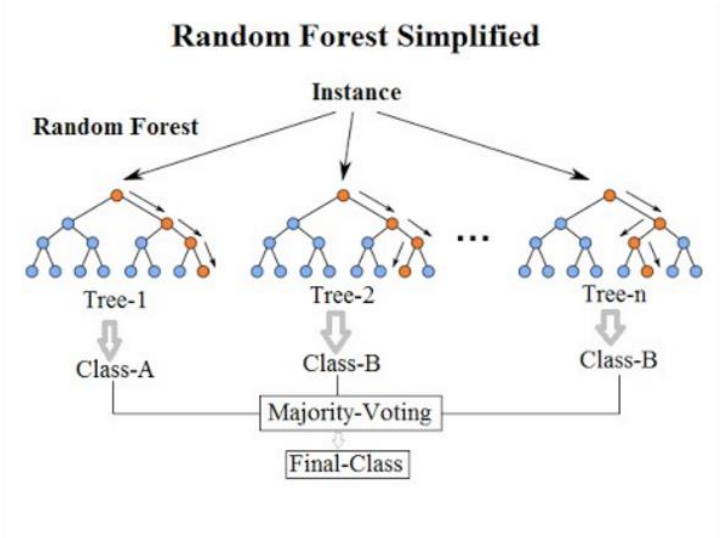

Рисунок 2.3 – Схема Random Forest для задачі класифікації

Random Forest досить сильний алгоритм через те, що він має здатність до обробки великих обсягів даних та високої розмірності характеристик (оскільки обчислення можна розпаралелити і виконувати на багатьох процесорах), при цьому алгоритм показує досить велику точність прогнозів. Також можна оцінювати вплив

кожної характеристики на прийняття рішень. Тим не менш його може бути важко інтерпритувати коли в моделі багато дерев, також досить багато параметрів підбір яких займаю великий проміжок часу. Незважаючи на обмеження, Random Forest залишається потужним і популярним методом машинного навчання, здатним до рішення широкого спектру завдань прогнозування та класифікації.

2.1.4 Boosting

Бустинг є технікою побудови ансамблю, де моделі побудовані послідовно. У бустингу кожна наступна модель намагається скоригувати помилки, зроблені попередніми моделями. Один з найпопулярніших методів бустингу - Gradient Boosting, він використовує градієнтні методи оптимізації для побудови послідовної моделі. Основна ідея полягає в тому, що початкова модель модель використовується для отримання початкових прогнозів. Далі відбувається обчислюється різниця між прогнозами початкової моделі і фактичними значеннями, що визначає помилки. Нова модель додається до ансамблю, спрямована на зменшення помилок попередніх моделей. Це робиться шляхом навчання моделі на "залишкових помилках" - різниці між фактичними значеннями і прогнозами попередніх моделей. Прогнози ансамблю оновлюються шляхом додавання прогнозів нової моделі з попередніми прогнозами. Далі ці кроки повторюються декілька разів, додаючи нові моделі до ансамблю і покращуючи прогнози.

Одна з відмінностей Gradient Boosting полягає в тому, що він використовує градієнтні методи оптимізації для знаходження оптимальних ваг моделей, які мінімізують функцію втрати. Це дозволяє ефективно пристосовувати моделі до навчальних даних і покращувати їх продуктивність. Gradient Boosting включає в себе різні варіації, такі як XGBoost, LightGBM і CatBoost, які використовують різні оптимізаційні підходи та оптимізовані структури даних для поліпшення швидкості та продуктивнjcns.

Серед сильних сторін таких типів побудування моделей є висока точність, здатність до моделювання складних зв’язків, а також адаптивність. Як правило, такі

моделі показують себе найкраще серед методів машинного навчання, але за все приходиться платити. Через специфіку побудови моделей вони маються дуже велику витрату обчислювальної потужності, також через цю ж причину, дуже часто важко підбирати гіперпараметри. Тим не менш, маючи багато обчислювальних потужностей і достатньо даних ви можете побудувати дуже точну і адаптивну модель машинного навчання яка здатна до задач класифікації, регресії, та пошуку аномалій.

2.1.5 SVM

Метод опорних векторів (Support Vector Machines, SVM) - це метод машинного навчання, який використовується для вирішення завдань класифікації та регресії. Алгоритм базується на ідеї знаходження оптимальної гіперплощини, яка найкраще розділяє два класи даних. Основна мета полягає в максимізації межі між цією гіперплощиною і найближчими до неї зразками обох класів. Зразки, що знаходяться найближче до гіперплощини, називаються опорними векторами. Розгляньмо деякі задані точки даних, кожна з яких належить до одного з двох класів, а метою є вирішувати, в якому класі буде нова точка даних. У випадку опорно-векторних машин точку даних розглядають як p-вимірний вектор, і хочуть дізнатися, чи можливо розділити такі точки (p-1) вимірною гіперплощиною. Це називається лінійним класифікатором. Існує багато гіперплощин, які могли би розділяти одні й ті ж дані. Одним із варіантів розумного вибору найкращої гіперплощини є такий, який пропонує найбільший проміжок, або розділення між двома класами. Тож ми обираємо гіперплощину таким чином, щоби відстань від неї до найближчих точок даних з кожного боку була максимальною. Така гіперплощина, якщо вона існує, відома як максимально розділова гіперплощина, а лінійний класифікатор, що вона його визначає, — як максимально розділовий. На рисунку 1.4 ми можемо бачити простий приклад побудови площини яка буде розділяти зразки на 2 класи. Ми бачимо, що H1 не розділяє ці класи. H2 розділяє, але лише з невеликим розділенням. H3 розділяє їх із максимальним розділенням.

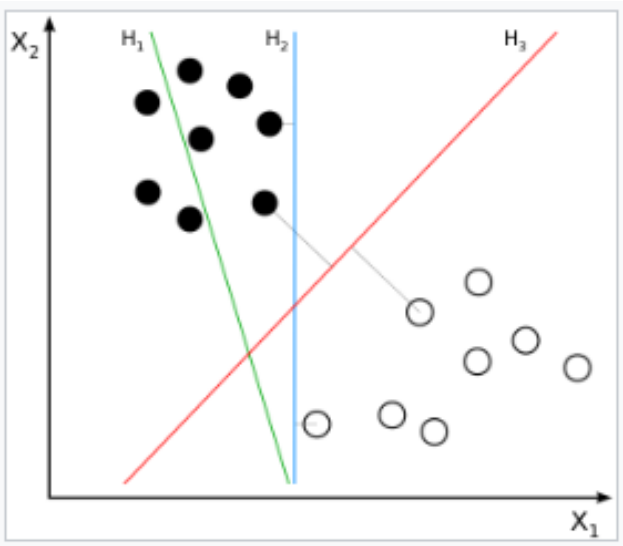

Рисунок 2.4 – побудова площини для векторів.

Для побудови площини використовується певний набір параметрів. Параметри ядра контролюють форму та комплексність розміщення границі розділення між класами. Вони можуть включати тип ядра (лінійне, поліноміальне, RBF тощо). Також використовуються параметр ширини ядра, регуляризації, а також додаткові параметри можуть включати тип оптимізатора, критерій зупинки, тип обробки категоріальних ознак та інші. Отож, основними перевагами алгоритму SVM є його здатність до ефективної роботи з великими наборами даних, можливість використання ядер для розв'язання нелінійних задач, а також висока точність класифікації. Однак, його недоліком є чутливість до високої розмірності даних та складність налаштування гіперпараметрів.

2.1.6 Баєсівський класифікатор

Наївний Баєсівський класифікатор (Naive Bayes Classifier) - це простий ймовірнісний класифікатор, який ґрунтується на теоремі Баєса та припущенні про незалежність ознак. Він використовує статистичні методи для прогнозування ймовірності належності зразка до певного класу на основі його ознак. Основна ідея класифікатора полягає у визначенні ймовірностей належності зразка до кожного класу та виборі класу з найвищою ймовірністю. За припущенням незалежності ознак, ймовірність належності до класу обчислюється шляхом множення ймовірностей належності кожної окремої ознаки до цього класу. Це надає

класифікатору швидкодіючість та простоту, але може призводити до невірних прогнозів, якщо припущення про незалежність ознак не виконується.

Інші схожі класифікатори, які також засновані на ймовірностях та використовують схожі припущення, включають:

- Багатокласовий наївний Баєсівський класифікатор (Multinomial Naive Bayes): Цей класифікатор використовується, коли ознаки є категоріальними або лічильними, і моделюється розподіл Мультиноміального типу.
- Баєсівський класифікатор з використанням вибіркового навчання (Bayesian Network Classifiers with Selective Learning): Цей класифікатор використовується для моделювання взаємозв'язків між ознаками та класами, використовуючи Баєсовські мережі. Він може розглядати залежність між ознаками, що робить його більш гнучким, ніж наївний Баєсівський класифікатор.
- Гауссівський наївний Баєсівський класифікатор (Gaussian Naive Bayes): Цей класифікатор використовується, коли ознаки мають нормальний розподіл. Він моделює ймовірності за допомогою гауссівських розподілів та розглядається як розширення наївного Баєсівського класифікатора для неперервних ознак.

2.2 Методи виявлення аномалій без вчителя

Пошук аномалій без учителя (Unsupervised Anomaly Detection) - це процес виявлення незвичайних або аномальних зразків або подій у даних, де невідомо які зразки є і можуть вважатись аномальними. Або якщо ми не маємо даних з попередньо позначених аномалій та нормальний зразків для навчання. У цьому варіанті машинного навчання, модель має сама на основі деяких ознак виділити ті значення, які сильно виділяються або їх поява є неочікуваною.

Основна мета пошуку аномалій без учителя - виявлення рідкісних, незвичайних або девіантних зразків, які відрізняються від загальної поведінки даних. Це може бути корисно для виявлення випадків шахрайства в операціях з кредитними картками, виявлення несправностей у промисловому обладнанні, виявлення загроз кібербезпеці, моніторингу медичних даних тощо.

У методах пошуку аномалій без учителя зазвичай використовуються різноманітні статистичні підходи. Вони зосереджені на виявленні зразків, які значно відрізняються від більшості даних, засновуючись на внутрішніх закономірностях і структурі даних. Незважаючи на свою потужність, пошук аномалій без учителя може також породжувати помилкові спрацьовування і не може гарантувати 100% точність. Тому важливо уважно оцінювати результати і використовувати додаткові методи для перевірки та підтвердження виявлених аномалій.

Цей метод пошуку аномальних зразків базуються на припущені того, що відсоток аномалій у наборі даних, зазвичай біля 1%. Оскільки аномалії рідкісні та невідомі користувачеві під час навчання, виявлення аномалії в більшості випадків зводиться до проблеми моделювання нормального розподілу даних і визначення вимірювання в цьому просторі, щоб класифікувати зразки як аномальні або нормальні. Існує декілька методів при пошуку аномалій без учителя далі ми розглянемо деякі з них.

2.2.1 Кластеризація

Кластеризація - це процес групування схожих об'єктів або даних разом у класи або кластери. Вона використовується для виявлення прихованих структур у даних, без наявності заздалегідь відомих міток або класів. Основна мета знайти схожість між об'єктами і розділити їх на групи, так щоб об'єкти всередині однієї групи були більш схожими між собою, ніж з об'єктами з інших груп. Кластеризація може бути використана для різних цілей, таких як розуміння структури даних, виявлення подібних груп споживачів або виявлення аномалій.

Точки даних, які належать до невеликих або розріджених кластерів, або які знаходяться далеко від свого призначеного кластеру, можуть вважатися аномаліями. Приклади методів кластеризації включають k-середні, ієрархічну кластеризацію та інші. Кластеризація має кілька переваг, таких як простота, інтуїтивність та масштабованість.

Давайте розглянемо найтиповіший метод кластеризації k-середні. Існує багато варіацій цього алгоритму, які можуть виконувати задачу на спеціфічних даних, такі як k-modes(кластеризація категоріальних), k-prototypes(кластеризація для змішаних даних) та інші. Але давайте розглянемо алгоритм на якому все базується.

2.2.1.1 K-means

Кластеризація K-means — це простий і широко використовуваний алгоритм кластеризації, який розбиває дані на заздалегідь визначену кількість кластерів з метою мінімізації суми квадратів відстаней між точками даних і призначеними центрами кластерів. Принцип алгоритму полягає в пошуку таких центрів кластерів та наборів елементів кожного кластера при наявності деякої функції Ф, що виражає якість поточного розбиття множини на k кластерів, коли сумарне квадратичне відхилення елементів кластерів від центрів цих кластерів буде найменшим:

$$\sum_{i=1}^{k} \sum_{xj \in Si} (xj - \mu i)^2 \qquad (2.3)$$

де k — число кластерів, S_i— отримані кластери, i={1,2,…, k}, μ_i— центри мас векторів $x_j \in S_i$.

Роботу алгоритму можна описати наступним чином:

1. Вибрати кількість кластерів (k), яку потрібно сформувати.

2. Ініціалізувати початкові центроїди кластерів. Це може бути зроблено випадковим вибором k точок з набору даних або за допомогою іншого методу ініціалізації.

3. Повторювати наступні кроки, поки не буде досягнуто зупинки:

- Присвоїти кожну точку до найближчого центроїда за допомогою обчислення відстані (найчастіше використовується евклідова відстань).
- Перерахувати нові центроїди для кожного кластеру, використовуючи середнє значення всіх точок, що належать до кластеру.
- Перевірити, чи змінилися центроїди. Якщо так, повторити попередні два кроки; в іншому випадку зупинити алгоритм.

4. В результаті отримуємо k кластерів, де кожна точка належить до одного з кластерів на основі її відстані до центроїдів.

У контексті виявлення аномалій k-середні можна використовувати для ідентифікації точок даних, які погано вписуються в жоден із кластерів або мають велику відстань до призначеного центру кластера. Також іноді аномалією можна вважати кластер з низькою кількістю елементів або найменший кластер.

K-means відноситься до методів кластеризації, де кластери формуються шляхом знаходження K центроїдів, які найкраще представляють групи даних. Але також існують ієрархічні алгоритми кластеризації - вони розподіляють дані у вигляді дерева (дендрограми), де кожен вузол представляє кластер або підкластер. Далі вони об'єднуються у все більші й більші кластери. Також існують алгоритми на основі моделей змішаних гаусів. Такі алгоритми базуються на ймовірнісній моделі, в якій кожен кластер представлений гаусівським розподілом, наприклад EM-кластеризація (Expectation-Maximization), нижче наглядне порівняння EM і K-means (рис 1.5). Тут слід зазначити те, що кожна окрема задача вимагає окремого підходу і підбору алгоритму.

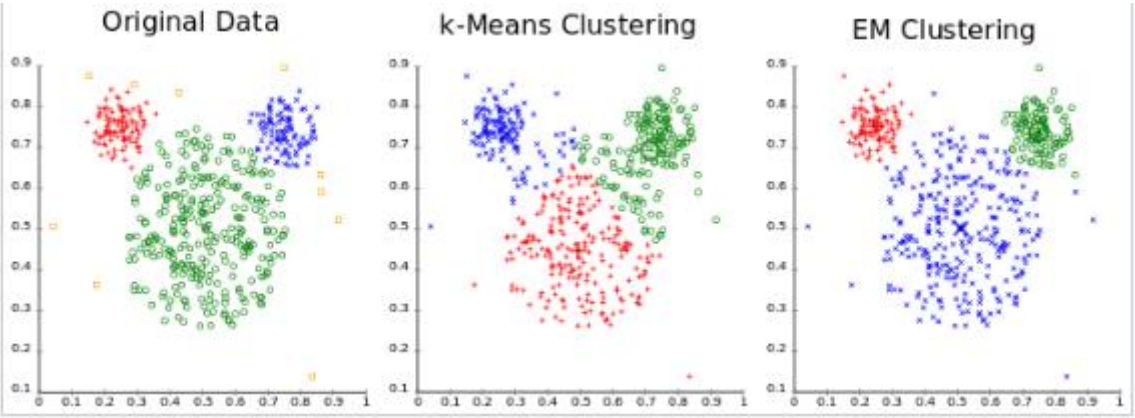

Рисунок 2.5 – порівняння EM і K-means

2.2.1.2 DBSCAN

Дуже корисним і цікавим методом є DBSCAN. DBSCAN (англ. density-based spatial clustering of applications with noise) - це алгоритм кластеризації на основі щільності, який об’єднує точки даних, які знаходяться близько одна до одної в просторі даних і мають мінімальну кількість сусідів на заданій відстані. Роботу алгоритма можна описати так:

- Вибір початкової точки: Випадково вибирається точка з невідвіданих даних або даних, які ще не були призначені до кластеру.
- Виявлення сусідів: Визначаються всі точки з заданого околу(epsilon), які є сусідами обраної початкової точки. Ці зразки включаються до сусідського околу обраної точки.
- Перевірка щільності: Якщо кількість точок в сусідському околі перевищує задану межу (minPts), то обрана точка вважається ядром кластера.
- Розповсюдження кластера: Розпочинаючи з ядра кластера, виконується пошук і додавання усіх сусідніх точок до кластера. Цей процес виконується рекурсивно для кожної нової точки, яка додається до кластера. Тобто якщо сусідні точки також мають достатню щільність, то процес розповсюдження кластера продовжується.
- Повторення кроків 2-4: Повторюються кроки 2-4 для кожної невідвіданої точки, поки всі точки не будуть пройдені і призначені до відповідних кластерів.

- Формування шуму: Точки, які не належать до жодного кластера і не мають достатньої щільності, вважаються шумом або аномаліями.

Нижче ми можемо бачити наглядний приклад того, що під кожну задачу потрібно обирати відповідний алгоритм кластеризації, а також те, що необхідно завжди оцінювати розподіл даних(рис 1.6).

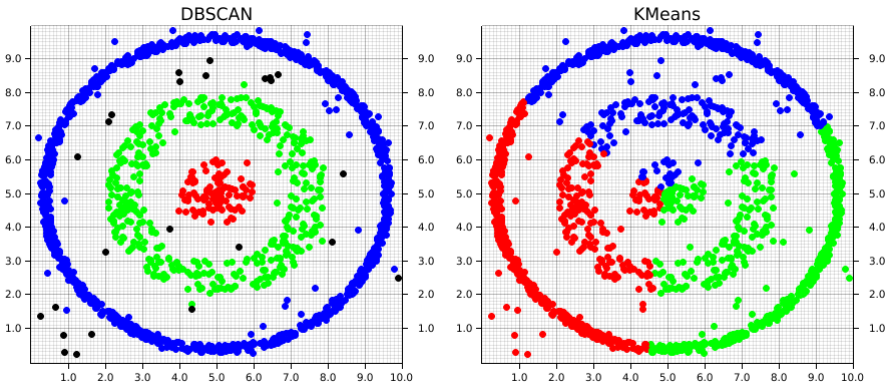

Рисунок 2.6 DBSCAN і K-means

У контексті виявлення аномалій DBSCAN можна використовувати для ідентифікації точок даних, які ізольовані або мають набагато меншу щільність, ніж навколишні точки даних. Як вже було написано вище алгоритм виділяє точки які не вдалось віднести до деякого кластеру(це відбувається через те що в деякому околі не достатньо точок для об’єднання в кластер), саме такі точки алгоритм позначає як аномальні.

Насправді, DBSCAN стоїть між двома основними методами для пошуку аномалій: методами на основі кластеризації та методами на основі щільності. Далі будуть розглядатись методи на основі щільності та похідні від них. Ці методи часто використовуються і вони є досить ефективним і потужним інструментом для нашої задачі.

Отож, методи кластеризації можуть бути корисними для виявлення аномалій у великих наборах даних, де важко вручну виявити закономірності або викиди. Вони можуть застосовуватись як до задач навчання без вчителя, так і до напівнаглядного навчання, що корисно в разі обмеженої кількості позначених

даних. Також здатні виявляти як глобальні, так і локальні аномалії, залежно від побудови моделі, надаючи комплексне уявлення про дані.

Тим не менш, кластеризація має деякі мінуси: Методи кластеризації можуть бути чутливими до вибору параметрів, таких як кількість кластерів, метрика відстані та алгоритм кластеризації. Неправильний вибір параметрів може призвести до поганої кластеризації та неточного виявлення аномалій. Окрім цього, кластеризація обчислювально витратна, особливо для великих наборів даних, що обмежує їх застосовність у реальних сценаріях.

2.2.2 Методи на основі щільності.

Метод виявлення викидів на основі щільності - це техніка, що досліджує співвідношення між щільністю об'єкта та його оточуючих сусідів. Використовуючи цей метод, ми можемо виділити дані, які знаходяться в областях зі збільшеною щільністю як нормальні, а ті, що знаходяться в областях з меншою щільністю, як аномальні. Цей підхід є ефективним, оскільки багато реальних наборів даних мають складну структуру, де об'єкти можуть бути відхиленнями від їхнього локального оточення, а не від глобального розподілу даних. Основне припущення цих методів полягає в тому, що щільність навколо нормального об'єкта подібна до щільності навколо його сусідів, тоді як щільність навколо відхиленого об'єкта значно відрізняється від щільності навколо його сусідів. Ці методи не потребують вказівки кількості кластерів або позначок, замість цього вони використовують поріг щільності або розмір сусідства для визначення рівня щільності. Приклади методів на основі щільності включають DBSCAN, LOF.

До сильних сторін можна віднести те, що методи на основі щільності можуть виявляти як глобальні, так і локальні аномалії в наборі даних, надаючи комплексний погляд на дані. Також ці методи можуть автоматично адаптуватися до змін щільності точок даних, що робить їх корисними для виявлення аномалій у динамічних середовищах. Вони не вимагають апріорної інформації про кількість кластерів або форму розподілу даних, що робить їх більш гнучкими та стійкими до різних типів даних.

Але в той же час методи на основі щільності можуть бути чутливими до вибору параметрів, таких як радіус сусідства або мінімальна кількість точок, необхідна для формування кластера(як в випадку DBSCAN). Поганий вибір параметрів може призвести до занадто великої або занадто малої кількості кластерів, що призведе до неточного виявлення аномалії. Методи на основі щільності можуть бути обчислювально витратними, особливо для великих наборів даних, що обмежує їх застосовність у реальних сценаріях.

2.2.2.1 Local outlier factor

Алгоритм локального викиду (Local outlier factor, LOF) — це метод виявлення аномалій без вчителя, який обчислює відхилення локальної щільності даної точки даних відносно її сусідів. LOF базується на співвідношенні локальної щільності області навколо точки та локальної щільності її сусідів. Він враховує відносну щільність точок даних. Простими словами, LOF порівнює локальну щільність точки з локальною щільністю k-найближчих сусідів і дає оцінку як кінцевий результат. Давайте детальніше подивимось як працює алгоритм.

Перш за все алгоритм використовуючи обрану метрику відстані (наприклад, Евклідову або Манхеттенську), обчислює відстань між кожним об'єктом X і його k найближчими сусідами.

Далі, нам необхідно визначити щільність локальної доступності(Local Reachability Density, LRD) для кожної точки даних. Для кожного об'єкта X обчислюється щільність, яка вимірюється відносно до його k найближчих сусідів. Фактично є оцінкою щільності оточення деякої точки і може бути сформульована як:

$$\mathrm{L}RD(X) = {}^{1}\Big/\frac{\sum_{y\in Nk(X)} Distance(X,y)}{|Nk(X)|} \quad (2.4)$$

де X – деякий об’єкт, Nk(X) – к-сусідів для об’єкта X.

Після того як ми визначили локальну щільність, обчислюємо LOF(X), використовуючи формулу:

$$LOF(X) = \frac{\sum_{y \in Nk(X)} \frac{LRD(y)}{LRD(X)}}{|Nk(X)|} \quad (2.5)$$

де X – деякий об’єкт, Nk(X) – к-сусідів для об’єкта X.

Ця формула обчислює середнє відношення локальної щільності кожного найближчого сусіда до локальної щільності об'єкта X. LOF(X) показує, наскільки аномальним є об'єкт X порівняно з його найближчими сусідами. Чим більше значення LOF(X), тим більша ймовірність, що об'єкт X є аномалією. Після отримання оцінок аномалій встановлюється поріг аномальності. Якщо значення LOF(X) перевищує цей поріг, об'єкт X вважається аномалією. Поріг може бути встановлений експертно або вирахований автоматично з використанням статистичних методів, таких як квантилі. На рисунку 1.7 ми можемо бачити наглядний приклад того, як оцінка аномальності залежить від щільності розподілу точок. Чорними точками позначається зразки даних. А червоним колом величину оцінки аномальності, причому саме радіус кола є значенням яке залежить від LOF.

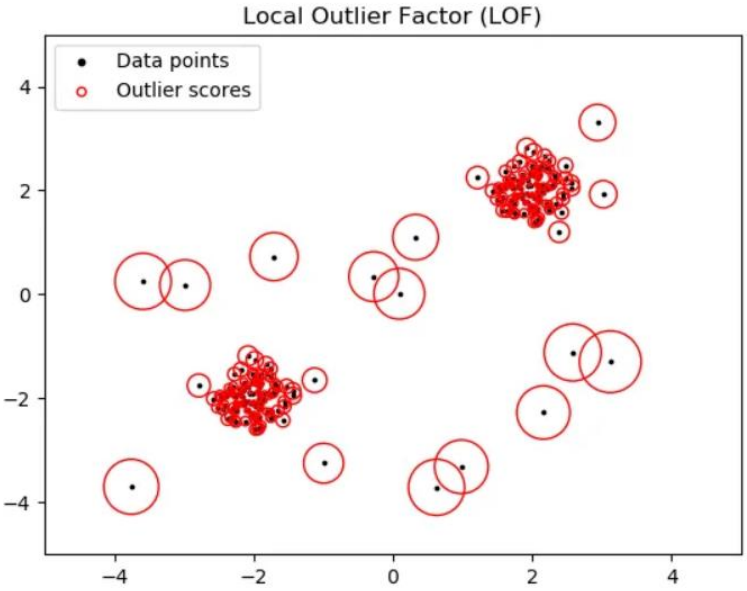

Рисунок 2.7 – Розподіл точок і значення LOF для них.

До сильних сторін LOF можна віднести його універсальність і те, що алгоритм враховує контекст даних. Алгоритм здатний працювати з різними типами даних, такими як числові, категоріальні або змішані дані. Він не залежить від конкретних припущень про розподіл даних і може бути застосований до широкого спектра завдань виявлення аномалій. LOF враховує контекст навколишніх об'єктів при визначенні аномалій. Він порівнює щільність кожного об'єкта зі щільністю його найближчих сусідів, що дозволяє виявляти аномалії в залежності від контексту, в якому вони знаходяться. Але в той же час алгоритм досить вимогливий до обчислювальних ресурсів, особливо при великій кількості об'єктів і вимогах до точності. Обчислення взаємний відстаней і локальних щільностей вимагає значних обчислювальних ресурсів і часу. Вибір оптимального значення k-сусідів може бути нетривіальною задачею і впливати на ефективність алгоритму. В тому числі як і більшість алгоритмів, LOF може стикатися з проблемою прокляття розмірності, коли високорозмірні дані стають менш ефективними для аналізу і виявлення аномалій.

2.2.3 Isolation forest

Ізоляційний ліс є це метод машинного навчання, який фактично є ансамблем бінарних дерев, де кожен зразок даних є листком дерева. Він базується на використанні дерев рішень для швидкого виділення аномальних спостережень з набору даних.

Основна ідея Isolation Forest полягає в тому, щоб ізолювати аномалії шляхом поділу набору даних на багато малих дерев рішень. Він ґрунтується на двох важливих припущеннях:

1. Аномальні точки є рідкісними і вони можуть бути ефективно виявлені за допомогою відносно коротких шляхів, в порівнянні зі шляхами нормальних точок. Тобто, аномалії можуть бути швидше виявлені, оскільки їх легше відділити від інших точок.

2. Нормальні точки потребують багато більше поділів для того, щоб бути ізольованими від інших точок. Тобто, нормальні дані потребують більшої кількості поділів для того, щоб їх виявити як аномальні.

На прикладі(рис 1.8) ми бачимо, що аномальні зразки виділяються деревом дуже швидко, в свою чергу зразки які ми вважаємо нормальними важко швидко ізолювати за допомогою дерева. Саме такий розподіл і виділення аномалій є нашою ціллю.

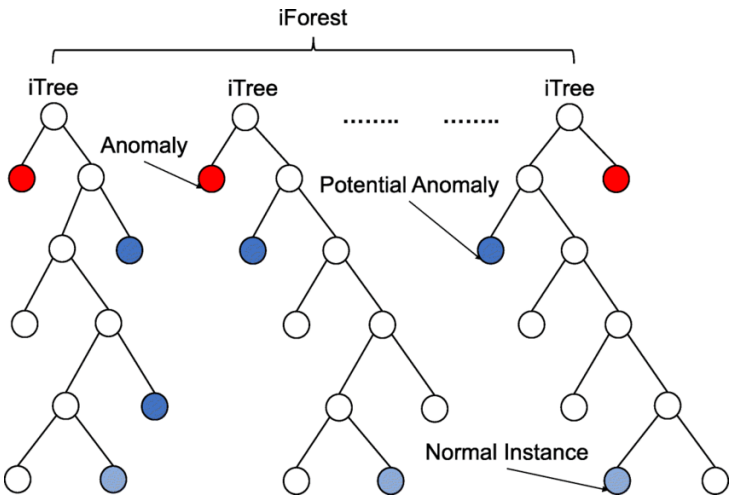

Рисунок 2.8 – Ізоляційний ліс

Основні кроки роботи Isolation Forest можна описати так:

1. Вибір випадкового піднабору даних з навчального набору
2. Побудова випадкового дерева: Для обраної підвибірки даних будується випадкове дерево. При цьому дерево будується рекурсивно, розбиваючи простір ознак на дві частини шляхом вибору випадкового розподілу ознаки і порогового значення. Розбиття продовжується досягнення заданого критерію зупинки, наприклад, коли досягнуто максимальну глибину дерева або коли кількість точок в листових вузлах становить одну або менше.
3. Повторення кроків 1 і 2. Перші два кроки повторюються для кожної підвибірки, але з кожним разом випадково обирається нова підвибірка.

4. Коли всі дерева побудовані, вони комбінуються для утворення ізоляційного лісу.

5. Для кожного вхідного зразка обчислюється середня довжина шляху досягнення його в лісі.

Після побудови всіх випадкових дерев алгоритм визначає оцінку аномальності для кожної точки даних. Оцінка аномальності визначається як середнє число розділень, необхідних для ізоляції цієї точки. Чим менше розділень потрібно, тим більш аномальною вважається точка. Аномаліями називають ті зразки, які мають довжину(оцінку аномалільності) шляху меншу за певний поріг. Цей поріг може як і в минулому алгоритмі визначитись експертами або за допомогою статистичних методів.

Ізоляційний ліс досить потужний алгоритм, він має ряд сильних сторін, наприклад те, що алгоритм працює швидко навіть з великими обсягами даних. Також він може працювати з високою розмірністю та не вимагає нормалізації або зменшення розмірності. Тим не менш, алгоритму іноді може бути важно знайти досить глибоку залежність між аномаліями і нормальними зразками.

Класифікувати і віднести Ізоляційий ліс до конкретного підходу або класу алгоритмів пошуку аномалій досить важко, але точно можна сказати, що це ансамблевий алгоритм. Крім того, Ізоляційний ліс може вважатися гібридним алгоритмом, оскільки поєднує елементи як distance-based (засновані на відстанях) так і density-based (засновані на щільності) підходів. Він використовує дерева для створення розбиття в даних, але враховує ітерації та динамічно оновлює значення аномальності.

2.3 Статистичні методи пошуку аномалій

Один з підходів до виявлення аномалій - статистичний методи які базується на отриманні оцінки статистичних характеристик даних та порівнянні зразків з

різними показниками. Слід зазначити, що такі методи не є машинним навчанням, але вони можуть допомогти у виділенні нормальних даних, а також працювати в комбінації з машинним навчанням

До типових статистичних методів відносяться :

Стандартне відхилення. Один з найпростіших і найбільш поширених методів пошуку аномалій - це використання стандартного відхилення. Цей метод базується на припущенні, що дані, які знаходяться в межах кількох стандартних відхилень від середнього значення, є нормальними, тоді як дані, що перевищують певний поріг, вважаються аномаліями.

Заснований на розподілі. Інший популярний підхід - це використання статистичних розподілів для виявлення аномалій. Зазвичай, використовуються розподіли, такі як нормальний розподіл або розподіл Гаусса, де відхилення від очікуваних значень вважаються аномаліями. Для цього можуть використовуватися різні статистичні метрики, такі як Z-оцінка або p-значення.

Але більш детальніше давайте розглянемо експериментальні методи, які розглядають входження даних з усіма стовпцями і можуть бути використані в комбінації з іншими методами пошуку аномалій.

2.3.1 Методи виявлення аномалій на основі відстані

В методах виявлення викидів на основі відстаней враховується оточення об'єкта, яке визначається заданим радіусом. Об'єкт вважається викидом, якщо його оточення не містить достатню кількість інших точок. Це називається методами виявлення викидів на основі відстаней. Нижче ми коротко розглянемо методи які можна застосовувати в експериментальному форматі для вирішення проблеми аномалій.

Один з підходів, називається "Менше, ніж p зразків". В цьому методі точки, які мають менше, ніж p сусідніх точок в деякому околі, класифікуються як аномальні. Для кожної точки обчислюється відстань до її найближчих сусідів, і якщо ця відстань менша за певний поріг p, то точка вважається аномальною. Цей

метод може бути налаштований на рівень чутливості до аномалій шляхом вибору значення p.

Ще один розглянутий підхід - це "kNN-методи". Вони використовують відстань до всіх точок для виявлення аномалій. Наприклад, точка вважається аномальною, якщо відстань до її найближчої точки більша за певний поріг. Для визначення аномалій можуть використовуватись різні метрики відстаней, такі як сума відстаней до всіх точок або середня відстань до k найближчих сусідів.

Для зменшення обчислювальної складності існує ще один підхід - "Методи обрізки". Вони розглядаються як розширення до попередніх методів і спрямовані на зменшення складності обчислень. Методи обрізки спочатку поділяють простір вхідних даних на окремі регіони і використовують статистику, таку як мінімальний обмежуючий прямокутник чи кількість точок, для кожного регіону. Під час пошуку найближчих сусідів тестового прикладу порівнюється з обмежуючим прямокутником, в якому воно знаходиться, щоб визначити, чи можливо, що в неподалік розташованому регіоні є сусіди. Якщо це неможливо, регіон відкидається, що зменшує складність пошуку близьких точок.

Методи виявлення аномалій на основі відстані відноситься до статистичного пошуку аномалій та методів дата-майнінгу. Виявлення викидів на основі відстаней не є безпосередньо пов'язаним з машинним навчанням, хоча деякі алгоритми можуть використовувати підхід навчання з учителем, наприклад, при побудові моделей на основі зразків викидів. Однак, в загальному розумінні, методи, що базуються на відстанях, зазвичай використовуються для аналізу даних і виявлення викидів без використання класифікації або навчання моделей. Це включає розрахунок відстаней між точками даних і порівняння їх з заданим порогом аномальності. Тому, хоча виявлення викидів на основі відстаней може використовувати деякі статистичні методи і інструменти дата-майнінгу, воно не є суворою технікою машинного навчання.

2.3.2 Ентропійні методи

Пошук аномалій на основі інформаційної ентропії (Entropy-based anomaly detection) є методом виявлення аномалій в наборах даних, що ґрунтується на понятті ентропії. Ентропія є мірою невизначеності або випадковості в наборі даних. У виявленні аномалій ідея полягає в тому, що аномалії часто вносять несподівані або нерегулярні закономірності, що призводить до вищих значень ентропії порівняно з нормальними точками даних.

Підхід на основі ентропії обчислює ентропію кожної точки даних або підмножини точок даних та порівнює її з пороговим значенням. Якщо ентропія перевищує поріг, то точка даних класифікується як аномалія.

Процес виявлення аномалій на основі ентропії зазвичай включає наступні кроки:

1. Вибір ознак: З набору даних вибираються відповідні ознаки або характеристики, які відображають особливості або поведінку точок даних. Ці ознаки можуть бути числовими, категоріальними або поєднанням обох.
2. Оцінка ймовірностей: Оцінюється розподіл ймовірностей вибраних ознак. Це може бути зроблено за допомогою різних методів, таких як підгонка параметричних розподілів або непараметричні методи оцінки щільності.
3. Обчислення ентропії: Обчислюється ентропія для кожної точки даних на основі її розподілу ймовірностей. Ентропія часто вимірюється за допомогою метрик, таких як ентропія Шеннона або дивергенція Кульбака-Лейблера. Вищі значення ентропії вказують на більшу невизначеність або нерегулярність.
4. Встановлення порогу: Визначається порогове значення для класифікації точок даних як нормальних або аномальних. Цей поріг може бути встановлений вручну на основі доменних знань або визначений автоматично за допомогою статистичних методів або технік перевірки.

5. Виявлення аномалій: Виділяються точки даних, у яких значення ентропії перевищують поріг. Ці точки вважаються відхиленнями від очікуваних закономірностей або розподілів нормальних точок даних.

2.4 Напівнаглядовані методи виявлення аномалій

Напівнаглядований пошук аномалій полягає в тому, що модель навчається на нормальних зразках, але без знання про аномальні зразки. Метою є побудова моделі, яка здатна виявляти аномалії в нових зразках, які можуть відрізнятися від нормальної поведінки.

Одним з популярних підходів до напівнаглядованого пошуку аномалій є метод побудови моделі під назвою "One-Class Classifier", наприклад One-Class SVM або One-Class в нейронних мережах. Цей метод використовує лише нормальні дані для побудови моделі, яка може визначати, наскільки нові зразки близькі до нормального розподілу. Напівнаглядований пошук аномалій є потужним інструментом для виявлення аномалій у випадках, коли доступні лише нормальні дані.

Він дозволяє побудувати модель, яка може виявляти незвичайні зразки на основі відхилення від нормального розподілу, що є корисним у багатьох областях, включаючи кібербезпеку, фінансовий аналіз, медицину та багато інших.

2.4.1 One-Class SVM

One-Class SVM (Support Vector Machine) це алгоритм на основі SVM, який ми розбирали раніше, але навчається тільки на нормальних даних. Цей алгоритм використовується для виявлення аномалій шляхом визначення границі нормального класу в просторі ознак. В процесі навчанн One-Class SVM прагне знайти гіперплощину, яка максимізує відступ навколо нормальних точок даних. Відступ - це регіон між гіперплощиною та найближчими нормальними точками даних. Шляхом максимізації відступу алгоритм створює границю, яка охоплює

якомога більше нормальних екземплярів. One-Class SVM не вимагає позначених аномальних зразків для навчання. Алгоритм лише отримує доступ до набору нормальних даних і на основі цього набору будує модель, яка може класифікувати нові зразки як нормальні або аномальні.

Припустімо, ми навчаємо SVM одного класу на 100 точках даних із двовимірного розподілу Гаусса. Після цього ми додаємо 20 випадкових точок даних і 20 точок даних, взятих із рівномірного розподілу. Наша мета полягає в тому, щоб правильно класифікувати точки даних Гауса, ігноруючи дані, що викидаються з рівномірного розподілу. На рисунку(рис 1.9) ви бачите результат роботи такого алгоритму, побудовану границю, навчальні точки і точки для оцінки роботи алгоритму.

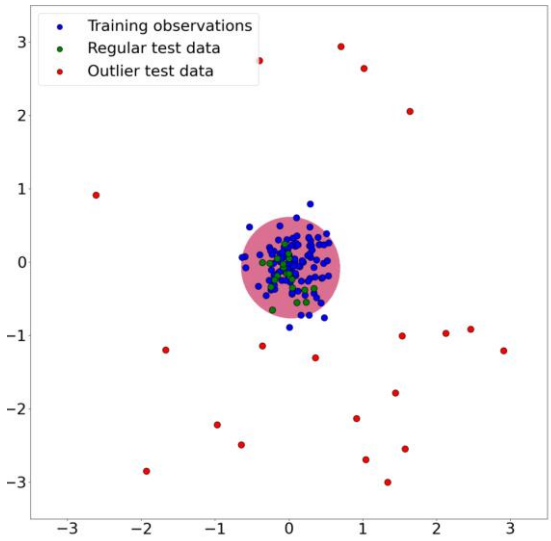

Рисунок 2.9 - One-Class SVM.

Цей метод є надзвичайно корисним в ситуаціях, коли ви маєте обмежену кількість аномальних даних або коли аномалії не представлені в навчальному наборі. One-Class SVM може бути застосований в різних областях, таких як виявлення шахрайства, кібербезпека, виявлення вторгнень та інші, де важливо виявити незвичайні та аномальні зразки.

Висновки до розділу 2

У другому розділі ми розглянули основні методи машинного навчання для вирішення проблеми пошуку аномалій в даних. Серед них було розглянуто 2 основних методів машинного навчання: навчання з учителем та без учителя, а також один спеціальних метод машинного навчання напівнаглядовий метод машинного навчання. Для навчання з учителем ми розглянули такі методи як: Логістична регресія, Дерева рішень, Випадковий ліс, бустинг, а також Метод опорних векторів. В свою чергу для методів без вчителя ми розглянули кластеризацію, методи на основі щільності, а також Ізоляційний ліс. Окрім цього хоч статистичні методи не є машинним навчання, але ці методи можуть бути корисними в вирішенні проблем аномалій в даних. Це можуть бути методи на основі стандартного відхилення, розподілу, відстані між точками або на основі ентропії. Для напівнаглядового методу машинного навчання ми запропонували метод One-Class SVM. Загалом було розглянуто досить багато методів навчання, кожен з них є корисним, але вибір методу залежить від задачі і формату даних.

3 ПОБУДОВА СИСТЕМИ ПОШУКУ АНОМАЛІЙ

3.1 Основні етапи побудови моделі машинного навчання

Процес побудови моделі машинного навчання може включати кілька етапів, які мають на меті створити ефективну та точну модель для вирішення конкретної задачі. Зазвичай все починається отримання і підготовки даних для нашого набору. Далі наступні етапи побудови моделі:

1) Дослідження даних для виявлення аномалій

Дослідження даних є початковою фазою під час процесу виявлення аномалій, оскільки воно допомагає провести початкове дослідження даних для виявлення закономірностей та викидів у даних. Воно визначає, як найкраще обробляти джерела даних, щоб отримати інформацію про аномалії та взаємозв'язки між ними. Під час етапу дослідження даних з використанням візуальних методів ми досліджуємо дані та виявляємо можливі викиди/аномалії. Ці викиди можуть потенційно вказувати на шахрайство, несанкціонований доступ і тд. Зловмисні дії зазвичай мають своєрідні характеристики, такі як кількість транзакцій/входів, нетипові часові зразки поведінки і т.д. Під час проведення дослідження даних перевіряють гіпотези та припущення, а потім визначати найбільш придатні техніки виявлення аномалій.

2) Обробка даних

Обробка даних передбачає покращення та доповнення набору даних релевантною додатковою інформацією. Це включає доповнення відсутніх або неповних даних, а також підготовку та структурування даних для алгоритмів машинного навчання. Неможливі комбінації даних, значення поза діапазоном і відсутні дані можуть призводити до неправильних результатів. Правильна попередня обробка даних гарантує високу якість даних, що, в свою чергу, забезпечує точність результатів.

3) Вибір алгоритмів машинного навчання для виявлення аномалій

Після кращого розуміння, який формат даних і зв’язків між ними, необхідно визначити найбільш відповідні алгоритми машинного навчання, які можуть допомогти з виявленням аномалій. Варто відзначити, що аномалії в наборах з високою розмірністю даних можуть відповідати досить складним закономірностям і можуть бути складними для виявлення. Тому алгоритми машинного навчання є дуже ефективними у визначенні аномалій у великих наборах даних.

4) Навчання моделі

Навчання моделі виявлення аномалій включає пошук закономірностей та патернів у вхідних даних, які вказують на наявність аномалій. У випадку навчання з наглядом, модель навчається на основі маркованих даних, де відомо, які екземпляри є нормальними, а які є аномальними. Модель навчається визначати відмінності та виокремлювати аномалії в майбутніх невидимих даних. У випадку навчання без нагляду, модель навчається на основі безмаркованих даних і шукає незвичайні патерни або аномалії, які відрізняються від загального контексту даних. Цей підхід дозволяє виявляти аномалії, не маючи точних маркувань, але вимагає більш глибокого аналізу та експертної оцінки результатів.

Навчальний набір використовується для тренування моделі, валідаційний набір використовується для налаштування параметрів моделі та оцінки її продуктивності, а тестовий набір використовується для оцінки загальної продуктивності моделі на нових даних яких модель ще не бачила.

5) Оцінка моделі

Після того, як модель виявлення аномалій навчена, її результати потрібно оцінити, щоб визначити точність продуктивності моделі для майбутніх, нових входжень даних. Продуктивність моделі оцінюється, використовуючи метрики, такі як точність, влучність, повнота, F-міра та інші. Це допомагає зрозуміти, наскільки добре модель впоралася зі своєю задачею виявлення аномалій і чи можна розраховувати на її результати в майбутньому. Обробляючись оцінки якості моделі

можуть прийматись різні рішення зв’язані з побудової моделі і формату даних. Наприклад, зміна/корекція гіперпараметрів моделі або створення нових стовпців на основі вже існуючих, або навпаки видалення. Всі ці націлені на побудови точно і ефективної моделі. Для підбору гіперпараметрів можуть бути використати автоматизовані методи типу GridSearch або RandomSearch.

Під час аналізу і побудові моделей для наших даних ми зустрілись з певними проблемами в обробці наших даних, перш за все це було те, що дані містять категоріальний формат, а саме типи входу, процеси входу, підвищені привілеї та інші. Особливість полягає не тільки в їх наявності або кількість, а насамперед в тому, що наш набір цілком складається з виключно подібних ознак. Обробка категоріальних ознак не є тривіальною задачею, нижче буде наведено можливі способи обробки категоріальних даних для побудови моделей і аналізу.

3.2 Особливість використання категоріальних даних для машинного навчання

При побудові моделей, аналізі та інших процесах в Data Science необхідно брати до уваги тип і формат даних. Категоріальні дані — це тип даних, який використовується для групування інформації зі схожими характеристиками, тоді як числові дані — це тип даних, який виражає інформацію у формі чисел. Прикладом категоріальних даних може бути кольори, типи автомобілів, стать і тд. В нашому випадку, тип входів, використана станція, процес, час та інші параметри наших даних.

Особливість використання категоріальних даних для машинного навчання полягає у тому, що ці дані не можуть бути безпосередньо використані алгоритмами машинного навчання, які очікують числові значення. По перше не можливість оцінки значення «синій», «червоний» та інші для алгоритму.

Категоріальні дані в свою чергу можна розділити на два типи(рис. 3.1):

- Номінальний: без особливого порядку.
- Порядковий: існує певний порядок між значеннями, тобто між значеннями ми можемо виконувати порівняння, наприклад чудово> добре>погано.

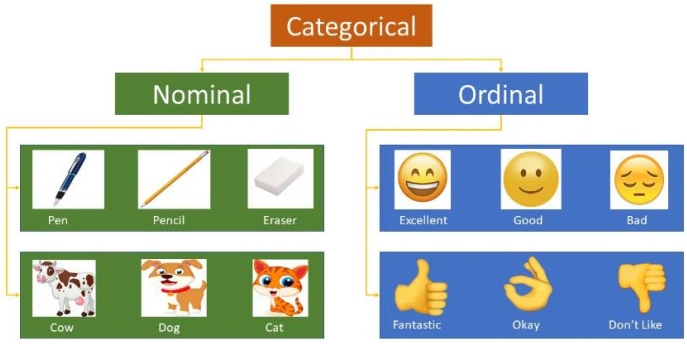

Рисунок 3.1 – Номінальні і порядкові дані

Існує дуже багато способів, якими ми можемо закодувати ці категоричні змінні як числа та використовувати їх в алгоритмі. Далі ми розповімо про деякі із них. Давайте розглянемо приклад для таблиці 3.1. Нехай нам треба закодувати таблицю для подальшого навчання моделі. Ознаку ресторан будемо вважати за індексну і не будемо використовувати в навчанні.

Таблиця 3.1 – Приклад набору даних

Ресторан	Кухня	Сер. Оцінка клієнтів	Середній чек	Рекомандація туристам(цільова змінна)
Діжонський смак(1)	Французька	Погано	25$	0
Мама Італія(2)	Італійська	Добре	23$	1
Балалайка(3)	Українська	Чудово	20$	1
Піца на Подолі(4)	Італійська	Жахливо	18$	0

3.2.1 Кодування категоріальних даних

One-hot Encoding

Отож нам необхідно спочатку закодувати ознаку(стовпець) Кухня. Через те, що ця ознака має номінальні категорії нам можна використовувати One-hot Encoding(OHE).У цьому методі кожна категорія відображається у векторі, який містить 1 і 0, що позначає наявність або відсутність функції. Кількість векторів залежить від кількості категорій для об’єктів.

Після застосування до стовпця кухня вона буде виглядати наступним чином:

Таблиця 3.2 - One-hot Encoding

Ресторан	Французька_кухня	Італійська кухня	Українська кухня
Діжонський смак(1)	1	0	0
Мама Італія(2)	0	1	0
Балалайка(3)	0	0	1
Піца на Подолі(4)	0	1	0

LabelEncoding

У цьому кодуванні кожній категорії призначається значення від 1 до N (де N - кількість категорій для ознаки). Одна з основних проблем з цим підходом полягає в тому, що між цими класами немає відношень або порядку, алгоритм може сприймати їх як певний порядок або відношення. У наведеному нижче прикладі для нашого набору це може виглядати так: (Погано < Добре < Чудово< Жахливо... 0 < 1 < 2 < 3), що скоріш за все не є бажаним результатом, хоча можливо для інший категорій або специфічних алгоритмів, може бути використано.

Ordinal encoding

Кодування з урахуванням порядку (Ordinal encoding) зберігає порядок (упорядковану природу) змінної у закодованому вигляді. Воно схоже на кодування

міток (label encoding), єдиний різниця полягає в тому, що для цього алгоритму ми маємо задати порядок категорій. Відповідно, для кодування середньої оцінки, нам необхідно використати даний алгоритм і очікуваний результат має бути(таблиця 3.3):

Таблиця 3.3 - Ordinal encoding

Ресторан	Сер. Оцінка клієнтів
Діжонський смак(1)	1
Мама Італія(2)	2
Балалайка(3)	3
Піца на Подолі(4)	0

Слід зазначити, що використання LabelEncoding та Ordinal encoding є недоцільним для прикладу коли ми кодуємо номінальні категорії, хоч і таке використання можливе, але можуть відбутись таке, що алгоритм буде сприймати Францію(2) > Італія(1), що буде просто недоцільно, адже така інтерпретація даних є помилковою та не має сенсу. Тим не менш, наведена трійка методів є основними для використання і найтиповішими, але іноді варто вийти за кордони типових і звичайних методів для забезпечення кращої роботи моделі. Нижче ми наведемо декілька інший варіантів екнодінгу, які використовуються рідкіше, але теж є гарним інструментом для цієї задачі.

Frequency Encoding (Кодування за частотою): У цьому методі кожна категорія замінюється її частотою в даних. Таким чином, категорії, які зустрічаються частіше, отримують більші значення, а категорії, які зустрічаються рідше, отримують менші значення. Для нашої таблиці італійська кухня отримає відповідне значення 0.5, а українська і французька по 0.25.

Target Encoding (Кодування за цільовою змінною): У цьому методі кожна категорія замінюється статистичною мірою цільової змінної в межах цієї категорії. Наприклад, для бінарної цільової змінної може використовуватися середнє

значення цільової змінної для кожної категорії. Цей метод може допомогти моделі усвідомити зв'язок між категорією і цільовою змінною.

Helmert Encoding: У цьому методі кожна категорія замінюється різницею між середнім значенням цільової змінної для цієї категорії і середнім значенням цільової змінної для всіх попередніх категорій. Цей підхід особливо корисний, коли існує лінійний тренд між категоріями і цільовою змінною.

Binary Encoding (Бінарне кодування): В даному кодуванні кожна категорія представлена у вигляді бінарного коду, де кожен біт відповідає наявності або відсутності певної властивості. Це зменшує кількість створених ознак і допомагає уникнути проблеми "перетину категорій".

Backward Difference Encoding (Кодування за різницею від заднього значення): У цьому методі кожна категорія замінюється різницею між середнім значенням цільової змінної для цієї категорії і середнім значенням цільової змінної для наступної категорії. Цей метод добре працює, коли категорії мають лінійний тренд.

Leave One Out Encoding (LOO Encoding): LOO Encoding є методом кодування категоріальних змінних, який використовує інформацію про цільову змінну для створення числового представлення категорії. У LOO Encoding для кожної категорії розраховується середнє значення цільової змінної, за винятком поточного спостереження. Це означає, що для кожного спостереження значення кодування буде залежати від інших спостережень з тією самою категорією. LOO Encoding може бути особливо корисним у випадках, коли категоріальна змінна сильно залежить від цільової змінної.

Hashing Encoding: Hashing Encoding є методом кодування категоріальних змінних, який використовує хеш-функції для створення числового представлення категорії. У цьому методі кожій категорії призначається числове значення, яке обчислюється шляхом застосування хеш-функції до назви категорії. Важливою особливістю Hashing Encoding є те, що воно використовує фіксований розмір кодування, незалежно від кількості унікальних значень категорії. Оскільки

розрахунок хеш-функції є необоротним процесом, втрати інформації можуть виникати, особливо якщо кількість унікальних значень велика.

Ці методи кодування надають різні способи представлення категоріальних змінних у числовому форматі, що допомагає моделям машинного навчання краще розуміти ці дані та зв'язок між категоріями та цільовою змінною. Важливо експериментувати з різними методами кодування та враховувати особливості даних та потреби моделі. Кращий метод кодування може залежати від конкретного завдання та властивостей даних.

Але бувають такі випадки коли використання методів кодування категорій в принципі не є доцільним, це можуть бути випадки коли такі дані становлять половину або навіть увесь набір даних, або коли певний стовпець має дуже багато різних значень, в такому випадку OHE розширить базу даних занадто широкого, через що буде занадто довгі обчислення. Саме для таких проблем існують два підходи рішення:

1)Оцінка схожості/відмінності. Цей підхід використовує спеціальні метрики, такі як метрика/дистанція Говера (gower distance), Жаккарда(Jaccard index), Хеммінга та інші. Для вимірювання схожості або відмінності між об'єктами чи множинами даних. Це дозволяє оцінити ступінь подібності між об'єктами та визначити їх відмінності. Ці метрики можуть бути корисними для порівняння категоріальних змінних чи інших типів даних, де важливо визначити ступінь схожості між об'єктами.

2)Методи зменшення розмірності. Такими методами є PCA (Principal Component Analysis) і t-SNE (t-Distributed Stochastic Neighbor Embedding). За допомогою них ми отримуємо інше відображення наших даних в н-вимірний простір, при цьому зберігаючи основні зв’язки. Також додамо, що для деяких методів зменшення розмірності варто використовувати оцінки які описані в пункті 1.

Нижче давайте більш детально розглянемо методи оцінки схожості та методи зменшення розмірності, адже саме за допомогою цих методів ми будували моделі машинного навчання в практичній частині.

3.2.2 Метрики схожості/відмінності

Евклідова відстань є метрикою, яка використовується для вимірювання відстані між двома точками у просторі. У випадку векторів ознак, евклідова відстань визначається як відстань між двома точками у n-вимірному просторі.

Формула для обчислення евклідової відстані між двома векторами a і b довжини n виглядає наступним чином:

$$euclidean = \sqrt[2]{(\mathrm{a}_1 - b_1)^2 + \cdots + (\mathrm{a}_n - b_n)^2} \quad (3.1)$$

Хоча евклідова відстань є популярною метрикою для числових даних, вона не підходить для категоріальних або змішаних даних. Це пояснюється тим, що евклідова відстань базується на числових розмірах та числових відношеннях між точками. Категоріальні дані, такі як мітки класів або категорії, не мають числових значень або відношень, тому не можуть бути вірно виміряні за допомогою евклідової відстані.

Треба зазначити, що відмова від евклідової відстані це основа при роботі з категоріальними даними, при роботі з алгоритмами які використовуються евклідову відстань необхідно її замінити на іншу метрику яка прийнятна для роботи з категоріальними або змішаними даними. Після використання цих методів ми отримуємо квадратну матрицю в взаємними відстанями між зразками. Давайте ж розглянемо деякі з них.

Відстань Говера(Gower’s Distance) можна використовувати, щоб визначити, наскільки різні два записи. Записи можуть містити комбінацію логічних, категоріальних, числових даних. Відстань визначається числом від 0 (ідентичний)

до 1 (максимально несхожий). Метрики, які використовуються для кожного типу даних, описані нижче:

- Числові (Неперервні) дані: нормалізована манхеттенська відстань
- порядковий: спочатку ранжується змінна, потім використовується манхеттенська відстань зі спеціальним коригуванням для зв'язків
- номінальний: змінні k категорій спочатку перетворюються в k двійкових стовпців, а потім використовується коефіцієнт Дайса

Відстань Говера обчислюється як середнє значення часткових відмінностей між індивідами. Загальний вигляд коефіцієнта такий:

$$Dgower=1-(\frac{1}{p}\sum_{j=1}^{p} sj(x1,x2)) \quad (3.2)$$

Формула відстані Говера з sj(x1,x2) як функцією часткової подібності, обчисленою окремо для кожного дескриптора.Формула для sj(x1,x2), коли дані є числовими:

$$sj(x1,x2) = 1 - \frac{|y1j-y2j|}{Rj} \quad (3.3)$$

Для якісних дескрипторів обчислюється відстань Dice. Щоразу, коли значення рівні, Dice Distance = 0(тобто відстань нульова), а коли вони не рівні ми обчислюємо наступним чином.

$$Dice = 2 * \frac{|X \cap Y|}{|X| + |Y|} \quad (3.4)$$

де X і Y множини, $|X|$-кількість елементів.

Отже ця метрика застосовується для порівняння схожості між об'єктами, які містять якісні та кількісні ознаки. Ця метрика підходить для змішаних типів даних, де ми можемо мати якісні (категоріальні) та кількісні (числові) ознаки.

Метрика Жаккара (іноді називається коефіцієнтом Жаккара) використовується для вимірювання подібності між двома множинами елементів, які можуть перетинатися. Ця відстань використовується для обчислення розбіжності між множинами шляхом порівняння кількості спільних елементів з загальною кількістю унікальних елементів у двох множинах. Метрика розраховується шляхом розподілу розміру перетину розмір об'єднання множин. Використовується для порівняння схожості між множинами або бінарними векторами. Ця метрика застосовується зазвичай до категоріальних даних, де важлива наявність або відсутність певних ознак. Формула для метрики Жаккара виглядає так:

$$J = \frac{|A \cap B|}{|A \cup B|} \tag{3.5}$$

де A і B - дві множини.

Метрика Хеммінга використовується для виміру відмінностей між двома послідовностями однакової довжини. Метрика Хеммінга розраховується шляхом підрахунку числа позицій, у яких елементи двох послідовностей розрізняються. Використовується для порівняння схожості між послідовностями бінарних символів. Ця метрика широко використовується в генетиці та телекомунікаціях, де важливість різниці в позиціях символів має велике значення. Формула для метрики Хеммінга виглядає так:

$$Hamming = \sum_{i=0}^{n} ai \neq bi \tag{3.6}$$

де A і B – дві послідовності однакової довжини, n – довжина послідовності.

Косинусна відстань (Cosine distance), також відома як косинусна міра схожості, є метрикою, яка використовується для вимірювання кутової розбіжності між двома векторами в n-вимірному просторі.

Косинусна відстань визначається за допомогою косинуса кута між двома векторами. Ця відстань знаходиться в діапазоні від 0 до 1, де 0 вказує на повну схожість (вектори однакові), а 1 вказує на повну відмінність (вектори ортогональні). Використовується для порівняння схожості між числовими векторами ознак. Ця метрика особливо ефективна для текстових даних та вимагає числових векторів для порівняння. Відстань можна описати як:

$$cosine\ distance = 1 - \frac{A * B}{\left(||A|| * ||B||\right)} \tag{3.7}$$

де A*B представляє скалярний добуток (проекцію) векторів A і B, а ||A|| та ||B|| представляють норми (довжини) векторів A і B.

3.3 Методи зменшення розмірності

В нашій роботі ми працюємо з набором даних який має дуже багато ознак(стовпців), які в свою чергу мають дуже багато різних категоріальних значень. Методи енкодингу навряд дадуть змогу отримати набір меншою ширини. Але як вже було сказано раніше, є варіант який дозволяє зменшувати розмірність наших даних.

Зменшення розмірності - це техніка, яка використовується для скорочення кількості ознак в наборі даних з одночасним збереженням якомога більше важливої інформації. Іншими словами, це процес перетворення високорозмірних даних в простір меншої розмірності, який все ще зберігає суть початкових даних. Це може бути зроблено з різних причин, таких як скорочення складності моделі, покращення продуктивності алгоритму навчання або полегшення візуалізації даних. Існує кілька методів зменшення розмірності, включаючи аналіз головних компонент (PCA), сингулярне розкладання (SVD) та лінійний дискримінантний аналіз (LDA) та інші. Кожний метод використовує різний підхід до проектування даних на простір меншої розмірності, зберігаючи важливу інформацію.

У машинному навчанні високорозмірні дані відносяться до даних з великою кількістю ознак або змінних. Проблема "прокляття розмірності" є поширеною у машинному навчанні, де продуктивність моделі погіршується зі зростанням кількості ознак. Це тому, що складність моделі зростає з кількістю ознак, і стає складніше знайти хороше рішення. Крім того, високорозмірні дані також можуть призводити до перенавчання, коли модель надто точно підлаштовується під тренувальні дані і погано узагальнюється на нові дані.

Зменшення розмірності допомагає подолати ці проблеми, зменшуючи складність моделі та поліпшуючи її узагальнюючу здатність. Існують два основних підходи до зменшення розмірності: відбір ознак(Feature Selection) і витягування ознак(Feature Extraction).

Feature Selection: Відбір ознак включає вибір підмножини початкових ознак, які найбільш відповідають задачі. Мета полягає в тому, щоб зменшити розмірність набору даних, зберігаючи найважливіші ознаки. Існують кілька методів відбору ознак, таких як фільтрування, засновані на моделі, і вбудовані методи. Фільтрування ранжирує ознаки за їх значимістю для цільової змінної, методи на основі моделі використовують показники продуктивності моделі як критерій для вибору ознак, а вбудовані методи поєднують відбір ознак з процесом тренування моделі.

Feature Extraction: Витягування ознак передбачає створення нових ознак шляхом комбінування або перетворення початкових ознак. Мета полягає в створенні набору ознак, який захоплює суть початкових даних в просторі меншої розмірності. Існують кілька методів витягування ознак, таких як аналіз головних компонент (PCA), лінійний дискримінантний аналіз (LDA) та вкладений метод навчання.

В свою чергу для пошуку аномалій ми будемо використовувати і розглядати детальніше Feature Extraction. Цей метод можна поділити на лінійні та нелінійні методи. Нелінійні методи відомі як методи вивчення множини (manifold learning).

Лінійні методи зменшення розмірності, такі як Principal Component Analysis (PCA), Factor Analysis (FA), Linear Discriminant Analysis (LDA) і Truncated Singular Value Decomposition (SVD), використовують лінійні перетворення для проекції початкових даних на простір низької розмірності. Ці методи шукають нові ознаки, які є лінійними комбінаціями початкових ознак.

Нелінійні методи зменшення розмірності, такі як Kernel PCA, t-distributed Stochastic Neighbor Embedding (t-SNE), Multidimensional Scaling (MDS) і Isometric mapping (Isomap), використовують нелінійні перетворення для проекції початкових даних на простір низької розмірності. Ці методи здатні виявляти складні залежності між ознаками та знаходити нелінійні структури у даних. Нелінійні методи зменшення розмірності можуть бути потужними для виявлення складних структур у даних, але вони можуть бути більш обчислювально витратними в порівнянні з лінійними методами. Далі, давайте детальніше розглянемо деякі з цих методів.

Principal Component Analysis (PCA) - це метод зменшення розмірності, який використовується для аналізу неперекладних числових змінних. Він допомагає зменшити кількість змінних у наборі даних, зберігаючи при цьому якомога більше інформації. PCA базується на лінійних алгоритмах та передбачає, що залежності між змінними можуть бути описані лінійними комбінаціями. Він дозволяє виявити головні фактори, які пояснюють найбільшу частину варіації в даних. PCA є потужним інструментом для виявлення складних взаємозв'язків і використовується для зменшення розмірності даних, візуалізації, побудови моделей та багатьох інших завдань аналізу даних.

Factor Analysis (FA) - це метод зменшення розмірності, який використовується для розуміння складної структури залежностей між змінними і виявлення латентних (неперекладних) факторів, які пояснюють змінність у наборі даних. FA базується на припущенні про існування латентних факторів, які не спостерігаються безпосередньо, але впливають на спостережувані змінні. Використовуючи FA, можна зменшити розмірність даних і отримати більш просту структуру, що дозволяє краще розуміти і аналізувати дані.

t-SNE (t-distributed Stochastic Neighbor Embedding) - це метод зменшення розмірності, який використовується для відображення складних даних в низькорозмірному просторі. Він особливо ефективний для виявлення складних структур і відношень між точками даних.

Основна ідея t-SNE полягає в перетворенні відстаней між точками відповідної вибірки на ймовірності. Відстані між близькими точками у вихідному просторі перетворюються на високі ймовірності, тоді як далекі точки мають низькі ймовірності. Це досягається шляхом використання функції Гаусса для обчислення ймовірностей.

Алгоритм t-SNE має два основних кроки:

- Обчислення схожості: На першому кроці вимірюється схожість між всіма парами точок вихідного простору. Це може бути виконано за допомогою різних метрик схожості, таких як Евклідова відстань або косинусна схожість.
- Відображення точок: На другому кроці t-SNE використовує оптимізаційний алгоритм для знаходження оптимального розташування точок в низькорозмірному просторі. Він спробує зберегти схожості між точками у вихідному і низькорозмірному просторах якомога краще.

Одна з важливих особливостей t-SNE полягає в тому, що він зберігає локальну структуру даних. Це означає, що близькі вихідні точки будуть наближені і в низькорозмірному просторі. Таким чином, t-SNE здатний виявляти складні шаблони, кластери і групи в даних.

Важливо зазначити, що t-SNE є стохастичним алгоритмом, тому при кожному запуску можуть отримуватися трохи різні результати. Додатково, при використанні t-SNE важливо правильно налаштувати параметри, такі як кількість ітерацій і розмірність цільового простору, для досягнення найкращих результатів.

t-SNE є потужним інструментом для візуалізації даних, особливо для виявлення складних залежностей і структур. Він часто використовується в

областях, таких як машинне навчання, обробка природної мови, геноміка та інші, де важливо розуміння взаємозв'язків між даними.

4 В цій дипломній роботі ми використовували різні методи зменшення розмірності, такі які UMAP, Multiple Correspondence Analysis (MCA) метод, що є розширенням класичного аналізу кореспонденції (Correspondence Analysis, CA), факторний аналіз, а також багато інших. Але враховуючи специфіку даних і вихідний результат, ми зрозуміли що найкращім рішенням для нашого набору даних є t-SNE.

3.4 Аналіз даних

Через те, що дані містять конфіденційну інформацію, ми не можемо навести детальний огляд даних в цьому дослідження, але для кращого розуміння давайте розглянемо розподіл даних, а також проаналізуємо ознаки які має набір. В роботі ми використовуємо дані з логів RDP-автентифікації сформовані в набір у вигляді таблиці. Цей набір містить інформацію про входи користувачів в систему, а саме логи входу в систему Windows з відповідним кодом 4624(S), який є подією яка описується як «В обліковий запис успішно ввійшли». Ця подія генерується, коли створюється сеанс входу (на машині призначення). Він генерується на комп'ютері, до якого було звернено доступ, де було створено сеанс. Ця подія має певний набір параметрів, які містять різну інформацію про вхід в систему. Серед яких є:

1. 'logonProcessName': Цей параметр вказує на ім'я процесу, який відповідає за автентифікацію користувача під час входу в систему. Він вказує на конкретний процес, який перевіряє введені облікові дані користувача і надає йому доступ до системи(такими процесами можуть бути NtLmSsp, Kerberos, User32 та інші).

2. 'elevatedToken': Цей параметр вказує, чи було вище підняте (elevated) право доступу для даного входу в систему. Він може мати значення True або False.

3. 'targetUserSid': Цей параметр містить ідентифікатор безпеки (SID) цільового користувача, тобто ідентифікатор, який унікально ідентифікує користувача в системі.

4. 'impersonationLevel': Цей параметр вказує на рівень підміни, який було використано під час входу в систему. Він може мати різні значення, такі як "Anonymous" (анонімний), "Identification" (ідентифікація), "Impersonation" (підміна) або "Delegation" (делегування).

5. 'workstationName': Цей параметр містить ім'я робочої станції (комп'ютера), з якого був здійснений вхід в систему. Він вказує на конкретний комп'ютер або пристрій, з якого користувач здійснив вхід.

6. 'timestamp': Цей параметр вказує на час (дату і час) здійснення входу в систему.

7. 'computer': Цей параметр містить ім'я комп'ютера або сервера, на якому було зареєстровано цей вхід.

8. 'processName': Цей параметр вказує на ім'я процесу, з якого було здійснено вхід в систему. Він вказує на конкретний процес, який був запущений користувачем або автоматично при вході в систему.

9. 'logonType': Цей параметр вказує на тип входу в систему. Він може мати різні значення, такі як 2 (вхід з інтерактивної робочої станції), 3 (мережевий вхід) або 5 (службовий вхід) та інші.

10. 'targetUserName': Цей параметр містить ім'я цільового користувача, тобто ім'я користувача, який здійснив вхід в систему.

Набір містить близько мільйона зареєстрованих входів для 850-ти унікальних користувачів. Серед користувачів, можна виділити три групи по кількості входів.

Перша група це 4 користувача, які мають 500-100 тисяч входів, ця група покриває майже 80% набору. Друга група 100-10 тисяч входів, яка налічує близько 15 користувачів, а також третя, до 10 тисяч з близько 25 користувачами. Також давайте розглянемо інші показники, наприклад SID. SID - це унікальний ідентифікатор безпеки, що використовується в операційних системах Windows для ідентифікації користувачів, груп і об'єктів безпеки. Кожен об'єкт в системі, такий як користувач, група, комп'ютер або ресурс, має свій власний SID. В нашому наборі представленні декілька видів цих значень, а саме:

- Користувацькі, зазвичай мають звичайну довжину і компоненти;
- Доменні, мають додатковий компонент який вказує домен, зазвичай це віддалені входи;
- Службові та анонімні SID, спеціальні типи SID, які використовуються в системах Windows для ідентифікації служб та анонімних користувачів.

logonType(тип входу) – це інформація, яка описує яким користувач увійшов в персональний запис. Існує декілька типів входів в систему, вони приймати різні значення відповідно до того як користувач увійшов в систему. Нижче ми коротко наведемо всі існуючі типи входів в автентифікації Windows.

2 - Інтерактивний (вхід за допомогою клавіатури та екрана системи).

3 - Мережевий (наприклад, підключення до загальної папки на цьому комп'ютері з іншого місця в мережі)

4 - Пакетний (наприклад, запланованe завдання)

5 - Службовий (запуск служби)

7 - Розблокування (наприклад, автоматично блокований робочий стіл з паролем)

8 - Мережевий без шифрування (вхід з обліковими даними, відправленими у відкритому тексті)

9 - Нові облікові дані

10 - Віддалено-інтерактивний (Terminal Services, Remote Desktop або Remote Assistance)

11 - Кешований інтерактивний (вхід з кешованими обліковими даними домену, наприклад, під час входу на ноутбук поза мережею)

3.5 Обробка та заміна значень в даних

Перш за все були оброблені коди в деяких стовпцях (ознаках), для того щоб їх можна було інтерпретувати. Наприклад, в стовпці elevatedToken коди Windows переводяться в формат 0 і 1. Також для стовпця impersonationLevel відповідно до документації коди відноситься до рівня підміни (impersonation) привілеїв, вони були переведені в класи Identification, Impersonation, Delegation.

Підміна привілеїв (impersonation) - це процес, коли процес або об'єкт виконує дії в імені та з привілеями іншого користувача або об'єкта. В операційних системах, які підтримують механізми безпеки, такі як Windows, цей процес може бути виконаний за допомогою підміни контексту безпеки.

1) 'Identification' (Ідентифікація): Цей рівень підміни вказує, що процес або об'єкт діє в своєму власному ідентифікаційному контексті і не підміняється під жодного іншого користувача або об'єкт. У цьому режимі процес виконує дії власного користувача без підміни або отримання доступу до ресурсів в контексті інших користувачів.

2) 'Impersonation' (Підміна): Цей рівень підміни вказує, що процес або об'єкт може підмінятися (перехоплювати) під іншим користувачем або об'єктом. У цьому режимі процес може виконувати дії в контексті іншого користувача з правами доступу, які надаються для цього користувача або об'єкта.

3) 'Delegation' (Делегування): Цей рівень підміни вказує, що процес або об'єкт може делегувати свої привілеї іншому об'єкту або сервісу. У цьому режимі процес може передати свої привілеї на інші об'єкти або сервіси, які виконують дії від його імені.

Окрім цього, був оброблений параметр processName, цей параметр містив повний файл ініціалізації входу (типу C:\\Windows\\.....\\SOMEFILE.exe) ми прибрали повний шлях і залишили лише кінцевий файл. Також для ряду входжень це значення було пропущеним, через те, що кількість таких значень було значимою частиною нашого набору, ми вирішили не видаляти і не замінювати такі входження іншими. Замість цього ми присвоїли таким рядкам значення NanVal.exe

Для обробки мітки часу був застосований метод отримання з них часу дня тижня. Це зумовлено тим, що ми маємо на меті вибудувати нормальну поведінку для кожного користувача і виділяти нетипові входження для нього. Отже, на виході кожен вхід в систему має параметри день тижня і година дня. На виході ми маємо 11 стовпців серед яких всі є категоріальними, а саме номінальними і порядковими.

3.5.1 Зменшення розмірності і візуалізація для набору даних

Перш за все, було прийняте рішення щодо застосування зменшення розмірності. Це дозволить нам візуалізувати наші дані. Візуалізація може допомогти виявити шаблони, тренди та аномалії в наборі даних. Також візуалізація може бути корисною для виявлення аномалій або випадків, які виділяються з нормального розподілу даних. Це дозволяє швидко виділити відхилення та незвичайні залежності.

Після побудови моделі машинного навчання, візуалізація може бути використана для представлення результатів. Наприклад, візуалізація кластерів може допомогти зрозуміти, як дані були розділені на групи, або графік розподілу може показати, як модель прогнозує значення в порівнянні з реальними даними. Це може бути корисним для оцінки якості моделі та розуміння її працездатності. Так само це може працювати для алгоритмів пошуку аномалій та класифікації.

Для виконання цієї задачі використовувалися декілька алгоритмів, а саме FAMD (Factor analysis of mixed data, Факторний аналіз змішаних даних), MCA (Multiple Correspondence Analysis, Аналіз множинної відповідності), а також t-SNE (t-distributed Stochastic Neighbour Embedding, T-розподілене вкладення стохастичної близькості). Ці алгоритми були застосовані до наших даних, але відображення, яке нас задовольняло, вдалося досягнути тільки за допомогою t-SNE. Відображення, яке було отримано з інших алгоритмів, було занадто сильно звужене і відображало всі дані в досить маленькому щільному кластері, що створювало проблему для оцінки роботи алгоритмів. При роботі з t-SNE ми отримали відображення даних в задовільному для нас форматі, де ми бачимо чіткі виділення у формі круглих кластерів, відображення відстані, а також деякі розсіяні точки, які відображають входження (рис. 3.2).

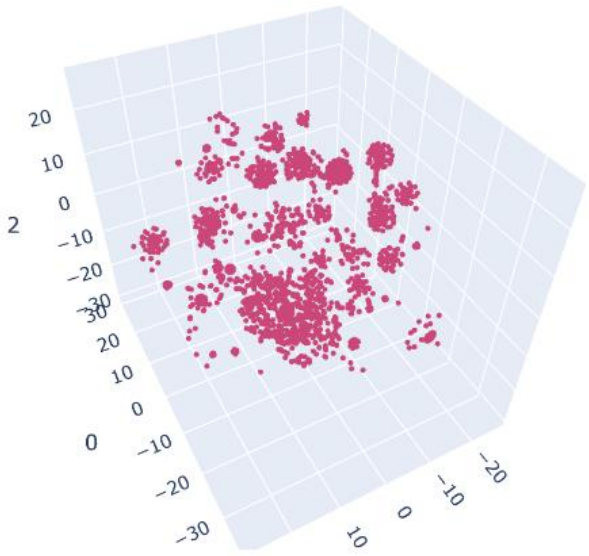

Рисунок 3.2 – візуалізація набору за допомогою t-SNE.

Для отримання даного відображення ми застосовували t-SNE в тривимірний простір. Для цього була використана зважена матриця Говера, де ми обрали певні ваги для всіх ознак, і після отримали квадратну матрицю, яка показує наскільки подібні зразки один одному. Для t-SNE, значення perplexity зазвичай вибирається в діапазоні від 5 до 50, і це відображає кількість сусідів, які враховуються при

розрахунку ймовірностей сусідства. Визначення цього значення є експериментальним. Після кількох спроб візуалізації ми вирішили присвоїти цьому параметру змінну 27, яка насправді є досить близькою до стандартного значення 30. Слід зазначити, що незалежно від того, як ми намагаємося зберегти інформацію при проектуванні в тривимірний простір, все ж відбувається втрата інформації при зменшенні розмірності. Це відображення слугує скоріше додатковим засобом оцінки, ніж основним.

Оскільки ми не маємо розмічених даних, ми не можемо використовувати класифікаційні моделі з розділу навчання з учителем, такі як логістична регресія, SVM, бустинги та інші методи навчання з учителем. Це значно обмежує кількість можливих підходів до пошуку аномалій, тому ми можемо звернутися до пошуку аномалій без учителя, а також статистичним методам. Однак, ми маємо досить специфічний набір даних у власному форматі, тому нам потрібно дуже уважно розглядати кожен алгоритм, який ми застосовуємо до набору даних.

Дуже важливим є зазначити те, що ми маємо нерозмічені дані, і не можемо оцінювати модель за допомогою стандартних методів оцінки моделей машинного навчання, таких як точність, повнота, F1 та інші метрики. Виходячи з цього, ми насправді маємо дещо іншу задачу. Нам необхідно знайти якомога більше аномальних зразків, які в свою чергу є ідентифікатором ризику та можуть бути ідентифікатором загрози.

3.6 Моделі машинного навчання

3.6.1 DBSCAN

При побудові моделей була використана підвибірка даних в 10000 екземплярів, це зумовлено обмеженнями обчислювальної потужності. Завдяки тому, що наш набір відсортований по даті і часу, ми візьмемо найперші 10000 входів одного проміжку часу.

Отож давайте розглянемо побудову моделі за допомогою алгоритму DBSCAN. Як вже було сказано алгоритм об’єднує точки в кластери, якщо в деякому околі для певної точки існує н сусідів. Саме тому, перш за все, давайте розглянемо параметри нашого алгоритму. Алгоритм використовує два основним параметра: epsilon(eps, окіл точки) та min_samples(мінімальна кількість зразків). Також важливою для нас є параметр metric, який за замовчуванням має Евклідову відстань. Давайте більш детально поговоримо про параметри.

Почнемо з метрики, фактично, в цьому алгоритмі метрика необхідна для визначення дистанції між точками і розуміння, чи є вони сусідами. Як вже було описано в попередніх розділах, ми не можемо працювати з евклідовою відстанню в нашому дослідженні. Саме тому ми використовуємо метрику Говера, визначаючи відмінність між зразками як відстань. Також ми масштабуємо матрицю Говера, де для кожного елемента максимальна відстань визначена як 1, а мінімальна - як 0.

Epsilon (eps): Це параметр, який визначає радіус околу навколо кожної точки. Всі точки, що знаходяться на відстані eps або менше від даної точки, вважаються сусідами цієї точки.

Min_samples: Це параметр, який визначає мінімальну кількість сусідів, необхідну для того, щоб точка була визнана як ядром (core point). Ядро - це точка, яка має щонайменше min_samples сусідів всередині свого околу.

Саме за допомогою цих параметрів точки об’єднуються в кластери. Цікаво, що змінюючи і підбираючи точки ми можемо отримати бажане розбиття на кластери. Наприклад якщо ми виставимо великий eps, то ймовірно отримаємо 2-3 кластери, які будуть покривати весь набір даних. В той же час, при високому параметрів Min_samples елементам важче об’єднуватись в кластери, оскільки це вимагає більшої кількості елементів в околі. Відповідним наслідком є або велика кількість малих кластерів, або велика кількість аномалій. Тому необхідно обирати ці параметри залежно від задачі, формату і кількості даних.

Далі ми обрали радіус окола 0.27 та кількість елементів для об’єднання в кластер – 25. Як вже було сказано, для тренування моделі використовуємо масштабовану матрицю Говера з відповідним параметром для алгоритму metric='precomputed'. Нижче подивимось на результати кластеризації, для візуалізації використовуємо t-SNE (рис 3.3).

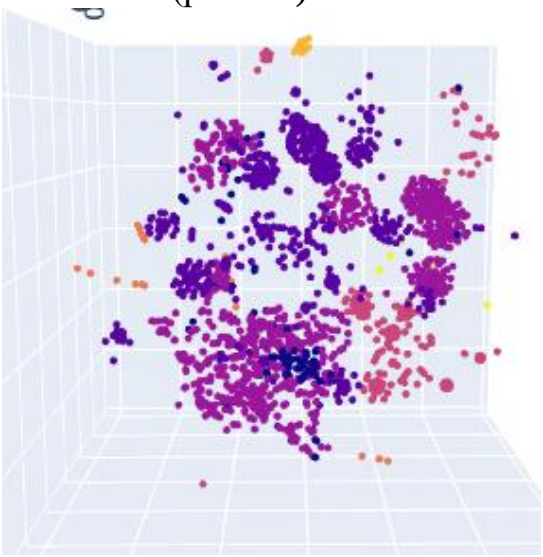

Рисунок 3.3 – Результат роботи алгоритму DBSCAN

Отож, ми бачимо, що кластеризація виділила 6 різних кластерів, серед яких є 3 кластери розмірів 5000 (темно-фіолетові, кластер 0), 3000 (фіолетові, кластер 1) та 900 (персикові, нижній правий кут, кластер 2). Також ми маємо аномальні зразки (темно-сині точки, позначені як -1), а також кластери з низькою чисельністю: жовтогарячі (кластер 4), жовті (кластер 5) та помаранчеві (кластер 3). Здається, що саме кластери 3-5 можуть бути потенційними аномаліями на рівні з точками, які ми позначено як -1.

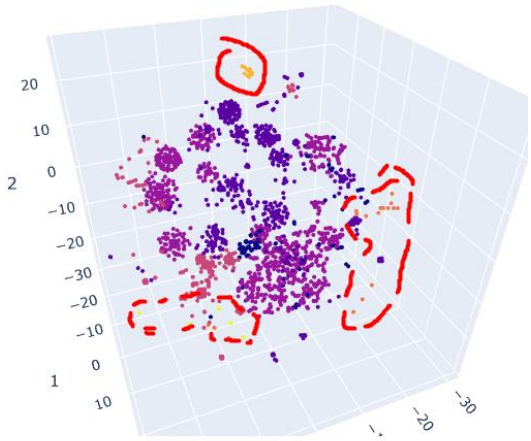

Рисунок 3.4 – Малі кластери

При більш детальному розгляді кластерів 3 і 5, ми помітили, що це входи двох конкретних користувачів, які насправді просто відрізняються від інших типових входів для цієї системи. Тим не менш, немає жодних ознак аномальності у цих входах, вони утворюють окремий патерн поведінки для себе. З іншого боку, кластер 4 є підмножиною входів для певного користувача. При аналізі цієї групи ми помітили, що входи були ініційовані нетиповими процесами для даного користувача, і також мали інші привілеї та параметри входу. Теоретично, кількість цих входів може бути достатньою, щоб вважати це патерном, але ми, як дослідники, маємо звернути на це особливу увагу, оскільки такі входи можуть вказувати на атаки на систему. Тому ми позначили їх як аномальні.

Давайте тепер детальніше розглянемо інші аномальні приклади. Під час аналізу цих прикладів ми помітили входи користувачів, які зустрічаються всього кілька разів у всьому наборі. Їх вважають аномаліями через те, що їхня кількість не дозволяє сформувати повноцінний кластер, що відображає їх поведінку. Алгоритм також позначав як аномалії елементи, які є підмножиною входів користувачів з нетиповою поведінкою, тобто більш малих структур даних, подібних до кластеру 4.

Підсумовуючи, якщо позначити всі точки, які ми назвали аномальними (точки з позначеннями -1 та кластер 4), то отримаємо наступний результат (див. рисунок 3.5). Таким чином, у нас є 10 000 випадків, серед яких 278 визначено як аномальні, а 9722 як нормальні. Однак ми вважаємо, що алгоритм не виділив достатню кількість окремих точок у просторі або малих структур даних. Саме з цього огляду ми можемо оцінити роботу алгоритму як задовільну.

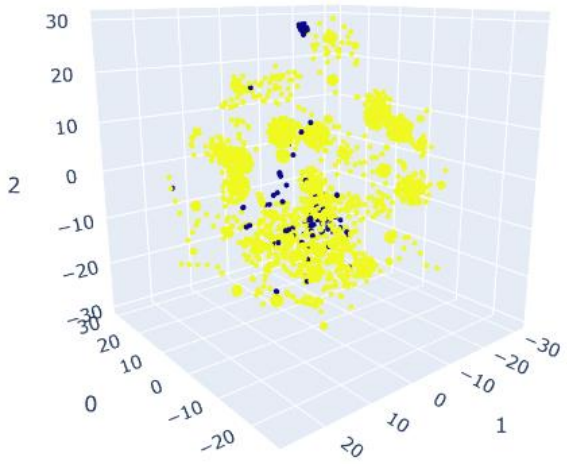

Рисунок 3.5 – Розбиття нормальних/аномальний на основі DBSCAN (сині точки відображають аномальні зразки)

3.6.2 LOF

Далі розглянемо досить потужний алгоритм для пошуку аномалій під назвою Local Outlier Factor. Ми вже описували, як працює цей алгоритм у теоретичному блоку, тому не будемо детально розбирати роботу алгоритму. Тим не менш, основна ідея полягає в побудові оцінки ступеня відхилення кожної точки від її сусідів, що дозволяє виявляти аномальні об'єкти, які мають значно нижчу щільність, ніж їх сусіди. Як і для попереднього алгоритму, давайте розглянемо детальніше параметри Local Outlier Factor. Цей алгоритм має основний параметр - кількість сусідів, що визначає кількість найближчих сусідів, які використовуються для обчислення локальної щільності кожної точки. Зазвичай значення k вибирається залежно від розміру даних та характеру набору даних. LOF, як і DBSCAN, використовує параметр метрики для визначення відстані між точок. Аналогічно до попереднього пункту, ми використовуємо масштабовану матрицю Говера. Також LOF має досить цікавий параметр contamination (англ. – забруднення), параметр впливає на поріг LOF алгоритму. Всі точки, які мають значення LOF більше порогу, вважаються аномальними. Тобто, чим більше значення "contamination", тим більше точок буде віднесено до категорії аномальних. Наприклад значення 0.1 означає, що алгоритм визначає аномальними 10% входжень з найбільшою оцінкою аномальності.

Значення "auto" для параметра "contamination" вказує на автоматичне визначення очікуваної частки аномалій у наборі даних. Алгоритм самостійно оцінює це значення на основі характеристик набору даних, зокрема щільності точок та розподілу даних. Зазвичай це виконується шляхом порівняння LOF значень з певними квантилями розподілу.

Отож, для прикладу, ми розглянемо дві моделі, які матимуть спільний параметр сусідів (40) та прийматимуть матрицю Говера на вхід. Однак, одна модель буде приймати чітку межу аномальності на вхід, а інша автоматично визначатиме кількість аномальних значень. Зазвичай коли не відома точна кількість аномалій у наборі даних кількість таких зразків визначаються як 1%, але з урахуванням аналізу даних ми припустимо, що маємо 5-7% аномальних значень.

Алгоритм з автоматичним визначенням границі аномалій виділив у наборі даних 1263 аномальних входжень, що становить 12.6% від всього набору даних. Візуальний результат можна побачити на рисунку 3.6. Ми бачимо, що аномальними точками є окремі точки поза кластерами, а також точки, які знаходяться в менш щільному середовищі, наприклад, в нижній частині рисунку 3.6.

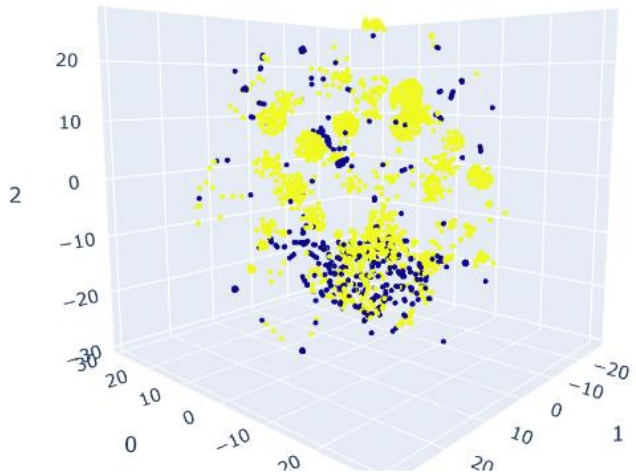

Рисунок 3.6 – Результат виділення аномалій алгоритму LOF з автоматичним параметром забруднення

Тим не менш, нам здається, що нижня частина графіку має занадто багато аномальних значень, які насправді можуть не бути ними. Тому давайте спробуємо власноруч визначити оптимальний параметр забрудненості. Згідно з нашими припущеннями, візьмемо значення з проміжку 5-7%. Після навчання, на рисунку 3.7 ми можемо бачити, що алгоритм видає більш збалансований результат. Ми бачимо, що багато точок, які знаходили в помірно щільному середовищі перестали позначатись як аномалії. Це означає, що значення їх оцінки аномальності насправді не було високим і відпадає, коли ми зменшили значення порогу аномальності. Однак, важливо зазначити, що далеко не всі віддалені точки є аномаліями. Це може бути через те, що відображення не є досконалим, але також можливо це вказує на необхідність зміни параметрів моделі.

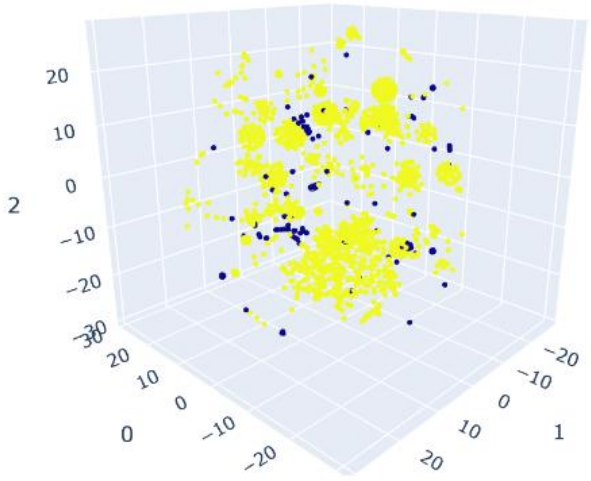

Рисунок 3.7 – Результат виділення аномалій алгоритму LOF з параметром забруднення 6%

При збільшенні кількості сусідів не відбулося суттєвих змін, в той же час зменшення цього параметра зовсім не є доцільним. Адже ми маємо досить великий набір даних, в якому зразки мають дуже різні значення. Аномальні зразки можуть не виявлятись через те, що вони порівнюються лише з такими самими аномаліями.

Давайте розглянемо аномальні зразки детальніше. При уважному аналізі цих зразків ми помітили, що деякі входи користувачів мають менше ніж 10 входів і вважаються аномальними через те, що їхня кількість ще недостатня для

формування достатньо великої структури даних. Також ми помітили, що входи користувачів А, Б і В широко представлені у наборі даних. Аномалії становлять певну частину входів для цих користувачів.

Наприклад, розглянемо користувача А. Він має 144 аномальних входи, що становить 3% від загальної кількості входів. Аналіз цих зразків показав, що вони здійснювалися в нетипові дні та час для даного користувача. Також були використані інші файли для ініціалізації входу, включаючи файли, які були використані лише 1-2 рази. В цілому, це може бути окремим патерном поведінки, але на фоні загальної структури даних він є досить малим і не має унікальних ознак. Цей патерн може розвиватися з часом і стати повноцінним, або ж вказувати на потенційну атаку.

У випадку інших користувачів, у яких вже сформувалися патерни, ми помітили входи, що здійснювалися з нових робочих станцій, окремі входи за допомогою інших файлів ініціалізації, а також нетиповий час. Це також може бути підозрілим і мати ознаки атаки. Незважаючи на те, що алгоритм не є ідеальним, він добре виявляє віддалені одиночні входи. Загальною оцінкою роботи алгоритму є добре, враховуючи всі зазначені фактори.

3.6.3 Isolation Forest

Через те, що алгоритм Isolation Forest не може працювати з категоріальними даними, нам необхідно вдатись до якоїсь іншої обробки даних. Ми вже згадували про розширений алгоритм Ізоляційного лісу, але його ідея полягала в використанні саме порядкових категоріальних змінних, що в нашому варіанті є не зовсім доцільним, оскільки ми маємо як порядкові, так і номінальні. Саме тому ми вирішили спробувати перевести наші входи в вектори довжини 3 і застосувати до них стандартний алгоритм IF. Відображення в 3-х вимірний простір відбувалось за допомогою того самого алгоритму t-SNE. Отримавши наші дані в 3-х вимірному форматі ми застосовуємо до них алгоритм Ізоляційного лісу. Отож, давайте розглянемо параметри алгоритму.

n_estimators - це кількість оцінювачів в ансамблі. Цей параметр визначає кількість дерев, що будуть побудовані в ізоляційному лісі. Більше значення дозволяє отримати більш точні результати, але при цьому збільшується час обчислення. Зазвичай рекомендовані значення для цього параметра рухаються в діапазоні від 50 до 100.

contamination: Цей параметр вказує очікувану частку аномалій у наборі даних. Він допомагає визначити поріг для визнання точок як аномальних. Цей параметр є таким самий як і параметр contamination в попередній моделі.

Також алгоритм має параметри max_features і max_samples. Зазвичай це значення дорівнює 1.0 і варіюється від 0 до 1.0. Ці параметри визначаються максимальну кількість ознак та кількість вибірок при побудові кожного дерева в лісі. Зазвичай ці параметри зменшуються для підвищення швидкості алгоритму, за ціну точності. Тобто фактично існують для знаходження компромісу між продуктивністю і точністю виявлення аномалій. В побудові моделі ми залишимо стандартні значення 1.0.

Аналогічно до попереднього алгоритму ми побудуємо дві моделі. Першу з автоматичним виявленням кількості аномалій, а другу з виставленим нами на тій самі межу 6%. Кількість дерев було експериментальним методом обрано значення 100.

При автоматичному виявленні межі аномальних зразків алгоритм визначив 2603 зразків як аномальні, що фактично означає 26% забруднення даних. На рисунку 3.8 ми можемо бачити, що до аномальних віднесений великий кластер даних, який фактично є окремою моделлю поведінки групи користувачів, через його розмір і подібність даних в ньому, ми вважаємо що він явно не є аномальним, а тому автоматичне виявлення межі не зовсім доцільним. Окрім цього дуже багато розсіяних точок класифіковані як аномалії, що є позитивним в роботі цього алгоритму.

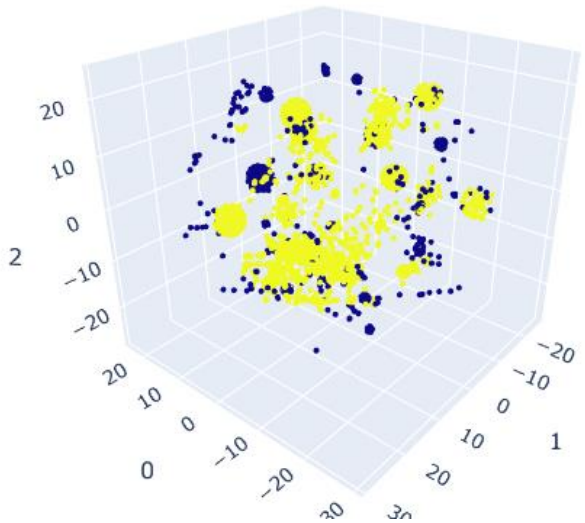

Рисунок 3.8 – Результат роботи алгоритму IF з автоматичним виявленням параметру забруднення

Далі давайте подивимось які результати ми маємо при визначенні цього порогу власноруч, а саме 6%. Модель виділяє 596 аномальних значень, на рисунку 3.9 ми можемо побачити, що аномальними є деякі окремі точки в просторі, а також невеликі кластери в просторі. Тим не менш, візуально алгоритм виділяє досить скупчені кластери, які насправді можуть не бути аномальними, а також досить мало окремих точок в просторі.

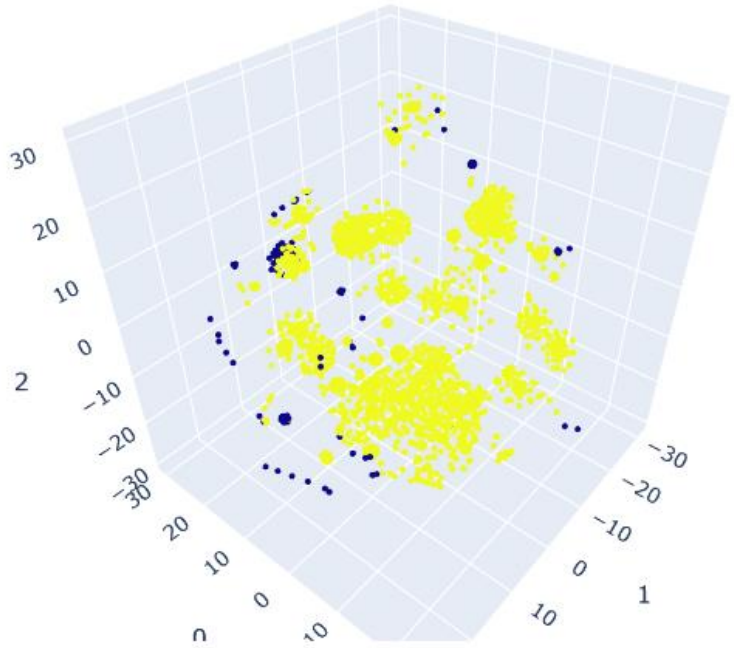

Рисунок 3.9 – Результат роботи алгоритму IF з параметром забруднення 6%

При аналізі ми виявили, що алгоритм IF так само, як і LOF, виділив схожий піднабір входів в систему для користувача A. Проте, він не виділив окремі входження з певним файлом ініціалізації. Замість цього ми отримали підмножини в середніх кластерах, які мали нетипові входи у часі. Це також може бути корисним для виявлення аномалій. Аналогічні результати були отримані для інших користувачів.

Крім того, IF відмітив кластери 3 і 5 як аномальні, які ми перевіряли під час аналізу роботи DBSCAN. Як вже зазначалося, ці входи не є аномальними, але представляють окремий патерн поведінки для певних користувачів. В той же час, візуально ми бачимо, що алгоритм виділяє частину.

3.7 Аналіз результатів моделей

Перш за все, хочу зазначити, що оцінка моделей проводилась за допомогою візуального аналізу розподілу даних, а також аналізу самих даних. Під час цього аналізу ми виявили, що алгоритм LOF показав досить гарні результати. Він не виділяв середньо-малі кластери як аномалії. Однак, було помічено багато виділених аномалій серед точок, які знаходилися в менш щільних регіонах, а також

серед точок, які створювали маленькі структури або знаходились поза основною структурою даних. Загалом, алгоритм не виділив всі ізольовані точки в просторі, але його роботу можна оцінити як добре виконану.

Щодо DBSCAN, ми отримали досить непоганий обсяг даних, включаючи аналіз тих самих середньо-малих структур даних, які алгоритм IF виділяв як аномалії. За допомогою DBSCAN ми змогли об'єднати їх в один кластер і встановити, що ці дані не є аномальними. Крім того, алгоритм непогано впорався з областями помірної щільності. Однак він часто об'єднував малі структури або окремі точки з більшими кластерами, що робило оцінку аномальності важкою. Загалом, алгоритм показав себе непогано, але менш ефективним у порівнянні з LOF. Тим не менш, ми оцінюємо його роботу як задовільну.

Використання алгоритму IF з використанням t-SNE було експериментальним рішенням, оскільки ми розуміємо, що зменшення розмірності набору даних призводить до втрати частини інформації. Однак, ми вирішили спробувати альтернативний підхід. Алгоритм ізольованого лісу непогано виділяв аномалії, які представляли нетипові входи або ще не сформовану поведінку деяких користувачів. Він також виділяв підмножину входів певного користувача, що відбувалась в нетиповий час. Однак загальний результат не був дуже хорошим, оскільки багато окремих точок в просторі не були виділені як аномалії. Ймовірно, це сталося через те, що алгоритм більше уваги приділяв глобальним відхиленням, ніж локальним. Загалом, нам вдалося виділити декілька цікавих структур даних за допомогою цього алгоритму, тому ми оцінюємо роботу цього алгоритму як задовільну.

Після виділення потенційно аномальних зразків, ми створили набір даних, який містить ці зразки, відібрані нашими алгоритмами як аномальні. Подальші кроки включали передачу цього набору даних компанії, яка надала нам дані для пошуку аномалій. Ми припускали, що ці зразки входів в систему можуть мати ознаки атак на систему. Проте, ми не можемо однозначно стверджувати чи вказувати на наявність ідентифікаторів компрометації, оскільки для цього потрібно

значно більше інформацію про систему. Але ми вважаємо, що такі логи аномальної поведінки можуть містити ідентифікатори ризиків, а часом навіть атак.

Після аналізу результатів нашого дослідження, компанія повідомила, що серед зразків, виявлених нашими алгоритмами як аномальні, є ідентифікатори загрози та атаки, але не було знайдено ознак компрометації в системі.

Висновки до розділу 3

В третьому розділі ми розглянули основні етапи побудови моделі машинного навчання серед яких аналіз і обробка даних, далі вибір алгоритму машинного навчання, навчання моделі та оцінки результатів моделі. Через те, що ми маємо досить специфічний набір в якому всі показники є категоріальним, ми розкрили в дипломі проблему категоріальних даних в машинному навчанні. Було наведено декілька методів обробки категоріальних даних, а саме кодування та метрики схожості. Також було розглянуті методи зменшення розмірності, їх можна використовувати для візуалізації результатів, або для використання в навчанні моделі.

Після цього був проведений аналіз даних нашого набору, описані основні ознаки для нашого набору. Відбулась обробка та заміна деяких значень, після чого застосувався алгоритм t-SNE. Цей алгоритм дав нам змогу отримати наші дані в вигляді векторів довжини 3, за допомогою нього ми будемо здійснювати візуалізацію та навчати модель в одному з алгоритмів. Наступним етапом було навчання моделей, через те, що ми не знаємо заздалегідь які зразки є аномальним ми вимушені використовувати лише методи навчання без вчителя. Окрім цього, ми не можемо оцінювати моделі за допомогою стандартних методів, таких як точність, повнота, влучність та інші. Саме тому, ми аналітично будемо намагатись отримати найоптимальніший розподіл по аномаліям, після чого відправимо на аналіз в компанію на пошук індикаторів атак.

Для побудови моделей машинного навчання було обрано 3 моделі, а саме DBSCAN, LOF, та комбінацію t-SNE та Isolation Forest. Ми провели підбір параметрів експериментальним способом, після чого навчали моделі і оцінювали їх результати за допомогою візуалізації, а також аналізу зразків які алгоритми позначили як аномальні. Після, ми сформували набір даних і відправили компанії для пошуку індикаторів атак або ризику. В свою чергу компанія підтвердила, що такі логи містять індикатор ризику, а також було знайдемо декілька індикаторів атак, про індикатори атак не повідомлялось.

ВИСНОВКИ

В дипломній роботі ми проводили дослідження та застосовували різні методи машинного навчання для виявлення аномалій у логах входів RDP автентифікації. Основною метою роботи було з'ясувати, наскільки ефективними можуть бути методи машинного навчання для виявлення потенційних атак та ризиків безпеки у системі. У роботі були розглянуті основні аспекти протоколу RDP, його потенційні загрози та можливі наслідки несанкціонованого доступу. Після аналізу цих аспектів було визначено, що важливим етапом є виявлення аномальної поведінки у логах входів RDP автентифікації, яка можуть свідчити про атаки або ризики для системи.

У дослідженні було детально розглянуто різні методи машинного навчання, такі як навчання з учителем, без учителя, напівнаглядовий метод, а також статистичні методи. Для навчання з учителем були розглянуті різні алгоритми, включаючи логістичну регресію, дерева рішень, випадковий ліс, бустинг та метод опорних векторів. Для методів без учителя були розглянуті кластеризація, методи на основі щільності та ізоляційний ліс.

Побудовано та навчано кілька моделей машинного навчання, а саме DBSCAN, LOF та комбінацію t-SNE та Isolation Forest. Ці моделі були використані для виявлення аномальних входів у логах RDP автентифікації. Результати оцінювалися за допомогою візуалізації та аналізу зразків, визначених як аномальні. Загальний аналіз результатів дослідження підтвердив потенціал методів машинного навчання для виявлення аномалій у логах входів RDP автентифікації. Окрім того, компанія замовник підтвердила виявлення індикаторів ризику та декілька індикаторів атак в логах які ми ідентифікували як аномальні, що свідчить про можливі загрози безпеці системи.

Результати дослідження підтверджують, що використання методів машинного навчання дозволяє виявляти потенційні атаки та ідентифікувати ризики безпеки у системах RDP автентифікації. Також, результати дослідження можуть мати практичне застосування в сфері кібербезпеки, допомагаючи організаціям

вдосконалювати свої системи безпеки, розробляти ефективні заходи протидії атакам та зменшувати ризики. Враховуючи постійно зростаючу кількість кіберзагроз у сучасному світі, результати цієї дипломної роботи можуть мати вагому значимість для покращення кібербезпеки як на рівні окремих організацій, так і на загальному рівні мережі Інтернет.

Отже, дипломна робота підтверджує значущість та ефективність використання методів машинного навчання для виявлення потенційних атак та ризиків безпеки у системах RDP автентифікації. Результати дослідження можуть бути використані як основа для подальшого розвитку та вдосконалення систем безпеки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Forecasting Suspicious Account Activity at Large-Scale Online Service Providers by Hassan Halawa, Matei Ripeanu and Konstantin Beznosov, University of British Columbia Vancouver, British Columbia, Canada 2018 Режим доступу до ресурсу: https://arxiv.org/pdf/1801.08629v1.pdf
2. Anomaly Detection in Login Data Bachelor's thesis as part of the IT security course at the University of Lübeck, submitted by Johannes Liebenow 2019 Режим доступу до ресурсу: https://dl.acm.org/doi/pdf/10.1145/1269880.1269886
3. Detecting Anomalous User Behavior Using an Extended Isolation Forest Algorithm: An Enterprise Case Study by Li Sun, Steven Versteeg, Serdar Boztas1 and Asha Rao School of Mathematical and Geospatial Sciences, RMIT University, Melbourne, Australia 2016 Режим доступу до ресурсу: https://arxiv.org/pdf/1609.06676.pdf
4. Authentication Anomaly Detection: A Case Study On AVirtual Private Network Michael J. Chapple, Nitesh Chawla and Aaron Striegel Department of Computer Science and Engineering, University of Notre Dame, IN, USA 2007 Режим доступу до ресурсу: https://dl.acm.org/doi/pdf/10.1145/1269880.1269886
5. A Machine Learning Approach for RDP-based Lateral Movement Detection by Zhenyu Bai A thesis presented to the University of Waterloo in fulfillment of the thesis requirement for the degree of Master of Mathematics in Computer Science Waterloo, Ontario, Canada, 2019
6. Remote desktop protocol (RDP) by Wesley Chai [Електронний ресурс] – Режим доступу до ресурсу: https://www.techtarget.com/searchenterprisedesktop/definition/Remote-Desktop-Protocol-RDP
7. HOW THE LOGISTIC REGRESSION MODEL WORKS [Електронний ресурс] Saimadhu Polamuri 2017 – Режим доступу до ресурсу https://dataaspirant.com/how-logistic-regression-model-works/
8. Decision Tree GeeksforGeeks 2023 [Електронний ресурс] – Режим доступу до ресурсу https://www.geeksforgeeks.org/decision-tree/

9. Boosting in Machine Learning by raman_257 GeeksforGeeks 2023 [Електронний ресурс] – Режим доступу до ресурсу https://www.geeksforgeeks.org/boosting-in-machine-learning-boosting-and-adaboost/
10. What are the pros and cons of using clustering vs. density-based methods for anomaly detection? Limkedin [Електронний ресурс] – Режим доступу до ресурсу https://www.linkedin.com/advice/3/what-pros-cons-using-clustering-vs-density-based-methods
11. Local Outlier Factor (LOF) — Algorithm for outlier identification Vaibhav Jayaswal Published in Towards Data Science 2020 Режим доступу до ресурсу https://towardsdatascience.com/local-outlier-factor-lof-algorithm-for-outlier-identification-8efb887d9843
12. How to develop an anomaly detection system by Olena Domanska 2021 Режим доступу до ресурсу https://www.avenga.com/magazine/anomaly-detection
13. Gower's Distance by Divyanshu Anand Published in Analytics Vidhya 2020 Режим доступу до ресурсу https://medium.com/analytics-vidhya/gowers-distance-899f9c4bd553
14. 4624(S): An account was successfully logged on. Microsoft documentation 2023 Режим доступу до ресурсу https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4624
15. Introduction to Dimensionality Reduction by GeeksforGeeks 2023 Режим доступу до ресурсу: https://www.geeksforgeeks.org/dimensionality-reduction/

ДОДАТОК А – КОД ПРОГРАМИ

```
import pandas as pd
import warnings
from scipy.spatial.distance import pdist, squareform
from sklearn.manifold import TSNE
import plotly.express as px
import gower
import pandas as pd
from scipy.spatial.distance import pdist, squareform
from prince import FAMD,MCA
from sklearn.cluster import KMeans
import numpy as np
warnings.filterwarnings('ignore')

log=pd.read_csv('logons.csv')

# elevatedToken
log['elevatedToken']=log['elevatedToken'].replace('%%1843',0)
log['elevatedToken']=log['elevatedToken'].replace('%%1842',1)
# impersonationLevel
log['impersonationLevel']=log['impersonationLevel'].replace('%%1832','Identification')
log['impersonationLevel']=log['impersonationLevel'].replace('%%1833','Impersonation')
log['impersonationLevel']=log['impersonationLevel'].replace('%%1840','Delegation')

log['targetUserName'] = log['targetUserName'].str.lower()

log['processName']=log['processName'].fillna("NanVal.exe")
```

```
import os
def get_filename(path_str):
    path = os.path.normpath(path_str)  # normalize path string
    return os.path.basename(path)

# apply the function to the "path" column and create a new "filename" column
log['processFilename'] = log['processName'].apply(get_filename)

log['timestamp'] = pd.to_datetime(log['timestamp'])

from datetime import datetime
date_format = "%Y-%m-%d %H:%M:%S.%f%z"
log['timestamp'] = pd.to_datetime(log['timestamp'], format='%b %d, %Y @ %H:%M:%S.%f')
log['dayoweek'] = log['timestamp'].dt.weekday+1
log['hour'] = log['timestamp'].dt.strftime('%H').astype(int)

log1=log.copy()
log1=log1.drop(columns=['timestamp','processName'])
sample1=log1.sample(frac=0.005)
print('amount of samples',len(sample1))

sample2=log1[:10000]

import seaborn as sns
import matplotlib.pyplot as plt
fig, axs = plt.subplots(ncols=2)
sns.countplot(data=log1,x=log1.hour, ax=axs[0])
sns.countplot(data=log1,x=log1.dayoweek,ax=axs[1])
```

```
grouped_data = log1.groupby(['dayoweek', 'hour']).size().reset_index(name='count')

sns.set_style("whitegrid")
plt.figure(figsize=(7,9))
sns.countplot(x="dayoweek", hue="hour", data=log1)
plt.title('Frequency of logins by hour and day of week')
plt.xlabel('Day of week')
plt.ylabel('Number of logins')
plt.show()

import matplotlib.pyplot as plt
fig, axs = plt.subplots(ncols=2)

sns.countplot(data=log, y=log.processFilename, ax=axs[0])
sns.countplot(data=log,y=log.logonProcessName,ax=axs[1])

plt.show()

fig, axs = plt.subplots(ncols=3)
sns.countplot(data=log,x=log.impersonationLevel,ax=axs[0])
sns.countplot(data=log,x=log.elevatedToken,ax=axs[1])
sns.countplot(data=log,x=log.logonType,ax=axs[1])
plt.figure(figsize=(10,6))
plt.show()

weights=np.array([0.06,0.05,0.07, 0.05,0.13,0.13,0.06, 0.13,0.06,0.13,0.13])
gower_matrix=gower.gower_matrix(sample2.values,weight=weights)
```

```
scaled_gower_matrix = (gower_matrix - np.min(gower_matrix)) /
(np.max(gower_matrix) - np.min(gower_matrix))



def vis_tsne(data, cluster_result):
    datav = data.copy()


    tsne = TSNE(n_components=3,
random_state=3,metric='precomputed',perplexity=27)
#     weights=np.array([0.06,0.05,0.07, 0.05,0.13,0.13,0.06, 0.13,0.06,0.13,0.13])
#     gower_matrix=gower.gower_matrix(datav.values,weight=weights)
#     scaled_gower_matrix = (gower_matrix - np.min(gower_matrix)) /
(np.max(gower_matrix) - np.min(gower_matrix))


#    jaccard_distance_matrix = squareform(pdist(datav.values, metric='jaccard'))
    tsne_res = pd.DataFrame(tsne.fit_transform(scaled_gower_matrix),
index=datav.index)
    datav['cluster']=cluster_result
    fig = px.scatter_3d(
        tsne_res, x=0, y=1, z=2,
        color=datav.cluster, labels=datav["cluster"].astype(object),
        hover_data = {"index": datav.index} # Add the hover data dictionary
    )
    fig.update_traces(marker_size=2)
    fig.show()



from prince import FAMD
def vis_famd(data, cluster_result):
```

```
    datav = data.copy()

    famd = FAMD(n_components=3, n_iter=5, check_input=True,random_state=3)
    famd = famd.fit(datav)
    print(famd.column_coordinates_)
    famdres=famd.row_coordinates(datav)

    datav['cluster']=cluster_result
    fig = px.scatter_3d(famdres, x=0, y=1, z=2,
                        color=datav.cluster,
                        labels=datav["cluster"].astype(object),
                        hover_data = {"index": datav.index}
    )
    fig.update_traces(marker_size=3)
    fig.show()

def vis_mca(data, cluster_result):

    datav = data.copy()

    mca = MCA(n_components=3, n_iter=5, check_input=True,random_state=3)
    mca = mca.fit(datav)
    mcares=mca.row_coordinates(datav)

    datav['cluster']=cluster_result
    fig = px.scatter_3d(mcares, x=0, y=1, z=2,
                        color=datav.cluster,
                        labels=datav["cluster"].astype(object),
```

```
                hover_data = {"index": datav.index}
    )
    fig.update_traces(marker_size=3)
    fig.show()

from sklearn.cluster._dbscan import DBSCAN
dbscan = DBSCAN(eps=0.27 , min_samples=25,metric='precomputed')
dbscan=dbscan.fit_predict(scaled_gower_matrix)
print('Clustering result(розподіл по кластерам)')
print(pd.DataFrame(dbscan).value_counts())
anDB=sample2[dbscan==-1].index
vis_tsne(sample2,dbscan)

anDB=sample2[dbscan==-1].index
# anDB=anDB.append(sample2[dbscan==3].index)
anDB=anDB.append(sample2[dbscan==4].index)
# anDB=anDB.append(sample2[dbscan==5].index)
resDBSCAN = pd.Series(1,index=sample2.index)
resDBSCAN[anDB]=-1
print(resDBSCAN.value_counts())
vis_tsne(sample2,resDBSCAN)


from sklearn.neighbors import LocalOutlierFactor
lof = LocalOutlierFactor(metric='precomputed',n_neighbors=40)
lof_scores = lof.fit_predict(scaled_gower_matrix)
print(pd.DataFrame(lof_scores).value_counts())
anLOF=sample2[lof_scores==-1].index
```

```
# len(set(anLOF))


lof2 = LocalOutlierFactor(metric='precomputed',n_neighbors=40,contamination=0.06)
lof_scores2 = lof2.fit_predict(scaled_gower_matrix)
print(pd.DataFrame(lof_scores2).value_counts())
anLOF2=sample2[lof_scores2==-1].index
vis_tsne(sample2,lof_scores2)

from sklearn.ensemble import IsolationForest
tsne = TSNE(n_components=3, random_state=3,metric='precomputed',perplexity=27)
tsne_res = pd.DataFrame(tsne.fit_transform(scaled_gower_matrix),
index=sample2.index)

IF=IsolationForest(n_estimators=100,contamination='auto')
tsne_if=IF.fit_predict(tsne_res)
pd.Series(tsne_if).value_counts()

vis_tsne(sample2,tsne_if)

IF2=IsolationForest(n_estimators=75, contamination=0.06)
tsne_if2=IF2.fit_predict(tsne_res)
pd.Series(tsne_if2).value_counts()
vis_tsne(sample2,tsne_if2)
```