ДОДАТОК А

Посібник користувача

ЗМІСТ

ВСТУП

А. 1 Призначення застосунку

Застосунок «RuleEngine» призначений для процесу прогнозування поломок виробничого обладнання шляхом аналізу великих масивів даних. Основною метою є виявлення потенційно небезпечних аномалій, визначення закономірностей, які передують несправностям, та попередження аварійних ситуацій на підприємствах. Використання алгоритмів машинного навчання, таких як DBSCAN для кластеризації та Apriori для формування асоціативних правил, дозволяє досягти високої точності в аналізі даних та створити надійну основу для прийняття управлінських рішень. Завдяки інтеграції функцій візуалізації та генерації звітів, система не лише ідентифікує проблеми, але й формує зрозумілі інтерпретації отриманих результатів, що сприяє підвищенню ефективності виробничих процесів. Основна цінність застосунку полягає у його здатності адаптуватися до умов різних підприємств, забезпечуючи гнучкість, масштабованість і зручність використання.

А. 2 Технічні вимоги роботи застосунку

Для забезпечення коректної роботи застосунку «RuleEngine» передбачено дотримання низки технічних вимог, які охоплюють апаратні, програмні та інфраструктурні аспекти. Застосунок розроблено для роботи на сучасних персональних комп’ютерах або серверних системах, що використовують операційну систему Windows 10 або новіші версії. Мінімальні апаратні вимоги включають процесор із тактовою частотою не менше 2,0 ГГц (рекомендовано багатоядерний), оперативну пам’ять об’ємом не менше 8 ГБ та наявність щонайменше 10 ГБ вільного місця на жорсткому диску для зберігання даних і роботи застосунку. Для ефективної візуалізації результатів аналізу бажано використовувати монітор із роздільною здатністю не менше 1920×1080.

З програмного забезпечення обов’язковою є наявність Python версії 3.8 або новішої, а також встановлених бібліотек для роботи з даними, зокрема NumPy, Pandas, sys, os, Matplotlib, Scikit-learn та Mlxtend. Інші залежності автоматично встановлюються під час початкової конфігурації.

Застосунок не потребує підключення до мережі. Це рішення було прийнято з метою максимально убезпечити важливі дані від можливих кібер-атак.

Дотримання цих технічних вимог є важливим для гарантування стабільної роботи застосунку, точності результатів аналізу та безперебійного функціонування системи в умовах реального виробництва.

А. 3 Опис основних функцій застосунку

Застосунок «RuleEngine» призначений для автоматизованого прогнозування поломок на виробничих підприємствах шляхом аналізу даних із сенсорів обладнання.

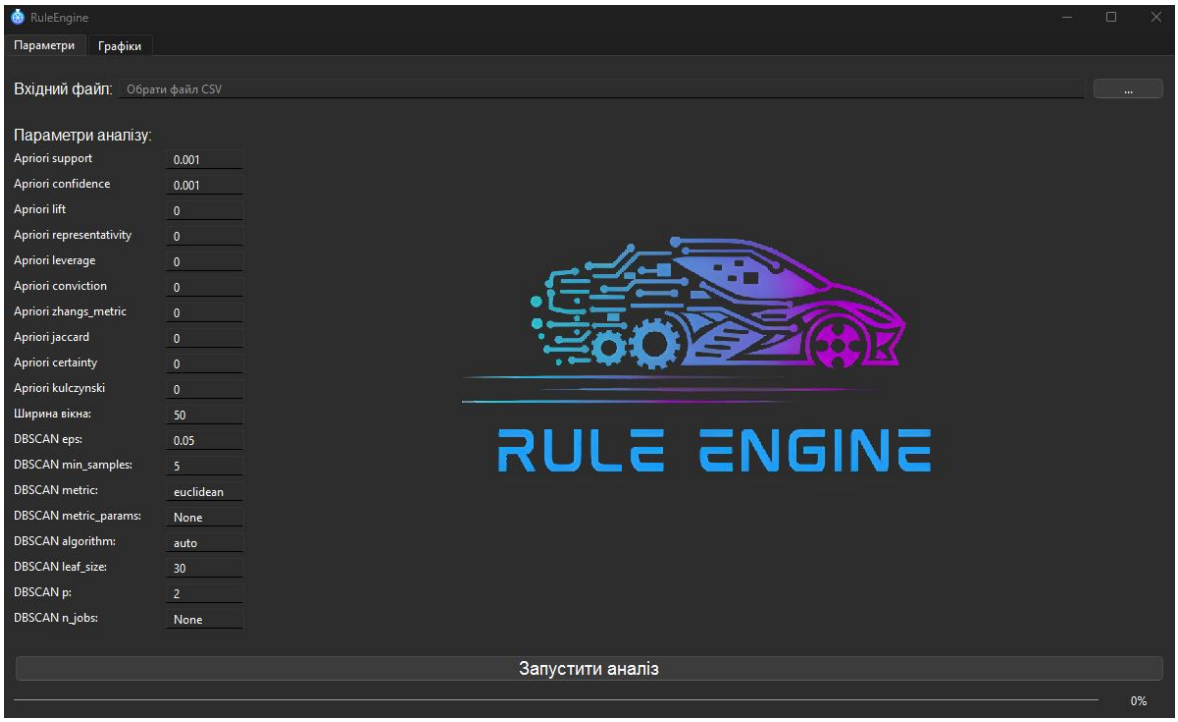

Рисунок А.3.1 – початковий інтерфейс застосунку «RuleEngine»

Застосунок передбачає можливість завантаження файлу формату .csv з показниками датчиків. Значення повинні мати подібне форматування (відіграє роль лише точна назва колонок їх розташування лишається на погляд оператора).

Таблиця А.3.1 – Приклад вхідних даних

timestamp	device	metric	value	manual_anomaly
01.01.2025 0:00	Device_1	Metric_1	0.31021438410706514	0
01.01.2025 0:01	Device_1	Metric_1	1.9460992578528598	0
01.01.2025 0:02	Device_1	Metric_1	15.420585183739963	0
01.01.2025 0:03	Device_1	Metric_1	29.626047568285284	0
01.01.2025 0:04	Device_1	Metric_1	24.7036032919286	0
01.01.2025 0:05	Device_1	Metric_1	27.49054540344458	0

На основі цих даних відбувається весь подальшой розрахунок, враховуючи заповнені критеріальні поля застосунку. Значення по замовчуванню в цих полях проставлені на основі отримання максимально доступної інформації з даних. Бо у випадку аналізу аномалій навіть менш ймовірні значення усе одно несуть в собі цінність і потребують врахування.

В ході виконання застосунку на вкладці «Графіки» генеруються базові необхідні відображення закономірностей показників та їх агрегацій від часу вимірювання.

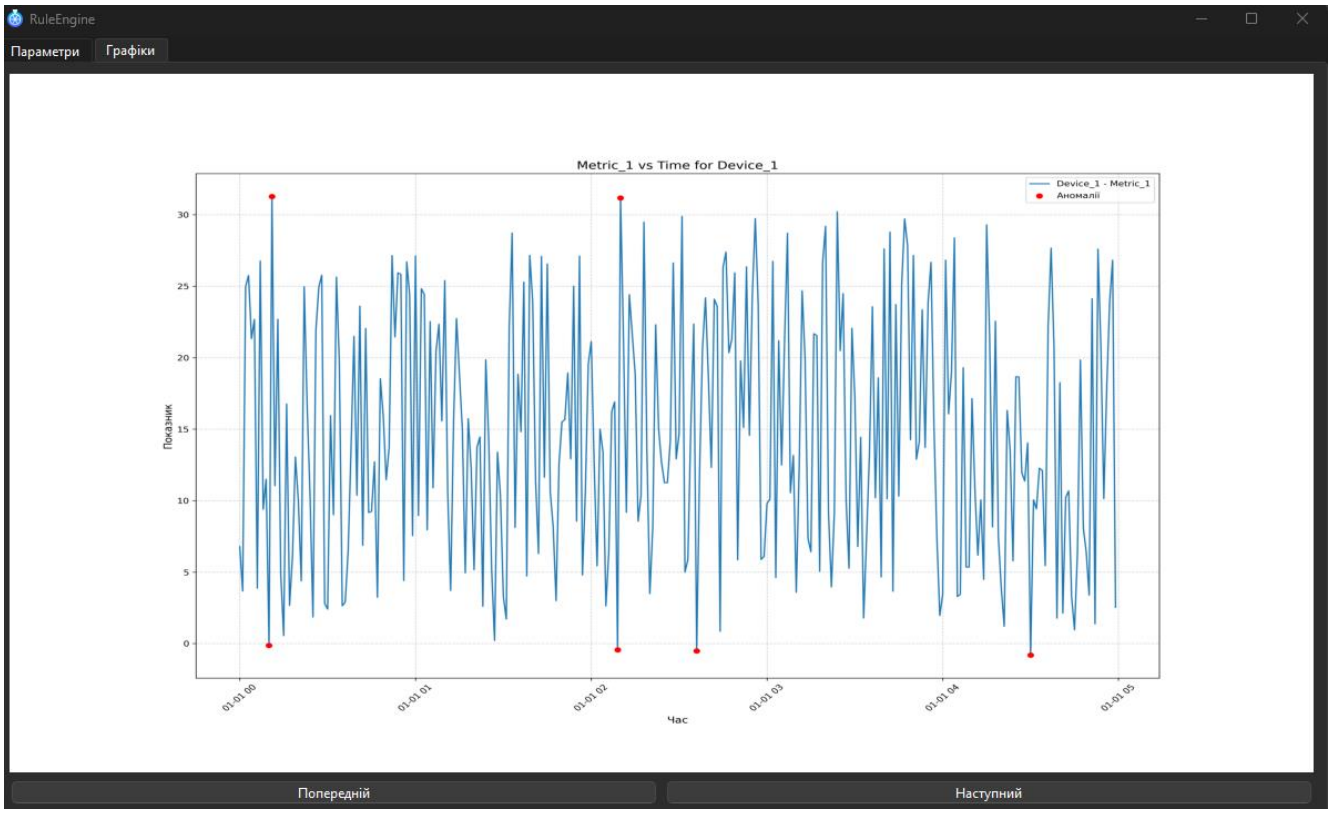

Рисунок А.3.2 – Приклад графічного зображення залежності на вкладці «Графіки»

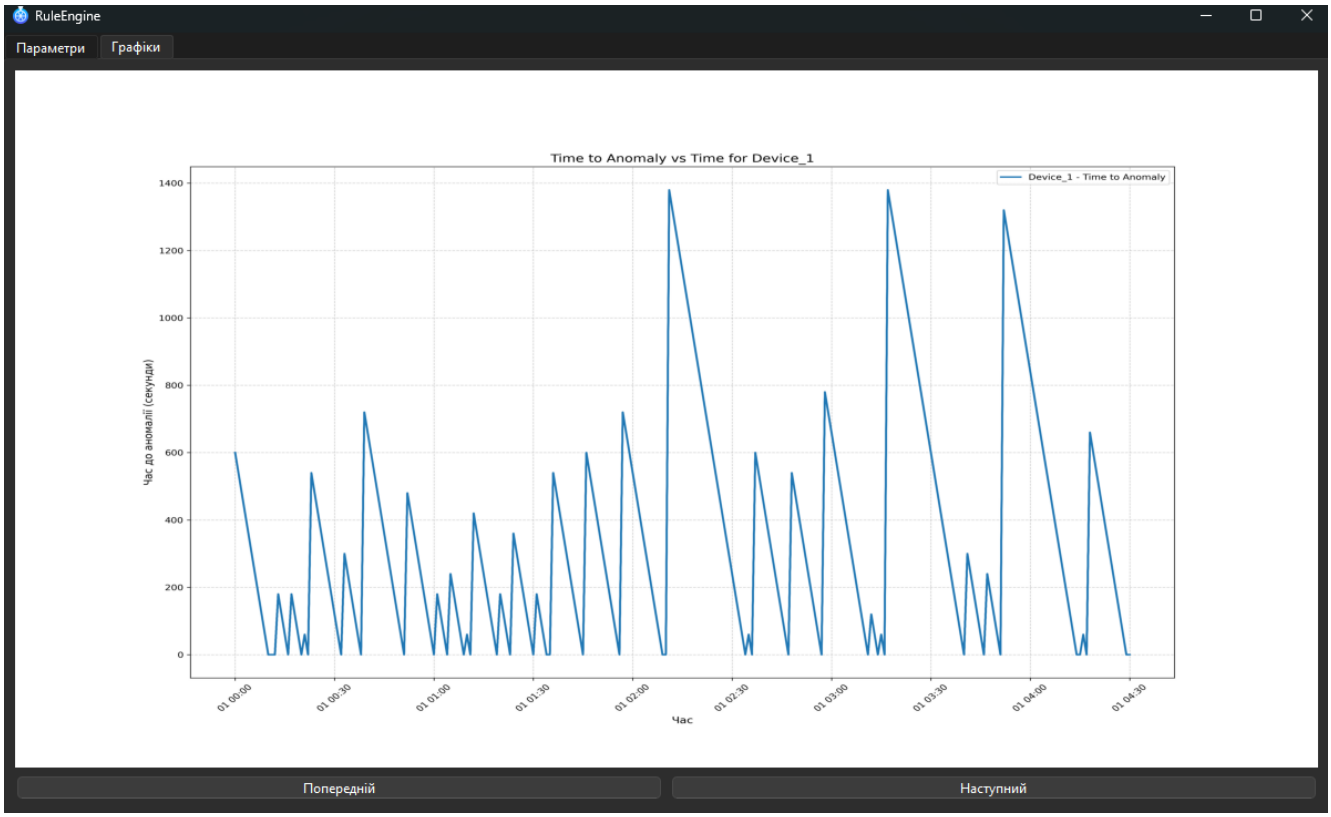

Рисунок А.3.3 – Приклад графічного зображення залежності на вкладці «Графіки»

Завдяки цим функціям застосунок надає користувачам інструменти для моніторингу стану обладнання, аналізу потенційних ризиків та планування технічного обслуговування, що забезпечує зниження простоїв, підвищення ефективності виробництва та мінімізацію витрат.

А.4 Рекомендації по експлуатації застосунку

Для забезпечення максимальної ефективності використання застосунку з прогнозування поломок необхідно дотримуватися кількох ключових рекомендацій щодо його експлуатації. Насамперед, перед початком роботи із застосунком слід переконатися, що всі необхідні технічні вимоги виконані, а системи збору даних із сенсорів підключені до відповідної інфраструктури. Це включає перевірку коректного функціонування апаратного забезпечення, стабільності мережевого з’єднання та наявності прав доступу до потрібних баз даних.

Рекомендується починати роботу із попередньої перевірки вхідних даних. Це допоможе уникнути ситуацій, коли в систему завантажуються некоректні або неповні дані, що може знизити якість прогнозування.

Користувачам рекомендується ретельно налаштовувати параметри алгоритмів DBSCAN і Apriori, виходячи зі специфіки обладнання та виробничих процесів. Наприклад, значення параметра eps в DBSCAN слід вибирати відповідно до характеру відхилень, які мають бути ідентифіковані, а мінімальну підтримку та впевненість для Apriori варто коригувати залежно від обсягу та якості даних. Для зручності застосунок містить початкові рекомендовані налаштування, які можуть бути адаптовані під конкретні умови.

Під час аналізу даних необхідно звертати увагу на інтерактивні графіки, що демонструють динаміку параметрів, виявлені аномалії та прогнозовані закономірності. Ці графіки дають змогу швидко оцінити стан обладнання та виявити потенційні ризики. Для більш глибокого аналізу можна скористатися функцією деталізації окремих ділянок графіків або експортувати результати у форматі CSV для подальшого опрацювання.

Щоб уникнути перевантаження системи та збоїв у роботі, рекомендується виконувати регулярне технічне обслуговування програмного забезпечення. Це включає періодичне очищення тимчасових файлів, оновлення бібліотек Python, що використовуються, та перевірку сумісності з оновленнями операційної системи. Застосунок має вбудовані механізми перевірки версій, які сповіщають користувача про необхідність оновлення.

Користувачам також рекомендується регулярно ознайомлюватися з документацією застосунку, особливо у разі додавання нових функцій або оновлення алгоритмів.

Дотримання цих рекомендацій сприятиме стабільній роботі застосунку, підвищенню точності прогнозування та зменшенню ризиків, пов’язаних із потенційними поломками обладнання.

ДОДАТОК Б

Апробація результатів кваліфікаційної роботи

МАТЕРІАЛИ
IV ВСЕУКРАЇНСЬКОЇ СТУДЕНТСЬКОЇ НАУКОВОЇ
КОНФЕРЕНЦІЇ

15 ГРУДНЯ 2023 РІК • М. ІВАНО-ФРАНКІВСЬК, УКРАЇНА

НАУКОВИЙ ПРОСТІР: АНАЛІЗ, СУЧАСНИЙ СТАН, ТРЕНДИ ТА ПЕРСПЕКТИВИ

ISBN 978-617-8126-73-5
DOI 10.36074/liga-ukr-15.12.2023

МОЛОДІЖНА НАУКОВА ЛІГА

МАТЕРІАЛИ КОНФЕРЕНЦІЇ

IV ВСЕУКРАЇНСЬКА СТУДЕНТСЬКА НАУКОВА КОНФЕРЕНЦІЯ

НАУКОВИЙ ПРОСТІР: АНАЛІЗ, СУЧАСНИЙ СТАН, ТРЕНДИ ТА ПЕРСПЕКТИВИ

15 ГРУДНЯ 2023 РІК

м. ІВАНО-ФРАНКІВСЬК, УКРАЇНА

Вінниця, Україна
«UKRLOGOS Group»
2023

УДК 082:001
Н 34

Голова оргкомітету: Коренюк І.О.
Верстка: Зрада С.І.
Дизайн: Бондаренко І.В.

Конференцію зареєстровано Державною науковою установою «УкрІНТЕІ» в базі даних науково-технічних заходів України та інформаційному бюлетені «План проведення наукових, науково-технічних заходів в Україні» (Посвідчення №331 від 16.06.2023).

Н 34 **Науковий простір: аналіз, сучасний стан, тренди та перспективи**: матеріали IV Всеукраїнської студентської наукової конференції, м. Івано-Франківськ, 15 грудня, 2023 рік / ГО «Молодіжна наукова ліга». — Вінниця: ТОВ «УКРЛОГОС Груп», 2023. — 728 с.

ISBN 978-617-8126-73-5
DOI 10.36074/liga-ukr-15.12.2023

Викладено матеріали учасників IV Всеукраїнської мультидисциплінарної студентської наукової конференції «Науковий простір: аналіз, сучасний стан, тренди та перспективи», яка відбулася 15 грудня 2023 року у місті Івано-Франківськ, Україна.

УДК 082:001

ISBN 978-617-8126-63-6

СЕКЦІЯ 20.
СИСТЕМНИЙ АНАЛІЗ, МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЯ

Іщенко Михайло Дмитрович, здобувач вищої освіти
факультету автоматизації та комп'ютерних технологій
Харківський національний університет радіоелектроніки, Україна

Науковий керівник: Максимова Світлана Святославівна, канд. техн. наук,
доцент, доцент кафедри КІТАР
Харківський національний університет радіоелектроніки, Україна

АНАЛІЗ ВИКОРИСТАННЯ АСОЦІАТИВНИХ ПРАВИЛ З МЕТОЮ ПЕРЕДБАЧЕННЯ ПОЛОМОК НА ВИРОБНИЦТВІ

В сучасному виробництві наголос на запобіганні поломкам стає все більш важливим. Завчасне виявлення технічних недоліків та забезпечення безперебійності виробничих процесів визначає успіх бізнесу і впливає на безпеку споживачів. Це особливо актуально у великих виробничих підприємствах, де навіть найменша поломка може призвести до серйозних проблем та великих збитків. Можливість заздалегідь спрогнозувати поломку та підготуватися до неї є вкрай важливою для бізнесу.

Використання цього підходу ідеально вкладається у концепцію Industry 4.0.

Industry 4.0, яку також називають четвертою промисловою революцією або 4IR, - це наступний етап оцифрування виробничого сектору, зумовлений проривними тенденціями, серед яких зростання обсягів даних і зв'язку, аналітики та людино-машинна взаємодія зі вдосконаленням робототехніки [1].

Основою прогнозування поломок є дані з датчиків, які фіксують стан обладнання та процесів виробництва. Інтеграція цих даних у систему аналізу дозволяє спостерігати за показниками працездатності, температурою, тиском та іншими факторами, що є ключовими для визначення стану обладнання. Здебільшого ці дані втрачаються одразу після їх першочергово аналізу, що є великою помилкою. Довготривалий збір даних з датчиків дозволить проводити додатковий аналіз щодо можливих поломок у найближчому майбутньому.

Сучасні системи збору даних невпинно удосконалюються завдяки розробці нових датчиків із підвищеною чутливістю до зовнішніх умов та розширеними можливостями фіксації різних параметрів. Підвищення точності та швидкості збору інформації дозволяє отримувати найактуальніші дані моніторингу в реальному часі.

Системи збору даних постійно вдосконалюються через розробку нових датчиків з вищою чутливістю до навколишніх умов і додатковими можливостями фіксації різних параметрів. Покращення точності та швидкості збору інформації дозволяє отримувати найактуальніші дані моніторингу в реальному часі. Крім того, використання сучасних інструментів для обробки та аналізу великих обсягів даних, які

ще називають Big Data сприяє здійсненню більш точних прогнозів та виявленню відхилень у роботі устаткування.

Big Data (укр. великі дані) — це великий масив структурованої та неструктурованої інформації, а також інструменти, підходи, методи обробки та зберігання даних. Важливість великих даних залежить не тільки від їхньої кількості, а й від того, як компанія їх інтерпретує та використовує. Через об’єм та різноманітність даних обробляти їх традиційним програмним забезпеченням неможливо [2].

Новітні системи збору даних оснащені передовими технологіями зв'язку, такими як бездротові мережі чи Інтернет, які в свою чергу забезпечують миттєвий доступ до даних з будь-якої точки виробництва. Це дозволяє оперативно реагувати на потенційні проблеми, а також стежити за роботою обладнання в реальному часі, що максимально спрощує управління та контроль за виробництвом.

Отримана в ході цього аналізу інформація дозволить передбачати потенційні поломки задовго до їх виникнення. Наприклад, зміни в показниках температури чи вібрації можуть вказувати на несправність обладнання. Виявлення цих відхилень у показниках дозволяє приймати заходи щодо запобігання поломок шляхом проведення планового обслуговування чи заміни деталей, ще до моменту поломки самого елементу.

Для більш детального розгляду проблеми було побудовано IDEF0 з її подальшою декомпозицією.

IDEF0 – методологія функціонального моделювання. Використовується для створення функціональної моделі, що відображає структуру і функції системи, а також потоки інформації і матеріальних об'єктів, що зв'язують ці функції [3].

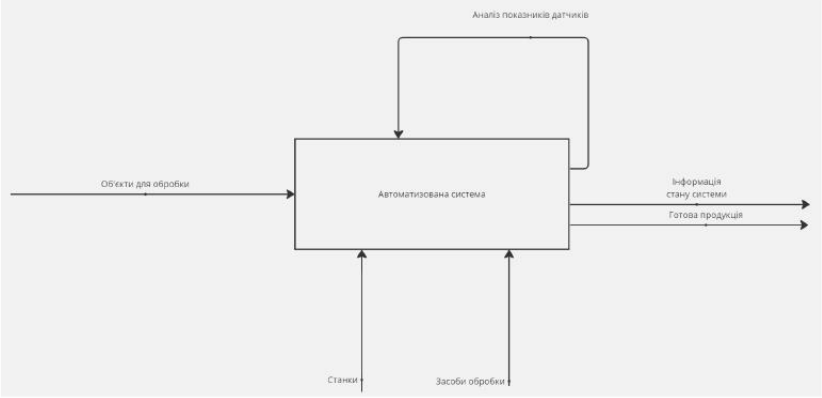

Рис. 1. **Контекстна діаграма IDEF0**

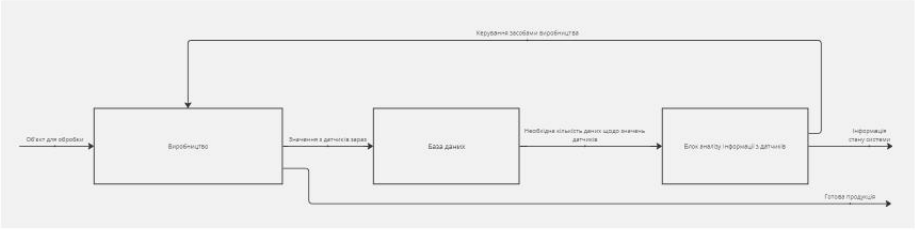

Рис. 2. **Перший рівень декомпозиції діаграми IDEF0**

Цього рівня декомпозиції цілком достатньо для детального огляду проблеми. З першого рівня декомпозиції можна побачити блок «Виробництво», який виконує роботу над заготовками та в ньому ж знаходяться датчики. Ці датчики заносять інформацію у базу даних. Чим більша база даних, тим більш точним може бути прогнозування поломок на виробництві. Наступний потік даних бере необхідні значення датчиків та заносить у блок аналізу інформації де відбувається пошук асоціативних правил.

База даних (БД) — це організована структура, призначена для зберігання, зміни й обробки взаємопов'язаної інформації, переважно великих обсягів [4].

Асоціативні правила – це аналог операторів if/then, які допомагають виявити зв'язки між незв'язаними даними в базі даних. Пошук виявляє приховані зв'язки у на перший погляд, ніяк незв'язаних даних. Ці зв'язки – правила. Ті, які перевищують певний поріг, вважаються цікавими. Такі правила дають можливість виконувати дії ґрунтуючись на певних шаблонах. Вони так само допомагають в прийнятті і поясненні рішень. Як і більшість методів Data Mining, даний метод дозволяє перетворити потенційно неосяжну кількість інформації в невеликий і зрозумілий набір статистичних показників [5].

На основі цих асоціативних правил створюється автоматичний керуючий вплив на виріб (наприклад за умови ймовірності поломки елемента виробництва >95% понизити темпи виробництва) та загальна інформація виводиться з системи для подальшого розгляду.

Ефективне управління ризиками у виробництві є ключовим аспектом для забезпечення стабільності та безперебійної роботи підприємства. Сучасні підходи до управління ризиками включають вдосконалені методики аналізу, прогнозування та мінімізації потенційних загроз. Розробка та застосування новітніх систем оцінки ризиків дозволяє ідентифікувати, аналізувати та управляти ризиками ефективніше.

Управління ризиками виробництва базується на використанні інформації, отриманої з різноманітних джерел, зокрема з систем моніторингу, датчиків та даних процесів виробництва. Впровадження технологій Big Data та штучного інтелекту (AI) дозволяє проводити аналіз великих обсягів даних, щоб визначити, класифікувати та прогнозувати можливі ризики.

Ефективне управління ризиками ґрунтується на системному підході до ідентифікації та оцінки ризиків у всіх сферах виробництва. Це включає постійний моніторинг даних з датчиків, аналіз виникнення відхилень в роботі устаткування, виявлення можливих збоїв та прийняття заходів щодо їх запобігання.

Основними аспектами ефективного управління ризиками є системи раннього попередження, що базуються на аналізі даних з датчиків. Це дозволяє оперативно реагувати на можливі негативні сценарії та зменшує вплив потенційних ризиків на виробничі процеси.

Таких підхід дозволить значно зменшити кількість простоїв системи, аварійних ситуацій та загалом проблем на виробництві.

Підбиваючи підсумки, аналіз інформації з датчиків для передбачення поломок на виробництві є критично важливим елементом, який впливає на ефективність виробництва та безпеку. Цей підхід є ключовим для забезпечення безперебійності та надійності виробництва.

Список використаних джерел:

1. McKinsey&Company: [Інтернет-портал]. URL: https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-are-industry-4-0-the-fourth-industrial-revolution-and-4ir (дата звернення: 06.12.2023).
2. Kyivstar business hub: [Інтернет-портал]. URL: https://hub.kyivstar.ua/articles/shho-take-big-data.
3. Черняк О. І., Захарченко П. В. Інтелектуальний аналіз даних. Київ, 2010. 320 с.
4. HostIQ: [Інтернет портал] URL: https://hostiq.ua/wiki/ukr/database/.
5. Ситніков Д.Е., Ситнікова П.Е., Тітов С.В., Тітова О.В. Фільтрація результуючого набору асоціативних правил з точки зору оцінки цікавості. Системи обробки інформації. 2021. № 1(164). С. 83-88. https://doi.org/10.30748/soi.2021.164.09.

НАУКОВЕ ВИДАННЯ

МАТЕРІАЛИ IV ВСЕУКРАЇНСЬКОЇ
СТУДЕНТСЬКОЇ НАУКОВОЇ КОНФЕРЕНЦІЇ

«НАУКОВИЙ ПРОСТІР: АНАЛІЗ, СУЧАСНИЙ СТАН, ТРЕНДИ ТА ПЕРСПЕКТИВИ»

15 грудня 2023 рік • м. Івано-Франківськ, Україна

Українською та англійською мовами

Всі матеріали пройшли перевірку на плагіат та експертизу за формальними ознаками (форматування, стиль мови, оформлення цитувань та списку використаних джерел). За точність викладеного матеріалу відповідальність несуть автори та їх наукові керівники. Організаційний комітет не завжди поділяє позицію авторів.

Підписано до друку 15.12.2023.
Папір офсетний. Цифровий друк. Формат 60×84/16.
Гарнітура Times New Roman, Poiret One та Arial.
Умовно-друк. арк. 42,31. Замовлення № 23/004.
Тираж: 50 екземплярів. Віддруковано з готового оригінал-макету.

Контактна інформація організаційного комітету:
Громадська організація «Молодіжна наукова ліга»
21037, Україна, м. Вінниця, вул. Зодчих, 40, офіс 103
Телефони: +38 098 1948380; +38 098 1526044
E-mail: info@liga.science | URL: www.liga.science

Видавець: ТОВ «УКРЛОГОС Груп».
21037, Україна, м. Вінниця, вул. Зодчих, 18, офіс 81. E-mail: info@ukrlogos.in.ua
Свідоцтво суб'єкта видавничої справи: ДК № 7860 від 22.06.2023.

ДОДАТОК В

Висвітлення результатів кваліфікаційної роботи

АНАЛІЗ ІМПЛЕМЕНТАЦІЇ ЗНАЧЕННЯ КІЛЬКОСТІ В АЛГОРИТМИ ПОШУКУ АСОЦІАТИВНИХ ПРАВИЛ

M. Ishchenko, D. Sitnikov
Kharkov National University of Radio Electronics, Ukraine

В цій статті пропонується розглянути можливість використання показника кількості для розширення інформативності асоціативних правил. Використання цього показника є інтуїтивно зрозумілим, оскільки майже завжди при пошуку асоціативних правил ми маємо справу з кількісними значеннями. Використовуючи запропоновані методи можна «вичавити» додатково інформацію з вже отриманих асоціативних правил, що збільшить їх ефективність.

Ключові слова: *асоціативні правила, підтримка, довіра, Apriori, кількість, коефіцієнт, аналіз, інформація.*

Вступ

Постановка проблеми. Рішення проблеми стосується сфери Data Mining, яка має практичне використання у вигляді витягування корисної інформації з великого обсягу даних. Data Mining знаходить можливості та методи пошуку прихованої інформації, яку неможливо зрозуміти вручну через занадто великі обсяги даних. Головні задачі Data Mining включають у себе:

1) кластерний аналіз, який розв'язує проблему пов'язану з угрупуванням об'єктів у схожі класи або кластери на основі спільних ознак;
2) класифікацію, яка розв'язує проблему визначення категорій чи класів об'єктів на основі кореляції сприятливих властивостей;
3) прогнозування, яке розв'язує проблему передбачення майбутніх значень змінних на основі даних про попередні значення;
4) асоціативний аналіз, який розв'язує проблему виявлення відносин між рядами ознак з метою визначення додаткової інформації про споживачів товарів чи послуг;
5) аналіз тексту, який розв'язує проблему виявлення різних завдань, пов'язаних з обробкою текстових даних, зокрема, інтерпретацію великих обсягів документації;
6) візуалізація даних, яка розв'язує проблему перетворення великих мас поставлених даних в графічну форму для зручної інтерпретації.

Аналіз імплементації значень кількості у алгоритми пошуку асоціативних правил відбувається при вирішенні задачі асоціативного аналізу (у списку вище вона знаходиться під номером 4) після безпосереднього пошуку асоціативних правил.

Пошук асоціативних правил є методом асоціативного аналізу для виявлення статистичного взаємозв'язку між різними елементами в наборах даних.

Спочатку завдання пошуку асоціативних правил трактувалось, як прикладне у сфері маркетингу (так званий «кошик покупок»), тобто виявлялися товари, які покупці намагались купувати разом. Згодом завдання пошуку асоціативних правил було сформульовано в більш загальному вигляді, як завдання знаходження логічних закономірностей в даних[1].

Аналіз останніх досліджень та публікацій. Асоціативне правило – це аналог оператора if/then, який допомагає виявити зв'язки між на перший погляд незв'язними даними в базі даних.

Нехай $L=I_1,I_2,\ldots,I_m$ – множина ознак об'єктів. Нехай T-множина записів. Кожен запис t представлений бінарним вектором t[k]. За умови, що t не містить ознаку I_k (k=1,m), бінарний вектор t[k] =0. У протилежному випадку, якщо t містить ознаку I_k (k=1,m),бінарний вектор t[k]=1. Нехай X – підмножина деяких ознак з L, тобто $X \subseteq L$. Вважається, що запис t

задовольняє X, якщо ∀ $I_k \subseteq X$, t[k]=1. Під асоціативним правилом розуміється вираз виду X→Y, де X⊆L. При чому X⋂Y=∅[2].

Існують характеристики, які дозволяють оцінити якість отриманого правила та істотно обмежити кількість генерованих залежностей [3-6].

-значення підтримки *(support)*, що дорівнює ймовірності наявності у записі значення X та Y одночасно. Таким чином правило $X \to Y$ має підтримку *supp*, якщо воно справедливо для *supp%* взятих випадків:

$$support(X \to Y) = P(X \cup Y); \quad (1)$$

-значення достовірності (confidence), що відображає відношення наявності наслідку *(Y)* за наявності передумови *(X)* правила:

$$confidence(X \to Y) = \frac{P(X \cup Y)}{P(X)} = \frac{support(X \to Y)}{support(X)}; \quad (2)$$

Виходячи з цих основних критеріїв можна побачити, що значення кількості ознак у транзакціях ніяким чином не враховується.

Мета статті – описати метод імплементації значення кількості ознак у транзакціях для покращення якості асоціативних правил.

Імплементація значення кількості у алгоритми пошуку асоціативних правил

Перший метод. Розглянемо ситуацію з генерацією асоціативних правил, наприклад, за алгоритмом Аргіогі. Першочергово визначаються обмежувальні значення підтримки *(1)* достовірності *(2)*. Після чого починаючи зі значення підтримки відбувається поетапний пошук з відсіюванням «неважливих» правил. На останньому етапі відбувається отримання «цікавих» правил, наприклад: X→Y.

Безпосередньо після цього пропонується почати етап імплементації значення кількості. Першочергово необхідно створити хеш-таблицю передумови *(X)*. У таблицю заносяться можливі значення кількості передумови (наприклад *X* зустрічається у кількостях 1,2,3,4,7,10,…). На даному етапі множина значень формується серед усіх транзакцій. Наступним етапом розраховується підтримка для кожного зі значень хеш-таблиці, отримуючи при цьому наступні пари: кількість – її підтримка.

$$\sup(X) = P(X \cup DSrule), \quad (3)$$

де $DSrule$ – множина транзакцій.

Граничне значення може бути змінено спеціально для цього етапу. Усі значення підтримок, які менше граничного відсіюються, оскільки за антимонотонною властивістю серед них не буде знайдено нічого «цікавого». Після чого формуються хеш-таблиці для всіх значень наслідку зі значенням кількостей, які знаходяться в одній транзакції з вже визначеними кількостями передумов. Тепер розраховується значення підтримок, де вказані передумови та наслідки:

$$\sup(X \to Y) = P(X \cup Y), \quad (4)$$

Отримуємо значення підтримок, які перебираємо та відсіюємо «нецікаві». Використовуючи формулу *(2)* розраховуємо достовірність для значень підтримок, що залишилися. Таким чином отримується кінцева інформація на основі якої відбувається розширення асоціативного правила, наприклад 5X→10Y, дозволяє стверджувати, що за умови купівлі 5 одиниць X, буде відносно велика ймовірність купівлі 10 одиниць Y.

Другий метод. Аналогічно першому методу імплементація кількості відбувається після стандартної генерації асоціативних правил з метою розширення інформативності останніх. Наприклад, ми вже маємо правило $X \to Y$ і нас цікавить середньостатистичне співвідношення кількостей item у ньому. На початку зі сформованої множини транзакцій де в наявності одночасно передумова з наслідком отримуємо кількісне значення (сумарна кількість у транзакціях) значень передумови та ділимо на кількість транзакцій n:

$$x = \frac{X_{\text{кільк}}}{n}, \quad (5)$$

де $X_{кільк}$ – кількість елементу X у транзакціях n;

n – транзакції, які включають в себе передумову X та наслідок Y.

Повторюємо для значень наслідку й отримуємо x_2. Це дозволяє нам зафіксувати середньостатистичне співвідношення передумови та наслідку у межах правила, як значення xX→ x_2Y. Таким чином можна стверджувати, що середньостатистично на x значень X приходиться x_2 значень Y.

Експериментальне дослідження

Перший метод. Розглянемо ситуацію коли вже знайдені асоціативні правила для транзакцій з рисунка 1.

IdTransaction	1	2	3	4	5	6	7	8	9	10
ItemA	1	2	2	1	6	0	0	1	0	1
ItemB	2	2	3	4	5	0	0	2	0	2
ItemC	5	1	2	3	3	0	0	5	0	0
ItemD	0	4	4	0	0	0	5	0	0	0
ItemE	0	0	5	0	0	0	0	0	5	0
ItemF	0	0	0	0	3	2	6	0	0	0
ItemG	0	0	0	0	0	0	0	1	2	0

Рис 1. **Транзакції з вказаними кількостями item**

Відповідно при встановленні параметрів *support* >= 20% та *confidence* >= 70% було отримано наступні правила(рис 2).

ItemId			Confidence
A	B	C	0,857142857
B	C	A	0,857142857
C	B	A	1
A	B	C	0,857142857
A	C	B	1
B	C	A	1

Рис 2. **Асоціативні правила**

З рисунка 2 можна однозначно побачити, що буде корисно отримати співвідношення кількостей itemA, itemB та itemC між собою. Першим етапом буде створення хеш-таблиць значень кількостей для кожного item окремо та розрахування для них значень підтримки (рис 3) за формулою (1):

де X – кількість транзакцій;

itemN дорівнює вказаному значенню кількості;

Y – кількість всіх транзакцій.

quantity A	Support
1	0,00%
2	20,00%
6	10,00%

quantity B	Support
2	40,00%
3	10,00%
4	10,00%
5	10,00%

quantity C	Support
1	10,00%
2	10,00%
3	20,00%
5	20,00%

Рис 3. **Хеш-таблиця значень кількостей з їх підтримками**

Як можна побачити на цьому етапі вже відсіюються деякі значення кількостей згідно антимонотонної властивості алгоритму Apriori. Наступним етапом будуються хеш таблиці для 2х елементних комбінацій елементів з подальшим визначенням значення підтримки (рис. 4).

quantity A,B		Support
1	2	30,00%
1	4	16,67%
2	2	16,67%
2	3	16,67%
6	5	16,67%

quantity B,C		Support
2	0	10,00%
2	1	10,00%
2	5	20,00%
3	2	10,00%
4	3	10,00%
5	3	10,00%

quantity C,A		Support
1	2	10,00%
2	2	10,00%
3	1	10,00%
3	6	10,00%
5	1	20,00%

Рис 4. **Хеш-таблиця значень кількостей з їх підтримками**

Продовжуємо, поки є можливість отримати значення підтримки більше або дорівнює заданому (рис. 5).

quantity A,B,C			Support
1	2	5	20,00%
2	2	1	10,00%
2	3	2	10,00%
1	4	3	10,00%
6	5	3	10,00%
1	2	3	0,00%

Рис 5. **Хеш-таблиця значень кількостей з їх підтримками**

Наступним етапом буде розрахування значення confidence для отриманих значень, де підтримка задовольняє умові (рис. 6).

ItemId	quantity			Confidence
{C}->{A}	5		1	1
{A}->{C}	1		5	0,5
{A}->{B}	1		2	0,75
{B}->{A}	2		1	0,75
{A}->{B, C}	1	2	5	0,5
{B}->{A,C}	2	1	5	0,5
{C}->{A,B}	5	1	2	1
{A,B}->{C}	1	2	5	0,666666667
{B,C}->{A}	2	5	1	1
{C,A}->{B}	5	1	2	1

Рис 6. **Асоціативні правила зі значень кількостей**

Проаналізувавши отримані дані, можна побачити багато цікавої та важливої інформації стосовно відношення кількостей продуктів. Наприклад, існує правило, що за умови взяття itemC у кількості 5 одиниць, велика ймовірність, що кількість itemA буде дорівнювати 1, а itemB – 2.

Цю інформацію пропонується використовувати у сукупності з іншими даними для більш обширного огляду на асоціативні правила але і використання виключно цих правил також допустиме. Наприклад можна ефективно робити «акційні пакети», коли окремий набір продуктів коштуватиме менше, ніж звичайна купівля їх окремо.

Другий метод. Аналогічно першому методу етап імплементації значень кількостей починається вже після отримання асоціативних правил. Наприклад, проаналізуємо вже отримане правило {C, A}→{B}.

Першочергово необхідно отримати сумарну кількість кожного придбаного item з транзакцій у межах правила (транзакції, які мають усіх учасників правила) (рис.7).

IdTransaction	1	2	3	4	5	8	
ItemA	1	2	2	1	6	1	13
ItemB	2	2	3	4	5	2	18
ItemC	5	1	2	3	3	5	19
ItemD	0	4	4	0	0	0	
ItemE	0	0	5	0	0	0	
ItemF	0	0	0	0	3	0	
ItemG	0	0	0	0	0	1	

Рис 7. **Транзакції, які мають усіх учасників правила та значення кількості цікавих item**

Після чого кожне з цих значень ділиться на кількість транзакцій у межах правила. Це дозволяє отримати необхідні середньостатистичні коефіцієнти. Значення коефіцієнтів наведено на рисунку 8.

IdTransaction	1	2	3	4	5	8	
ItemA	1	2	2	1	6	1	2,166667
ItemB	2	2	3	4	5	2	3
ItemC	5	1	2	3	3	5	3,166667
ItemD	0	4	4	0	0	0	
ItemE	0	0	5	0	0	0	
ItemF	0	0	0	0	3	0	
ItemG	0	0	0	0	0	1	

Рис 8. **Транзакції, які мають усіх учасників правила та отримані значення коефіцієнтів**

Таким чином можна розширити вже існуюче правило {C,A}→{B} отриманими коефіцієнтами, тобто {3,1(6) C, 2,1(6) A}→{3 B}. Розширене правило можна трактувати наступним чином: «У межах виконання правила, середньостатистично, на 3,1(6) одиниць itemC та 2,1(6) одиниць itemA приходиться 3 одиниці itemB». Поєднуючи отриману інформацію з цього методу та попереднього можна прийти до більш обширної інформації. Тобто середньостатистично у межах правила ми маємо співвідношення кількостей, але при зустрічі конкретного значення кількості item (наприклад itemC = 5) ці коефіцієнти перевизначаються згідно першого методу.

Висновки

У статті було запропоновано методи для імплементації значень кількості у алгоритми пошуку асоціативних правил. Перший метод полягає у створенні нової множини даних з вже існуючою та подальшому її аналізу. Аналіз включає у себе поетапний пошук підтримок, що дозволяє зменшити кількість операцій відкидаючи нецікаві значення за допомогою антимонотонної властивості з подальшим визначенням достовірності. Використовуючи вказані методі можна «вичавити» з вже отриманого асоціативного правила більшу користь.

Список літератури

1. Ситніков, Д. Е., Ситнікова, П. Е., Тітов, С. В., & Тітова, О. В. Фільтрація результуючого набору асоціативних правил з точки зору оцінки цікавості. Системи обробки інформації. – 2021. Р. 83–88.

2. Ситніков, Д. Е., Ситнікова, П. Е., Тітов, С. В., & Тітова. Визначення параметрів узагальнених асоціативних правил методом декомпозиції. Системи обробки інформації. – 2019. Р. 58-63.

3. Agrawal R. Mining association rules between sets of items in large databases / R. Agrawal, T. Imielinski, A. Swami // Proc. Of the ACM SIGMOD Conference Washington DC, USA, May 1993. – P.207-216.

4. Agrawal R. R. Strikant. Fast algorithms for mining association rules / R. Agrawal, R. Srikant // Proc. Of the 20th VLDB Conference Satiago, Chile, September 1994.

5. Srikant R. Mining generalized association rules / R. Srikant, R. Agrawal // Proc. Of the 21th VLDB Conference Zurich, Swizerland, September 1995. – P.407-419/

6. Amir A., R.Feldman, R.Kashi. A new and versatile method for association generation / A. Amir, R. Feldman, R. Kashi. // Information Systems. – 1997. – Vol. 22, №6/7.–P.333–347.

ДОДАТОК Г

Текст програми

```
import numpy as np
import pandas as pd
import sys
import os
from sklearn.cluster import DBSCAN
from mlxtend.frequent_patterns import apriori, association_rules
from PyQt6.QtWidgets import (
    QApplication, QMainWindow, QTabWidget, QWidget, QVBoxLayout, QHBoxLayout,
    QLabel, QLineEdit, QPushButton, QSlider, QFileDialog, QProgressBar, QSpinBox,
)
from PyQt6.QtWidgets import QSizePolicy
from PyQt6.QtGui import (QGuiApplication, QFont, QPixmap, QIcon)
from PyQt6.QtCore import Qt
import matplotlib.pyplot as plt
from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg as FigureCanvas

################################################################################
# 1. Generate Synthetic Data with “Staggered” Anomalies
################################################################################
def generate_synthetic_data(num_devices=3, num_points=300, metrics_per_device=3, seed=42):
    ««««
    Generates data across multiple devices & metrics, with anomalies
    inserted in the earliest ~50 and latest ~50 timestamps for each.
    ««««
    # np.random.seed(seed)
    time_index = pd.date_range(«2025-01-01 00:00:00», periods=num_points, freq=«min»)

    all_data = []
    for d_id in range(num_devices):
        device_name = f»Device_{d_id + 1}»
        for m_id in range(metrics_per_device):
            metric_name = f»Metric_{m_id + 1}»

            base = np.random.uniform(0, 30, num_points)
```

```
        noise = np.random.normal(0, 0.8, num_points)
        values = base + noise

        # Manually place anomalies in earliest ~50 and latest ~50
        anomaly_flags = np.zeros(num_points, dtype=int)

        tmp_df = pd.DataFrame({
            «timestamp»: time_index,
            «device»: device_name,
            «metric»: metric_name,
            «value»: values,
            «manual_anomaly»: anomaly_flags
        })
        all_data.append(tmp_df)

    df = pd.concat(all_data).reset_index(drop=True)
    df.to_csv('1. InputData.csv', index=False)
    return df

################################################################################
# 2. DBSCAN => cluster => anomaly
################################################################################
def dbscan_anomaly(df, eps, min_samples, metric, metric_params, algorithm, leaf_size, p, n_jobs):
    ««««
    Performs DBSCAN on (timestamp_ordinal, value).
    cluster=-1 => anomaly.
    ««««
    # Sort & interpolate
    df = df.sort_values([«device», «metric», «timestamp»]).copy()
    df[«value»] = df.groupby([«device», «metric»])[«value»].transform(lambda x: x.interpolate(method=«linear»))
    df = df.dropna().copy()

    # DBSCAN
    df[«timestamp_ordinal»] = df[«timestamp»].map(pd.Timestamp.toordinal)
```

```
    dbs_model = DBSCAN(eps=eps, min_samples=min_samples, metric=metric,
                       metric_params=metric_params, algorithm=algorithm,
                       leaf_size=leaf_size, p=p, n_jobs=n_jobs)
    df[«cluster»] = dbs_model.fit_predict(df[[«timestamp_ordinal», «value»]])
    df[«anomaly»] = (df[«cluster»] == -1).astype(int)
    return df

def relabel_small_clusters_as_anomaly(df, min_cluster_size=5):
    ««««
    If any cluster has fewer than min_cluster_size points, label it as -1 => anomaly.
    ««««
    cluster_counts = df[«cluster»].value_counts()
    small_clusters = cluster_counts[cluster_counts < min_cluster_size].index
    df.loc[df[«cluster»].isin(small_clusters), «cluster»] = -1
    df[«anomaly»] = (df[«cluster»] == -1).astype(int)
    return df

def two_cluster_labels(df):
    ««««
    Collapse all non--1 clusters into 0 => only 'Anomaly' (-1) and 'Normal' (0).
    ««««
    df[«cluster»] = df[«cluster»].apply(lambda c: -1 if c == -1 else 0)
    df[«anomaly»] = (df[«cluster»] == -1).astype(int)
    return df

def assign_two_names(df):
    ««««
    cluster=-1 => 'Anomaly', cluster=0 => 'Normal'.
    ««««
    df[«ClusterName»] = df[«cluster»].map({-1: «Anomaly», 0: «Normal»})
    df.to_csv('2. DBSCANData.csv', index=False)
    return df

################################################################################
```

```
# 3. Future Anomaly: AnomalyFutureEvent
################################################################################
def mark_time_to_anomaly(df, time_threshold):
    ««««
    Mark AnomalyFutureEvent=1 if next anomaly is within 'time_threshold' seconds.
    ««««
    df_sorted = df.sort_values(«timestamp»)
    timestamps = df_sorted[«timestamp»].unique()
    anomaly_times = df_sorted.loc[df_sorted[«anomaly»] == 1, «timestamp»].unique()
    time_map = {}
    for t in timestamps:
        future_anoms = anomaly_times[anomaly_times >= t]
        if len(future_anoms) == 0:
            tta = np.inf
        else:
            tta = (future_anoms[0] - t).total_seconds()
        time_map[t] = tta

    df[«time_to_anomaly»] = df[«timestamp»].map(time_map)
    df[«AnomalyFutureEvent»] = (df[«time_to_anomaly»] <= time_threshold).astype(int)
    df.to_csv('3. AnomalyFutureEventData.csv', index=False)
    return df

################################################################################
# 4. Custom Binary Events
################################################################################
def create_custom_events(df):
    ««««
    For demonstration: Device_1/Metric_1 > 25, Device_2/Metric_3 < 10, etc.
    plus 'AnomalyNow' = current anomaly.
    ««««
    df[«Dev1_M1_Above25»] = df.apply(lambda r: (
            r[«device»] == «Device_1» and r[«metric»] == «Metric_1» and r[«value»] > 25
    ), axis=1).astype(int)
```

```
    df[«Dev2_M3_Below10»] = df.apply(lambda r: (
        r[«device»] == «Device_2» and r[«metric»] == «Metric_3» and r[«value»] < 10
    ), axis=1).astype(int)

    df[«Dev3_M2_Above20»] = df.apply(lambda r: (
        r[«device»] == «Device_3» and r[«metric»] == «Metric_2» and r[«value»] > 20
    ), axis=1).astype(int)

    df[«AnomalyNow»] = df[«anomaly»]
    df.to_csv('4. CustomEventsData.csv', index=False)
    return df

################################################################################
# 5. Pivot for Apriori
################################################################################
def pivot_for_apriori(df):
    bin_cols = [c for c in df.columns if c.startswith(«Dev») or c.startswith(«Anomaly»)]
    pivot_data = df.groupby(«timestamp»)[bin_cols].max().reset_index()
    pivot_data.to_csv('5. DataForApriori.csv', index=False)
    return pivot_data

################################################################################
# 6. Apriori
################################################################################
def run_apriori(pivot_data, support, confidence, lift=0, leverage=0, conviction=0, zhangs_metric=0, jaccard=0, certainty=0,
kulczynski=0):
    data_for_apriori = pivot_data.drop(columns=[«timestamp»]).astype(bool)
    freq_itemsets = apriori(data_for_apriori, min_support=support, use_colnames=True)
    if freq_itemsets.empty:
        return pd.DataFrame()

    rules_ = association_rules(
        freq_itemsets,
```

```
        metric=«confidence»,
        min_threshold=confidence,
        num_itemsets=len(freq_itemsets)
    )

    if lift > 0:
        rules_ = rules_[rules_[«lift»] >= lift]
    if leverage > 0:
        rules_ = rules_[rules_[«leverage»] >= leverage]
    if conviction > 0:
        rules_ = rules_[rules_[«conviction»] >= conviction]
    if zhangs_metric > 0:
        rules_ = rules_[rules_[«zhangs_metric»] >= zhangs_metric]
    if jaccard > 0:
        rules_ = rules_[rules_[«jaccard»] >= jaccard]
    if certainty > 0:
        rules_ = rules_[rules_[«certainty»] >= certainty]
    if kulczynski > 0:
        rules_ = rules_[rules_[«kulczynski»] >= kulczynski]

    return rules_.sort_values(«confidence», ascending=False)

################################################################################
# 7. Description Map for Rule Items
################################################################################
def create_description_map(time_threshold=None):
    if time_threshold is not None:
        anomaly_future_label = f»AnomalyFutureEvent({time_threshold}s)»
    else:
        anomaly_future_label = «AnomalyFutureEvent»

    desc_map = {
        «Dev1_M1_Above25»: «Device_1/Metric_1 > 25»,
        «Dev2_M3_Below10»: «Device_2/Metric_3 < 10»,
```

```
        «Dev3_M2_Above20»: «Device_3/Metric_2 > 20»,
        «AnomalyNow»: «Current Anomaly»,
        «AnomalyFutureEvent»: anomaly_future_label,
    }
    return desc_map

def rename_itemset(itemset, desc_map):
    new_set = []
    for elem in itemset:
        if elem in desc_map:
            new_set.append(desc_map[elem])
        else:
            new_set.append(elem)
    return set(new_set)

def transform_rules_to_human(rules_df, desc_map):
    if rules_df.empty:
        return rules_df

    copy_ = rules_df.copy()

    def rename_fs(fs):
        return rename_itemset(fs, desc_map)

    def itemset_str(s):
        return « & «.join(sorted(list(s)))

    copy_[«antecedents»] = copy_[«antecedents»].apply(rename_fs)
    copy_[«consequents»] = copy_[«consequents»].apply(rename_fs)

    rule_texts = []
    for i, row in copy_.iterrows():
        left_ = itemset_str(row[«antecedents»])
        right_ = itemset_str(row[«consequents»])
```

```
        c_ = row[«confidence»]
        s_ = row[«support»]
        txt = f»If [{left_}] => [{right_}] (conf={c_:.3f}, sup={s_:.3f})»
        rule_texts.append(txt)

    copy_[«rule_text»] = rule_texts
    return copy_

def backend(support, confidence, eps, min_samples, metric, metric_params, algorithm, leaf_size, p, n_jobs, window):
    # 8.1 Generate data
    # if df_raw.empty:
    df_raw = generate_synthetic_data()

    # 8.2 DBSCAN => cluster => anomaly
    df_dbscan = dbscan_anomaly(df_raw, eps=eps, min_samples=min_samples, metric=metric,
                               metric_params=metric_params, algorithm=algorithm,
                               leaf_size=leaf_size, p=p, n_jobs=n_jobs)
    df_dbscan = relabel_small_clusters_as_anomaly(df_dbscan, min_cluster_size=5)

    # 8.3 Convert all non--1 clusters to 0 => only two clusters
    df_dbscan = two_cluster_labels(df_dbscan)
    df_dbscan = assign_two_names(df_dbscan)

    # 8.4 Save wide format input (time/device/cluster, metrics)
    df_wide = df_dbscan.pivot_table(
        index=[«timestamp», «device», «cluster», «ClusterName»],
        columns=«metric»,
        values=«value»
    ).reset_index()

    # 8.5 Loop over different windows for future anomaly
    windows = [1, 3, 5, 10, 25, 50, 100, 250, 500, 1000, 5000]
    all_rules_final = pd.DataFrame()
```

```
for w in windows:
    df_temp = df_dbscan.copy()
    df_temp = mark_time_to_anomaly(df_temp, w)
    df_temp = create_custom_events(df_temp)

    pivot_data = pivot_for_apriori(df_temp)
    rules_ = run_apriori(pivot_data, support, confidence)
    if rules_.empty:
        print(f»[window={w}] No rules found.»)
        continue

    dmap = create_description_map(time_threshold=w)
    rules_human = transform_rules_to_human(rules_, dmap)
    if rules_human.empty:
        print(f»[window={w}] All rules empty after transformation.»)
        continue

    def has_anomaly(s):
        return any(«Anomaly» in x for x in s)

    mask = (rules_human[«antecedents»].apply(has_anomaly) |
            rules_human[«consequents»].apply(has_anomaly))
    rules_anomaly = rules_human[mask].copy()
    rules_anomaly[«time_window»] = w

    if rules_anomaly.empty:
        print(f»[window={w}] Some rules, but none with anomaly.»)
        continue

    if all_rules_final.empty:
        all_rules_final = rules_anomaly
    else:
        all_rules_final = pd.concat([all_rules_final, rules_anomaly], ignore_index=True)
```

```
        print(f»[window={w}] => {len(rules_anomaly)} anomaly rules»)

    if all_rules_final.empty:
        print(«No anomaly-related rules in any window!»)
    else:
        all_rules_final.to_csv(«7. AssociationRules.csv», index=False)
        print(f»\nTotal {len(all_rules_final)} rules saved to 'anomaly_rules.csv'.»)

class ModernGUI(QMainWindow):
    def __init__(self):
        super().__init__()

        self.setWindowIcon(QIcon(«ruleEngineIcon.png»))

        self.setWindowTitle(«RuleEngine»)
        # Встановлення розміру вікна
        window_width = 1200
        window_height = 500

        # Встановлення положення по центру
        self.center_window(window_width, window_height)

        self.tabs = QTabWidget()
        self.setCentralWidget(self.tabs)

        # Create tabs
        self.param_tab = QWidget()
        self.graph_tab = QWidget()

        self.tabs.addTab(self.param_tab, «Параметри»)
        self.tabs.addTab(self.graph_tab, «Графіки»)

        # Initialize tabs
        self.init_param_tab()
```

```
    self.init_graph_tab()

def center_window(self, width, height):
    # Отримання геометрії доступного екрана
    screen = QGuiApplication.primaryScreen()
    screen_geometry = screen.availableGeometry()

    # Розрахунок координат для центрування
    x = (screen_geometry.width() - width) // 2
    y = (screen_geometry.height() - height-125) // 2

    # Встановлення геометрії вікна
    self.setGeometry(x, y, width, height)

def get_parameters(self):
    ««««
    Зчитує значення параметрів із GUI.
    ««««
    params = {
        # «df»: None,  # або завантажений DataFrame
        «support»: float(self.support_input.text()),
        «confidence»: float(self.confidence_input.text()),
        «eps»: float(self.eps_input.text()),
        «min_samples»: int(self.min_samples_input.text()),
        «metric»: self.metric.text(),
        «metric_params»: None if self.metric_params.text() == «None» else self.metric_params.text(),
        «algorithm»: self.algorithm.text(),
        «leaf_size»: int(self.leaf_size.text()),
        «p»: int(self.p.text()),
        «n_jobs»: None if self.n_jobs.text() == «None» else int(self.n_jobs.text()),
        «window»: int(self.window_input.text()),
    }
    return params
```

```
def run_analysis(self):
    ««««
    Викликає backend із параметрами, отриманими з GUI.
    ««««
    params = self.get_parameters()
    backend(**params)

    self.graph_paths = self.generate_graph_data()  # Оновлюємо шляхи графіків
    self.current_graph_index = 0  # Скидаємо індекс графіка
    self.update_graph()
    print(«Аналіз завершено.»)

def init_param_tab(self):
    layout = QVBoxLayout()

    # Add top margin
    layout.setContentsMargins(10, 20, 10, 10)
    layout.setAlignment(Qt.AlignmentFlag.AlignTop | Qt.AlignmentFlag.AlignLeft)

    # Input file
    file_layout = QHBoxLayout()
    self.file_label = QLabel(«Вхідний файл:»)
    self.file_label.setFont(QFont(«Arial», 12))
    self.file_input = QLineEdit()
    self.file_input.setPlaceholderText(«Обрати файл CSV»)
    self.file_input.setStyleSheet(«padding-left: 5px;»)
    self.file_browse = QPushButton(«...»)
    self.file_browse.clicked.connect(self.browse_file)
    file_layout.addWidget(self.file_label)
    file_layout.addWidget(self.file_input)
    file_layout.addWidget(self.file_browse)
    layout.addLayout(file_layout)

    layout.addSpacing(20)
```

```
# Parameters layout
param_layout = QHBoxLayout()

# Left column with labels and inputs
left_column_layout = QVBoxLayout()
param_label = QLabel(«Параметри аналізу:»)
param_label.setFont(QFont(«Arial», 12))
left_column_layout.addWidget(param_label)

self.support_container, self.support_input = self.create_line_edit_with_label(«Apriori support», «0.001»)
self.confidence_container, self.confidence_input = self.create_line_edit_with_label(«Apriori confidence»,
                                                                                     «0.001»)
self.lift_container, self.lift_input = self.create_line_edit_with_label(«Apriori lift», «0»)
self.representativity_container, self.representativity_input = self.create_line_edit_with_label(
    «Apriori representativity», «0»)
self.leverage_container, self.leverage_input = self.create_line_edit_with_label(«Apriori leverage», «0»)
self.conviction_container, self.conviction_input = self.create_line_edit_with_label(«Apriori conviction», «0»)
self.zhangs_metric_container, self.zhangs_metric_input = self.create_line_edit_with_label(
    «Apriori zhangs_metric», «0»)
self.jaccard_container, self.jaccard_input = self.create_line_edit_with_label(«Apriori jaccard», «0»)
self.certainty_container, self.certainty_input = self.create_line_edit_with_label(«Apriori certainty», «0»)
self.kulczynski_container, self.kulczynski_input = self.create_line_edit_with_label(«Apriori kulczynski», «0»)
self.window_container, self.window_input = self.create_line_edit_with_label(«Ширина вікна:», «50»)
self.eps_input_container, self.eps_input = self.create_line_edit_with_label(«DBSCAN eps:», «0.05»)
self.min_samples_container, self.min_samples_input = self.create_line_edit_with_label(«DBSCAN min_samples:»,
                                                                                       «5»)
self.metric_container, self.metric = self.create_line_edit_with_label(«DBSCAN metric:», «euclidean»)
self.metric_params_container, self.metric_params = self.create_line_edit_with_label(«DBSCAN metric_params:»,
                                                                                     «None»)
self.algorithm_container, self.algorithm = self.create_line_edit_with_label(«DBSCAN algorithm:», «auto»)
self.leaf_size_container, self.leaf_size = self.create_line_edit_with_label(«DBSCAN leaf_size:», «30»)
self.p_container, self.p = self.create_line_edit_with_label(«DBSCAN p:», «2»)
self.n_jobs_container, self.n_jobs = self.create_line_edit_with_label(«DBSCAN n_jobs:», «None»)
```

```
left_column_layout.addWidget(self.support_container)
left_column_layout.addWidget(self.confidence_container)
left_column_layout.addWidget(self.lift_container)
left_column_layout.addWidget(self.representativity_container)
left_column_layout.addWidget(self.leverage_container)
left_column_layout.addWidget(self.conviction_container)
left_column_layout.addWidget(self.zhangs_metric_container)
left_column_layout.addWidget(self.jaccard_container)
left_column_layout.addWidget(self.certainty_container)
left_column_layout.addWidget(self.kulczynski_container)
left_column_layout.addWidget(self.window_container)
left_column_layout.addWidget(self.eps_input_container)
left_column_layout.addWidget(self.min_samples_container)
left_column_layout.addWidget(self.metric_container)
left_column_layout.addWidget(self.metric_params_container)
left_column_layout.addWidget(self.algorithm_container)
left_column_layout.addWidget(self.leaf_size_container)
left_column_layout.addWidget(self.p_container)
left_column_layout.addWidget(self.n_jobs_container)

# Add logo to the right column
right_column_layout = QVBoxLayout()
logo_label = QLabel()
logo_label.setPixmap(QPixmap(«RuleEngineLogo.png»).scaled(500, 500, Qt.AspectRatioMode.KeepAspectRatio))
logo_label.setAlignment(Qt.AlignmentFlag.AlignCenter)
right_column_layout.addStretch()
right_column_layout.addWidget(logo_label)
right_column_layout.addStretch()

# Combine left and right columns
param_layout.addLayout(left_column_layout)
param_layout.addLayout(right_column_layout)
layout.addLayout(param_layout)
```

```
    # Add spacing above the button
    layout.addSpacing(20)

    # Run button
    self.run_button = QPushButton(«Запустити аналіз»)
    self.run_button.setFont(QFont(«Arial», 14))

    # Progress bar
    self.progress_bar = QProgressBar()
    self.progress_bar.setValue(0)

    layout.addWidget(self.run_button)
    layout.addWidget(self.progress_bar)
    self.param_tab.setLayout(layout)

def init_graph_tab(self):
    layout = QVBoxLayout()

    # Graph viewer
    self.graph_canvas = FigureCanvas(plt.figure())
    size_policy = self.graph_canvas.sizePolicy()
    size_policy.setHorizontalPolicy(QSizePolicy.Policy.Expanding)
    size_policy.setVerticalPolicy(QSizePolicy.Policy.Expanding)
    self.graph_canvas.setSizePolicy(size_policy)
    layout.addWidget(self.graph_canvas)

    # Navigation buttons
    nav_layout = QHBoxLayout()
    self.prev_button = QPushButton(«Попередній»)
    self.prev_button.clicked.connect(self.show_previous_graph)
    self.next_button = QPushButton(«Наступний»)
    self.next_button.clicked.connect(self.show_next_graph)
```

```
    nav_layout.addWidget(self.prev_button)
    nav_layout.addWidget(self.next_button)

    layout.addLayout(nav_layout)
    self.graph_tab.setLayout(layout)

    # Створення даних для графіків
    self.graph_paths = []
    self.current_graph_index = 0
    self.update_graph()

def generate_graph_data(self):
    # Створення папки для графіків, якщо вона не існує
    charts_folder = «Charts»
    os.makedirs(charts_folder, exist_ok=True)

    # Завантаження даних для показників
    df = pd.read_csv('3. AnomalyFutureEventData.csv')  # Файл із даними про майбутні аномалії

    # Перетворення 'timestamp' у формат datetime
    if not pd.api.types.is_datetime64_any_dtype(df[«timestamp»]):
        df[«timestamp»] = pd.to_datetime(df[«timestamp»])

    # Конвертація значень 'value' і 'time_to_anomaly' у числовий формат
    df[«value»] = pd.to_numeric(df[«value»], errors=«coerce»)
    df[«time_to_anomaly»] = pd.to_numeric(df[«time_to_anomaly»], errors=«coerce»)

    # Видалення рядків із NaN
    df = df.dropna(subset=[«value», «timestamp», «time_to_anomaly»])

    # Список шляхів до збережених графіків
    file_paths = []

    # Визначення кольорів для кожного пристрою
```

```
unique_devices = df[«device»].unique()
colors = plt.cm.tab10.colors  # Використовуємо кольорову карту 'tab10'
device_colors = {device: colors[i % len(colors)] for i, device in enumerate(unique_devices)}

# Генерація графіків для кожного пристрою та метрики
for device in unique_devices:
    device_data = df[df[«device»] == device]

    for metric in device_data[«metric»].unique():
        metric_data = device_data[device_data[«metric»] == metric]

        # Створення графіка залежності показника від часу з аномаліями
        fig, ax = plt.subplots(figsize=(18, 12))
        ax.plot(metric_data[«timestamp»], metric_data[«value»], label=f»{device} - {metric}»,
                color=device_colors[device])

        # Позначення аномалій
        anomalies = metric_data[metric_data[«anomaly»] == 1]
        ax.scatter(anomalies[«timestamp»], anomalies[«value»], color=«red», label=«Аномалії», zorder=3)

        # Налаштування графіка
        ax.set_title(f»{metric} vs Time for {device}», fontsize=14)
        ax.set_xlabel(«Час», fontsize=12)
        ax.set_ylabel(«Показник», fontsize=12)
        ax.legend()
        ax.grid(True, linestyle=«--», alpha=0.6)
        plt.xticks(rotation=45)

        # Збереження графіка в папку
        file_path = os.path.join(charts_folder, f»{device}_{metric}_vs_time.png»)
        fig.savefig(file_path, dpi=300)
        plt.close(fig)  # Закриваємо графік після збереження
        file_paths.append(file_path)
```

```
        # Генерація агрегованого графіка time_to_anomaly vs time для кожного пристрою
        fig, ax = plt.subplots(figsize=(18, 12))
        ax.plot(device_data[«timestamp»], device_data[«time_to_anomaly»], label=f»{device} - Time to Anomaly»,
                color=device_colors[device], linewidth=2)

        # Налаштування графіка
        ax.set_title(f»Time to Anomaly vs Time for {device}», fontsize=14)
        ax.set_xlabel(«Час», fontsize=12)
        ax.set_ylabel(«Час до аномалії (секунди)», fontsize=12)
        ax.legend()
        ax.grid(True, linestyle=«--», alpha=0.6)
        plt.xticks(rotation=45)

        # Збереження графіка в папку
        file_path = os.path.join(charts_folder, f»{device}_time_to_anomaly_vs_time.png»)
        fig.savefig(file_path, dpi=300)
        plt.close(fig)  # Закриваємо графік після збереження
        file_paths.append(file_path)

    return file_paths

def update_graph(self):
    if self.graph_paths and 0 <= self.current_graph_index < len(self.graph_paths):
        file_path = self.graph_paths[self.current_graph_index]

        # Очищення поточного графіка
        self.graph_canvas.figure.clear()

        # Завантаження графіка
        ax = self.graph_canvas.figure.add_subplot(111)
        img = plt.imread(file_path)
        ax.imshow(img, aspect=«auto»)  # Автоматичне масштабування
        ax.axis(«off»)  # Вимкнення осей
```

```
        self.graph_canvas.figure.tight_layout()  # Оптимізація компонування
        self.graph_canvas.draw()

def resizeEvent(self, event):
    ««««
    Викликається під час зміни розміру вікна.
    Оновлює графік для відповідності новому розміру.
    ««««
    self.update_graph()
    super().resizeEvent(event)

def browse_file(self):
    file_path, _ = QFileDialog.getOpenFileName(self, «Обрати CSV файл», ««, «CSV Files (*.csv)»)
    if file_path:
        self.file_input.setText(file_path)

def create_slider_with_spinbox(self, label_text, min_value, max_value, step, default_value):
    layout = QHBoxLayout()
    label = QLabel(f»{label_text}:»)
    slider = QSlider(Qt.Orientation.Horizontal)
    slider.setMinimum(int(min_value * 1000))
    slider.setMaximum(int(max_value * 1000))
    slider.setValue(int(default_value * 1000))
    slider.setTickInterval(int(step * 1000))
    slider.setTickPosition(QSlider.TickPosition.TicksBelow)

    spinbox = QSpinBox()
    spinbox.setRange(int(min_value * 1000), int(max_value * 1000))
    spinbox.setSingleStep(int(step * 1000))
    spinbox.setValue(int(default_value * 1000))

    slider.valueChanged.connect(lambda v: spinbox.setValue(v))
    spinbox.valueChanged.connect(lambda v: slider.setValue(v))
```

```
    layout.addWidget(label)
    layout.addWidget(slider)
    layout.addWidget(spinbox)
    return layout

def save_graphs_to_folder(self):
    # Створення папки «Charts», якщо вона не існує
    charts_folder = «Charts»
    if not os.path.exists(charts_folder):
        os.makedirs(charts_folder)

    # Завантаження даних для графіків
    self.graph_data = self.generate_graph_data()

    # Збереження кожного графіка у папку
    for i, fig in enumerate(self.graph_data):
        file_path = os.path.join(charts_folder, f»chart_{i + 1}.png»)
        try:
            fig.savefig(file_path)
            plt.close(fig)  # Закриття графіка після збереження
            print(f»Графік {i + 1} збережено як {file_path}»)
        except Exception as e:
            print(f»Помилка збереження графіка {i + 1}: {e}»)

    print(«Усі графіки збережено у папку Charts.»)

def create_line_edit_with_label(self, label_text, default_value):
    # Створюємо контейнерний віджет
    container = QWidget()
    layout = QHBoxLayout(container)
    layout.setContentsMargins(0, 0, 0, 0)  # Мінімальні відступи
    layout.setSpacing(5)  # Відстань між лейблом і текстовим полем
    layout.setAlignment(Qt.AlignmentFlag.AlignLeft)  # Вирівнювання по лівому краю
```

```
        # Створюємо лейбл
        label = QLabel(label_text)
        label.setAlignment(Qt.AlignmentFlag.AlignLeft)
        label.setFixedWidth(150)  # Фіксована ширина лейбла для вирівнювання

        # Створюємо текстове поле
        line_edit = QLineEdit()
        line_edit.setText(default_value)
        line_edit.setFixedWidth(80)  # Фіксована ширина текстового поля
        line_edit.setStyleSheet(«padding-left: 5px; text-align: left;»)  # Вирівнювання тексту по лівому краю

        # Додаємо віджети до макета
        layout.addWidget(label)
        layout.addWidget(line_edit)

        # Встановлення політики розміру для контейнера
        container.setSizePolicy(QSizePolicy.Policy.Fixed, QSizePolicy.Policy.Fixed)
        return container, line_edit
    def show_previous_graph(self):
        if self.current_graph_index > 0:
            self.current_graph_index -= 1
            self.update_graph()
    def show_next_graph(self):
        if self.current_graph_index < len(self.graph_paths) - 1:
            self.current_graph_index += 1
            self.update_graph()
if __name__ == «__main__»:
    app = QApplication(sys.argv)
    gui = ModernGUI()

    gui.run_button.clicked.connect(gui.run_analysis)

    gui.show()
    sys.exit(app.exec())
```

№	Позначення	Найменування	Дод. відомості
		Текстові документи	
1	ГЮІК. 508100.010 ПЗ	Пояснювальна записка	73 с., ф. А4
		Графічні документи	
		Презентація	у форматі
			*.ppt

					ГЮІК. 508100.010 ВД		
Змін.	Арк.	№ документа	Підп.	Дата			
Розроб.		Іщенко М.Д.				Літ.	Аркуш	Аркушів
Перевір.		Максимова С.С.					1	1
Н. контр.		Бронніков А.І.				ХНУРЕ		
Затвер.		Невлюдов І.Ш				Кафедра КІТАР		