

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

До захисту допущено:

В. о. завідувач кафедри

Артем ВОЛОКИТА

(підпис)

“ ” 2026 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою “Комп’ютерні системи та мережі”

спеціальності 123 “Комп’ютерна інженерія”

на тему: Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання

Виконав: студент 4 курсу, групи ІО-24
(шифр групи)

Бородій Володимир Юрійович

(прізвище, ім’я, по батькові)

(підпис)

Керівник асистент Меленчуков М. Є.

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Консультант (нормоконтроль) ст. викл., д. ф. Міщенко Л. Д.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

Рецензент

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2026 р.

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”**

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Рівень вищої освіти – перший (бакалавр)

Освітньо-професійна програма

“Комп’ютерні системи та мережі”

спеціальності 123 “Комп’ютерна інженерія”

ЗАТВЕРДЖУЮ

В. о. Завідувач кафедри

Артем ВОЛОКИТА

(підпис)

“ ” _____ 2026 р.

ЗАВДАННЯ

на бакалаврський дипломний проєкт студента

Бородій Володимир Юрійович

1. Тема проєкту Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання
керівник проєкту Меленчуков Микита Євгенович, асистент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
- затверджені наказом по університету від 3 червня 2026 року №2055-с
2. Термін здачі студентом закінченого проєкту _____
3. Вихідні дані до проєкту див. технічне завдання
4. Зміст розрахунково-пояснювальної записки: Аналіз предметної області та методів виявлення аномалій. Обґрунтування вибору технологій розробки. Реалізація програмного комплексу виявлення аномалій. Аналіз результатів роботи програмного комплексу.
5. Перелік графічного матеріалу (назви креслень з точним позначенням обов’язкових креслень) Діаграма архітектури автоенкодера. Схема обробки сенсорних даних. Архітектурна схема програмного комплексу. Матриці помилок моделей. Графіки результатів тестування моделей.

6. Консультанта проєкту, з вказівкою розділів проєкту, які до них вносяться

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Нормоконтроль	Міщенко Л. Д.		

7. Дата видачі завдання 20 листопада 2025 р.

Календарний план

№ п/п	Найменування етапів дипломного проєкту	Терміни виконання етапів проєкту	Примітки
1.	<i>Затвердження теми проєкту</i>	<i>15.12.2025-31.12.2025</i>	
2.	<i>Аналіз предметної області та постановка задачі виявлення аномалій у сенсорних системах</i>	<i>01.01.2026-24.01.2026</i>	
3.	<i>Дослідження існуючих методів anomaly detection та алгоритмів машинного навчання</i>	<i>25.01.2026–18.02.2026</i>	
4.	<i>Проектування архітектури програмного комплексу та моделі Autoencoder</i>	<i>19.02.2026–12.03.2026</i>	
5.	<i>Реалізація програмного комплексу виявлення аномалій у сенсорних даних</i>	<i>13.03.2026–23.05.2026</i>	
6.	<i>Оформлення пояснювальної записки</i>	<i>24.05.2026-31.05.2026</i>	
7.	<i>Проходження нормоконтролю</i>	<i>01.06.2026-06.06.2026</i>	
8.	<i>Захист</i>	<i>20.06.2026</i>	

Студент-дипломник _____ Бородій В. Ю.
(підпис)

Керівник проєкту _____ Меленчуков М. Є.
(підпис)

АНОТАЦІЯ

Було розроблено програмний комплекс для виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання. Реалізовано систему обробки сенсорних даних, виконано очищення та нормалізацію даних, а також побудовано моделі для автоматизованого виявлення аномальних станів. Для аналізу даних використано алгоритми Isolation Forest, Logistic Regression та нейромережеву модель Autoencoder..

Система забезпечує аналіз сенсорних даних, виявлення аномалій та візуалізацію результатів роботи моделей машинного навчання. Для оцінки ефективності моделей використано метрики Accuracy, Precision, Recall та F1-score. Проведено порівняння базової моделі та автоенкодера, що дозволило оцінити ефективність різних підходів до anomaly detection.

Ключові слова: anomaly detection, автоенкодер, машинне навчання, сенсорні системи, нейронні мережі, Isolation Forest.

ANNOTATION

A software system for anomaly detection in sensor system data using machine learning algorithms was developed. A sensor data processing pipeline was implemented, including data cleaning, normalization, and models for automated anomaly detection. Isolation Forest, Logistic Regression, and a neural network Autoencoder model were used for data analysis..

The system provides sensor data analysis, anomaly detection, and visualization of machine learning model results. Accuracy, Precision, Recall, and F1-score metrics were used to evaluate model performance. A comparison between the baseline model and the Autoencoder was conducted to evaluate the effectiveness of different anomaly detection approaches.

Keywords: anomaly detection, autoencoder, machine learning, sensor systems, neural networks, Isolation Forest.

справки	Формат	Значення			Найменування	Кіл. листів	№ екзем- пляра	Додаток
					Документація загальна			
					Знову розроблена			
	A4	ІАЛЦ.467200.002 ТЗ			Програмний комплекс виявлення	4		
					аномалій у даних сенсорних систем із			
					використанням алгоритмів			
					машинного навчання			
					Технічне завдання			
	A4	ІАЛЦ.467200.003 ПЗ			Програмний комплекс виявлення	73		
					аномалій у даних сенсорних систем із			
					використанням алгоритмів			
					машинного навчання			
					Пояснювальна записка			
	A4	ІАЛЦ.467200.004 ДІ			Програмний комплекс виявлення	1		
					аномалій у даних сенсорних систем із			
					використанням алгоритмів			
					машинного навчання			
					Архітектурна схема системи			
					(Принципова схема)			
	A4	ІАЛЦ. 467200.005 Д2			Програмний комплекс виявлення	1		
					аномалій у даних сенсорних систем із			
					використанням алгоритмів			
					машинного навчання			
					Структура програмного комплексу			
					виявлення аномалій			
					(Структурна схема)			
	A4	ІАЛЦ.467200.006 ДЗ			Програмний комплекс виявлення	1		
					аномалій у даних сенсорних систем із			
					використанням алгоритмів			
					машинного навчання			
					Алгоритм роботи програмного			
					комплексу виявлення аномалій			
					(Функціональна схема)			
	A4	ІАЛЦ.467200.007 Д4			Текст програмного коду	11		
					ІАЛЦ.467200.001 ОА			
Зм	Лист	№ докум.	Підп	Дата	Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання Опис альбому	Літ.	Аркуш	Аркушів
Розроб		Бородій В. Ю.					1	1
Перев		Меленчуков М. Є.						
Реценз.						КПІ ім. Ігоря Сікорського, ФІОТ, ІО-24		
Н. контр.		Мищенко Л. Д.						
Затв.		Волокита А. М.						

ТЕХНІЧНЕ ЗАВДАННЯ ДО ДИПЛОМНОГО ПРОЄКТУ

**на тему: «Програмний комплекс виявлення аномалій у
даних сенсорних систем із використанням алгоритмів
машинного навчання»**

Київ – 2026 р.

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ.....	2
2. ПІДСТАВИ ДЛЯ РОЗРОБКИ.....	2
3. МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ	2
4. ДЖЕРЕЛА РОЗРОБКИ	3
5. ТЕХНІЧНІ ВИМОГИ.....	3
5.1. Вимоги до продукту	3
5.2. Вимоги до програмного забезпечення	3
5.3. Вимоги до апаратної частини.....	4
6. ЕТАПИ РОЗРОБКИ	4

					ІАЛЦ.467200.002 ТЗ			
Зм.	Арк.	№ докум.	Підпис	Дата				
Розробив		Бородій В. Ю.			Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання Технічне завдання	Літ.	Аркуш	Аркушів
Перевірив		Меленчуков М. Є.					1	4
Реценз.						КПІ ім. Ігоря Сікорського, ФІОТ, ІО-24		
Н. Контр.		Міщенко Л. Д.						
Затвердив		Волокита А. М.						

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

Технічне завдання поширюється на розробку програмного комплексу виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання. Система призначена для аналізу сенсорних даних, автоматизованого виявлення аномальних станів та оцінки технічного стану обладнання.

Область застосування: промислові сенсорні системи, системи моніторингу технічного стану обладнання та системи anomaly detection.

2. ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки системи є завдання на виконання бакалаврського дипломного проєкту за спеціальністю 123 «Комп'ютерна інженерія» освітньо-професійної програми «Комп'ютерні системи та мережі». Розробка виконується відповідно до теми дипломного проєкту «Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання».

3. МЕТА ТА ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою розробки є підвищення ефективності виявлення аномалій у сенсорних системах шляхом використання алгоритмів машинного навчання та нейронних мереж. Система дозволяє автоматизувати аналіз сенсорних даних, виявляти аномальні стани обладнання та оцінювати ефективність моделей anomaly detection.

					ІАЛЦ.467200.002 ТЗ	Арк.
						2
Зм.	Арк.	№ докум.	Підпис	Дата		

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелом розробки є науково-технічна література, документація бібліотек машинного навчання, електронні публікації та матеріали, пов'язані з anomaly detection, нейронними мережами та аналізом сенсорних даних.

5. ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до продукту

- Забезпечувати обробку сенсорних даних;
- Виконувати виявлення аномалій у даних;
- Підтримувати роботу моделей машинного навчання;
- Забезпечувати візуалізацію результатів роботи моделей;
- Проводити аналіз ефективності алгоритмів anomaly detection.

5.2. Вимоги до програмного забезпечення

- Операційна система Windows, MacOS чи Linux;
- Python 3.12 або новіше;
- TensorFlow;
- Scikit-learn;
- Pandas, NumPy, Matplotlib, Seaborn;
- Середовище розробки PyCharm або аналогічне.

					ІАЛЦ.467200.002 ТЗ	Арк.
						3
Зм.	Арк.	№ докум.	Підпис	Дата		

5.3. Вимоги до апаратної частини

- Оперативної пам'яті не менше 8 GB;
- Процесор не нижче Intel Core i5 або аналогічний;
- Наявність стабільного підключення до мережі Інтернет.

6. ЕТАПИ РОЗРОБКИ

	Дата
Затвердження теми дипломного проєкту	15.12.2025-31.12.2025
Аналіз предметної області та постановка задачі виявлення аномалій у сенсорних системах	01.01.2026-24.01.2026
Дослідження існуючих методів anomaly detection та вибір алгоритмів машинного навчання	25.01.2026–18.02.2026
Проектування архітектури програмного комплексу та моделі Autoencoder	19.02.2026–12.03.2026
Підготовка сенсорного датасету та реалізація попередньої обробки даних	13.03.2026–18.04.2026
Реалізація базової моделі та нейромережевого автоенкодера	19.04.2026–10.05.2026
Тестування моделей, оцінювання ефективності та аналіз результатів	11.05.2026–23.05.2026
Оформлення пояснювальної записки та підготовка дипломної роботи	24.05.2026-31.05.2026

					ІАЛЦ.467200.002 ТЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		4

ПОЯСНЮВАЛЬНА ЗАПISKA ДО ДИПЛОМНОГО ПРОЄКТУ

**на тему: «Програмний комплекс виявлення аномалій у
даних сенсорних систем із використанням алгоритмів
машинного навчання»**

Київ – 2026 р.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ.....	3
ВСТУП	4
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА МЕТОДІВ ВИЯВЛЕННЯ АНОМАЛІЙ У СЕНСОРНИХ СИСТЕМАХ.....	5
1.1. Сенсорні системи та їх роль у сучасних інформаційних технологіях	5
1.2. Методи виявлення аномалій у сенсорних даних.....	7
1.3. Автоенкодери як метод anomaly detection	8
1.4. Постановка задачі дослідження	10
ВИСНОВОК ДО РОЗДІЛУ 1	13
РОЗДІЛ 2. ВИБІР ТЕХНОЛОГІЙ ТА ЗАСОБІВ РОЗРОБКИ СИСТЕМИ.....	14
2.1. Формалізація задачі виявлення аномалій.....	14
2.2. Архітектура та принцип роботи автоенкодера.....	17
2.3. Вибір Попередня обробка сенсорних даних.....	21
2.4. Реалізація базової моделі anomaly detection	23
2.5. Реалізація моделі Autoencoder.....	26
2.6. Оцінювання ефективності моделей	32
ВИСНОВОК ДО РОЗДІЛУ 2	36
РОЗДІЛ 3. ДЕТАЛІ РОЗРОБКИ ПРОГРАМНОГО КОМПЛЕКСУ.....	38
3.1. Реалізація програмної частини системи.....	38
3.1.1. Структура програмного комплексу	40
3.1.2. Реалізація обробки даних та навчання моделей.....	41
3.2. Проектування та підготовка набору даних	47
3.2.1. Загальна структура набору даних.....	48
3.2.2. Опис основних ознак набору даних	50
3.3. Реалізація оцінювання та аналіз результатів	52
3.3.1. Оцінювання моделі Logistic Regression	53
3.3.2. Аналіз результатів моделі Logistic Regression	55
3.3.3. Аналіз процесу навчання та помилки реконструкції Autoencoder..	58
ВИСНОВОК ДО РОЗДІЛУ 3	61
РОЗДІЛ 4. ПРЕДСТАВЛЕННЯ РОЗРОБЛЕНОЇ СИСТЕМИ.....	62

					ІАЛЦ.467200.002 ПЗ			
Зм.	Арк.	№ докум.	Підпис	Дата	Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання Пояснювальня записка	Літ.	Аркуш	Аркушів
Розробив	Бородій В. Ю.						1	73
Перевірив	Меленчуков М. Є.							
Реценз.								
Н. Контр.	Міщенко Л. Д.							
Затвердив	Волокита А. М.					КПІ ім. Ігоря Сікорського, ФІОТ, ІО-24		

4.1. Демонстрація роботи розробленої системи	62
4.2. Аналіз результатів роботи системи	64
ВИСНОВОК ДО РОЗДІЛУ 4	69
ВИСНОВКИ.....	70
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	72

					ІАЛЦ.467200.003 ПЗ	Арк.
						2
Зм.	Арк.	№ докум.	Підпис	Дата		

ПЕРЕЛІК СКОРОЧЕНЬ

F1-score	Гармонійне середнє між Precision та Recall
Recall	Метрика повноти виявлення аномалій
CSV	(Comma-Separated Values) формат зберігання табличних даних у текстовому файлі
IoT	(Internet of Things) інтернет речей
IQR	(Interquartile Range) міжквартильний розмах
Keras	Бібліотека для створення та навчання нейронних мереж
MSE	(Mean Squared Error) середньоквадратична помилка
Precision	Метрика точності визначення аномалій
ReLU	(Rectified Linear Unit) функція активації нейронної мережі
Accuracy	Метрика загальної точності моделі

ВСТУП

Сучасний розвиток промислових та інформаційних систем супроводжується зростанням обсягів сенсорних даних, що генеруються різноманітними технічними пристроями та системами моніторингу. У зв'язку з підвищенням рівня автоматизації виробничих процесів виникає необхідність у створенні ефективних засобів контролю та аналізу стану обладнання в режимі реального часу. Одним із важливих завдань у цьому напрямі є своєчасне виявлення аномалій, які можуть свідчити про несправності або нестандартні режими роботи систем.

Традиційні методи аналізу часто не забезпечують достатньої ефективності при роботі з великими обсягами високовимірних сенсорних даних. Саме тому все більшої популярності набувають методи машинного навчання та нейронні мережі, здатні автоматично виявляти приховані закономірності та аномальні відхилення у даних. Особливу увагу привертають автоенкодери — нейронні мережі, що дозволяють визначати аномальні стани на основі похибки реконструкції вхідних даних.

Актуальність теми дослідження обумовлена потребою у створенні ефективних програмних засобів для автоматизованого виявлення аномалій у сенсорних системах. Об'єктом дослідження є процес аналізу сенсорних даних, а предметом — використання алгоритмів машинного навчання та нейронних мереж для задач anomaly detection..

Метою дипломної роботи є розробка програмного комплексу виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання та дослідження ефективності моделей anomaly detection..

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		4

РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА МЕТОДІВ ВИЯВЛЕННЯ АНОМАЛІЙ У СЕНСОРНИХ СИСТЕМАХ

1.1. Сенсорні системи та їх роль у сучасних інформаційних технологіях

Сучасний розвиток інформаційних технологій, автоматизації та цифровізації виробничих процесів супроводжується активним використанням сенсорних систем у різних сферах діяльності людини. Сенсорні системи застосовуються у промисловості, транспорті, медицині, енергетиці, системах безпеки та автоматизованого керування технологічними процесами. Основним завданням таких систем є збір інформації про стан обладнання або навколишнього середовища з подальшою передачею даних до програмних комплексів для аналізу та прийняття рішень.

Сенсори дозволяють вимірювати широкий спектр фізичних параметрів: температуру, вологість, тиск, швидкість, вібрацію, електричні показники та інші характеристики. Отримані значення передаються у цифровому вигляді та можуть аналізуватись у режимі реального часу [1]. Завдяки цьому сенсорні системи забезпечують безперервний моніторинг технічного стану обладнання та дозволяють своєчасно реагувати на відхилення від нормального режиму роботи.

У сучасних умовах обсяги сенсорних даних постійно зростають. Великі промислові комплекси можуть генерувати тисячі показників щосекунди, що створює необхідність використання автоматизованих засобів обробки інформації. Традиційні методи аналізу часто не дозволяють ефективно працювати з багатовимірними даними та виявляти складні закономірності у

					ІАЛЦ.467200.003 ПЗ	Арк.
						5
Зм.	Арк.	№ докум.	Підпис	Дата		

роботі систем. Саме тому важливого значення набувають алгоритми машинного навчання та нейронні мережі.

Однією з основних проблем сенсорних систем є поява аномальних значень або нестандартних режимів функціонування [2]. Такі ситуації можуть бути спричинені технічними несправностями, помилками вимірювання, зношенням обладнання або впливом зовнішніх факторів [3]. Несвоєчасне виявлення аномалій може призвести до зниження ефективності роботи обладнання, втрати даних, аварійних ситуацій та фінансових збитків.

Важливою перевагою сучасних систем моніторингу є можливість інтеграції із технологіями Інтернету речей (IoT), хмарними сервісами та системами штучного інтелекту. Це дозволяє виконувати централізований збір та аналіз інформації, автоматично формувати повідомлення про небезпечні ситуації та прогнозувати потенційні несправності обладнання. Таким чином, сенсорні системи стають невід’ємною складовою сучасних інформаційних технологій та автоматизованих виробничих процесів.

Крім промисловості, сенсорні системи активно використовуються у медичних інформаційних системах, системах «розумного дому», транспортній інфраструктурі та системах екологічного моніторингу. У медичній сфері сенсори застосовуються для контролю стану пацієнтів та збору фізіологічних показників. У транспорті сенсорні технології дозволяють контролювати технічний стан транспортних засобів та аналізувати дорожню ситуацію. Це свідчить про універсальність сенсорних систем та широкий спектр їх практичного застосування.

					ІАЛЦ.467200.003 ПЗ	Арк.
						6
Зм.	Арк.	№ докум.	Підпис	Дата		

1.2. Методи виявлення аномалій у сенсорних даних

Виявлення аномалій є одним із найважливіших напрямів аналізу даних у сучасних інформаційних системах. Аномалія являє собою значення або набір значень, які суттєво відрізняються від нормальної поведінки системи. У сенсорних системах аномальні значення можуть свідчити про несправності обладнання, збої у роботі програмного забезпечення або помилки передачі інформації.

Традиційні методи виявлення аномалій базуються на статистичному аналізі даних. До таких методів належать аналіз середнього значення, стандартного відхилення, міжквартильного розмаху та інші математичні підходи. Основною перевагою статистичних методів є простота реалізації та невисокі вимоги до обчислювальних ресурсів. Проте вони мають обмеження при роботі зі складними високорозмірними даними та не завжди дозволяють ефективно виявляти приховані закономірності.

З розвитком технологій машинного навчання для задач anomaly detection почали активно використовуватися інтелектуальні алгоритми аналізу даних. Одним із популярних підходів є кластеризація, яка дозволяє групувати схожі об'єкти та виявляти значення, що не належать до жодного з кластерів. Також широко застосовуються методи класифікації та алгоритми пошуку викидів.

Для аналізу сенсорних даних ефективно використовуються алгоритми Isolation Forest, One-Class SVM, Random Forest та нейронні мережі [4]. Алгоритм Isolation Forest базується на випадковому поділі даних та дозволяє швидко виявляти аномальні об'єкти без необхідності попереднього маркування даних. Однією з його переваг є висока швидкість роботи та здатність обробляти великі обсяги інформації [5].

У даній дипломній роботі Isolation Forest використовується для формування псевдоміток аномалій. Отримані результати застосовуються для

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		7

навчання базової моделі логістичної регресії та подальшого аналізу ефективності алгоритмів машинного навчання. Такий підхід дозволяє автоматизувати процес виявлення аномальних станів без необхідності створення великого набору розмічених даних [6].

Останніми роками значного поширення набули методи глибокого навчання. Нейронні мережі здатні працювати з великими масивами даних та автоматично знаходити складні залежності між параметрами. Особливо ефективними для задач anomaly detection є автоенкодери, які дозволяють оцінювати ступінь відхилення даних від нормального стану системи на основі похибки реконструкції [7].

1.3. Автоенкодери як метод anomaly detection

Автоенкодер є різновидом нейронної мережі, призначеної для автоматичного навчання компактного представлення вхідних даних. Основною особливістю автоенкодера є здатність відновлювати початкові значення після їх стискання у прихованому просторі. Архітектура моделі складається з двох основних частин: енкодера та декодера [8].

Енкодер виконує зменшення розмірності даних та формує приховане представлення, яке містить найбільш важливі характеристики вхідної інформації. Декодер, у свою чергу, намагається відновити початкові дані із цього компактного представлення. Під час навчання модель мінімізує різницю між вхідними та відновленими даними [9].

Автоенкодери широко використовуються для задач anomaly detection, оскільки вони добре відновлюють лише ті дані, на яких були навчені. Якщо модель навчається на нормальних даних, то для стандартних сигналів похибка реконструкції буде невеликою. У випадку появи аномалій або нестандартних значень похибка суттєво збільшується, що дозволяє використовувати її як

					ІАЛЦ.467200.003 ПЗ	Арк.
						8
Зм.	Арк.	№ докум.	Підпис	Дата		

критерій виявлення відхилень [10].

Перевагою автоенкодерів є можливість роботи без великої кількості аномальних прикладів. У більшості практичних задач дані про несправності є обмеженими або взагалі відсутніми. У такому випадку модель навчається лише на нормальних даних та автоматично визначає нетипові сигнали.

У рамках даної дипломної роботи для реалізації автоенкодера використовується бібліотека TensorFlow та Keras. Модель навчається на сенсорних даних SensorA, SensorB та SensorC після виконання попередньої обробки [11]. Перед навчанням виконувалося заповнення пропущених значень, видалення викидів та нормалізація параметрів за допомогою StandardScaler [12].

Для покращення якості роботи моделі були створені додаткові ознаки: середнє значення сенсорів, стандартне відхилення, мінімальні та максимальні значення, а також різниця між сусідніми показниками [13]. Такий підхід дозволив збільшити інформативність набору даних та підвищити ефективність виявлення аномалій.

Навчання моделі здійснювалося із використанням механізму EarlyStopping, який дозволяє автоматично завершувати навчання при відсутності покращення функції втрат [14]. Це дозволяє уникнути перенавчання моделі та зменшити час виконання алгоритму.

Для оцінки ефективності автоенкодера використовувалися метрики Accuracy, Precision, Recall та F1-score [15]. Також було побудовано матрицю помилок (Confusion Matrix) та графіки реконструктивної помилки [16]. Отримані результати показали, що автоенкодер забезпечує кращу точність виявлення аномалій у порівнянні з базовою моделлю логістичної регресії.

1.4. Постановка задачі дослідження

Стрімкий розвиток інформаційних технологій та систем автоматизації призводить до постійного збільшення кількості сенсорних пристроїв, що використовуються для моніторингу технічного стану обладнання та контролю технологічних процесів. У сучасних сенсорних системах щосекунди генерується значний обсяг даних, який потребує оперативного аналізу та автоматизованої обробки. У зв'язку з цим особливої актуальності набуває задача своєчасного виявлення аномальних станів, які можуть свідчити про несправності, помилки роботи обладнання або відхилення від нормального режиму функціонування системи.

Традиційні методи аналізу сенсорних даних часто не забезпечують достатньої ефективності при роботі з великими обсягами високотовірної інформації [17]. Крім того, використання фіксованих порогових значень або експертних правил не дозволяє враховувати складні залежності між параметрами сенсорів та адаптуватися до змін у роботі системи. Саме тому виникає необхідність застосування сучасних алгоритмів машинного навчання та нейронних мереж для автоматизованого виявлення аномалій.

Одним із найбільш перспективних підходів до вирішення задач anomaly detection є використання автоенкодерів — нейронних мереж, здатних навчатися прихованому представленню даних та визначати аномальні стани на основі похибки реконструкції. Використання автоенкодерів дозволяє аналізувати складні багатовимірні сенсорні дані та автоматично виявляти нетипові значення без необхідності створення великого набору розмічених аномальних прикладів.

У рамках дипломної роботи планується розробка програмного комплексу для автоматизованого виявлення аномалій у даних сенсорних систем. Для реалізації системи використовуються алгоритми машинного навчання Isolation Forest, Logistic Regression та нейромережевий Autoencoder [18]. Перед

					ІАЛЦ.467200.003 ПЗ	Арк.
						10
Зм.	Арк.	№ докум.	Підпис	Дата		

навчанням моделей виконується попередня обробка даних, що включає очищення набору даних, заповнення пропущених значень, видалення викидів та нормалізацію параметрів [19]. Також виконується створення додаткових ознак для підвищення ефективності моделей anomaly detection.

Основною метою роботи є підвищення точності та ефективності автоматизованого виявлення аномалій у сенсорних системах шляхом використання сучасних алгоритмів машинного навчання. Для досягнення поставленої мети необхідно вирішити такі задачі:

- провести аналіз предметної області та існуючих методів виявлення аномалій;
- дослідити сучасні алгоритми машинного навчання для аналізу сенсорних даних;
- виконати попередню обробку та підготовку набору даних;
- реалізувати базову модель anomaly detection;
- розробити нейромережеву модель Autoencoder;
- провести навчання та тестування моделей;
- оцінити ефективність моделей за допомогою метрик Accuracy, Precision, Recall та F1-score;
- виконати порівняльний аналіз отриманих результатів;
- побудувати графічне відображення результатів роботи моделей.

Об'єктом дослідження є процес аналізу сенсорних даних у системах моніторингу технічного стану обладнання.

Предметом дослідження є використання алгоритмів машинного навчання та нейронних мереж для автоматизованого виявлення аномалій у сенсорних системах.

Практичне значення роботи полягає у можливості використання розробленого програмного комплексу для автоматизованого моніторингу обладнання, своєчасного виявлення несправностей та зниження ризику

					ІАЛЦ.467200.003 ПЗ	Арк.
						11
Зм.	Арк.	№ докум.	Підпис	Дата		

виникнення аварійних ситуацій. Результати дослідження можуть бути використані у промислових інформаційних системах, системах технічної діагностики та інших сферах, де необхідний аналіз великих обсягів сенсорних даних у режимі реального часу.

ВИСНОВОК ДО РОЗДІЛУ 1

На підставі проведеного у першому розділі аналізу було визначено, що сучасні сенсорні системи генерують значні обсяги даних, обробка яких потребує використання ефективних методів автоматизованого аналізу та виявлення аномалій. Розглянуто основні підходи до anomaly detection, проаналізовано переваги та недоліки традиційних статистичних методів і алгоритмів машинного навчання.

У результаті дослідження встановлено, що використання нейронних мереж типу Autoencoder є перспективним підходом для задач виявлення аномалій у сенсорних даних завдяки здатності моделі автоматично визначати відхилення на основі похибки реконструкції. Також було обґрунтовано доцільність використання алгоритмів Isolation Forest та Logistic Regression для побудови базової моделі anomaly detection.

Проведений аналіз предметної області підтвердив актуальність розробки програмного комплексу виявлення аномалій у сенсорних системах із використанням алгоритмів машинного навчання та визначив основні задачі подальшого дослідження й реалізації програмної системи.

					ІАЛЦ.467200.003 ПЗ	Арк.
						13
Зм.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2 ВИБІР ТЕХНОЛОГІЙ ТА ЗАСОБІВ РОЗРОБКИ СИСТЕМИ

2.1. Формалізація задачі виявлення аномалій

У сучасних сенсорних системах велика кількість даних формується у вигляді безперервного потоку вимірювань, що надходять від різноманітних датчиків та сенсорів. Такі дані можуть містити інформацію про температуру, тиск, вібрацію, електричні параметри або інші характеристики технічного обладнання. Аналіз подібних даних дозволяє оцінювати поточний стан системи та своєчасно виявляти відхилення від нормального режиму функціонування..

У рамках даної дипломної роботи розглядається задача автоматизованого виявлення аномалій у сенсорних даних із використанням алгоритмів машинного навчання. Під аномалією у даному випадку розуміється значення або група значень, які суттєво відрізняються від нормальної поведінки системи та можуть свідчити про наявність технічної несправності, помилки вимірювання або нестандартного режиму роботи обладнання.

Вхідні дані представляються у вигляді набору сенсорних показників SensorA, SensorB та SensorC, що змінюються у часі. Кожен запис набору даних містить значення сенсорів у певний момент часу та формує багатовимірний вектор ознак. Таким чином, потік сенсорних даних може бути представлений як послідовність спостережень:

$$X = \{x_1, x_2, x_3, \dots, x_n\}, \quad (2.1)$$

Де:

- X — потік сенсорних даних
- x_n — фактичне значення n -ї ознаки;
- n — індекс елемента вектора

Особливістю сенсорних даних є наявність шумів, пропущених значень та викидів, що можуть негативно впливати на якість роботи моделей машинного навчання. У зв'язку з цим перед навчанням моделей необхідно виконати попередню обробку даних, яка включає очищення набору даних, заповнення пропущених значень, видалення аномальних викидів та нормалізацію ознак.

Для побудови системи виявлення аномалій у роботі використовуються алгоритми Isolation Forest, Logistic Regression та нейромережева модель Autoencoder. На першому етапі Isolation Forest застосовується для автоматичного формування псевдоміток аномалій. Отримані результати використовуються для навчання базової моделі логістичної регресії та оцінки ефективності алгоритмів anomaly detection.

Основним методом виявлення аномалій у роботі є Autoencoder — нейронна мережа, призначена для реконструкції вхідних даних. Принцип роботи автоенкодера базується на припущенні, що модель добре відновлює лише нормальні дані, на яких була навчена. У випадку появи аномальних значень похибка реконструкції суттєво зростає.

Архітектура автоенкодера складається з двох основних частин: енкодера та декодера. Енкодер виконує стискання вхідних даних у латентний простір меншої розмірності, а декодер відновлює початкові значення на основі отриманого представлення. Процес реконструкції можна подати у вигляді:

$$\hat{x} = D(E(x)) \quad (2.2)$$

Де:

- $E(x)$ — енкодер;
- $D(x)$ — декодер;
- $\hat{x}$ — реконструйований вектор даних.

Для оцінювання якості реконструкції використовується функція середньоквадратичної помилки:

$$MSE = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2, \quad (2.3)$$

Де:

- MSE — середньоквадратична помилка реконструкції;;
- n — кількість елементів (ознак) у векторі даних;;
- i — індекс елемента вектора.
- x_i — фактичне значення i -ї ознаки;.
- $\hat{x}_i$ — реконструйоване автоенкодером значення i -ї ознаки;.
- $(x_i - \hat{x}_i)^2$ — квадрат похибки реконструкції для i — ї ознаки;
- $\sum_{i=1}^n$ — сума квадратів похибок для всіх ознак вектора;
- $\frac{1}{n}$ — усереднення отриманої суми.

Чим більшою є похибка реконструкції, тим вищою є ймовірність того, що поточний стан системи є аномальним. Для класифікації спостережень використовується порогове значення reconstruction error, яке визначається експериментальним шляхом під час тестування моделі.

У процесі розробки програмного комплексу також виконується створення додаткових ознак для підвищення ефективності алгоритмів машинного навчання. До таких ознак належать середнє значення сенсорів, стандартне відхилення, мінімальні та максимальні значення, а також різниця між сусідніми

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		16

вимірюваннями. Використання feature engineering дозволяє покращити якість навчання моделей та підвищити точність виявлення аномалій.

Для оцінювання ефективності реалізованих моделей використовуються метрики Accuracy, Precision, Recall та F1-score. Також виконується побудова матриці помилок та графічне відображення результатів роботи моделей anomaly detection.

Таким чином, формалізація задачі виявлення аномалій у даній роботі базується на аналізі багатовимірних сенсорних даних, використанні алгоритмів машинного навчання та оцінюванні похибки реконструкції автоенкодера для автоматизованого визначення аномальних станів системи.

2.2. Архітектура та принцип роботи автоенкодера

Архітектура автоенкодера, використаного у даній дипломній роботі, представлена на рисунку 2.1. Основним призначенням моделі є автоматизоване виявлення аномалій у сенсорних даних шляхом аналізу похибки реконструкції вхідних параметрів. Для реалізації моделі використано багатошаровий глибокий автоенкодер, який складається з енкодерної та декодерної частин.

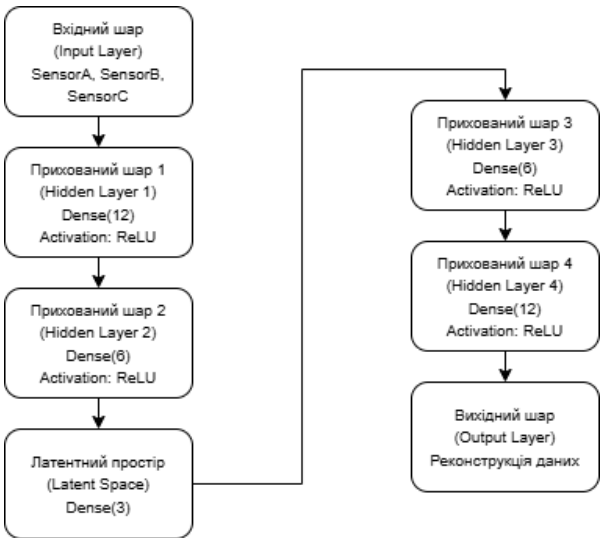


Рисунок 2.1 – Архітектура автоенкодера

У структурі автоенкодера вхідний шар виконує функцію прийому та передачі вхідних даних до прихованих шарів нейронної мережі. На цьому етапі до моделі надходять попередньо оброблені сенсорні показники SensorA, SensorB та SensorC, а також сформовані додаткові ознаки, що характеризують стан системи. Розмірність вхідного шару відповідає кількості ознак у наборі даних після виконання feature engineering та нормалізації даних. Основним завданням вхідного шару є забезпечення коректного представлення інформації для подальшої обробки нейронною мережею.

Після вхідного шару дані передаються до першого прихованого шару енкодера. Його основною функцією є початкове зменшення розмірності вхідного простору та виділення найбільш важливих характеристик даних. На цьому етапі модель починає формувати узагальнене представлення нормальної поведінки сенсорної системи. Зменшення кількості нейронів у прихованому шарі дозволяє усунути надлишкову інформацію та зосередити увагу на найбільш значущих закономірностях.

Другий прихований шар енкодера виконує подальшу компресію даних та формує більш компактне представлення сенсорної інформації. У процесі навчання модель автоматично визначає внутрішні залежності між параметрами системи та формує латентні ознаки, що характеризують нормальний стан об'єкта моніторингу. Саме на цьому рівні відбувається поступове виділення прихованих закономірностей, які складно визначити за допомогою традиційних статистичних методів.

Центральним елементом архітектури є латентний простір. Він являє собою компактне представлення вхідних даних, яке містить найбільш суттєві характеристики сенсорних сигналів. Латентний простір є результатом роботи енкодера та використовується декодером для відновлення початкового вектору ознак. У задачах anomaly detection саме латентне представлення дозволяє моделі відокремлювати типову поведінку системи від аномальних відхилень.

					ІАЛЦ.467200.003 ПЗ	Арк.
						18
Зм.	Арк.	№ докум.	Підпис	Дата		

Латентний простір відіграє ключову роль у процесі роботи автоенкодера, оскільки саме у ньому формується компактне представлення вхідних сенсорних даних. Під час навчання нейронна мережа автоматично визначає найбільш суттєві характеристики нормального стану системи та зберігає їх у стислому вигляді. Такий підхід дозволяє зменшити розмірність даних без втрати критично важливої інформації про поведінку сенсорної системи.

Однією з головних переваг використання латентного простору є здатність моделі відокремлювати типові закономірності від випадкових шумів та аномальних відхилень. У процесі реконструкції нормальні сигнали відновлюються з невеликою похибкою, оскільки їх структура добре представлена у латентному просторі. Водночас аномальні стани, які відрізняються від навчальних даних, не можуть бути якісно реконструйовані моделлю, що призводить до збільшення reconstruction error.

Крім цього, використання компактного латентного представлення позитивно впливає на швидкість роботи моделі та зменшує обчислювальні витрати під час навчання. Це є особливо важливим для задач аналізу великих потоків сенсорних даних у режимі реального часу, де швидкість обробки інформації має критичне значення.

Після проходження через латентний простір дані передаються до декодерної частини автоенкодера. Третій прихований шар виконує функцію початкового відновлення інформації зі стислого представлення. На цьому етапі модель намагається реконструювати структуру нормальних сенсорних даних на основі інформації, отриманої у латентному просторі.

Четвертий прихований шар виконує подальше розгортання даних та відновлення взаємозв'язків між ознаками. Архітектура декодера є симетричною відносно енкодера, що дозволяє забезпечити більш точну реконструкцію вхідних параметрів. Завдяки цьому модель здатна ефективно відтворювати нормальні сигнали та формувати підвищену похибку реконструкції для

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		19

аномальних значень.

Вихідний шар автоенкодера має ту саму розмірність, що й вхідний, оскільки його основним завданням є відновлення початкового набору сенсорних даних. Після завершення реконструкції виконується порівняння між реальними та відновленими значеннями. Для цього використовується середньоквадратична помилка реконструкції, яка дозволяє оцінити ступінь відхилення поточного стану системи від нормальної поведінки.

У процесі навчання автоенкодера використовувалася функція втрат Mean Squared Error (MSE), яка мінімізує різницю між вхідними та реконструйованими даними. Для оптимізації ваг нейронної мережі застосовано алгоритм Adam, що забезпечує стабільне та ефективне навчання моделі. Також у роботі використовувався механізм EarlyStopping, який дозволяє автоматично завершувати навчання при відсутності покращення функції втрат та запобігає перенавчанню моделі.

Вибір саме глибокого автоенкодера обумовлений його здатністю працювати з багатовимірними сенсорними даними та автоматично виявляти складні нелінійні залежності між параметрами системи. На відміну від традиційних алгоритмів anomaly detection, автоенкодер не потребує великої кількості розмічених аномальних прикладів та здатний ефективно навчатися на нормальних даних.

Таким чином, обрана архітектура автоенкодера дозволяє реалізувати ефективну систему автоматизованого виявлення аномалій у сенсорних даних та забезпечує можливість подальшого масштабування програмного комплексу для використання у промислових системах моніторингу.

					ІАЛЦ.467200.003 ПЗ	Арк.
						20
Зм.	Арк.	№ докум.	Підпис	Дата		

2.3. Попередня обробка сенсорних даних

Одним із найважливіших етапів побудови системи виявлення аномалій є попередня обробка вхідних даних. Якість навчання моделі автоенкодера безпосередньо залежить від коректності підготовки сенсорної інформації, оскільки необроблені або зашумлені дані можуть негативно впливати на точність реконструкції та стабільність роботи нейронної мережі.

У межах даної дипломної роботи в якості джерела інформації використовувався набір сенсорних даних `sensor_data.csv`, що містить часові ряди показників `SensorA`, `SensorB` та `SensorC`. Дані характеризуються наявністю пропущених значень, шумів та окремих аномальних сплесків, що потребувало виконання комплексу процедур попередньої обробки.

На початковому етапі було виконано аналіз структури набору даних із використанням бібліотеки `pandas`. Після завантаження файлу проводилась перевірка розмірності датасету, типів даних та кількості пропущених значень у кожному сенсорному каналі. Отримані результати показали, що частина вимірювань містить відсутні значення, які могли негативно вплинути на подальше навчання моделі.

Для усунення пропусків використовувався метод заповнення медіанними значеннями відповідних ознак. Такий підхід дозволяє мінімізувати вплив викидів та зберегти статистичну структуру розподілу сенсорних параметрів. На відміну від середнього значення, медіана є більш стійкою до аномальних сплесків, що особливо важливо у задачах `anomaly detection`.

Після обробки пропущених значень виконувалось очищення даних від викидів. Для цього використовувався метод міжквартильного розмаху `IQR` (`Interquartile Range`). На основі першого та третього квартилів для кожної ознаки обчислювався допустимий інтервал значень, після чого рядки з надмірними відхиленнями видалялися з набору даних. Застосування такого підходу

					ІАЛЦ.467200.003 ПЗ	Арк.
						21
Зм.	Арк.	№ докум.	Підпис	Дата		

дозволило зменшити вплив випадкових аномальних значень на процес навчання автоенкодера.

Після очищення даних було реалізовано етап формування додаткових ознак. Оскільки початковий набір сенсорних параметрів містив лише базові значення трьох сенсорів, для покращення якості моделі виконувалось feature engineering. Зокрема, для кожного запису обчислювалися:

- середнє значення сенсорів;
- стандартне відхилення;
- мінімальне та максимальне значення;
- діапазон між максимальним і мінімальним показником.

Крім цього, для кожного сенсора формувалися похідні характеристики, що описують відхилення поточного значення від середнього рівня системи, а також різниця між сусідніми часовими вимірюваннями. Використання таких ознак дозволило підвищити інформативність вхідних даних та забезпечити кращу здатність моделі до виявлення прихованих аномалій.

Наступним етапом стала нормалізація даних. Оскільки різні сенсорні параметри мають різні масштаби вимірювання, без попереднього масштабування окремі ознаки могли домінувати у процесі навчання нейронної мережі. Для усунення цього ефекту використовувався алгоритм StandardScaler із бібліотеки scikit-learn.

У процесі стандартизації кожна ознака перетворювалася таким чином, щоб її середнє значення дорівнювало нулю, а стандартне відхилення — одиниці. Завдяки цьому всі параметри набували єдиного масштабу, що позитивно вплинуло на стабільність градієнтного спуску та швидкість навчання автоенкодера.

Попередня обробка даних є одним із найбільш критичних етапів побудови систем машинного навчання, оскільки якість вхідної інформації безпосередньо впливає на точність та стабільність роботи моделей. У задачах

					ІАЛЦ.467200.003 ПЗ	Арк.
						22
Зм.	Арк.	№ докум.	Підпис	Дата		

anomaly detection сенсорні дані часто містять випадкові шуми, пропущені значення та нестандартні відхилення, що можуть спотворювати процес навчання нейронної мережі. Саме тому коректна підготовка датасету є необхідною умовою для ефективного функціонування автоенкодера.

Особливу увагу у роботі було приділено процесу нормалізації ознак. Без масштабування даних окремі параметри сенсорів могли мати значно більший вплив на функцію втрат, що призводило б до нестабільного навчання нейронної мережі. Використання StandardScaler дозволило привести всі ознаки до єдиного масштабу та забезпечити більш стабільний процес оптимізації ваг моделі.

Крім цього, feature engineering дозволив суттєво розширити початковий набір сенсорних ознак. Додаткові статистичні характеристики надали моделі більше інформації про поточний стан системи та дозволили ефективніше виявляти приховані аномальні закономірності у сенсорних даних. Завдяки цьому вдалося підвищити інформативність вхідного простору та покращити результати anomaly detection.

Після завершення нормалізації було сформовано фінальний набір ознак, який використовувався для навчання моделей машинного навчання. Підготовлені дані надалі застосовувалися як для базової моделі Logistic Regression, так і для моделі глибокого автоенкодера.

Таким чином, реалізований етап попередньої обробки даних дозволив підвищити якість вхідної інформації, зменшити вплив шумів та аномальних значень, а також забезпечити стабільне навчання моделей виявлення аномалій у сенсорних системах.

2.4. Реалізація базової моделі anomaly detection

Після виконання попередньої обробки сенсорних даних було реалізовано базову модель виявлення аномалій із використанням алгоритмів Isolation Forest

					ІАЛЦ.467200.003 ПЗ	Арк.
						23
Зм.	Арк.	№ докум.	Підпис	Дата		

та Logistic Regression. Основною метою створення базової моделі стало формування початкового підходу до anomaly detection та подальше порівняння її ефективності з нейромережевою моделлю Autoencoder.

На першому етапі для автоматичного визначення аномальних значень використовувався алгоритм Isolation Forest. Даний метод належить до алгоритмів unsupervised learning та дозволяє виявляти аномалії без необхідності використання попередньо розміченого набору даних. Принцип роботи Isolation Forest базується на випадковому поділі простору ознак та ізоляції об'єктів, які суттєво відрізняються від основної маси даних.

Для реалізації алгоритму використовувалась бібліотека scikit-learn [20]. У процесі налаштування моделі було задано параметр contamination, який визначає приблизну частку аномальних значень у наборі даних. У рамках даної роботи значення contamination було встановлено на рівні 0.05, що відповідає приблизно 5% потенційних аномалій у сенсорних даних.

Після навчання Isolation Forest кожному запису набору даних було присвоєно псевдомітку. Нормальні значення отримували мітку 0, а аномальні — 1. Сформовані псевдомітки надалі використовувалися для навчання базової моделі класифікації.

Для побудови базової supervised-моделі використовувалась Logistic Regression [21]. Даний алгоритм є одним із найбільш поширених методів класифікації та дозволяє оцінити ефективність лінійного підходу до задач anomaly detection. Logistic Regression забезпечує швидке навчання моделі та добре підходить для формування baseline-рішення.

Використання базової моделі є важливим етапом побудови систем anomaly detection, оскільки дозволяє сформувати початковий рівень оцінювання ефективності алгоритмів машинного навчання. У рамках даної роботи Logistic Regression використовувалась як baseline-модель для подальшого порівняння з нейромережевим автоенкодером.

Перевагою Logistic Regression є простота реалізації, невисокі обчислювальні витрати та здатність працювати з багатовимірними наборами даних. Алгоритм дозволяє оцінити ймовірність належності об'єкта до класу аномалій та формує лінійну межу розділення між нормальними та аномальними станами системи.

Разом із тим лінійна природа Logistic Regression створює певні обмеження при роботі зі складними сенсорними даними. У випадку нелінійних залежностей між параметрами система може втрачати здатність ефективно відокремлювати аномальні сигнали від нормальної поведінки. Саме це стало однією з причин подальшого використання глибокого автоенкодера у роботі.

Перед навчанням моделі набір даних було розділено на тренувальну та тестову вибірки із використанням функції `train_test_split`. Для забезпечення коректного співвідношення нормальних та аномальних записів застосовувалась стратифікація вибірки за допомогою параметра `stratify`.

У процесі навчання Logistic Regression використовувався параметр `class_weight="balanced"`, який дозволяє компенсувати дисбаланс класів у наборі даних. Оскільки кількість аномалій у сенсорних даних є значно меншою за кількість нормальних значень, застосування балансування класів дозволило покращити якість виявлення аномальних станів.

Після завершення навчання модель використовувалась для прогнозування аномалій у тестовій вибірці. Для оцінювання ефективності роботи алгоритму були використані метрики Accuracy, Precision, Recall та F1-score. Крім цього, будувалась матриця помилок (Confusion Matrix), що дозволило виконати детальний аналіз результатів класифікації.

Отримані результати показали, що базова модель Logistic Regression забезпечує достатньо високе значення Recall, що свідчить про здатність алгоритму виявляти значну кількість аномальних станів. Водночас значення Precision залишалося відносно невисоким через наявність великої кількості

					ІАЛЦ.467200.003 ПЗ	Арк.
						25
Зм.	Арк.	№ докум.	Підпис	Дата		

хибнопозитивних спрацювань.

Для додаткового аналізу стабільності моделі використовувався метод часової крос-валідації TimeSeriesSplit. Даний підхід дозволив оцінити якість роботи алгоритму на різних часових інтервалах та перевірити здатність моделі адаптуватися до змін у структурі сенсорних даних.

Крім цього, у роботі було виконано аналіз залишків моделі шляхом побудови гістограми residual distribution. Це дозволило оцінити характер помилок прогнозування та визначити особливості розподілу реконструктивних відхилень у тестовій вибірці.

Таким чином, реалізована базова модель anomaly detection дозволила сформулювати початковий підхід до автоматизованого виявлення аномалій у сенсорних даних та стала основою для подальшого порівняння з моделлю Autoencoder.

2.5. Реалізація моделі Autoencoder

Розроблювана Після побудови базової моделі було реалізовано нейромережеву модель Autoencoder для більш ефективного виявлення аномалій у сенсорних даних. Основною перевагою автоенкодера є здатність автоматично навчатися прихованому представленню нормального стану системи та визначати аномальні значення на основі похибки реконструкції.

Для реалізації моделі використовувались бібліотеки TensorFlow та Keras. Архітектура автоенкодера складалася з енкодерної та декодерної частин. Енкодер виконував поступове зменшення розмірності вхідних даних, формуючи компактне латентне представлення сенсорної інформації. Декодер, у свою чергу, виконував реконструкцію початкових даних із латентного простору.

У моделі використовувались повнозв'язні шари Dense із функцією

					ІАЛЦ.467200.003 ПЗ	Арк.
						26
Зм.	Арк.	№ докум.	Підпис	Дата		

активації ReLU. Архітектура нейронної мережі включала кілька прихованих шарів із поступовим зменшенням кількості нейронів у центральній частині моделі. Це дозволило забезпечити ефективне стискання вхідних ознак та виділення найбільш важливих закономірностей у сенсорних даних.

На вхід автоенкодера подавалися попередньо нормалізовані сенсорні параметри та додаткові ознаки, сформовані на етапі feature engineering. Перед навчанням моделі всі ознаки проходили стандартизацію за допомогою StandardScaler, що дозволило забезпечити стабільність процесу оптимізації ваг нейронної мережі.

Навчання моделі здійснювалося із використанням функції втрат Mean Squared Error (MSE), яка мінімізує різницю між вхідними та реконструйованими значеннями. Для оптимізації ваг застосовувався алгоритм Adam, що забезпечує ефективне налаштування параметрів нейронної мережі та швидку збіжність процесу навчання.

Вибір функції втрат Mean Squared Error обумовлений специфікою задачі реконструкції сенсорних даних. Дана функція дозволяє оцінити різницю між початковими та відновленими значеннями ознак і забезпечує коректне налаштування ваг нейронної мережі під час навчання. Мінімізація середньоквадратичної помилки сприяє формуванню стабільного латентного представлення нормального стану системи та дозволяє ефективно виявляти аномальні відхилення.

У процесі навчання модель поступово адаптувалася до структури сенсорних даних та формувала внутрішні залежності між параметрами системи. На початкових етапах навчання reconstruction error залишався відносно високим, оскільки нейронна мережа ще не мала достатньо точного уявлення про закономірності у наборі даних. Після кількох епох спостерігалось поступове зменшення функції втрат, що свідчило про покращення здатності моделі до реконструкції нормальних сигналів.

					ІАЛЦ.467200.003 ПЗ	Арк.
						27
Зм.	Арк.	№ докум.	Підпис	Дата		

Особливістю використання автоенкодера у задачах anomaly detection є відсутність необхідності у великому наборі розмічених аномальних даних. Модель навчається переважно на нормальних станах системи та формує уявлення про типову поведінку сенсорних параметрів. У випадку появи нетипових або аномальних сигналів автоенкодер не може якісно реконструювати такі дані, що призводить до помітного збільшення reconstruction error.

У рамках дипломної роботи було використано багатосарову архітектуру нейронної мережі, що дозволило підвищити здатність моделі до виявлення складних нелінійних закономірностей у сенсорних даних. Глибока архітектура забезпечує поступове формування ознак різного рівня абстракції, що позитивно впливає на якість anomaly detection у порівнянні з простими лінійними моделями.

Під час експериментального тестування особливу увагу було приділено аналізу процесу навчання нейронної мережі. Для контролю стабільності моделі використовувались графіки функції втрат Training Loss. Аналіз отриманих результатів показав, що автоенкодер демонструє стабільне зближення функції втрат без різких коливань або ознак нестабільного навчання. Це свідчить про коректний вибір архітектури моделі та параметрів оптимізації.

Використання механізму EarlyStopping дозволило автоматично завершувати навчання після припинення покращення функції втрат. Даний підхід є особливо важливим для нейронних мереж, оскільки надмірна кількість епох може призводити до перенавчання моделі. У випадку overfitting автоенкодер починає надто точно відтворювати навчальні дані та втрачає здатність ефективно працювати з новими сенсорними сигналами. Застосування EarlyStopping дозволило зменшити ризик перенавчання та покращити узагальнюючу здатність моделі.

Окрему увагу у роботі було приділено аналізу reconstruction error як

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		28

основного критерію anomaly detection. Для кожного запису набору даних обчислювалась похибка реконструкції між вхідними та відновленими ознаками. Отримані значення використовувались для подальшого визначення аномальних станів системи. Нормальні сигнали характеризувалися низьким reconstruction error, тоді як аномальні відхилення супроводжувалися помітним збільшенням похибки реконструкції.

Крім цього, у роботі було досліджено вплив розмірності латентного простору на якість роботи моделі. Надто малий латентний простір призводить до втрати важливої інформації про структуру сенсорних даних, тоді як надто великий — зменшує здатність моделі до ефективного виявлення аномалій. У результаті експериментального тестування було обрано оптимальну архітектуру, яка забезпечує баланс між якістю реконструкції та ефективністю anomaly detection.

Загалом використання автоенкодера показало високу перспективність застосування глибокого навчання для задач автоматизованого моніторингу сенсорних систем. Реалізована модель дозволяє ефективно працювати з багатовимірними даними, автоматично визначати складні аномальні закономірності та забезпечувати стабільну якість виявлення несправностей у технічних системах.

Для запобігання перенавчанню використовувався механізм EarlyStopping. Даний підхід дозволяв автоматично завершувати процес навчання у випадку відсутності покращення функції втрат протягом кількох епох. Завдяки цьому вдалося зменшити ризик overfitting та підвищити узагальнюючу здатність моделі.

Однією з важливих особливостей реалізованої моделі є використання глибокої архітектури нейронної мережі. На відміну від класичних алгоритмів машинного навчання, автоенкодер здатний автоматично формувати приховане представлення сенсорних даних та виявляти складні нелінійні залежності між

					ІАЛЦ.467200.003 ПЗ	Арк.
						29
Зм.	Арк.	№ докум.	Підпис	Дата		

параметрами системи. Завдяки цьому модель може ефективно працювати навіть у випадках, коли аномальні відхилення мають неявний характер і не можуть бути визначені за допомогою простих статистичних методів.

У процесі навчання автоенкодер поступово навчається реконструювати нормальні сенсорні сигнали з мінімальною похибкою. При цьому нейронна мережа формує компактне латентне представлення даних, яке містить найбільш важливі характеристики нормального стану системи. Використання латентного простору дозволяє зменшити розмірність вхідних ознак та усунути надлишкову інформацію, що позитивно впливає на стабільність роботи моделі.

Важливу роль у роботі нейронної мережі відіграє функція активації ReLU, яка використовувалась у прихованих шарах автоенкодера. Її застосування дозволяє підвищити швидкість навчання моделі та забезпечити ефективне проходження градієнтів під час оптимізації ваг нейронної мережі. Крім цього, використання ReLU дозволяє зменшити ризик виникнення проблеми затухаючого градієнта, що є важливим для глибоких архітектур.

Для оптимізації параметрів моделі використовувався алгоритм Adam, який поєднує переваги адаптивного налаштування швидкості навчання та градієнтного спуску. Даний оптимізатор забезпечує стабільне та швидке зближення функції втрат, що дозволяє ефективно навчати модель навіть при роботі з багатовимірними сенсорними даними.

Під час експериментального тестування було встановлено, що збільшення кількості прихованих шарів позитивно впливає на здатність моделі до виділення складних закономірностей у даних. Водночас надмірне ускладнення архітектури може призводити до перенавчання та збільшення часу навчання моделі. Саме тому у роботі було обрано компромісну архітектуру автоенкодера, яка забезпечує достатню точність anomaly detection при помірних обчислювальних витратах.

Після завершення навчання виконувалось обчислення reconstruction error

					ІАЛЦ.467200.003 ПЗ	Арк.
						30
Зм.	Арк.	№ докум.	Підпис	Дата		

— похибки реконструкції для кожного запису набору даних. Для нормальних значень похибка залишалась незначною, тоді як для аномальних станів спостерігалось суттєве зростання reconstruction error.

Для визначення аномалій використовувалось порогове значення threshold. У рамках роботи було реалізовано автоматичний підбір threshold шляхом перебору різних значень та вибору оптимального порогу за максимальним значенням F1-score. Такий підхід дозволив досягти кращого балансу між Precision та Recall.

Процес підбору threshold є одним із найважливіших етапів побудови систем anomaly detection, оскільки саме від вибраного порогового значення залежить баланс між кількістю правильно виявлених аномалій та кількістю хибнопозитивних спрацювань. Надто низьке значення threshold може призводити до великої кількості помилкових спрацювань системи, тоді як надто високе — до пропуску реальних аномальних станів.

У рамках дипломної роботи підбір threshold виконувався експериментальним шляхом із використанням перебору різних значень reconstruction error. Для кожного значення порогу обчислювались метрики Precision, Recall та F1-score, після чого вибиралось оптимальне значення threshold із максимальним показником F1-score. Такий підхід дозволив забезпечити більш збалансовану роботу моделі та підвищити загальну ефективність anomaly detection.

Додатково було проведено аналіз впливу threshold на поведінку системи. У результаті тестування встановлено, що зміна порогового значення безпосередньо впливає на співвідношення між Precision та Recall. Це підтверджує важливість правильного налаштування threshold для задач автоматизованого моніторингу технічного стану обладнання.

Після визначення оптимального threshold модель використовувалась для класифікації сенсорних даних на нормальні та аномальні стани. Результати

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		31

роботи автоенкодера оцінювались за допомогою метрик Accuracy, Precision, Recall та F1-score. Також будувалась матриця помилок та графіки reconstruction error.

У процесі експериментального тестування автоенкодер продемонстрував кращі результати у порівнянні з базовою моделлю Logistic Regression. Особливо помітним стало підвищення значення Precision та F1-score, що свідчить про зменшення кількості хибнопозитивних спрацювань та покращення якості anomaly detection.

Таким чином, реалізована модель Autoencoder підтвердила ефективність використання нейронних мереж для автоматизованого виявлення аномалій у сенсорних системах та показала перспективність застосування глибокого навчання у задачах аналізу сенсорних даних.

2.6. Оцінювання ефективності моделей

Для оцінювання ефективності реалізованих моделей anomaly detection у роботі використовувався комплекс метрик класифікації, що дозволяє детально проаналізувати якість виявлення аномальних станів у сенсорних даних.

Основними метриками оцінювання стали Accuracy, Precision, Recall та F1-score. Метрика Accuracy характеризує загальну частку правильно класифікованих об'єктів у тестовій вибірці. Проте у задачах anomaly detection використання лише Accuracy є недостатнім через значний дисбаланс класів між нормальними та аномальними значеннями.

У задачах виявлення аномалій особливо важливо враховувати дисбаланс класів у наборі даних. Як правило, кількість нормальних станів суттєво перевищує кількість аномальних значень, через що навіть модель із високим показником Accuracy може демонструвати низьку ефективність anomaly

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		32

detection. Наприклад, система може правильно класифікувати більшість нормальних записів, але пропускати значну частину аномалій.

Саме тому у роботі основна увага приділялася метрикам Precision, Recall та F1-score. Precision дозволяє оцінити кількість хибнопозитивних спрацювань моделі, тоді як Recall характеризує здатність системи виявляти реальні аномальні стани. Баланс між цими показниками є критично важливим для систем технічного моніторингу, оскільки надмірна кількість помилкових спрацювань може ускладнювати роботу системи, а пропуск аномалій — створювати ризик виникнення несправностей.

У процесі експериментального тестування було встановлено, що використання оптимізації threshold за F1-score дозволяє досягти більш збалансованої роботи автоенкодера та підвищити якість anomaly detection у порівнянні з використанням фіксованих порогових значень.

Для більш детального аналізу використовувалась метрика Precision, яка показує частку правильно визначених аномалій серед усіх об'єктів, класифікованих моделлю як аномальні. Високе значення Precision дозволяє зменшити кількість хибнопозитивних спрацювань системи.

Метрика Recall характеризує здатність моделі виявляти реальні аномальні стани серед усіх наявних аномалій у наборі даних. Високе значення Recall є особливо важливим для систем технічного моніторингу, оскільки пропуск аномалії може призвести до серйозних наслідків у роботі обладнання.

Для комплексного оцінювання якості моделей використовувався показник F1-score, який є гармонійним середнім між Precision та Recall. Саме F1-score використовувався як основний критерій підбору оптимального threshold для автоенкодера.

Крім числових метрик, у роботі використовувалась матриця помилок (Confusion Matrix), що дозволяє візуально оцінити співвідношення правильних та помилкових класифікацій. Аналіз матриці помилок дає можливість

визначити кількість хибнопозитивних та хибнонегативних прогнозів і виконати детальне порівняння ефективності моделей.

Для аналізу процесу навчання автоенкодера будувались графіки зміни функції втрат (Training Loss) залежно від кількості епох. Це дозволило оцінити стабільність навчання моделі та ефективність використання механізму EarlyStopping.

Також у роботі виконувалась побудова графіків reconstruction error, що відображають розподіл похибки реконструкції для нормальних та аномальних значень. Візуалізація reconstruction error дозволила наочно оцінити ефективність вибраного threshold та здатність моделі відокремлювати аномальні стани від нормальної поведінки системи.

Використання графічних методів аналізу дозволяє більш наочно оцінити ефективність роботи моделей anomaly detection та виявити особливості розподілу сенсорних даних. У межах дипломної роботи будувались графіки зміни функції втрат під час навчання автоенкодера, гістограми reconstruction error та матриці помилок для різних моделей.

Аналіз графіків Training Loss показав стабільне зменшення функції втрат у процесі навчання нейронної мережі. Це свідчить про успішне формування прихованого представлення нормальних сенсорних сигналів та ефективне налаштування ваг моделі. Використання механізму EarlyStopping дозволило автоматично завершувати навчання після досягнення стабільного значення функції втрат, що позитивно вплинуло на узагальнюючу здатність автоенкодера.

Побудова графіків reconstruction error дозволила візуально відокремити нормальні стани системи від аномальних відхилень. Для більшості нормальних записів похибка реконструкції залишалась незначною, тоді як аномальні сигнали характеризувалися помітним зростанням reconstruction error. Це підтвердило ефективність використання автоенкодера для задач anomaly

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		34

detection у сенсорних системах.

Крім цього, у процесі тестування було виконано порівняльний аналіз результатів базової моделі Logistic Regression та нейромережевого Autoencoder. Отримані результати показали, що автоенкодер забезпечує більш стабільну роботу системи та дозволяє зменшити кількість хибнопозитивних спрацювань при збереженні достатньо високого рівня виявлення аномалій.

У результаті проведеного тестування було встановлено, що модель Autoencoder демонструє кращу якість anomaly detection у порівнянні з базовою моделлю Logistic Regression. Зокрема, автоенкодер забезпечив вищі значення Precision та F1-score, а також зменшив кількість хибнопозитивних спрацювань.

Таким чином, проведене оцінювання підтвердило ефективність використання нейромережевого підходу для задач автоматизованого виявлення аномалій у сенсорних системах та доцільність використання Autoencoder у системах технічного моніторингу.

					ІАЛЦ.467200.003 ПЗ	Арк.
						35
Зм.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВОК ДО РОЗДІЛУ 2

У другому розділі дипломної роботи було розглянуто основні технології та методи, необхідні для реалізації програмного комплексу виявлення аномалій у сенсорних системах із використанням алгоритмів машинного навчання. Проведено формалізацію задачі anomaly detection у багатовимірних сенсорних даних та обґрунтовано доцільність застосування нейромережових методів для автоматизованого аналізу технічного стану систем.

У процесі дослідження було визначено особливості сенсорних даних, зокрема наявність шумів, пропущених значень та аномальних викидів, що потребує виконання попередньої обробки даних перед навчанням моделей. Для підготовки набору даних було використано методи заповнення пропущених значень, очищення даних від викидів за допомогою міжквартильного розмаху, нормалізацію ознак та feature engineering. Реалізований етап попередньої обробки дозволив підвищити якість вхідних даних та забезпечити стабільність навчання моделей машинного навчання.

У роботі було реалізовано базову модель anomaly detection із використанням алгоритмів Isolation Forest та Logistic Regression. Isolation Forest використовувався для автоматичного формування псевдоміток аномалій, а Logistic Regression — як базова supervised-модель для оцінки ефективності класифікації сенсорних даних. Проведене тестування дозволило визначити сильні та слабкі сторони класичних алгоритмів anomaly detection і сформувати baseline-рішення для подальшого порівняння.

Основну увагу у другому розділі було приділено реалізації моделі Autoencoder. Було обґрунтовано вибір архітектури нейронної мережі, що складається з енкодерної та декодерної частин, а також визначено особливості використання латентного простору для формування компактного представлення нормального стану системи. Для навчання моделі

					ІАЛЦ.467200.003 ПЗ	Арк.
						36
Зм.	Арк.	№ докум.	Підпис	Дата		

використовувалися TensorFlow та Keras, функція втрат Mean Squared Error та оптимізатор Adam. Додатково було застосовано механізм EarlyStopping для запобігання перенавчанню моделі.

Також у роботі було розглянуто методику оцінювання ефективності моделей anomaly detection. Для аналізу результатів використовувалися метрики Accuracy, Precision, Recall та F1-score, а також матриця помилок і графіки reconstruction error. Проведене тестування показало, що модель Autoencoder забезпечує кращу якість виявлення аномалій у порівнянні з базовою моделлю Logistic Regression, зокрема за показниками Precision та F1-score.

Таким чином, у другому розділі було сформовано технологічну та алгоритмічну основу програмного комплексу виявлення аномалій у сенсорних системах. Отримані результати підтвердили доцільність використання нейронних мереж типу Autoencoder для задач anomaly detection та створили підґрунтя для подальшого дослідження ефективності реалізованої системи у наступному розділі.

					ІАЛЦ.467200.003 ПЗ	Арк.
						37
Зм.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3 ДЕТАЛІ РОЗРОБКИ ПРОГРАМНОГО КОМПЛЕКСУ

У цьому розділі розглядається практична реалізація програмного комплексу для виявлення аномалій у даних сенсорних систем. Основна увага приділяється структурі програмної частини, організації обробки сенсорних даних, реалізації модуля попередньої обробки, формуванню додаткових ознак, побудові базової моделі машинного навчання та нейромережевої моделі Autoencoder.

3.1. Реалізація програмної частини системи

Серверна Програмна частина системи реалізована мовою програмування Python із використанням бібліотек для аналізу даних, машинного навчання та побудови нейронних мереж [22]. У межах проєкту було реалізовано два основні програмні модулі: модуль базової моделі anomaly detection та модуль нейромережевої моделі Autoencoder. Кожен із цих модулів виконує повний цикл обробки даних: завантаження вхідного набору, попередню підготовку, формування ознак, нормалізацію, навчання моделі, оцінювання ефективності та збереження графічних результатів.

Обробка даних у системі відбувається послідовно. Спочатку із файлу sensor_data.csv завантажуються часові ряди показників сенсорів SensorA, SensorB та SensorC. Після цього виконується аналіз структури набору даних, визначаються числові ознаки, перевіряється наявність пропущених значень та виконується їх заповнення. Наступним етапом є очищення даних від викидів за допомогою методу міжквартильного розмаху, що дозволяє зменшити вплив надмірних відхилень на процес навчання моделей.

Після очищення набору даних у програмному комплексі реалізовано етап

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		38

формування додаткових ознак. Для кожного запису обчислюються статистичні характеристики сенсорних показників, зокрема середнє значення, стандартне відхилення, мінімальне та максимальне значення, а також діапазон між ними [4]. Додатково для кожного сенсора формуються ознаки відхилення від середнього значення та різниці між сусідніми часовими вимірюваннями. Такий підхід дозволяє розширити вхідний простір ознак та надати моделям більше інформації про поведінку сенсорної системи.

У системі також виділено окремі етапи для нормалізації даних, формування псевдоміток аномалій, навчання моделей та візуалізації результатів. Для нормалізації використовується StandardScaler, який приводить усі числові ознаки до єдиного масштабу. Для початкового визначення аномальних значень застосовується алгоритм Isolation Forest, результати якого використовуються як псевдомітки для подальшого навчання та оцінювання моделей.

У програмній частині реалізовано два підходи до виявлення аномалій. Перший підхід базується на використанні Logistic Regression як базової моделі класифікації. Другий підхід використовує нейромережеву модель Autoencoder, яка визначає аномальні стани на основі похибки реконструкції вхідних даних. Наявність двох моделей дозволяє виконати порівняльний аналіз класичного алгоритму машинного навчання та нейромережевого підходу.

Завершальним етапом роботи програмного комплексу є формування результатів. Для кожної моделі виводяться однакові метрики якості, зокрема Accuracy, Precision, Recall та F1-score, а також будується матриця помилок. Крім числового виводу, система автоматично зберігає графічні матеріали у директорію results, що дозволяє використовувати отримані зображення для подальшого аналізу та оформлення дипломної роботи.

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		39

3.1.1. Структура програмного комплексу

Програмний комплекс реалізовано мовою програмування Python та організовано у вигляді набору окремих файлів і каталогів, кожен з яких виконує визначену функцію в процесі виявлення аномалій у даних сенсорних систем. Такий підхід забезпечує зрозумілу структуру проєкту, спрощує супровід програмного коду та дозволяє розділити окремі етапи обробки даних. Структуру програмного комплексу наведено на рисунку 3.1.

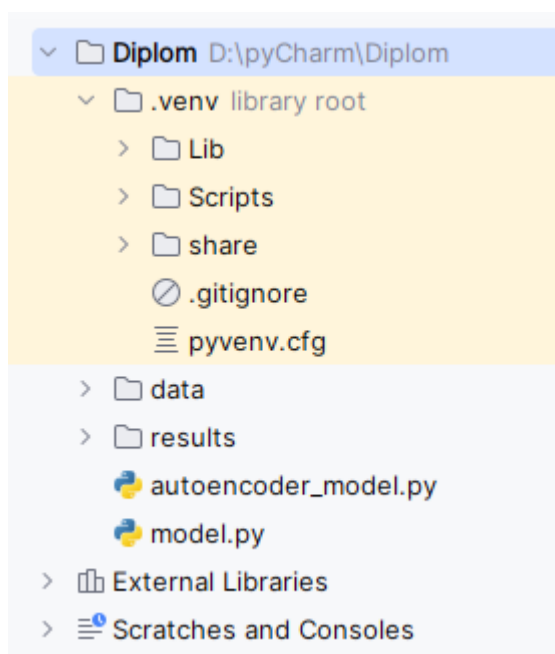


Рисунок 3.1 – Структура програмного комплексу

Каталог data використовується для зберігання вхідного набору даних sensor_data.csv, який містить показники сенсорів та використовується для навчання й тестування моделей машинного навчання.

Каталог results призначений для автоматичного збереження результатів роботи програмного комплексу. До нього записуються матриці помилок, графіки оцінювання моделей, результати навчання Autoencoder та інші візуалізації, необхідні для подальшого аналізу ефективності алгоритмів.

Файл `model.py` містить реалізацію базової моделі виявлення аномалій на основі алгоритму Logistic Regression. У межах даного модуля виконуються завантаження та попередня обробка даних, формування додаткових ознак, генерація псевдоміток за допомогою алгоритму Isolation Forest, навчання моделі та оцінювання її ефективності.

Файл `autoencoder_model.py` реалізує неймережевий підхід до виявлення аномалій із використанням архітектури Autoencoder. У цьому модулі здійснюється підготовка даних, побудова та навчання нейронної мережі, визначення порогового значення для класифікації аномалій, а також формування графічних матеріалів для аналізу результатів.

Організація проєкту у вигляді окремих модулів дозволяє незалежно досліджувати різні підходи до виявлення аномалій та виконувати їх подальше порівняння на однаковому наборі даних.

3.1.2. Реалізація обробки даних та навчання моделей

Робота програмного комплексу побудована як послідовність етапів обробки даних, починаючи із завантаження вхідного набору даних та завершуючи навчанням моделей машинного навчання і формуванням результатів. Такий підхід дозволяє забезпечити однакові умови для роботи різних алгоритмів виявлення аномалій та виконати їх подальше коректне порівняння.

На першому етапі роботи програмного комплексу виконується завантаження набору даних `sensor_data.csv`. Для цього використовується бібліотека Pandas, яка забезпечує зручні засоби роботи з табличними даними. Після зчитування файлу здійснюється аналіз структури набору даних, визначення типів ознак, кількості записів та наявності пропущених значень. Фрагмент програмного коду, що відповідає за завантаження набору даних,

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		41

наведено на рисунку 3.2.

```
print_header("DATA LOADING")

df = pd.read_csv(DATA_PATH)

print("Dataset shape before cleaning:", df.shape)
print("Columns:", df.columns.tolist())
```

Рисунок 3.2 – Завантаження набору даних

Після завантаження набору даних виконується попередня обробка інформації. Одним із важливих етапів є усунення пропущених значень, оскільки їх наявність може негативно впливати на процес навчання моделей машинного навчання. Для кожної числової ознаки пропущені значення замінюються медіанним значенням відповідного стовпця. Використання медіани дозволяє зменшити вплив можливих викидів та забезпечити більш стійку обробку даних. Реалізацію даного етапу наведено на рисунку 3.3.

```
numeric_cols = df.select_dtypes(include=[np.number]).columns
df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].median())
```

Рисунок 3.3 – Заповнення пропущених значень

Після усунення пропусків виконується очищення набору даних від аномально великих або малих значень. Для цього використовується метод міжквартильного розмаху (Interquartile Range, IQR). На основі першого та третього квартилів для кожної числової ознаки визначається допустимий діапазон значень. Записи, що виходять за межі цього діапазону, виключаються з подальшого аналізу. Такий підхід дозволяє зменшити вплив екстремальних значень на процес навчання моделей та покращити якість отриманих

результатів. Фрагмент програмного коду наведено на рисунку 3.4.обробки.

```
Q1 = numeric_df.quantile(0.25)
Q3 = numeric_df.quantile(0.75)
IQR = Q3 - Q1

outlier_mask = ~(
    (numeric_df < (Q1 - 1.5 * IQR)) |
    (numeric_df > (Q3 + 1.5 * IQR))
).any(axis=1)
```

Рисунок 3.4 –Видалення викидів методом IQR

Після очищення набору даних виконується формування додаткових ознак. Використання лише початкових значень сенсорів не завжди дозволяє моделі повною мірою виявити закономірності в даних, тому в програмному комплексі реалізовано етап *feature engineering*. Для кожного запису обчислюються статистичні характеристики, зокрема середнє значення показників сенсорів, стандартне відхилення, мінімальне та максимальне значення, а також діапазон між максимальним і мінімальним значеннями.

Крім цього, для кожного сенсора додатково обчислюється відхилення від середнього значення та різниця між сусідніми вимірюваннями. Такі ознаки дозволяють враховувати не лише абсолютні значення сенсорів, але й їхню поведінку в часі. Формування додаткових ознак підвищує інформативність набору даних та позитивно впливає на якість роботи моделей машинного навчання. Фрагмент програмного коду, що відповідає за створення додаткових ознак, наведено на рисунку 3.5.

					ІАЛЦ.467200.003 ПЗ	Арк.
						43
Зм.	Арк.	№ докум.	Підпис	Дата		

```
df_clean["sensor_mean"] = df_clean[sensor_cols].mean(axis=1)
df_clean["sensor_std"] = df_clean[sensor_cols].std(axis=1)
df_clean["sensor_min"] = df_clean[sensor_cols].min(axis=1)
df_clean["sensor_max"] = df_clean[sensor_cols].max(axis=1)
df_clean["sensor_range"] = df_clean["sensor_max"] - df_clean["sensor_min"]
```

Рисунок 3.5 – Формування додаткових ознак

Після завершення формування ознак виконується нормалізація даних. Необхідність цього етапу пов'язана з тим, що різні ознаки можуть мати різні діапазони значень, що негативно впливає на процес навчання моделей. Для усунення цієї проблеми використовується алгоритм StandardScaler, який виконує стандартизацію даних шляхом віднімання середнього значення та ділення на стандартне відхилення.

У результаті всі ознаки приводяться до єдиного масштабу, що забезпечує стабільніше навчання моделей та покращує якість отриманих результатів. Реалізацію нормалізації даних наведено на рисунку 3.6.

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Рисунок 3.6 – Нормалізація даних

Для формування початкових міток аномалій у програмному комплексі використовується алгоритм Isolation Forest. Даний алгоритм належить до методів неконтрольованого навчання та дозволяє визначати аномальні спостереження без необхідності використання заздалегідь розміченого набору даних.

Після навчання моделі кожному запису призначається мітка, яка визначає його належність до нормальних або аномальних спостережень. Отримані результати використовуються як псевдомітки для подальшого навчання та

оцінювання моделей виявлення аномалій. Фрагмент програмного коду, що реалізує генерацію псевдоміток, наведено на рисунку 3.7.

```
iso_model = IsolationForest(
    n_estimators=200,
    contamination=0.05,
    random_state=RANDOM_STATE
)

pseudo_labels = iso_model.fit_predict(X_scaled)
pseudo_labels = np.where(pseudo_labels == -1, x: 1, y: 0)
```

Рисунок 3.7 – Генерація псевдоміток аномалій

Після завершення підготовки даних виконується навчання базової моделі виявлення аномалій. У межах програмного комплексу як базову модель використано Logistic Regression. Перед початком навчання набір даних розділяється на тренувальну та тестову вибірки, що дозволяє виконати незалежне оцінювання якості моделі.

Навчена модель використовується для класифікації записів на нормальні та аномальні. Для оцінювання ефективності її роботи надалі використовуються метрики Accuracy, Precision, Recall та F1-score. Фрагмент програмного коду реалізації Logistic Regression наведено на рисунку 3.8.

					ІАЛЦ.467200.003 ПЗ	Арк.
						45
Зм.	Арк.	№ докум.	Підпис	Дата		

```

X_train, X_test, y_train, y_test = train_test_split(
    *arrays: X_scaled,
    df_clean["pseudo_target"],
    test_size=0.2,
    random_state=RANDOM_STATE,
    stratify=df_clean["pseudo_target"]
)

model = LogisticRegression(
    class_weight="balanced",
    max_iter=1000,
    random_state=RANDOM_STATE
)

```

Рисунок 3.8 – Реалізація моделі Logistic Regression

Окрім базової моделі, у програмному комплексі реалізовано неймережевий підхід до виявлення аномалій із використанням архітектури Autoencoder. На відміну від Logistic Regression, дана модель навчається відновлювати структуру нормальних даних. Під час роботи мережа отримує на вхід набір ознак та намагається відтворити їх на виході. Чим більшою є похибка реконструкції, тим вищою є ймовірність того, що спостереження належить до аномальних.

Архітектура реалізованої нейронної мережі складається з послідовності прихованих шарів із поступовим зменшенням та подальшим збільшенням кількості нейронів. Така структура дозволяє формувати компактне приховане представлення даних та виявляти нетипові патерни поведінки сенсорної системи. Реалізацію архітектури Autoencoder наведено на рисунку 3.9.

```

encoded = Dense(units=32, activation="relu")(input_layer)
encoded = BatchNormalization()(encoded)
encoded = Dense(units=16, activation="relu")(encoded)
encoded = BatchNormalization()(encoded)
encoded = Dense(units=8, activation="relu")(encoded)

decoded = Dense(units=16, activation="relu")(encoded)
decoded = BatchNormalization()(decoded)
decoded = Dense(units=32, activation="relu")(decoded)
decoded = BatchNormalization()(decoded)
decoded = Dense(input_dim, activation="linear")(decoded)

```

Рисунок 3.9 – Архітектура моделі Autoencoder

3.2. Проєктування та підготовка набору даних

Набір даних є основою роботи програмного комплексу, оскільки саме на його основі виконується навчання та тестування моделей виявлення аномалій. У межах дипломної роботи використовується файл `sensor_data.csv`, який містить часові ряди показників сенсорної системи. Дані представлені у табличному вигляді, де кожен рядок відповідає окремому вимірюванню, а кожен стовпець містить певну характеристику цього вимірювання.

Початковий набір даних складається з часової мітки та трьох сенсорних показників: `SensorA`, `SensorB` та `SensorC`. Така структура дозволяє аналізувати зміну параметрів системи у часі та виявляти нетипові відхилення у поведінці сенсорів. Оскільки дані сенсорних систем можуть містити пропущені значення, шуми та викиди, перед навчанням моделей необхідно виконати їх попередню підготовку.

Процес підготовки набору даних включає декілька послідовних етапів: аналіз структури вхідного файлу, заповнення пропущених значень, видалення викидів, формування додаткових ознак та нормалізацію числових параметрів. Виконання цих етапів дозволяє підвищити якість вхідної інформації та

забезпечити стабільну роботу моделей машинного навчання.

Особливістю використаного набору даних є те, що початкові сенсорні показники доповнюються новими ознаками, сформованими під час feature engineering. Завдяки цьому моделі отримують не лише початкові значення сенсорів, але й додаткову інформацію про їх статистичні характеристики та зміну у часі. Це дозволяє підвищити інформативність вхідних даних та покращити якість виявлення аномалій.

Таким чином, підготовка набору даних є важливим етапом реалізації програмного комплексу, оскільки від якості попередньої обробки залежить ефективність подальшого навчання моделей Logistic Regression та Autoencoder.

3.2.1. Загальна структура набору даних

Початковий набір даних має табличну структуру та містить чотири основні стовпці: time, SensorA, SensorB та SensorC. Стовпець time використовується для збереження часової мітки вимірювання, а SensorA, SensorB і SensorC містять числові значення відповідних сенсорів. Загальна структура початкового набору даних наведена у таблиці 3.1.чату.

Таблиця 3.1 – Основні поля початкового набору даних

Поле	Призначення
time	Часова мітка вимірювання
SensorA	Значення першого сенсора
SensorB	Значення другого сенсора
SensorC	Значення третього сенсора

У таблиці 3.1 наведено основні поля, що містяться у файлі sensor_data.csv. Часова мітка дозволяє зберігати порядок вимірювань, а числові сенсорні показники використовуються як основа для подальшого аналізу. Оскільки моделі машинного навчання працюють із числовими ознаками, основна увага під час обробки приділяється саме показникам SensorA, SensorB та SensorC.

Після завантаження набору даних у програмі виконується перевірка його розмірності, назв стовпців та типів даних. Це дозволяє переконатися, що файл має очікувану структуру та може бути використаний для подальшої обробки. У результаті аналізу було встановлено, що початковий набір даних містить 1440 записів і 4 стовпці.

Після етапу очищення даних від викидів кількість записів зменшилася до 1415. Це свідчить про те, що частина рядків містила надмірні відхилення та була виключена з подальшого аналізу. Такий підхід дозволяє зменшити вплив випадкових або некоректних значень на процес навчання моделей.

Для підвищення інформативності набору даних у програмному комплексі додатково формуються нові ознаки. Вони характеризують статистичні властивості сенсорних вимірювань та допомагають моделям краще виявляти приховані закономірності у даних. Перелік основних додаткових ознак наведено у таблиці 3.2

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		49

Таблиця 3.2 –Основні поля початкового набору даних

Ознака	Призначення
sensor_mean	Середнє значення показників сенсорів
sensor_std	Стандартне відхилення між показниками сенсорів
sensor_min	Мінімальне значення серед сенсорів
sensor_max	Максимальне значення серед сенсорів
sensor_range	Різниця між максимальним і мінімальним значенням
*_dev	Відхилення значення конкретного сенсора від середнього
*_delta	Різниця між поточним і попереднім значенням сенсора

Додаткові ознаки дозволяють моделі оцінювати не лише абсолютні значення окремих сенсорів, але й взаємозв'язки між ними. Наприклад, sensor_range дає змогу визначити, наскільки сильно відрізняються показники сенсорів у конкретний момент часу, а *_delta дозволяє враховувати зміну показників у часовій послідовності.

3.2.2. Опис основних ознак набору даних

Після завантаження та попередньої обробки набір даних містить як початкові сенсорні показники, так і додаткові ознаки, сформовані під час feature engineering. Кожна з ознак виконує окрему функцію та використовується моделями машинного навчання для виявлення аномалій.

Ознака time містить часову мітку вимірювання. Вона використовується

для збереження послідовності спостережень та дозволяє аналізувати зміну показників сенсорів у часі. Хоча дана ознака безпосередньо не використовується під час навчання моделей, вона є важливою для побудови часових графіків та аналізу результатів.

Ознаки SensorA, SensorB та SensorC містять початкові значення сенсорів. Вони є основними джерелами інформації про стан контрольованої системи. Значення кожного сенсора можуть змінюватися в залежності від умов роботи об'єкта спостереження. Саме на основі аналізу цих параметрів моделі виконують пошук аномальних станів.

Для підвищення інформативності набору даних додатково формується ознака `sensor_mean`, яка містить середнє значення всіх сенсорних показників для кожного вимірювання. Використання цієї ознаки дозволяє моделі оцінювати загальний рівень параметрів системи та виявляти випадки суттєвого відхилення окремих сенсорів від середнього значення.

Ознака `sensor_std` містить стандартне відхилення між показниками сенсорів. Вона характеризує ступінь розкиду значень та дозволяє виявляти ситуації, коли один із сенсорів починає поводитися нетипово порівняно з іншими. Зростання значення `sensor_std` може свідчити про виникнення аномальної ситуації або появу несправності в роботі сенсорної системи.

Ознаки `sensor_min` та `sensor_max` містять відповідно мінімальне та максимальне значення серед усіх сенсорів для конкретного вимірювання. Дані параметри використовуються для визначення меж зміни показників системи та дозволяють отримати додаткову інформацію про поточний стан об'єкта спостереження.

На основі ознак `sensor_min` та `sensor_max` формується показник `sensor_range`, який характеризує різницю між максимальним та мінімальним значенням сенсорів. Велике значення даного параметра може свідчити про значну невідповідність між окремими вимірюваннями та використовуватись як

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		51

додатковий індикатор аномальної поведінки.

Для кожного сенсора також формуються ознаки типу *_dev. Вони відображають різницю між значенням конкретного сенсора та середнім значенням усіх сенсорів у межах одного вимірювання. Такі ознаки дозволяють моделі оцінювати внесок кожного окремого сенсора у загальну картину спостереження.

Окрім цього, для кожного сенсора розраховуються ознаки типу *_delta. Дані параметри містять різницю між поточним та попереднім значенням сенсора. Використання таких ознак дозволяє враховувати динаміку зміни параметрів системи та підвищує здатність моделей виявляти різкі відхилення у часових рядах.

Застосування додаткових ознак дозволило розширити початковий простір даних та надати моделям більше інформації для аналізу. У результаті цього підвищується якість навчання алгоритмів та покращується ефективність виявлення аномальних спостережень.

3.3. Реалізація оцінювання та аналіз результатів

Після завершення навчання моделей виконується оцінювання якості їх роботи. Основною метою даного етапу є визначення ефективності виявлення аномалій та порівняння результатів різних підходів машинного навчання. У межах програмного комплексу реалізовано єдину систему оцінювання для моделей Logistic Regression та Autoencoder, що забезпечує можливість їх коректного порівняння.

Для оцінювання якості класифікації використовуються стандартні метрики машинного навчання: Accuracy, Precision, Recall та F1-score. Використання декількох показників дозволяє більш об'єктивно оцінити роботу моделей, оскільки кожна метрика характеризує окремий аспект якості

					ІАЛЦ.467200.003 ПЗ	Арк.
						52
Зм.	Арк.	№ докум.	Підпис	Дата		

класифікації.

Показник Accuracy визначає частку правильно класифікованих записів відносно загальної кількості спостережень. Дана метрика дозволяє оцінити загальну якість роботи моделі, проте в задачах виявлення аномалій її використання є недостатнім через нерівномірний розподіл класів.

Метрика Precision характеризує точність виявлення аномалій та показує, яка частка записів, визначених моделлю як аномальні, дійсно належить до класу аномалій. Високе значення Precision свідчить про невелику кількість хибних спрацьовувань системи.

Метрика Recall визначає здатність моделі знаходити реальні аномалії. Вона показує, яка частка фактичних аномалій була правильно виявлена моделлю. Даний показник є особливо важливим для задач моніторингу та діагностики, де пропуск аномалії може призвести до негативних наслідків.

Для комплексної оцінки використовується метрика F1-score, яка поєднує Precision та Recall в один показник. Саме F1-score використовується як основний критерій порівняння реалізованих моделей, оскільки він дозволяє оцінити баланс між точністю та повнотою виявлення аномалій.

3.3.1. Оцінювання моделі Logistic Regression

Після навчання моделі Logistic Regression виконується прогнозування для тестової вибірки та розрахунок основних показників якості класифікації. Отримані результати дозволяють оцінити здатність моделі відокремлювати нормальні спостереження від аномальних.

Для наочного аналізу результатів використовується матриця помилок (Confusion Matrix). Вона відображає кількість правильно та неправильно класифікованих записів для кожного класу. Матриця помилок моделі Logistic Regression наведена на рисунку 3.10.

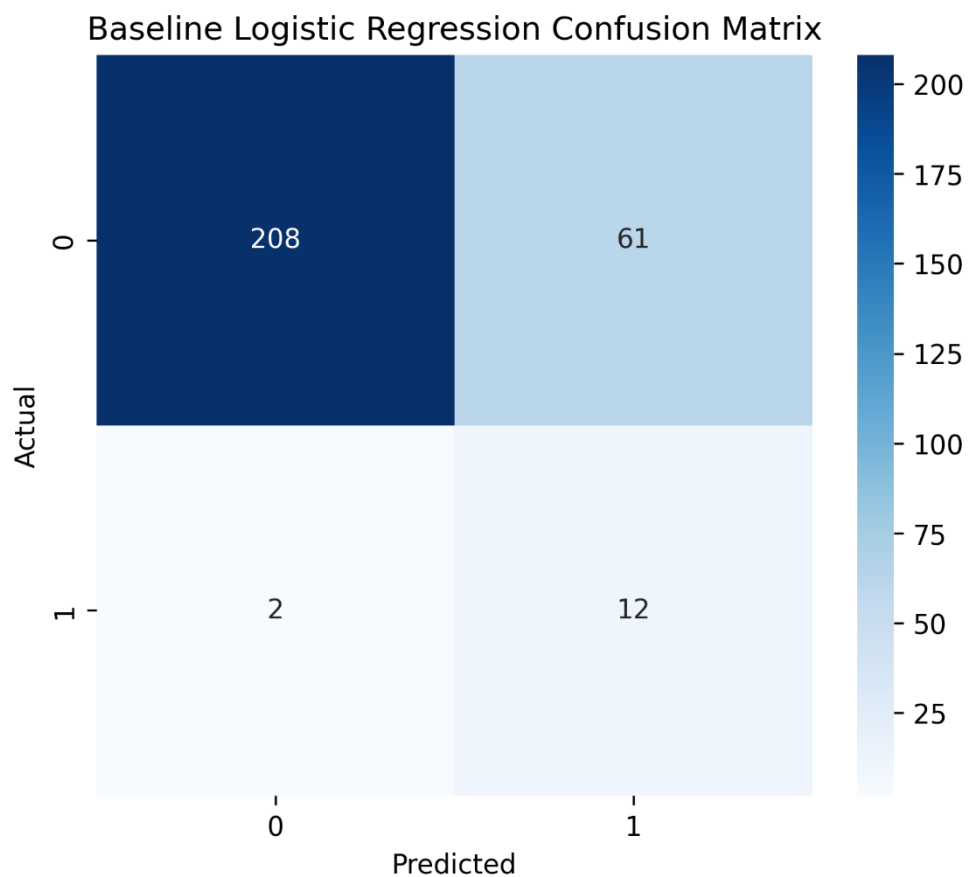


Рисунок 3.10 – Матриця помилок моделі Logistic Regression

На основі результатів класифікації обчислюються значення Accuracy, Precision, Recall та F1-score. Отримані показники наведено у таблиці 3.3.

Таблиця 3.3 – Метрики якості моделі Logistic Regression

Метрика	Значення
Accuracy	0.7774
Precision	0.1644
Recall	0.8571
F1-score	0.2759

З отриманих результатів видно, що модель забезпечує високе значення Recall, тобто успішно виявляє більшість аномальних спостережень. Водночас значення Precision залишається невисоким, що свідчить про наявність певної кількості хибних спрацьовувань. У результаті значення F1-score становить 0.2759, що використовується як базовий рівень для подальшого порівняння з нейромережевою моделлю.

3.3.2. Аналіз результатів моделі Logistic Regression

Після навчання нейромережевої моделі Autoencoder було проведено її оцінювання на тестовій вибірці. На відміну від Logistic Regression, дана модель визначає аномалії на основі помилки реконструкції вхідних даних. Чим більшою є різниця між початковими даними та результатом їх відновлення, тим вищою є ймовірність належності об'єкта до класу аномалій.

Для аналізу результатів класифікації було побудовано матрицю помилок, яка дозволяє оцінити кількість правильно та неправильно визначених спостережень. Результати роботи моделі наведено на рисунку 3.11.

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		55

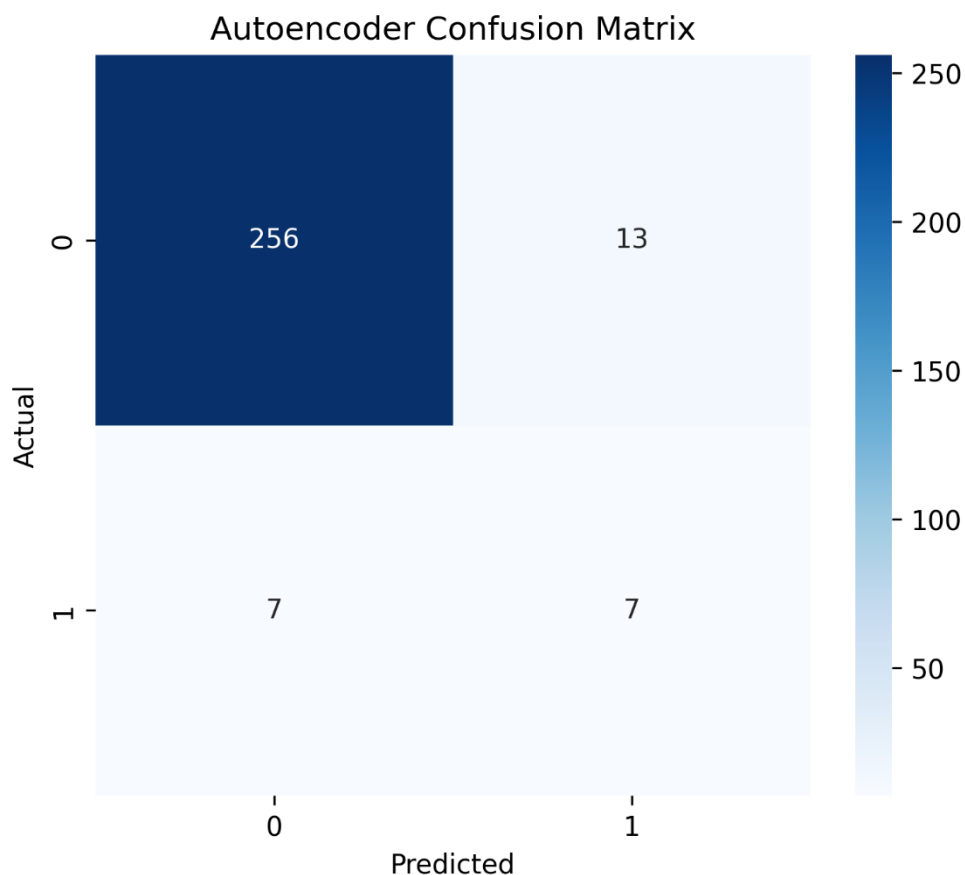


Рисунок 3.11 – Матриця помилок моделі Autoencoder

З рисунка 3.11 видно, що модель правильно класифікувала 256 нормальних об’єктів та 7 аномальних. Кількість хибнопозитивних спрацювань становить 13 випадків, тоді як кількість пропущених аномалій дорівнює 7. У порівнянні з Logistic Regression спостерігається суттєве зменшення кількості хибних спрацювань, що позитивно впливає на загальну якість класифікації.

Для більш детального аналізу було розраховано основні метрики оцінювання моделі. Отримані результати наведено на рисунку 3.12.

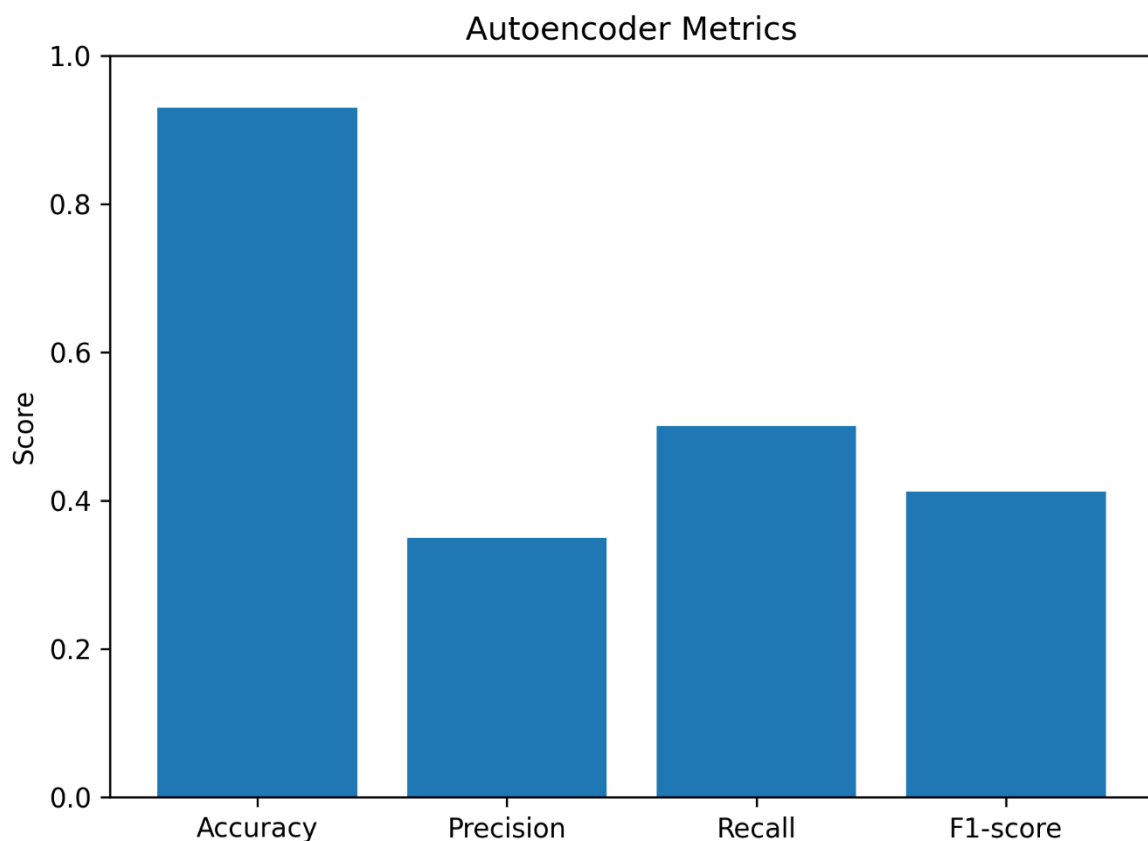


Рисунок 3.12 – Основні метрики моделі Autoencoder

Відповідно до отриманих результатів, модель забезпечила Accuracy на рівні близько 93 %, Precision — 35 %, Recall — 50 %, а значення F1-score становило близько 41 %. У порівнянні з базовою моделлю Logistic Regression спостерігається суттєве покращення показників Accuracy, Precision та F1-score.

Особливо важливим є зростання значення F1-score з 0,2759 до 0,4118 після вдосконалення архітектури Autoencoder. Це свідчить про більш збалансовану роботу моделі та кращу здатність одночасно забезпечувати високу точність і повноту виявлення аномалій.

Отримані результати підтверджують ефективність використання неймережевого підходу для задачі виявлення аномалій у сенсорних даних та демонструють перевагу Autoencoder над базовою моделлю Logistic Regression за більшістю показників якості.

3.3.3. Аналіз процесу навчання та помилки реконструкції Autoencoder

Однією з переваг неймережевої моделі Autoencoder є можливість контролювати процес її навчання та аналізувати якість реконструкції вхідних даних. Для цього під час навчання фіксуються значення функції втрат, а після завершення навчання обчислюється помилка реконструкції для кожного спостереження.

Функція втрат використовується для оцінювання різниці між початковими даними та результатом їх відновлення нейронною мережею. Чим меншим є значення функції втрат, тим точніше модель відтворює структуру вхідних даних. Для аналізу процесу навчання було побудовано графік зміни значень функції втрат на навчальній та валідаційній вибірках. Результати наведено на рисунку 3.13.

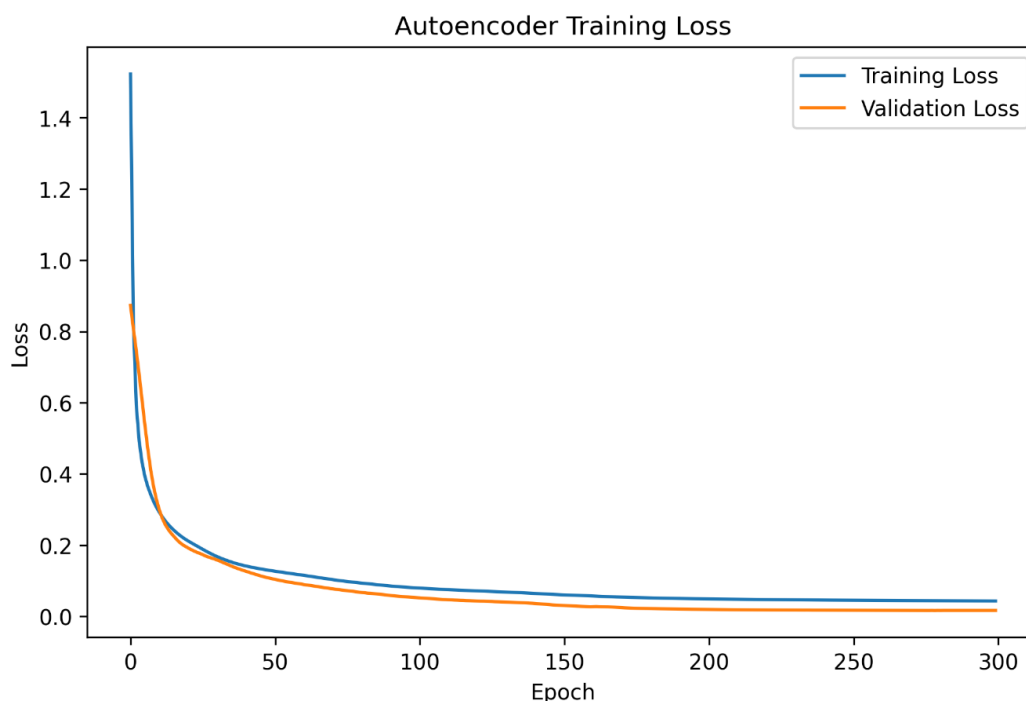


Рисунок 3.13 – Графік зміни функції втрат Autoencoder під час навчання

З рисунка 3.13 видно, що найбільше зменшення функції втрат відбувається на початкових етапах навчання. У перші епохи модель активно адаптується до структури вхідних даних та швидко покращує якість реконструкції. Надалі швидкість зниження помилки поступово зменшується, а криві навчальної та валідаційної вибірок наближаються до стабільних значень.

Важливою особливістю отриманого результату є відсутність значного розходження між навчальною та валідаційною помилками. Це свідчить про те, що модель не демонструє вираженого перенавчання та зберігає здатність до узагальнення на нових даних. Отриманий результат підтверджує коректність обраної архітектури Autoencoder та параметрів навчання.

Після завершення навчання для кожного спостереження було розраховано помилку реконструкції. Даний показник характеризує різницю між початковими значеннями ознак та результатом їх відновлення нейронною мережею. Передбачається, що для нормальних спостережень значення помилки реконструкції є невеликим, тоді як для аномальних об'єктів воно суттєво зростає.

Для аналізу отриманих результатів було побудовано розподіл помилки реконструкції. Відповідний графік наведено на рисунку 3.14.

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		59

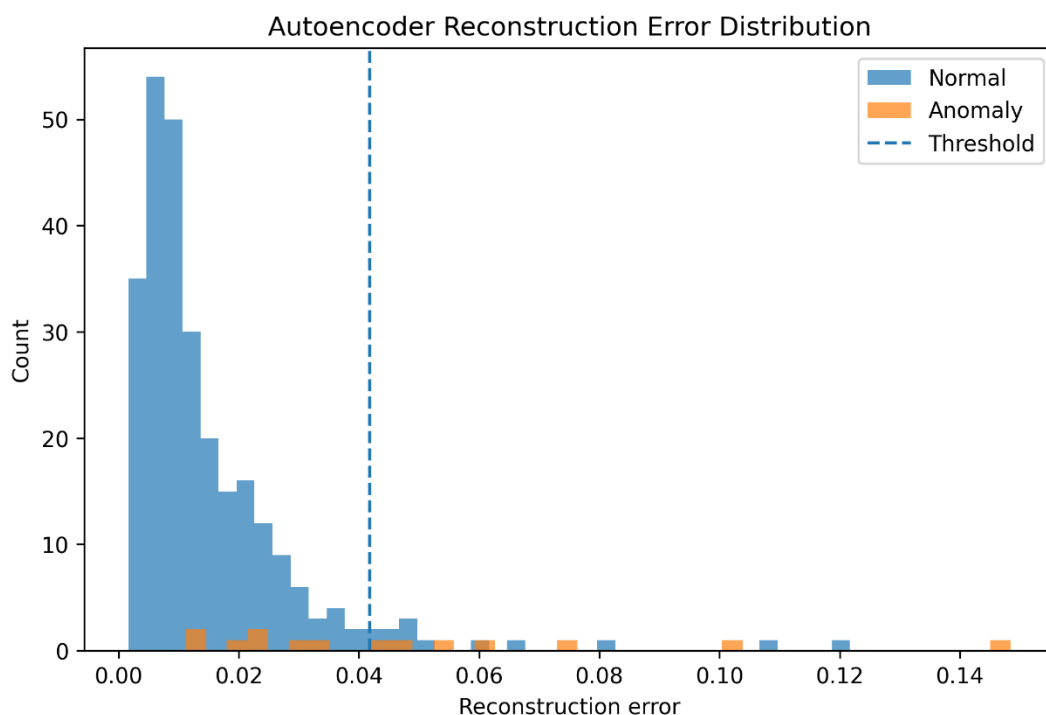


Рисунок 3.14 – Розподіл помилки реконструкції Autoencoder

На рисунку 3.14 видно, що більшість нормальних спостережень характеризується невеликими значеннями помилки реконструкції. Основна частина таких об'єктів зосереджена в області низьких значень помилки, що свідчить про успішне відтворення їх структури нейронною мережею.

Для аномальних спостережень характерні значно вищі значення помилки реконструкції. Саме ця особливість використовується для відокремлення нормальних та аномальних об'єктів. На основі аналізу розподілу визначається порогове значення, перевищення якого свідчить про наявність потенційної аномалії.

Отримані результати підтверджують ефективність використання помилки реконструкції як критерію виявлення аномалій. Поєднання нейромережевої архітектури Autoencoder та механізму аналізу помилки реконструкції дозволило досягти вищих показників якості порівняно з базовою моделлю Logistic Regression та забезпечити більш збалансоване співвідношення між точністю та повнотою виявлення аномальних спостережень.

ВИСНОВОК ДО РОЗДІЛУ 3

У третьому розділі було розглянуто практичну реалізацію програмного комплексу для виявлення аномалій у даних сенсорних систем. У ході роботи реалізовано модулі завантаження та попередньої обробки даних, формування додаткових ознак, навчання моделей машинного навчання та оцінювання отриманих результатів.

Було реалізовано дві моделі виявлення аномалій: Logistic Regression та Autoencoder. Для формування початкових міток аномалій використано алгоритм Isolation Forest. Також виконано очищення даних від пропущених значень і викидів, а також нормалізацію ознак для підвищення якості навчання моделей.

Проведене оцінювання показало, що модель Logistic Regression забезпечує високе значення Recall та успішно виявляє більшість аномалій. Водночас модель Autoencoder продемонструвала кращі показники Accuracy, Precision та F1-score, що свідчить про більш збалансовану якість класифікації.

Отримані результати підтверджують ефективність використання методів машинного навчання та нейронних мереж для задачі виявлення аномалій у сенсорних даних і можуть бути використані як основа для подальшого розвитку систем моніторингу та аналізу даних.

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		61

РОЗДІЛ 4 ПРЕДСТАВЛЕННЯ РОЗРОБЛЕНОЇ СИСТЕМИ

4.1. Демонстрація роботи розробленої системи

На рисунку 4.1 наведено основні метрики якості базової моделі логістичної регресії.

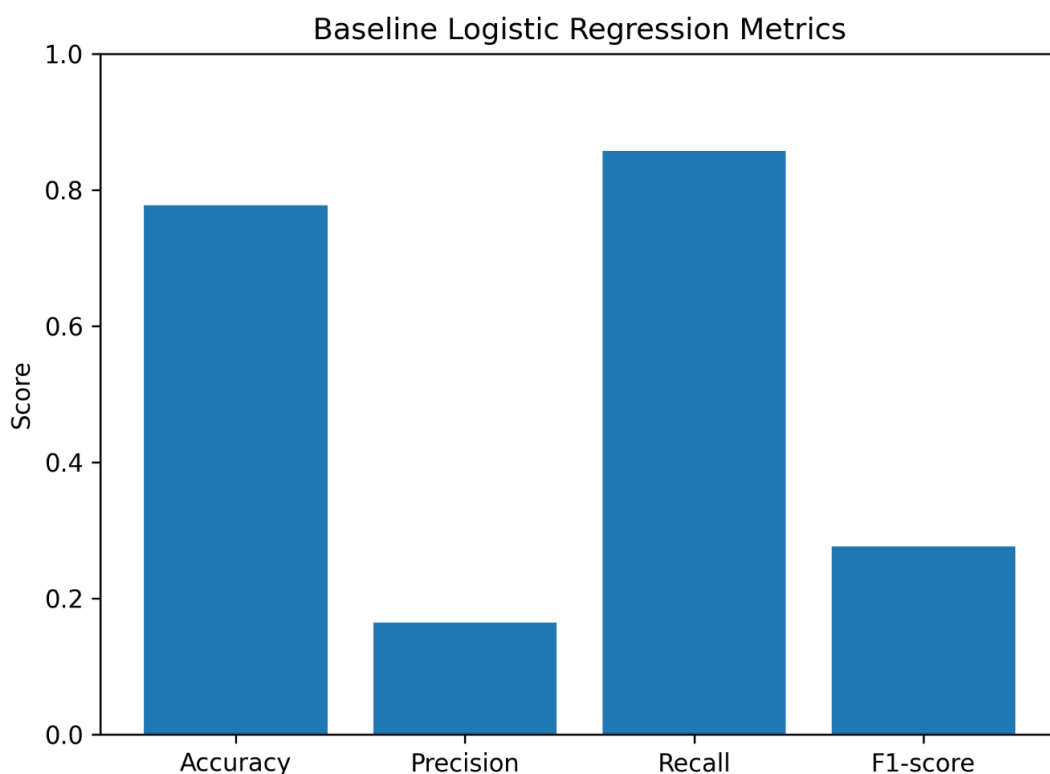


Рисунок 4.1 – Метрики якості моделі логістичної регресії

За результатами тестування модель досягла точності близько 78 %. Найкраще значення було отримано для показника Recall, що свідчить про здатність моделі виявляти більшість аномальних подій.

На рисунку 4.2 наведено метрики якості автоенкодера.

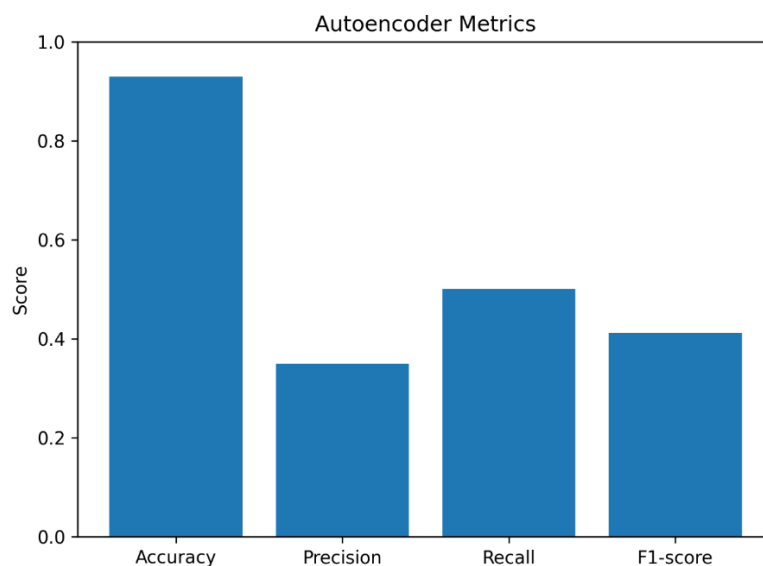


Рисунок 4.2 – Метрики якості автоенкодера

Порівняно з логістичною регресією автоенкодер продемонстрував вищу загальну точність класифікації. Це пояснюється здатністю нейронної мережі виявляти складні нелінійні залежності між ознаками.

На рисунку 4.3 наведено матрицю помилок логістичної регресії.

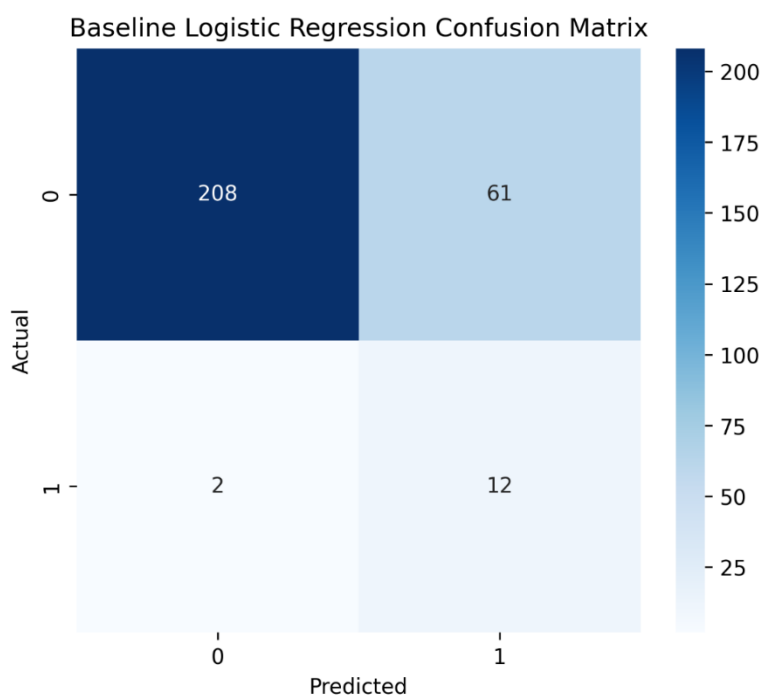


Рисунок 4.3 – Матриця помилок моделі логістичної регресії

Матриця помилок дозволяє оцінити кількість правильно та неправильно класифікованих об'єктів. Більшість нормальних спостережень була визначена коректно, однак частина аномалій залишилася невиявленою.

На рисунку 4.4 наведено матрицю помилок автоенкодера.

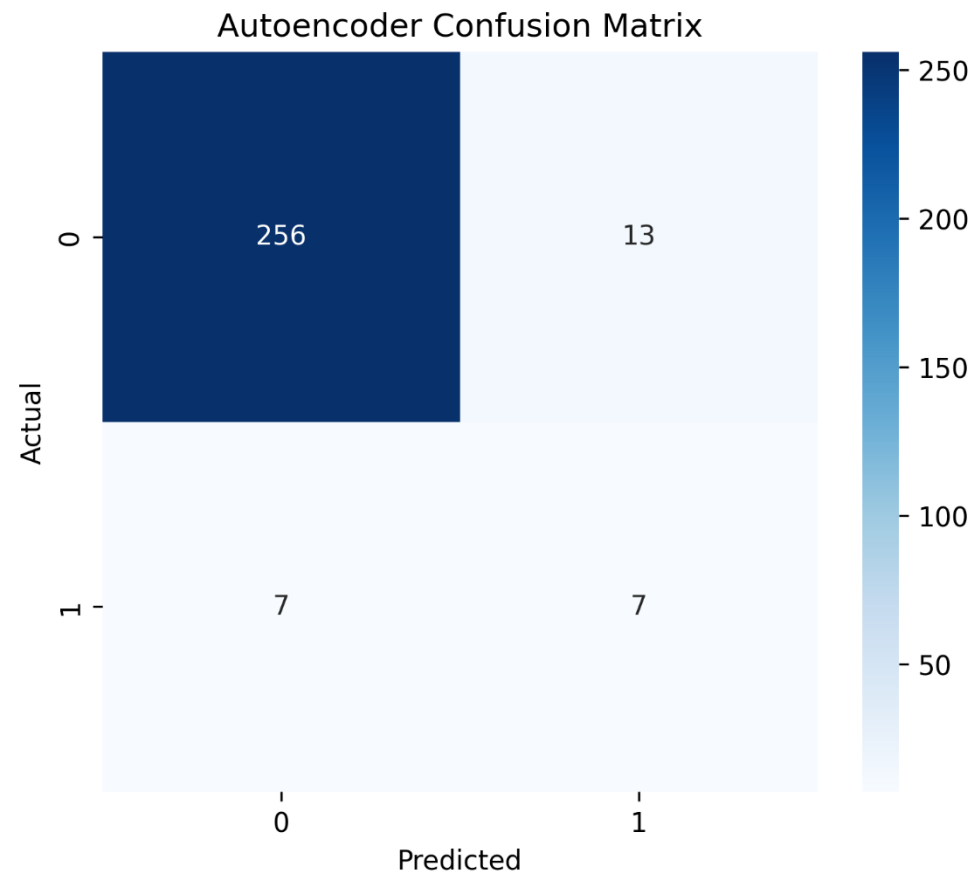


Рисунок 4.4 – Матриця помилок автоенкодера

Отримані результати свідчать про кращу здатність автоенкодера відокремлювати аномальні дані від нормальних порівняно з базовою моделлю.

4.2. Аналіз результатів роботи системи

На рисунку 4.5 наведено розподіл оцінок аномальності, отриманих за допомогою логістичної регресії.

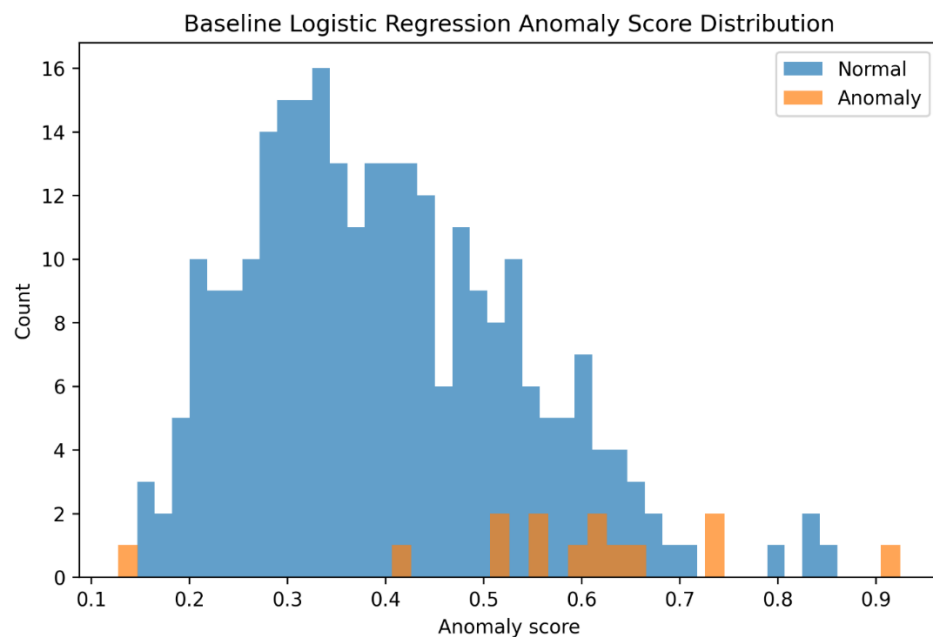


Рисунок 4.5 – Розподіл оцінок аномальності логістичної регресії

Графік демонструє розділення між нормальними та аномальними спостереженнями. Часткове накладання розподілів пояснює наявність помилок класифікації.

На рисунку 4.6 наведено розподіл помилок реконструкції автоенкодера.

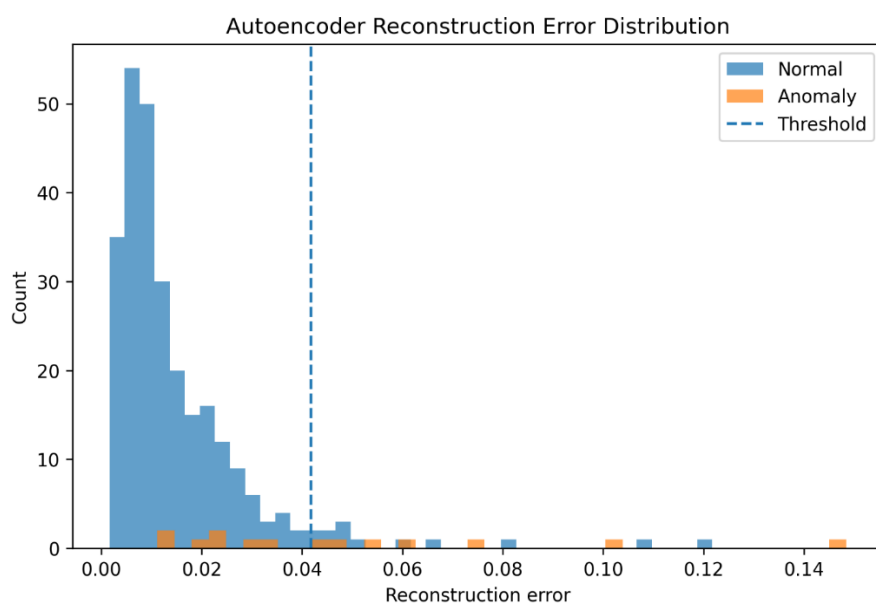


Рисунок 4.6 – Розподіл помилки реконструкції автоенкодера

Більшість нормальних спостережень характеризується низькою помилкою реконструкції, тоді як для аномальних даних значення помилки є суттєво вищими.

На рисунку 4.7 показано зміну помилки реконструкції для окремих тестових спостережень.

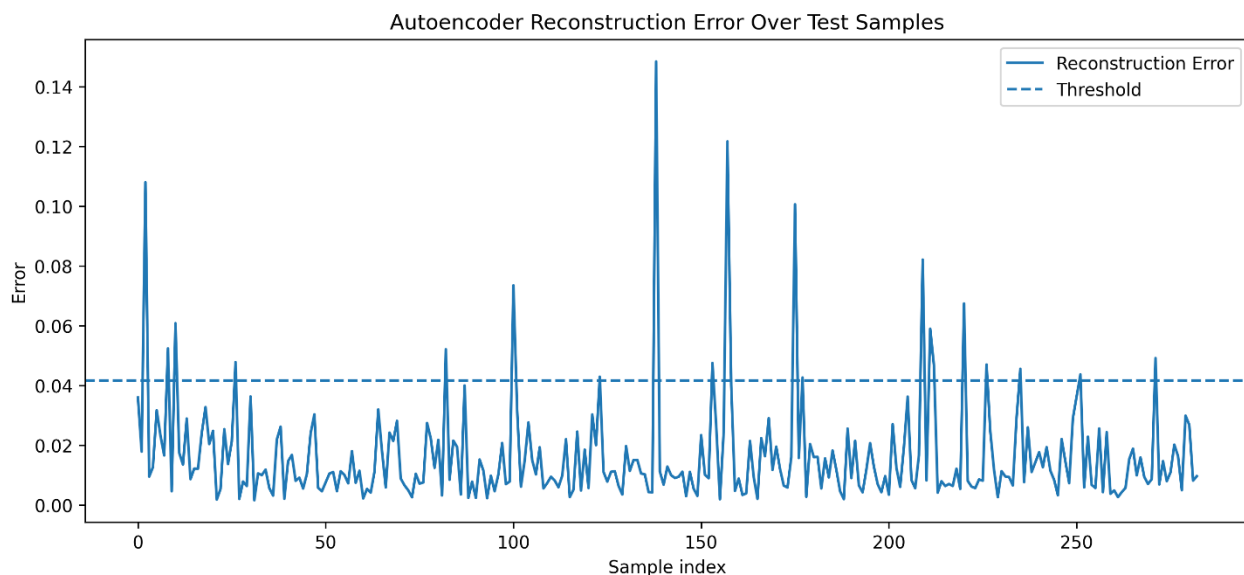


Рисунок 4.7 – Помилка реконструкції на тестовій вибірці

Перевищення встановленого порогового значення свідчить про наявність аномалії. Такий підхід дозволяє автоматично виявляти нетипові події у часовому ряді.

На рисунку 4.8 наведено процес навчання автоенкодера.

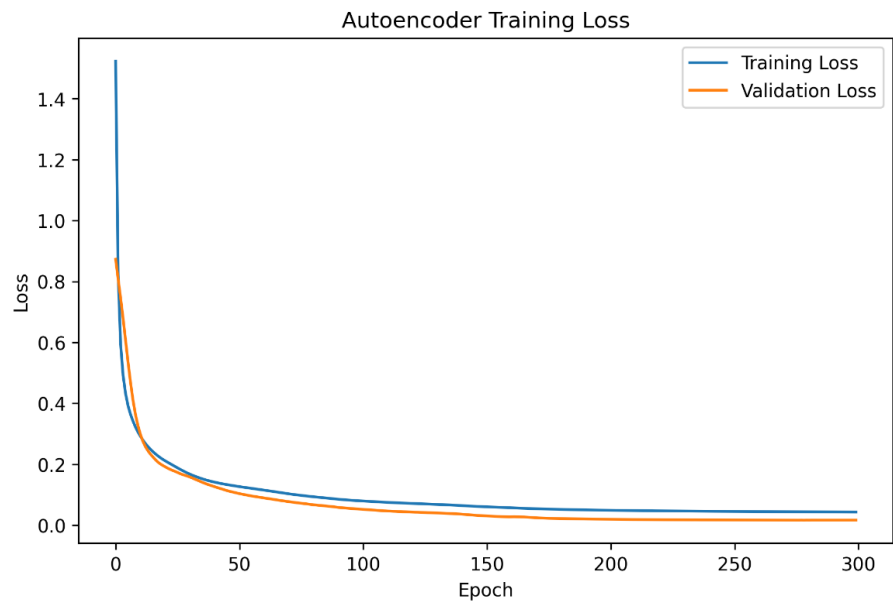


Рисунок 4.8 – Зміна функції втрат під час навчання автоенкодера

Зменшення значення функції втрат протягом епох навчання свідчить про успішне налаштування параметрів нейронної мережі та відсутність ознак нестабільності навчання.

На рисунку 4.9 наведено матрицю кореляції сформованих ознак.

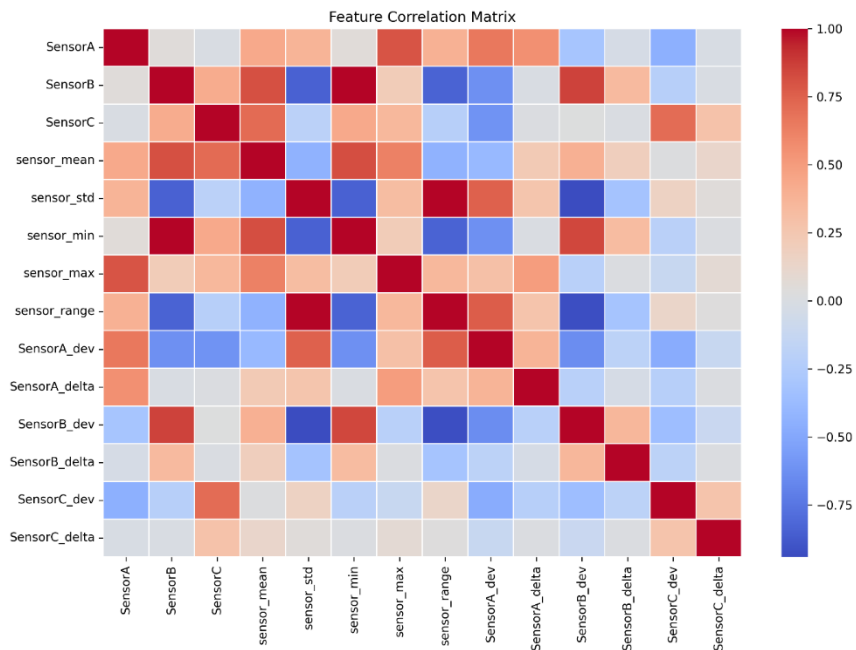


Рисунок 4.9 – Кореляційна матриця сформованих ознак

Аналіз кореляційної матриці підтверджує наявність взаємозв'язків між окремими параметрами, що дозволяє використовувати їх для побудови ефективних моделей виявлення аномалій.

					ІАЛЦ.467200.003 ПЗ	Арк.
						68
Зм.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВОК ДО РОЗДІЛУ 4

У четвертому розділі представлено результати роботи розробленої системи виявлення аномалій у часових рядах на основі методів машинного навчання. Для оцінювання ефективності системи було проведено тестування двох підходів: моделі логістичної регресії та автоенкодера.

Під час демонстрації роботи системи проаналізовано основні метрики якості моделей, матриці помилок, розподіли оцінок аномальності та процес навчання автоенкодера. Отримані результати показали, що обидві моделі здатні виявляти аномальні спостереження, проте автоенкодер продемонстрував вищу точність та кращу здатність працювати зі складними залежностями між ознаками.

Проведений аналіз також підтвердив коректність підготовки даних та ефективність сформованих ознак для задачі виявлення аномалій. Таким чином, розроблена система може бути використана для автоматизованого моніторингу даних та своєчасного виявлення нетипових подій у часових рядах.

					ІАЛЦ.467200.003 ПЗ	Арк.
						69
Зм.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

При виконанні дипломної роботи було розроблено систему виявлення аномалій у часових рядах із використанням методів машинного навчання. За результатами проведеного дослідження можна сформулювати такі висновки.

Проведено аналіз предметної області виявлення аномалій у часових рядах та розглянуто сучасні підходи до розв’язання даної задачі. Встановлено, що автоматичне виявлення аномальних подій є важливим елементом моніторингу технічних систем, промислового обладнання та інформаційних сервісів, оскільки дозволяє своєчасно виявляти відхилення від нормальної поведінки даних..

Проаналізовано існуючі методи машинного навчання для задачі виявлення аномалій. Особливу увагу приділено моделі логістичної регресії як базовому алгоритму класифікації та автоенкодеру як нейромережевому підходу до виявлення нетипових спостережень на основі помилки реконструкції.

Виконано підготовку та аналіз набору даних. Проведено попередню обробку даних, формування додаткових ознак, нормалізацію параметрів та дослідження взаємозв’язків між ознаками за допомогою кореляційного аналізу..

Розроблено програмну реалізацію системи виявлення аномалій мовою Python із використанням сучасних бібліотек машинного навчання та аналізу даних. Реалізовано повний цикл роботи з даними: завантаження, підготовку, навчання моделей, тестування та візуалізацію результатів.

Проведено навчання та оцінювання моделі логістичної регресії. Отримані результати показали здатність моделі виявляти аномальні спостереження та можуть використовуватися як базовий рівень для порівняння більш складних алгоритмів.

Реалізовано та досліджено модель автоенкодера для виявлення аномалій. Аналіз результатів показав, що автоенкодер ефективно відтворює нормальні

					ІАЛЦ.467200.003 ПЗ	Арк.
						70
Зм.	Арк.	№ докум.	Підпис	Дата		

спостереження та дозволяє виявляти аномалії за рахунок аналізу помилки реконструкції.

Проведено порівняння результатів роботи моделей за допомогою метрик якості та графічного аналізу. Отримані результати підтвердили можливість використання запропонованих підходів для автоматизованого виявлення аномалій у часових рядах та продемонстрували перспективність застосування нейромережових методів для розв’язання даної задачі.

Таким чином, поставлені у дипломній роботі завдання виконано, а розроблена система може бути використана як основа для побудови більш складних рішень моніторингу та аналізу часових рядів у реальних умовах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Han J. Data Mining: Concepts and Techniques / J. Han, M. Kamber, J. Pei. – 3rd ed. – Burlington : Morgan Kaufmann, 2011. – 703 p.
2. Chandola V. Anomaly Detection: A Survey / V. Chandola, A. Banerjee, V. Kumar // ACM Computing Surveys. – 2009. – Vol. 41, № 3. – P. 1–58.
3. Aggarwal C. C. Outlier Analysis / C. C. Aggarwal. – 2nd ed. – Cham : Springer, 2017. – 446 p.
4. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow / A. Géron. – 3rd ed. – Sebastopol : O'Reilly Media, 2022. – 851 p.
5. IsolationForest [Електронний ресурс] // Scikit-learn Documentation. – Режим доступу: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html>
6. Liu F. T. Isolation Forest / F. T. Liu, K. M. Ting, Z. H. Zhou // Proceedings of the IEEE International Conference on Data Mining. – 2008. – P. 413–422.
7. Goodfellow I. Deep Learning / I. Goodfellow, Y. Bengio, A. Courville. – Cambridge : MIT Press, 2016. – 775 p.
8. Keras Documentation [Електронний ресурс]. – Режим доступу: <https://keras.io/>
9. Chollet F. Deep Learning with Python / F. Chollet. – 2nd ed. – Shelter Island : Manning Publications, 2021. – 504 p.
10. An T. Variational Autoencoder based Anomaly Detection using Reconstruction Probability / T. An, S. Cho // Special Lecture on IE. – 2015. – P. 1–18.
11. TensorFlow Documentation [Електронний ресурс]. – Режим доступу: <https://www.tensorflow.org/>

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		72

12. StandardScaler [Електронний ресурс] // Scikit-learn Documentation. – Режим доступу: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html>.
13. McKinney W. Python for Data Analysis / W. McKinney. – 3rd ed. – Sebastopol : O'Reilly Media, 2022. – 579 p.
14. Keras Documentation [Електронний ресурс]. – Режим доступу: <https://keras.io/>
15. Scikit-learn Documentation [Електронний ресурс]. – Режим доступу: <https://scikit-learn.org/stable/>
16. Matplotlib Documentation [Електронний ресурс]. – Режим доступу: <https://matplotlib.org/stable/>
17. Bishop C. M. Pattern Recognition and Machine Learning / C. M. Bishop. – New York : Springer, 2006. – 738 p.
18. Hosmer D. W. Applied Logistic Regression / D. W. Hosmer, S. Lemeshow, R. X. Sturdivant. – 3rd ed. – Hoboken : Wiley, 2013. – 528 p.
19. Pandas Documentation [Електронний ресурс]. – Режим доступу: <https://pandas.pydata.org/docs/>
20. Pedregosa F. Scikit-learn: Machine Learning in Python / F. Pedregosa, G. Varoquaux, A. Gramfort та ін. // Journal of Machine Learning Research. – 2011. – Vol. 12. – P. 2825–2830
21. LogisticRegression [Електронний ресурс] // Scikit-learn Documentation. – Режим доступу: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
22. Python Documentation [Електронний ресурс]. – Режим доступу: <https://docs.python.org/3/>

ДОДАТОК А

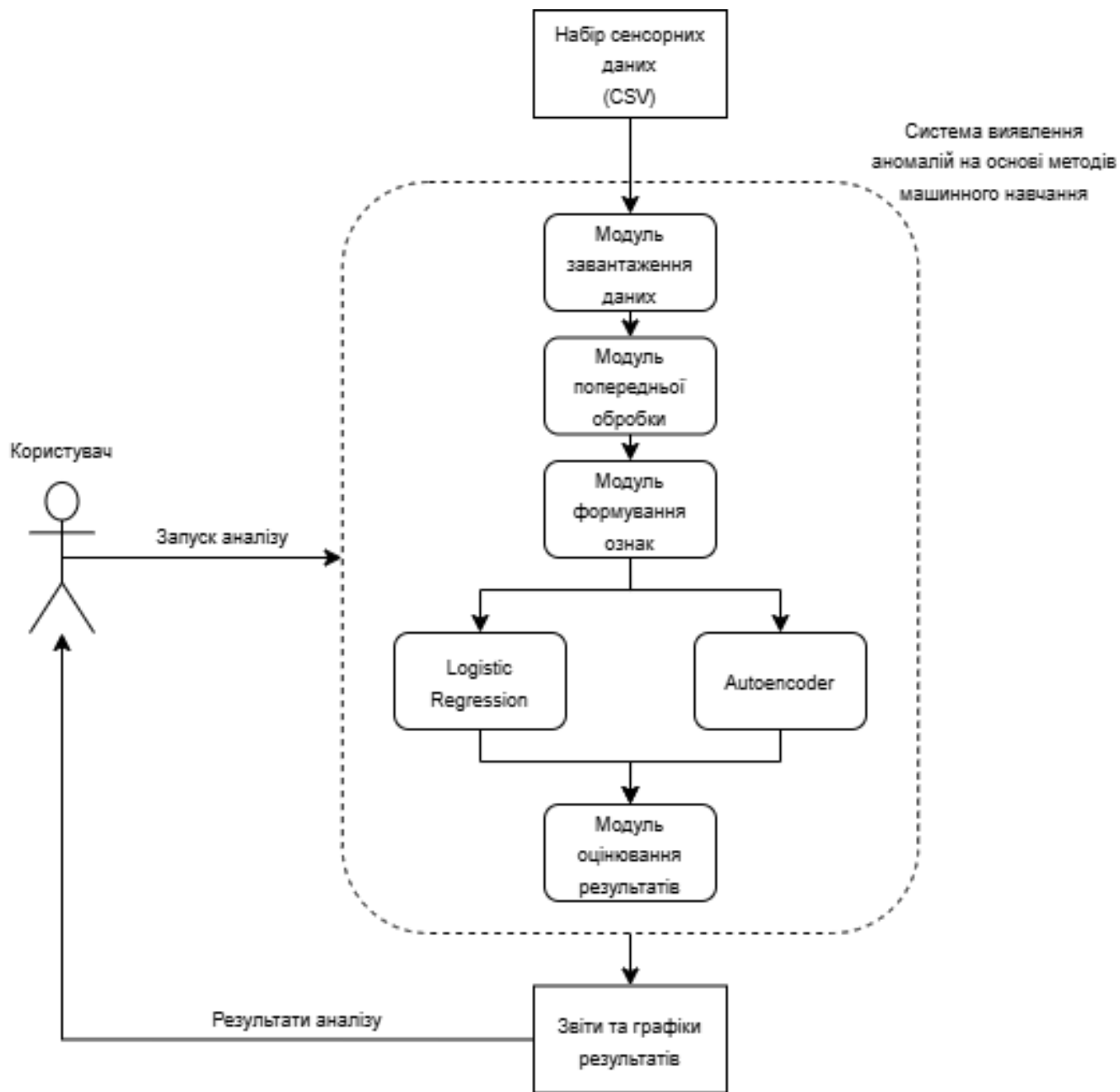
Програмний комплекс виявлення аномалій у даних сенсорних систем із
використанням алгоритмів машинного навчання

Архітектурна схема системи (принципова схема)

ІАЛЦ.467200.004 Д1

Аркушів 1

Київ – 2026 р.



					ІАЛЦ.467200.004 Д1			
					<i>Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання</i> Принципова схема	<i>Літ.</i>	<i>Маса</i>	<i>Масш.</i>
Зм.	Лист	№ докум.	Підпис	Дата				
Розробив		Бородій В. Ю.						
Перевірив		Меленчуков М. Є.						
Рецензія								
						Арк. 1	Аркушів 1	
Н. контр		Міщенко Л. Д.			<i>Дипломний проєкт</i>	<i>КПІ ім. Ігоря Сікорського, ФІОТ, ІО-24</i>		
Затвердив		Волокита А. М.						

ДОДАТОК Б

Програмний комплекс виявлення аномалій у даних сенсорних систем із
використанням алгоритмів машинного навчання

Структура програмного комплексу виявлення аномалій (схема
структурна)

ІАЛЦ.467200.005 Д2

Аркушів 1

Київ – 2026 р



					ІАЛЦ.467200.005 Д2			
					Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання Структурна схема	Літ.	Маса	Маси.
						Арк.	1	Аркушів 1
						КПІ ім. Ігоря Сікорського, ФІОТ, ІО-24		
Зм.	Лист	№ докум.	Підпис	Дата	Дипломний проєкт			
Розробив	Бородій В. Ю							
Перевірів	Меленчуков М. С.							
Рецензія								
Н. контр	Міщенко Л. Д.							
Затвердив	Волокита А. М.							

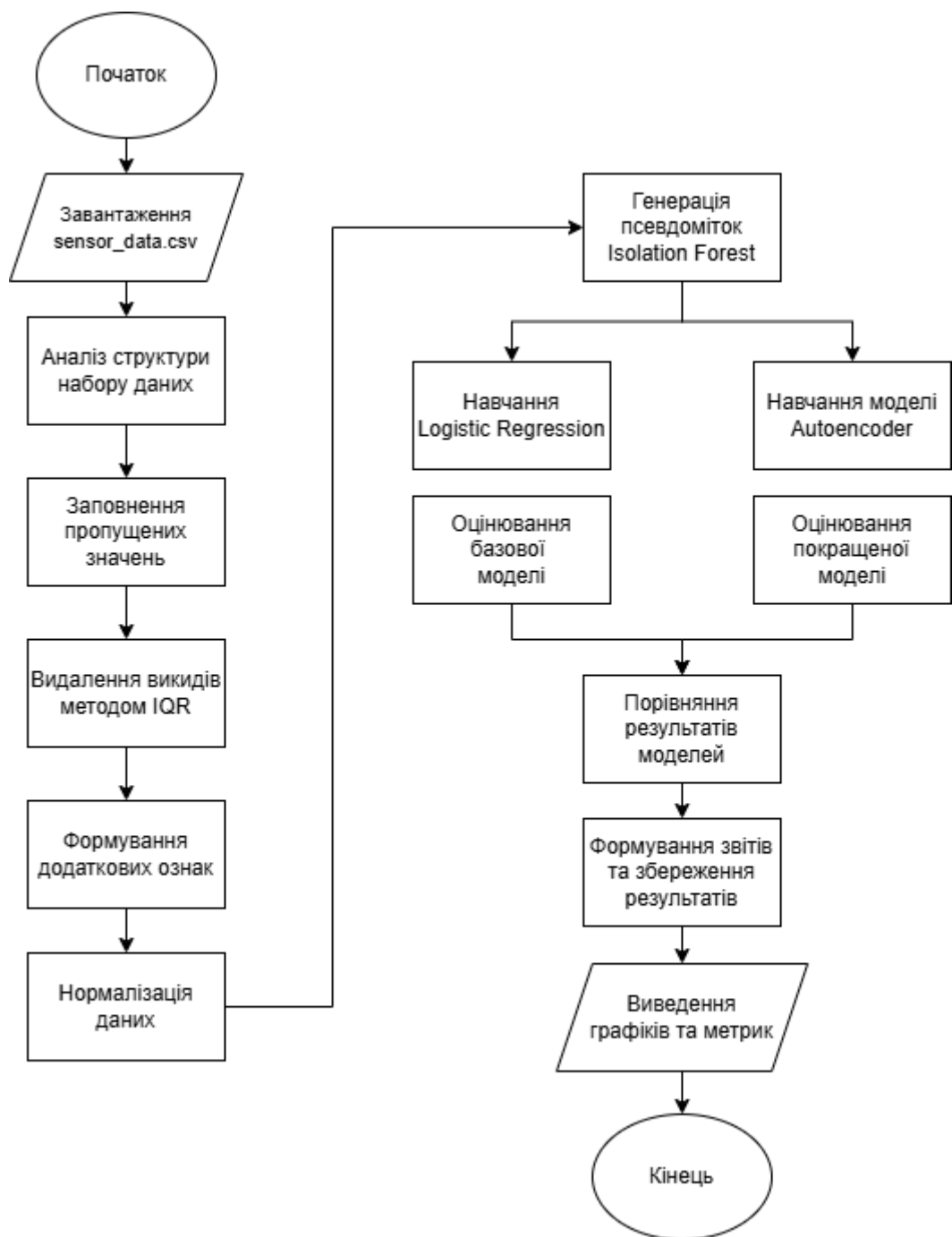
ДОДАТОК В

Програмний комплекс виявлення аномалій у даних сенсорних систем із
використанням алгоритмів машинного навчання

Алгоритм роботи програмного комплексу виявлення аномалій
(схема функціональна)
ІАЛЦ.467200.006 ДЗ

Аркушів 1

Київ – 2026 р.



					ІАЛЦ.467200.006 ДЗ			
					<i>Програмний комплекс виявлення аномалій у даних сенсорних систем із використанням алгоритмів машинного навчання</i> Функціональна схема	<i>Літ.</i>	<i>Маса</i>	<i>Масш.</i>
Зм.	Лист	№ докум.	Підпис	Дата				
Розробив	Бородій В. Ю							
Перевірів	Меленчуков М. Є.							
Рецензія						<i>Арк.</i> 1	<i>Аркушів</i> 1	
					<i>Дипломний проєкт</i>	<i>КПІ ім. Ігоря Сікорського, ФІОТ, ІО-24</i>		
Н. контр	Міщенко Л. Д.							
Затвердив	Волокита А. М.							

ДОДАТОК Г

Програмний комплекс виявлення аномалій у даних сенсорних систем із
використанням алгоритмів машинного навчання

Текст програмного коду

ІАЛЦ.467200.007 Д4

Аркушів 11

Київ – 2026 р.

Файл model.py

```
import os
from pathlib import Path

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import IsolationForest
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, TimeSeriesSplit
from sklearn.metrics import (
    accuracy_score,
    precision_score,
    recall_score,
    f1_score,
    classification_report,
    confusion_matrix
)

RANDOM_STATE = 42
DATA_PATH = "data/sensor_data.csv"
RESULTS_DIR = Path("results")
RESULTS_DIR.mkdir(exist_ok=True)

np.random.seed(RANDOM_STATE)

def print_header(title):
    print("\n" + "=" * 50)
    print(title)
    print("=" * 50)

def save_metrics_plot(metrics_dict, filename, title):
    plt.figure(figsize=(7, 5))
    plt.bar(metrics_dict.keys(), metrics_dict.values())
    plt.ylim(0, 1)
    plt.ylabel("Score")
    plt.title(title)
    plt.savefig(RESULTS_DIR / filename, dpi=300, bbox_inches="tight")
    plt.close()

def save_confusion_matrix(cm, filename, title):
    plt.figure(figsize=(6, 5))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
    plt.title(title)
    plt.savefig(RESULTS_DIR / filename, dpi=300, bbox_inches="tight")
    plt.close()

def save_score_distribution(scores, y_true, filename, title):
```

					ІАЛЦ.467200.003 ПЗ	Арк.
						1
Зм.	Арк.	№ докум.	Підпис	Дата		

```

plt.figure(figsize=(8, 5))
plt.hist(scores[y_true == 0], bins=40, alpha=0.7, label="Normal")
plt.hist(scores[y_true == 1], bins=40, alpha=0.7, label="Anomaly")
plt.xlabel("Anomaly score")
plt.ylabel("Count")
plt.title(title)
plt.legend()
plt.savefig(RESULTS_DIR / filename, dpi=300, bbox_inches="tight")
plt.close()

def save_predictions_timeline(y_true, y_pred, filename, title):
    plt.figure(figsize=(12, 5))
    plt.plot(y_true, label="Actual", marker="o", linewidth=1)
    plt.plot(y_pred, label="Predicted", marker="x", linewidth=1)
    plt.xlabel("Sample index")
    plt.ylabel("Class")
    plt.title(title)
    plt.legend()
    plt.savefig(RESULTS_DIR / filename, dpi=300, bbox_inches="tight")
    plt.close()

print_header("DATA LOADING")

df = pd.read_csv(DATA_PATH)

print("Dataset shape before cleaning:", df.shape)
print("Columns:", df.columns.tolist())
print(df.info())
print(df.describe(include="all"))

numeric_cols = df.select_dtypes(include=[np.number]).columns
df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].median())

numeric_df = df[numeric_cols]

Q1 = numeric_df.quantile(0.25)
Q3 = numeric_df.quantile(0.75)
IQR = Q3 - Q1

outlier_mask = ~(
    (numeric_df < (Q1 - 1.5 * IQR)) |
    (numeric_df > (Q3 + 1.5 * IQR))
).any(axis=1)

df_clean = df.loc[outlier_mask].reset_index(drop=True)

print("Dataset shape after cleaning:", df_clean.shape)

sensor_cols = df_clean.select_dtypes(include=[np.number]).columns

df_clean["sensor_mean"] = df_clean[sensor_cols].mean(axis=1)
df_clean["sensor_std"] = df_clean[sensor_cols].std(axis=1)
df_clean["sensor_min"] = df_clean[sensor_cols].min(axis=1)
df_clean["sensor_max"] = df_clean[sensor_cols].max(axis=1)
df_clean["sensor_range"] = df_clean["sensor_max"] - df_clean["sensor_min"]

for col in sensor_cols:
    df_clean[f"{col}_dev"] = df_clean[col] - df_clean["sensor_mean"]

```

					ІАЛЦ.467200.003 ПЗ	Арк.
						2
Зм.	Арк.	№ докум.	Підпис	Дата		

```

df_clean[f"{col}_delta"] = df_clean[col].diff()

df_clean = df_clean.fillna(0)

numeric_corr_df = df_clean.select_dtypes(include=[np.number])

plt.figure(figsize=(12, 8))
sns.heatmap(numeric_corr_df.corr(), cmap="coolwarm", linewidths=0.5)
plt.title("Feature Correlation Matrix")
plt.savefig(RESULTS_DIR / "baseline_correlation_matrix.png", dpi=300,
bbox_inches="tight")
plt.close()

X = df_clean.select_dtypes(include=[np.number])

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled = pd.DataFrame(X_scaled, columns=X.columns)

iso_model = IsolationForest(
    n_estimators=200,
    contamination=0.05,
    random_state=RANDOM_STATE
)

pseudo_labels = iso_model.fit_predict(X_scaled)
pseudo_labels = np.where(pseudo_labels == -1, 1, 0)

df_clean["pseudo_target"] = pseudo_labels

print_header("PSEUDO-LABEL DISTRIBUTION")
print(df_clean["pseudo_target"].value_counts())

X_train, X_test, y_train, y_test = train_test_split(
    X_scaled,
    df_clean["pseudo_target"],
    test_size=0.2,
    random_state=RANDOM_STATE,
    stratify=df_clean["pseudo_target"]
)

model = LogisticRegression(
    class_weight="balanced",
    max_iter=1000,
    random_state=RANDOM_STATE
)

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
y_score = model.predict_proba(X_test)[:, 1]

accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, zero_division=0)
recall = recall_score(y_test, y_pred, zero_division=0)
f1 = f1_score(y_test, y_pred, zero_division=0)

metrics = {
    "Accuracy": accuracy,
    "Precision": precision,

```

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		3

```

        "Recall": recall,
        "F1-score": f1
    }

    cm = confusion_matrix(y_test, y_pred)

    print_header("MODEL: BASELINE LOGISTIC REGRESSION")
    print("Train samples:", X_train.shape[0])
    print("Test samples:", X_test.shape[0])
    print("Normal samples:", int((df_clean["pseudo_target"] == 0).sum()))
    print("Anomaly samples:", int((df_clean["pseudo_target"] == 1).sum()))

    for name, value in metrics.items():
        print(f"{name}: {value:.4f}")

    print("\nConfusion Matrix:")
    print(cm)

    print("\nClassification Report:")
    print(classification_report(y_test, y_pred, zero_division=0))

    save_confusion_matrix(
        cm,
        "baseline_confusion_matrix.png",
        "Baseline Logistic Regression Confusion Matrix"
    )

    save_metrics_plot(
        metrics,
        "baseline_metrics.png",
        "Baseline Logistic Regression Metrics"
    )

    save_score_distribution(
        y_score,
        y_test.values,
        "baseline_score_distribution.png",
        "Baseline Logistic Regression Anomaly Score Distribution"
    )

    save_predictions_timeline(
        y_test.values,
        y_pred,
        "baseline_predictions_timeline.png",
        "Baseline Logistic Regression Predictions Timeline"
    )

    tscv = TimeSeriesSplit(n_splits=5)
    backtesting_scores = []

    for train_idx, test_idx in tscv.split(X_scaled):
        model.fit(
            X_scaled.iloc[train_idx],
            df_clean["pseudo_target"].iloc[train_idx]
        )

        preds = model.predict(X_scaled.iloc[test_idx])

        score = f1_score(
            df_clean["pseudo_target"].iloc[test_idx],

```

					ІАЛЦ.467200.003 ПЗ	Арк.
Зм.	Арк.	№ докум.	Підпис	Дата		4

```

        preds,
        zero_division=0
    )

    backtesting_scores.append(score)

print_header("BACKTESTING F1 SCORES")
for i, score in enumerate(backtesting_scores, start=1):
    print(f"Split {i}: {score:.4f}")

plt.figure(figsize=(8, 5))
plt.plot(backtesting_scores, marker="o")
plt.title("Baseline Backtesting F1 Scores")
plt.xlabel("Split")
plt.ylabel("F1 Score")
plt.ylim(0, 1)
plt.savefig(RESULTS_DIR / "baseline_backtesting_f1.png", dpi=300,
bbox_inches="tight")
plt.close()

pd.DataFrame([metrics]).to_csv(RESULTS_DIR / "baseline_metrics.csv",
index=False)

print_header("FILES SAVED")
print(f"Results saved to: {RESULTS_DIR.resolve()}")

```

					ІАЛЦ.467200.003 ПЗ	Арк.
						5
Зм.	Арк.	№ докум.	Підпис	Дата		

Файл autoencoder_model.py

```
import os
os.environ["TF_CPP_MIN_LOG_LEVEL"] = "2"
os.environ["TF_ENABLE_ONEDNN_OPTS"] = "0"

from pathlib import Path

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf

from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import IsolationForest
from sklearn.model_selection import train_test_split
from sklearn.metrics import (
    accuracy_score,
    precision_score,
    recall_score,
    f1_score,
    classification_report,
    confusion_matrix
)

from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dense, BatchNormalization
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping

RANDOM_STATE = 42
DATA_PATH = "data/sensor_data.csv"
RESULTS_DIR = Path("results")
RESULTS_DIR.mkdir(exist_ok=True)

np.random.seed(RANDOM_STATE)
tf.random.set_seed(RANDOM_STATE)

def print_header(title):
    print("\n" + "=" * 50)
    print(title)
    print("=" * 50)

def save_metrics_plot(metrics_dict, filename, title):
    plt.figure(figsize=(7, 5))
    plt.bar(metrics_dict.keys(), metrics_dict.values())
    plt.ylim(0, 1)
    plt.ylabel("Score")
    plt.title(title)
    plt.savefig(RESULTS_DIR / filename, dpi=300, bbox_inches="tight")
    plt.close()

def save_confusion_matrix(cm, filename, title):
```

					ІАЛЦ.467200.003 ПЗ	Арк.
						6
Зм.	Арк.	№ докум.	Підпис	Дата		

```

plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title(title)
plt.savefig(RESULTS_DIR / filename, dpi=300, bbox_inches="tight")
plt.close()

def save_score_distribution(scores, y_true, threshold, filename, title):
    plt.figure(figsize=(8, 5))
    plt.hist(scores[y_true == 0], bins=40, alpha=0.7, label="Normal")
    plt.hist(scores[y_true == 1], bins=40, alpha=0.7, label="Anomaly")
    plt.axvline(threshold, linestyle="--", label="Threshold")
    plt.xlabel("Reconstruction error")
    plt.ylabel("Count")
    plt.title(title)
    plt.legend()
    plt.savefig(RESULTS_DIR / filename, dpi=300, bbox_inches="tight")
    plt.close()

def save_predictions_timeline(y_true, y_pred, filename, title):
    plt.figure(figsize=(12, 5))
    plt.plot(y_true, label="Actual", marker="o", linewidth=1)
    plt.plot(y_pred, label="Predicted", marker="x", linewidth=1)
    plt.xlabel("Sample index")
    plt.ylabel("Class")
    plt.title(title)
    plt.legend()
    plt.savefig(RESULTS_DIR / filename, dpi=300, bbox_inches="tight")
    plt.close()

print_header("DATA LOADING")

df = pd.read_csv(DATA_PATH)

print("Dataset shape before cleaning:", df.shape)
print("Columns:", df.columns.tolist())

numeric_cols = df.select_dtypes(include=[np.number]).columns
df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].median())

numeric_df = df[numeric_cols]

Q1 = numeric_df.quantile(0.25)
Q3 = numeric_df.quantile(0.75)
IQR = Q3 - Q1

outlier_mask = ~(
    (numeric_df < (Q1 - 1.5 * IQR)) |
    (numeric_df > (Q3 + 1.5 * IQR))
).any(axis=1)

df_clean = df.loc[outlier_mask].reset_index(drop=True)

print("Dataset shape after cleaning:", df_clean.shape)

sensor_cols = df_clean.select_dtypes(include=[np.number]).columns

```

					ІАЛЦ.467200.003 ПЗ	Арк.
						7
Зм.	Арк.	№ докум.	Підпис	Дата		

```

df_clean["sensor_mean"] = df_clean[sensor_cols].mean(axis=1)
df_clean["sensor_std"] = df_clean[sensor_cols].std(axis=1)
df_clean["sensor_min"] = df_clean[sensor_cols].min(axis=1)
df_clean["sensor_max"] = df_clean[sensor_cols].max(axis=1)
df_clean["sensor_range"] = df_clean["sensor_max"] - df_clean["sensor_min"]

for col in sensor_cols:
    df_clean[f"{col}_dev"] = df_clean[col] - df_clean["sensor_mean"]
    df_clean[f"{col}_delta"] = df_clean[col].diff()

df_clean = df_clean.fillna(0)

X = df_clean.select_dtypes(include=[np.number])

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

iso_model = IsolationForest(
    n_estimators=200,
    contamination=0.05,
    random_state=RANDOM_STATE
)

pseudo_labels = iso_model.fit_predict(X_scaled)
pseudo_labels = np.where(pseudo_labels == -1, 1, 0)

df_clean["pseudo_target"] = pseudo_labels

print_header("PSEUDO-LABEL DISTRIBUTION")
print(df_clean["pseudo_target"].value_counts())

X_train, X_test, y_train, y_test = train_test_split(
    X_scaled,
    df_clean["pseudo_target"].values,
    test_size=0.2,
    random_state=RANDOM_STATE,
    stratify=df_clean["pseudo_target"].values
)

normal_train_data = X_train[y_train == 0]

input_dim = X_scaled.shape[1]

input_layer = Input(shape=(input_dim,))

encoded = Dense(32, activation="relu")(input_layer)
encoded = BatchNormalization()(encoded)
encoded = Dense(16, activation="relu")(encoded)
encoded = BatchNormalization()(encoded)
encoded = Dense(8, activation="relu")(encoded)

decoded = Dense(16, activation="relu")(encoded)
decoded = BatchNormalization()(decoded)
decoded = Dense(32, activation="relu")(decoded)
decoded = BatchNormalization()(decoded)
decoded = Dense(input_dim, activation="linear")(decoded)

autoencoder = Model(inputs=input_layer, outputs=decoded)

```

					ІАЛЦ.467200.003 ПЗ	Арк.
						8
Зм.	Арк.	№ докум.	Підпис	Дата		

```

autoencoder.compile(
    optimizer=Adam(learning_rate=0.001),
    loss="mse"
)

early_stop = EarlyStopping(
    monitor="val_loss",
    patience=30,
    restore_best_weights=True
)

print_header("MODEL TRAINING")

history = autoencoder.fit(
    normal_train_data,
    normal_train_data,
    epochs=300,
    batch_size=32,
    validation_split=0.2,
    callbacks=[early_stop],
    verbose=1
)

train_reconstructions = autoencoder.predict(X_train, verbose=0)
train_errors = np.mean(np.square(X_train - train_reconstructions), axis=1)

thresholds = np.linspace(
    np.percentile(train_errors, 50),
    np.percentile(train_errors, 99),
    500
)

best_f1 = 0
best_threshold = 0

for threshold in thresholds:
    train_preds = np.where(train_errors > threshold, 1, 0)
    current_f1 = f1_score(y_train, train_preds, zero_division=0)

    if current_f1 > best_f1:
        best_f1 = current_f1
        best_threshold = threshold

test_reconstructions = autoencoder.predict(X_test, verbose=0)
test_errors = np.mean(np.square(X_test - test_reconstructions), axis=1)

y_pred = np.where(test_errors > best_threshold, 1, 0)

accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, zero_division=0)
recall = recall_score(y_test, y_pred, zero_division=0)
f1 = f1_score(y_test, y_pred, zero_division=0)

metrics = {
    "Accuracy": accuracy,
    "Precision": precision,
    "Recall": recall,
    "F1-score": f1
}

```

					ІАЛЦ.467200.003 ПЗ	Арк.
						9
Зм.	Арк.	№ докум.	Підпис	Дата		

```

cm = confusion_matrix(y_test, y_pred)

print_header("MODEL: AUTOENCODER")
print("Train samples:", X_train.shape[0])
print("Test samples:", X_test.shape[0])
print("Normal samples:", int((df_clean["pseudo_target"] == 0).sum()))
print("Anomaly samples:", int((df_clean["pseudo_target"] == 1).sum()))
print(f"Best threshold: {best_threshold:.6f}")

for name, value in metrics.items():
    print(f"{name}: {value:.4f}")

print("\nConfusion Matrix:")
print(cm)

print("\nClassification Report:")
print(classification_report(y_test, y_pred, zero_division=0))

save_confusion_matrix(
    cm,
    "autoencoder_confusion_matrix.png",
    "Autoencoder Confusion Matrix"
)

save_metrics_plot(
    metrics,
    "autoencoder_metrics.png",
    "Autoencoder Metrics"
)

save_score_distribution(
    test_errors,
    y_test,
    best_threshold,
    "autoencoder_score_distribution.png",
    "Autoencoder Reconstruction Error Distribution"
)

save_predictions_timeline(
    y_test,
    y_pred,
    "autoencoder_predictions_timeline.png",
    "Autoencoder Predictions Timeline"
)

plt.figure(figsize=(8, 5))
plt.plot(history.history["loss"], label="Training Loss")
plt.plot(history.history["val_loss"], label="Validation Loss")
plt.title("Autoencoder Training Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.legend()
plt.savefig(RESULTS_DIR / "autoencoder_training_loss.png", dpi=300,
bbox_inches="tight")
plt.close()

plt.figure(figsize=(12, 5))
plt.plot(test_errors, label="Reconstruction Error")
plt.axhline(y=best_threshold, linestyle="--", label="Threshold")
plt.title("Autoencoder Reconstruction Error Over Test Samples")

```

					ІАЛЦ.467200.003 ПЗ	Арк.
						10
Зм.	Арк.	№ докум.	Підпис	Дата		

```

plt.xlabel("Sample index")
plt.ylabel("Error")
plt.legend()
plt.savefig(RESULTS_DIR / "autoencoder_reconstruction_error_timeline.png",
            dpi=300, bbox_inches="tight")
plt.close()

pd.DataFrame([metrics]).to_csv(RESULTS_DIR / "autoencoder_metrics.csv",
                                index=False)

print_header("FILES SAVED")
print(f"Results saved to: {RESULTS_DIR.resolve()}")

```

					ІАЛЦ.467200.003 ПЗ	Арк.
						11
Зм.	Арк.	№ докум.	Підпис	Дата		