МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

СТЕЦЮК Арсен Ігорович

Інтелектуальний модуль підтримки рішень щодо автоматизованого зрошення на базі даних IoT-сенсорів та ансамблевих моделей машинного навчання / Intelligent Decision Support Module for Automated Irrigation Based on IoT Sensor Data and Ensemble Machine Learning Models

спеціальність: 122 - Комп'ютерні науки
освітньо-професійна програма - Комп'ютерні науки

Кваліфікаційна робота

Виконав студент групи КН-41
А. І. Стецюк

_____

Науковий керівник:
к.т.н., доцент, І. М. Майків

_____

Кваліфікаційну роботу
допущено до захисту:
"___"_____ 20__ р.

Завідувач кафедри
_____ **Н.В. Дзюбановська**

Тернопіль – 2026

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність: 122 – Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.В. Дзюбановська
«____» _____20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
СТЕЦЮКУ Арсену Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Інтелектуальний модуль підтримки рішень щодо автоматизованого зрошення на базі даних IoT-сенсорів та ансамблевих моделей машинного навчання / Intelligent Decision Support Module for Automated Irrigation Based on IoT Sensor Data and Ensemble Machine Learning Models
керівник роботи к.т.н., доцент І.М. Майків
затверджені наказом по університету від 01 грудня 2025 року № 894.
2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2026 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.
4. Основні питання, які потрібно розробити
- проаналізувати предметну область автоматизованого зрошення, сучасні IoT-засоби моніторингу та підходи до інтелектуальної підтримки рішень;
- дослідити існуючі методи вимірювання вологості ґрунту, обробки сенсорних даних і моделі машинного навчання, придатні для задач зрошення;
- сформувати вимоги до інтелектуального модуля підтримки рішень щодо поливу;
- розробити архітектуру системи збору, передавання, збереження та аналітичної обробки даних IoT-сенсорів;
- сформувати ознаковий простір і підготувати дані для навчання ансамблевих моделей;
- розробити алгоритм підтримки рішень щодо автоматизованого зрошення;
- реалізувати програмний прототип модуля;
- провести експериментальне дослідження та оцінити ефективність запропонованого підходу.
5. Перелік графічного матеріалу в роботі
- узагальнена схема архітектури інтелектуального модуля;
- узагальнена структурна схема ознакового простору;

– схема алгоритму підтримки рішень щодо поливу;

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 01 грудня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи	до 01.01. 2026 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02. 2026 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 15.03.2026 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05. 2026 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2026 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2026 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2026 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2026 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06. 2026 р.	

Студент_____ А.І. Стецюк
підпис

Керівник роботи_____ к.т.н., доцент І.М. Майків
підпис

АНОТАЦІЯ

Кваліфікаційна робота на тему «Інтелектуальний модуль підтримки рішень щодо автоматизованого зрошення на базі даних IoT-сенсорів та ансамблевих моделей машинного навчання» на здобуття освітнього ступеня «бакалавр» зі спеціальності 122 «Комп'ютерні науки» освітньої програми «Комп'ютерні науки» написана обсягом у 83 сторінки і містить 12 ілюстрацій, 16 таблиць, 2 додатки та 35 використаних джерел.

Метою кваліфікаційної роботи є розроблення інтелектуального модуля підтримки рішень щодо автоматизованого зрошення, який використовує дані IoT-сенсорів і ансамблеві моделі машинного навчання для визначення доцільності поливу та підвищення ефективності використання водних ресурсів.

Методи досліджень включають метод аналізу для визначення особливостей автоматизованого зрошення, IoT-сенсорів і сучасних підходів до підтримки рішень; метод моделювання для розроблення архітектури модуля; методи попередньої обробки даних; методи машинного навчання; метод експериментальних досліджень; метод порівняльного аналізу.

Розроблено інтелектуальний модуль підтримки рішень щодо автоматизованого зрошення, який забезпечує завантаження даних IoT-сенсорів, їх попередню обробку, формування базових і похідних ознак, навчання ансамблевих моделей машинного навчання та формування рішення щодо доцільності поливу.

Результати дослідження можуть бути використані для побудови програмно-апаратних систем автоматизованого зрошення, розроблення інтелектуальних модулів моніторингу аграрного середовища, інтеграції IoT-сенсорів із системами підтримки рішень, а також для створення навчально-дослідних прототипів у сфері точного землеробства.

Ключові слова: АВТОМАТИЗОВАНЕ ЗРОШЕННЯ, ІНТЕРНЕТ РЕЧЕЙ, IOT-СЕНСОРИ, ВОЛОГІСТЬ ҐРУНТУ, МАШИННЕ НАВЧАННЯ, АНСАМБЛЕВІ МОДЕЛІ, RANDOM FOREST, ПІДТРИМКА РІШЕНЬ, ТОЧНЕ ЗЕМЛЕРОБСТВО.

ANNOTATION

Qualification work on the topic «Intelligent Decision Support Module for Automated Irrigation Based on IoT Sensor Data and Ensemble Machine Learning Models» for Bachelor's degree on speciality 122 «Computer Science» educational and professional program «Computer Science» is written on 83 pages and it contains 12 figures, 16 tables, 2 annexes and 35 sources.

The purpose of the qualification work is to develop an intelligent decision support module for automated irrigation that uses IoT sensor data and ensemble machine learning models to determine the feasibility of irrigation and improve the efficiency of water resource use.

The research methods include the method of analysis to identify the specific features of automated irrigation, IoT sensors, and modern approaches to decision support; the modeling method for developing the module architecture; data preprocessing methods; machine learning methods; the method of experimental research; and the method of comparative analysis.

An intelligent decision support module for automated irrigation has been developed. It provides loading of IoT sensor data, their preprocessing, the formation of basic and derived features, training of ensemble machine learning models, and the generation of a decision regarding the feasibility of irrigation.

The research results can be used for building software and hardware systems for automated irrigation, developing intelligent modules for monitoring the agricultural environment, integrating IoT sensors with decision support systems, and creating educational and research prototypes in the field of precision agriculture.

Keywords: AUTOMATED IRRIGATION, INTERNET OF THINGS, IOT SENSORS, SOIL MOISTURE, MACHINE LEARNING, ENSEMBLE MODELS, RANDOM FOREST, DECISION SUPPORT, PRECISION AGRICULTURE.

ЗМІСТ

ВСТУП

Сільське господарство дедалі активніше інтегрує цифрові технології, що дає змогу переходити від загальних, наперед заданих режимів поливу до точного керування зрошенням на основі фактичного стану ґрунту й середовища. У таких умовах особливого значення набуває використання сенсорних мереж, засобів Інтернету речей та методів машинного навчання, які забезпечують безперервний збір даних, їх аналітичну обробку та формування обґрунтованих керуючих рішень [6, 24, 33].

Актуальність теми зумовлена кількома чинниками. По-перше, у практиці зрошення досі поширені підходи, за яких полив виконується за фіксованим графіком або за спрощеними пороговими правилами. Такий підхід не враховує поточних змін вологості ґрунту, температури, вологості повітря, локальних мікрокліматичних умов і, як наслідок, може спричиняти або дефіцит вологи, або перевитрати водних ресурсів. По-друге, сучасні IoT-сенсори дають змогу організувати регулярний моніторинг параметрів середовища, однак сам по собі збір даних ще не гарантує ефективного керування. Необхідним стає інтелектуальний модуль, здатний перетворювати потік вимірювань на конкретне рішення щодо доцільності поливу. По-третє, ансамблеві моделі машинного навчання демонструють високу придатність до роботи з багатофакторними даними, шумом вимірювань і нелінійними залежностями, що є типовими для задач автоматизованого зрошення [10, 17, 21].

Проблема полягає в тому, що наявні системи автоматизованого зрошення часто або мають обмежену аналітичну складову, або орієнтовані переважно на апаратний рівень без належного інтелектуального механізму підтримки рішень. Частина рішень зводиться до простого порівняння поточного показника вологості з фіксованим порогом. Частина систем використовує віддалений моніторинг, але не реалізує повноцінного прогнозно-аналітичного шару. Унаслідок цього знижується точність реагування на зміни стану ґрунту, погіршується адаптивність системи та не повною мірою реалізується потенціал цифрових аграрних технологій [7, 9, 31].

Аналіз наукових джерел засвідчує, що значна увага приділяється питанням вимірювання вологості ґрунту, калібрування сенсорів, побудови бездротових мереж моніторингу та використання моделей машинного навчання для оцінювання або прогнозування стану середовища [1, 5, 8, 18]. Окремі дослідження присвячені огляду smart irrigation, сучасних технологій сенсорного контролю та методів оцінювання вологості на основі класичних і інтелектуальних підходів [2, 6, 19]. Разом із тим недостатньо системно опрацьованим залишається питання побудови саме програмного модуля підтримки рішень, який поєднує надходження даних від IoT-сенсорів, попередню обробку, формування ознакового простору, застосування ансамблевих моделей і видачу кінцевого рішення щодо поливу в межах єдиної логіки функціонування системи.

Ця тема є актуальною ще й тому, що вона розташована на перетині кількох важливих напрямів: аналізу даних, інтелектуальних інформаційних систем, вбудованих та розподілених засобів збору інформації, програмної інженерії й підтримки прийняття рішень. У межах такої постановки задачі важливим є не лише вибір окремого алгоритму, а й проєктування цілісної архітектури: від сенсорного рівня до рівня логіки прийняття рішення, журналювання подій, оцінювання якості та інтерпретації результатів.

Таким чином, розроблення інтелектуального модуля підтримки рішень щодо автоматизованого зрошення на базі даних IoT-сенсорів та ансамблевих моделей машинного навчання є актуальним науково-прикладним завданням. Його розв’язання дає змогу поєднати засоби моніторингу стану ґрунту й навколишнього середовища з методами інтелектуального аналізу даних для підвищення обґрунтованості рішень щодо поливу, зменшення надлишкового використання води та підвищення адаптивності системи.

Метою роботи є розроблення інтелектуального модуля підтримки рішень щодо автоматизованого зрошення, який використовує дані IoT-сенсорів і ансамблеві моделі машинного навчання для визначення доцільності поливу та підвищення ефективності використання водних ресурсів.

Для досягнення поставленої мети сформовано такі завдання дослідження:

– проаналізувати предметну область автоматизованого зрошення, сучасні IoT-засоби моніторингу та підходи до інтелектуальної підтримки рішень;

– дослідити існуючі методи вимірювання вологості ґрунту, обробки сенсорних даних і моделі машинного навчання, придатні для задач зрошення;

– сформувати вимоги до інтелектуального модуля підтримки рішень щодо поливу;

– розробити архітектуру системи збору, передавання, збереження та аналітичної обробки даних IoT-сенсорів;

– сформувати ознаковий простір і підготувати дані для навчання ансамблевих моделей;

– розробити алгоритм підтримки рішень щодо автоматизованого зрошення;

– реалізувати програмний прототип модуля;

– провести експериментальне дослідження та оцінити ефективність запропонованого підходу.

Об’єктом дослідження є процес автоматизованого зрошення на основі даних, отриманих від IoT-сенсорів у системах моніторингу стану ґрунту та навколишнього середовища.

Предметом дослідження є методи, моделі та програмні засоби підтримки рішень щодо автоматизованого зрошення на базі сенсорних даних і ансамблевих моделей машинного навчання.

Результати кваліфікаційної роботи апробовані на міжнародній науковій інтернет-конференції «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення», м. Тернопіль, Україна, м. Ополе, Польща, 7-8 травня 2026 р. (додаток Б).

Кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел та додатків.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Особливості автоматизованого зрошення в системах точного землеробства

Автоматизоване зрошення є одним із ключових напрямів розвитку точного землеробства, оскільки саме керування водним режимом безпосередньо впливає на стан рослин, стабільність урожайності та ефективність використання ресурсів. У традиційних підходах полив часто виконується за фіксованим графіком або на основі спрощеної оцінки стану ґрунту. Така організація процесу не враховує оперативних змін мікроклімату, різної інтенсивності випаровування, неоднорідності ділянок та динаміки вологості, через що частина води використовується нераціонально [6, 18].

У системах точного землеробства зрошення розглядається не як ізольована дія, а як керований процес, що спирається на безперервне спостереження за станом ґрунту та середовища. Його сутність полягає у визначенні моменту, тривалості та інтенсивності поливу на основі вимірюваних параметрів. Насамперед йдеться про вологість ґрунту, температуру повітря, відносну вологість повітря, а в окремих випадках також про освітленість, характеристики ґрунтового профілю та часові особливості розвитку культури. Такий підхід забезпечує перехід від статичних схем поливу до адаптивного керування, у межах якого рішення формується відповідно до фактичного стану об’єкта [2, 9, 24].

На рисунку 1.1 подано узагальнену схему місця автоматизованого зрошення в системі точного землеробства. У показано взаємозв’язок між сенсорним рівнем, рівнем передавання даних, аналітичним модулем та виконавчим механізмом поливу.

Основною передумовою автоматизації зрошення стало поширення технологій Інтернету речей. Завдяки їм сформовано можливість організувати постійний збір показників із розподілених сенсорів, передавати дані на сервер або хмарну платформу, накопичувати історію вимірювань і використовувати її для подальшого аналізу. У такій архітектурі сенсор перестає бути лише

вимірювальним елементом і стає джерелом даних для підтримки рішень. Саме це відрізняє сучасні системи smart irrigation від простих схем автоматичного вмикання насоса за одним порогом вологості [7, 16, 33].

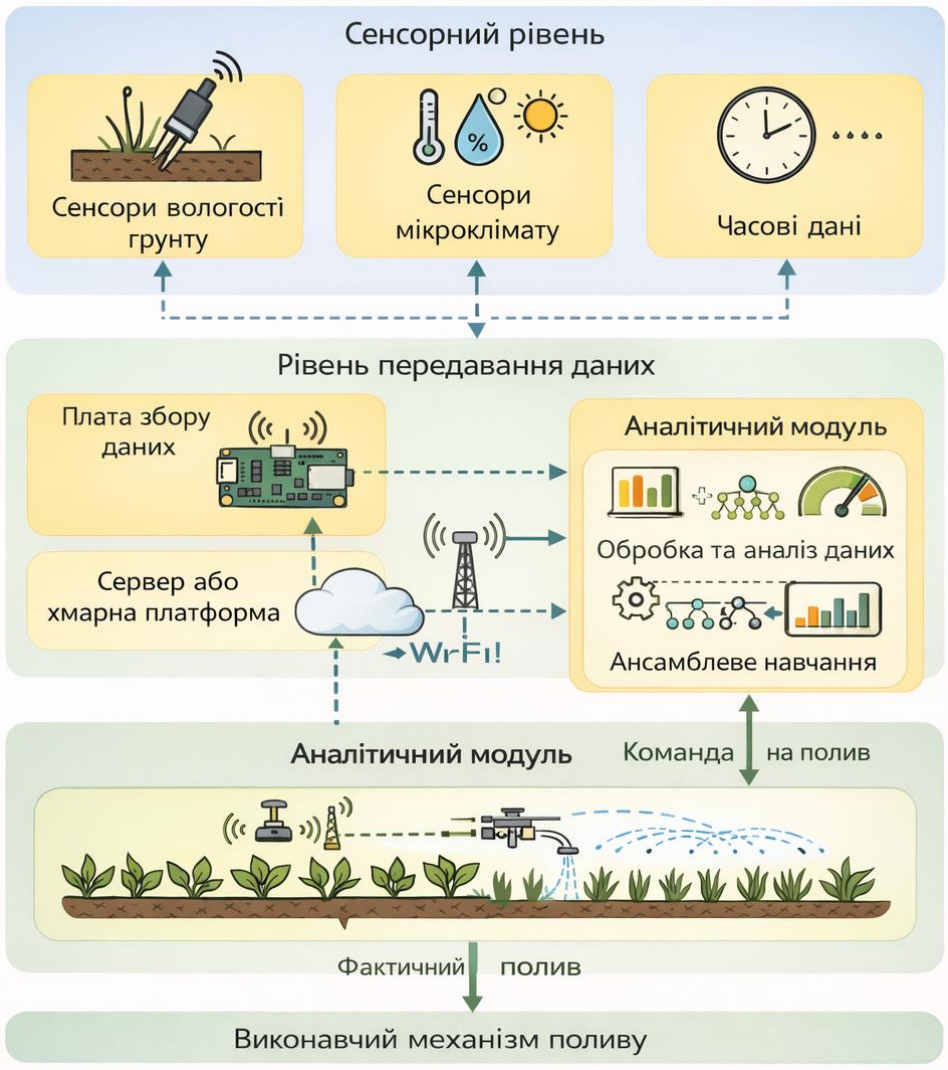

Рисунок 1.1 – Місце автоматизованого зрошення в системі точного землеробства

Визначальною особливістю автоматизованого зрошення є багатофакторність. Рішення щодо поливу не може вважатися достатньо обґрунтованим, якщо воно базується лише на одному параметрі. Навіть за однакового значення вологості ґрунту потреба в поливі може відрізнятися залежно від температури, поточної вологості повітря, інтенсивності випаровування, типу ґрунту та фази розвитку рослини. Через це сучасні системи мають враховувати сукупність ознак, а не одну локальну умову. Така постановка

задачі є природною для використання методів інтелектуального аналізу даних, оскільки між вхідними параметрами та доцільністю поливу існують нелінійні залежності, які складно описати жорсткими правилами [10, 17, 31].

Ще однією важливою особливістю є просторово-часова мінливість параметрів. Стан ґрунту змінюється не лише з часом, а й у межах різних зон однієї ділянки. Навіть на відносно невеликій площі можуть спостерігатися відмінності у вологоутримувальній здатності ґрунту, швидкості висихання, глибині зволоження та умовах доступу рослин до води. У зв'язку з цим система автоматизованого зрошення має або працювати з локальними зонами спостереження, або враховувати узагальнення даних із кількох точок вимірювання. Такий підхід підвищує точність керування, але водночас ускладнює структуру системи та вимоги до обробки даних [8, 19, 25].

Автоматизоване зрошення в межах точного землеробства має і виражену програмно-аналітичну складову. Якщо на ранніх етапах розвитку подібних систем основна увага приділялася апаратній реалізації, то сучасні рішення орієнтуються на створення інтелектуальних модулів, здатних інтерпретувати потік сенсорних даних. Це означає, що ефективність системи визначається не лише точністю сенсора чи надійністю каналу зв'язку, а й якістю алгоритму, який аналізує вхідні дані та формує керуюче рішення. Саме тому в задачах зрошення все частіше застосовуються методи машинного навчання, зокрема ансамблеві моделі, які краще працюють у випадках шуму, неповноти даних і складних залежностей між параметрами [12, 21].

У таблиці 1.1 узагальнено відмінності між традиційним і автоматизованим підходами до поливу.

Переваги автоматизованого зрошення не зводяться лише до економії води. Важливим результатом є також підвищення стабільності керування. Якщо система здатна регулярно зчитувати параметри та приймати рішення на основі узгоджених правил або моделі, зменшується залежність від суб'єктивного людського фактора. Крім того, накопичення історичних даних формує основу для подальшого прогнозування, виявлення типових сценаріїв висихання ґрунту, налаштування порогів і побудови моделей підтримки рішень. Саме

накопичувальний характер сенсорних даних створює умови для переходу від реактивного поливу до прогнозно-аналітичного керування [3, 6].

Таблиця 1.1 – Порівняння традиційного та автоматизованого підходів до зрошення

Ознака порівняння	Традиційне зрошення	Автоматизоване зрошення
Основа прийняття рішення	Фіксований графік або візуальна оцінка	Дані сенсорів і алгоритми аналізу
Частота реагування	Періодична	Оперативна, за станом середовища
Урахування змін мікроклімату	Обмежене	Безперервне або регулярне
Використання води	Часто надлишкове	Оптимізоване
Масштабованість	Невисока	Висока за наявності мережевої інфраструктури
Адаптивність	Низька	Вища завдяки аналітичному модулю

Разом із перевагами існує низка технічних обмежень. По-перше, точність системи суттєво залежить від якості вимірювання вологості ґрунту. Відомо, що дешеві сенсори можуть мати нестабільні показники, залежність від температури, особливостей монтажу та типу ґрунту. Унаслідок цього важливими стають калібрування, компенсація похибок і перевірка достовірності даних [5, 8, 22]. По-друге, у реальних умовах можливі пропуски даних, короткочасні збої зв’язку, аномальні значення та дрейф показників, що потребує реалізації процедур фільтрації й попередньої обробки. По-третє, сама логіка прийняття рішень не може бути надто спрощеною, оскільки це знижує адаптивність системи та робить її нестійкою до змін умов експлуатації.

Окрему увагу слід приділити ролі сенсорної підсистеми. Для задач автоматизованого зрошення найважливішими є датчики вологості ґрунту, проте в більш досконалих рішеннях використовуються також температурні та вологісні

сенсори повітря, а інколи й додаткові джерела контекстних даних. У наукових працях запропоновано різні варіанти реалізації таких засобів: від простих недорогих рішень до каліброваних сенсорів для точнішого вимірювання в польових умовах [1, 14, 26]. Це свідчить про те, що вибір апаратної конфігурації має здійснюватися з урахуванням не лише вартості, а й вимог до стабільності, точності та інтеграції з програмним модулем.

У сучасному розумінні автоматизоване зрошення є частиною ширшої цифрової трансформації аграрного виробництва. Воно поєднує сенсорний моніторинг, бездротові мережі, хмарну або локальну аналітику, інтелектуальні алгоритми та виконавчі механізми в єдину систему. Така інтеграція формує нову якість керування, за якої рішення приймається не інтуїтивно, а на основі даних і моделі [30]. У цьому контексті особливого значення набуває саме модуль підтримки рішень, оскільки він забезпечує перехід від набору виміряних значень до керуючої дії.

Отже, особливості автоматизованого зрошення в системах точного землеробства визначаються орієнтацією на дані, багатофакторністю, просторово-часовою мінливістю параметрів, необхідністю інтеграції IoT-інфраструктури з аналітичними алгоритмами та потребою в адаптивному прийнятті рішень. Це формує підґрунтя для розроблення інтелектуального модуля, здатного опрацьовувати дані сенсорів, виявляти закономірності та визначати доцільність поливу з урахуванням поточного стану середовища.

1.2 Аналіз IoT-сенсорів, даних моніторингу та методів машинного навчання для підтримки рішень щодо поливу

Ефективність автоматизованого зрошення безпосередньо визначається якістю даних, на основі яких формується рішення щодо поливу. Саме тому в сучасних системах точного землеробства центральне місце посідають IoT-сенсори, що забезпечують безперервне або періодичне вимірювання параметрів ґрунту та навколишнього середовища. Проте практична цінність таких засобів полягає не лише у фіксації окремих показників, а у формуванні цілісного потоку

даних, придатного для подальшого аналітичного опрацювання.

Основним параметром у задачах керування поливом є вологість ґрунту. Саме вона найбільш безпосередньо відображає потребу рослин у воді та є базовою ознакою для прийняття рішення щодо зрошення. У наукових джерелах описано різні підходи до вимірювання цього показника, зокрема резистивні, ємнісні, FDR- та інші типи сенсорів [1, 18, 22]. Кожен із них характеризується власними перевагами та обмеженнями. Резистивні датчики є простими й дешевими, проте схильні до корозії та зниження стабільності показників у процесі тривалої експлуатації. Ємнісні сенсори є практичнішими для тривалого використання, оскільки мають вищу стійкість до впливу середовища, однак також потребують калібрування. Складніші сенсорні рішення забезпечують кращу точність, але збільшують вартість системи та вимоги до інтеграції.

Суттєвою особливістю сенсорів вологості ґрунту є залежність точності вимірювання від фізичних властивостей ґрунту, температури, способу встановлення та умов експлуатації. Один і той самий датчик може по-різному поводитися у різних типах ґрунтів, а дешеві сенсори нерідко демонструють нестабільність або підвищену похибку. У зв’язку з цим у багатьох дослідженнях наголошено на важливості калібрування, температурної компенсації та перевірки достовірності зчитувань [5, 8, 26]. Це означає, що для побудови надійного програмного модуля підтримки рішень недостатньо просто підключити сенсор; необхідно враховувати якість даних, межі похибок і можливість появи аномальних значень.

Окрім вологості ґрунту, у системах автоматизованого поливу доцільно використовувати й інші параметри. До них належать температура повітря, відносна вологість повітря, а в окремих випадках – температура ґрунту, освітленість, час доби, характеристики попередніх циклів поливу та похідні показники, обчислені на основі історії вимірювань. Такі дані не завжди прямо визначають потребу в поливі, проте істотно впливають на швидкість висихання ґрунту, інтенсивність випаровування та динаміку зміни вологості. Унаслідок цього рішення, сформоване на основі сукупності ознак, зазвичай є більш точним, ніж рішення, побудоване лише на одному поточному значенні

вологості [6, 10, 21].

На рисунку 1.2 представлено класифікацію сенсорних даних, що використовуються в задачах автоматизованого зрошення. У схемі виділено три групи: базові сенсорні показники, службові дані системи та похідні аналітичні ознаки.

Рисунок 1.2 – Класифікація сенсорних даних для автоматизованого зрошення

У практичній реалізації IoT-система формує потік даних, який характеризується низкою особливостей. По-перше, дані мають часову природу, оскільки кожне вимірювання пов’язане з конкретним моментом зчитування. По-друге, вони можуть містити шуми, пропуски та аномалії, спричинені нестабільністю живлення, порушенням зв’язку, апаратними збоями або некоректною роботою датчика. По-третє, деякі параметри змінюються поступово, а деякі – різко, що ускладнює безпосереднє використання сирих значень для прийняття рішень. Саме тому перед застосуванням моделей машинного навчання необхідно виконувати попередню обробку даних: очищення, нормалізацію, усунення пропусків, агрегацію, формування часових ознак та іноді згладжування значень [3, 7].

У таблиці 1.2 узагальнено основні типи даних, що використовуються в задачі підтримки рішень щодо поливу.

Таблиця 1.2 – Основні типи даних для підтримки рішень щодо автоматизованого зрошення

Тип даних	Приклади параметрів	Призначення в системі
Дані стану ґрунту	Вологість ґрунту, температура ґрунту	Оцінювання фактичного стану зволоження
Дані мікроклімату	Температура повітря, відносна вологість повітря	Урахування умов висихання та випаровування
Часові дані	Час доби, дата, інтервал після попереднього поливу	Виявлення циклічності та часових закономірностей
Дані керування	Статус насоса, тривалість поливу	Формування навчальних прикладів і перевірка рішень
Похідні ознаки	Середні значення, зміна за інтервал, тренд	Підвищення інформативності моделі

Формування якісного набору даних є одним із найважливіших етапів побудови інтелектуального модуля. Сенсорні вимірювання мають бути не лише зібрані, а й структуровані таким чином, щоб їх можна було використовувати для навчання моделі. Для цього зазвичай формується таблиця спостережень, у якій кожний рядок відповідає певному моменту або інтервалу часу, а стовпці відображають набір ознак. Окремо визначається цільова змінна, тобто те, що саме має передбачати модель. У контексті автоматизованого зрошення це може бути або клас «полив потрібен / полив не потрібен», або прогноз рівня вологості на найближчий часовий горизонт.

З огляду на тему роботи доцільнішим є підхід, за якого цільова змінна визначає доцільність поливу. Така постановка задачі є зрозумілішою для практичного впровадження, оскільки результат моделі безпосередньо використовується у модулі підтримки рішень. Водночас для побудови якісної класифікаційної моделі важливо правильно сформувати навчальні приклади.

Якщо цільовий клас визначається надто спрощено, наприклад лише за одним жорстким порогом вологості, модель фактично відтворюватиме просте правило і не даватиме суттєвого інтелектуального ефекту. Тому доцільно враховувати комбіновані умови, які включають не лише рівень вологості, а й супровідні параметри середовища.

Методи машинного навчання в задачах автоматизованого зрошення застосовуються для виявлення прихованих залежностей між параметрами середовища та потребою у поливі. У наукових працях розглянуто як прості підходи, так і складніші моделі. Зокрема, використовуються KNN, логістична регресія, дерева рішень, випадковий ліс, градієнтні бустингові методи, нейронні мережі та гібридні інтелектуальні рішення [3, 12, 17]. Вибір конкретного алгоритму залежить від обсягу даних, структури ознак, вимог до інтерпретованості та обчислювальних ресурсів.

Прості моделі мають свої переваги. Логістична регресія забезпечує зрозумілу інтерпретацію впливу ознак, а метод k найближчих сусідів є концептуально простим для реалізації. Проте в умовах багатофакторних, нелінійних і частково шумних даних такі підходи часто поступаються точністю більш гнучким моделям. Саме тому в задачах, пов’язаних із сенсорними вимірюваннями, особливу увагу привертають ансамблеві методи машинного навчання [10, 12].

Ансамблеві моделі ґрунтуються на поєднанні кількох базових алгоритмів або кількох дерев рішень, що дає змогу підвищити узагальнювальну здатність, зменшити ризик перенавчання та краще працювати з неоднорідними ознаками. До найпоширеніших ансамблевих підходів належать Random Forest, Extra Trees, Gradient Boosting та споріднені методи. Їх перевага в задачах автоматизованого зрошення полягає у здатності враховувати нелінійні зв’язки між параметрами, оцінювати важливість ознак і стійко працювати за наявності часткового шуму в даних. Крім того, такі моделі зазвичай не потребують надто складного попереднього масштабування ознак, що є практичним для побудови прикладного програмного модуля [17, 21].

Разом із тим методи машинного навчання не усувають усіх проблем

автоматично. Їх ефективність напряму залежить від якості навчального набору даних. Якщо дані містять систематичні помилки, невідповідність між ознаками та цільовою змінною, недостатню кількість спостережень або нерепрезентативні сценарії, модель не зможе сформувати надійне рішення. У зв'язку з цим у системах підтримки рішень важливими стають не лише вибір алгоритму, а й процедури перевірки даних, балансування класів, валідації та порівняння кількох моделей за узгодженими метриками.

Для задачі підтримки рішень щодо поливу доцільно оцінювати моделі не лише за загальною точністю, а й за здатністю правильно виявляти ситуації, коли зрошення справді потрібне. Якщо система занадто часто видає хибнопозитивні рішення, це спричинятиме перевитрати води. Якщо, навпаки, модель пропускатиме необхідний полив, погіршуватиметься стан рослин. Тому під час аналізу якості слід враховувати не лише accuracy, а й precision, recall, F1-міру, матрицю помилок та, за потреби, ROC-криві. Такий підхід є більш обґрунтованим для практичної інтерпретації результатів.

Окремої уваги заслуговує поєднання IoT-інфраструктури та моделей машинного навчання в межах єдиної системи. У такій архітектурі сенсори забезпечують збирання первинних вимірювань, мережевий рівень виконує передавання даних, програмний модуль здійснює підготовку ознак, а модель формує прогноз або класифікаційне рішення. Далі це рішення або безпосередньо впливає на стан виконавчого механізму, або використовується як рекомендація для користувача. Саме така інтеграція дозволяє говорити про повноцінну систему підтримки рішень, а не лише про набір окремих технічних компонентів [7, 16, 23].

З практичної точки зору, для розроблюваного модуля найбільш доцільним є використання набору ознак, який включає поточну вологість ґрунту, температуру повітря, відносну вологість повітря, часові параметри та, за можливості, похідні показники зміни вологості за попередні інтервали. Такий підхід дає змогу формувати модель, яка враховує не лише статичний стан середовища, а й його динаміку. Як наслідок, рішення щодо поливу стає більш адаптивним і стійким до випадкових коливань окремих вимірювань.

Отже, IoT-сенсори формують первинну інформаційну основу автоматизованого зрошення, а дані моніторингу після відповідної підготовки стають базою для застосування методів машинного навчання. Аналіз джерел показав, що найбільшу практичну цінність у цій предметній області мають моделі, здатні працювати з багатофакторними, часово залежними та частково зашумленими даними. Серед них особливо доцільними є ансамблеві методи, які поєднують достатню точність, стійкість та придатність до інтеграції в програмний модуль підтримки рішень.

1.3 Аналіз існуючих систем автоматизованого зрошення та їх обмежень

Розвиток автоматизованого зрошення в останні роки відбувається досить активно, що зумовлено одночасно кількома факторами: поширенням технологій Інтернету речей, здешевленням сенсорного обладнання, підвищенням доступності бездротових засобів зв’язку та зростанням інтересу до методів інтелектуального аналізу даних у сільському господарстві. У результаті сформувалася значна кількість рішень, які відрізняються за рівнем автоматизації, апаратною складовою, складністю програмного забезпечення та способом формування рішення щодо поливу.

Найпростішу групу становлять системи порогового керування. У таких рішеннях до виконавчого пристрою, зазвичай насоса або електромагнітного клапана, підключається датчик вологості ґрунту, а запуск поливу відбувається тоді, коли значення показника стає нижчим за встановлений поріг. Перевагою такого підходу є простота реалізації, низька вартість та зрозуміла логіка функціонування. Саме з цієї причини порогові схеми залишаються поширеними в навчальних, демонстраційних і малобюджетних проєктах [7, 16]. Водночас їх функціональні можливості є обмеженими, оскільки рішення формується на основі одного параметра без належного врахування умов середовища, історії зміни вологості та особливостей конкретної ділянки.

Іншу групу становлять системи дистанційного моніторингу, в яких основна увага приділяється збору та передаванню даних, а керування поливом

виконується користувачем або за спрощеним правилом. У таких рішеннях застосовуються мікроконтролери, хмарні сервіси, мобільні застосунки та панелі візуалізації, що дає змогу спостерігати за поточним станом ґрунту й мікроклімату в реальному часі [6, 23, 33]. Перевага цього підходу полягає у зручності експлуатації та можливості накопичення історичних вимірювань. Проте, якщо аналітичний модуль відсутній або реалізований формально, така система фактично виконує лише функцію спостереження, а не повноцінної підтримки рішень.

Окрему групу формують системи smart irrigation, у яких поєднано сенсорний моніторинг, бездротовий зв’язок, локальну або хмарну аналітику та автоматичне керування поливом. У подібних рішеннях уже враховується кілька параметрів, а рішення щодо зрошення формується не лише за поточним значенням вологості, а й з урахуванням контексту середовища [6, 9]. Такі системи є значно ближчими до концепції точного землеробства, однак навіть у цій категорії часто спостерігається спрощений характер аналітичної частини. У багатьох випадках система називається інтелектуальною лише через наявність сенсорів і мережевого доступу, хоча фактична логіка прийняття рішення залишається правиловою або слабо адаптивною.

У наукових і прикладних роботах також трапляються системи, у яких використовуються методи машинного навчання для прогнозування рівня вологості ґрунту або визначення потреби в поливі. Такі підходи є більш перспективними, оскільки дозволяють враховувати багатофакторні залежності та працювати з історичними даними [3, 12, 17]. Разом із тим не всі з них можна вважати повноцінними системами підтримки рішень. У частині досліджень основна увага приділяється лише побудові моделі та демонстрації її точності, тоді як питання інтеграції моделі з реальною IoT-інфраструктурою, механізмом керування насосом, перевіркою достовірності вхідних даних і обробкою відмов залишаються поза межами розгляду. Така неповнота є суттєвим недоліком, оскільки практична цінність системи визначається саме узгодженістю всіх її компонентів.

Класифікація існуючих систем автоматизованого зрошення за рівнем

інтелектуалізації показує перехід від простих порогових систем до систем моніторингу, далі до багатопараметричних автоматизованих рішень і, нарешті, до інтелектуальних модулів підтримки рішень на основі машинного навчання.

Існуючі системи також відрізняються за архітектурою розгортання. Частина рішень побудована як автономні локальні вузли, у яких датчик, контролер і виконавчий механізм взаємодіють без зовнішньої інфраструктури. Такі системи є простими в реалізації, але мають обмежену масштабованість. Інша частина використовує хмарні сервіси для зберігання, візуалізації та аналітики даних, що полегшує централізований доступ до інформації, проте підвищує залежність від якості мережевого зв'язку та зовнішніх платформ [6, 7]. У більш складних реалізаціях застосовуються розподілені сенсорні мережі, які дають змогу охопити кілька зон спостереження та будувати більш точну модель зрошення для неоднорідних ділянок [8]. Проте зі зростанням кількості вузлів різко ускладнюються задачі синхронізації, збереження даних, обробки пропусків та інтеграції результатів у єдину логіку керування.

Істотною проблемою більшості наявних систем є обмежена якість сенсорних даних. Навіть за коректно реалізованій архітектурі система не може забезпечити надійного рішення, якщо вхідні дані нестабільні, мають значну похибку або не проходять попередньої перевірки. У літературі неодноразово наголошується, що сенсори вологості ґрунту, особливо недорогі, чутливі до особливостей ґрунту, умов монтажу, температурних впливів та тривалості експлуатації [5, 18, 22]. Через це велика частина прикладних рішень працює задовільно лише в лабораторних або демонстраційних умовах, але потребує суттєвого доопрацювання для стабільного використання в реальному середовищі.

Ще одним поширеним обмеженням є недостатня глибина програмно-аналітичної частини. Багато рішень реалізують лише ланцюг «зчитування – передавання – відображення – вмикання насоса», але не забезпечують повноцінного аналізу історичних даних, оцінювання важливості ознак, прогнозування майбутнього стану або адаптації до нових умов експлуатації. У таких системах інтелектуальний компонент або відсутній, або має формальний

характер. Відповідно, навіть за наявності IoT-інфраструктури рішення щодо поливу залишається спрощеним, а система не використовує повною мірою потенціал даних, що накопичуються [21, 24].

У таблиці 1.3 доцільно узагальнити типові переваги та обмеження існуючих підходів.

Таблиця 1.3 – Характеристика існуючих підходів до автоматизованого зрошення

Тип системи	Переваги	Основні обмеження
Порогова система	Простота, низька вартість, швидка реалізація	Використання одного параметра, низька адаптивність
Система дистанційного моніторингу	Збір історії даних, віддалений доступ, візуалізація	Відсутність або слабкість модуля підтримки рішень
Багатопараметрична автоматизована система	Урахування кількох сенсорів, вища точність реагування	Часто правилова логіка, складніше налаштування
Система з ML-моделлю	Здатність виявляти складні залежності, краща адаптивність	Високі вимоги до якості даних, потреба в навчанні та валідації

Важливим обмеженням значної частини існуючих рішень є також недостатня інтерпретованість рішень. У прикладних системах користувач або оператор повинен розуміти, чому саме система рекомендує або виконує полив. Якщо рішення формується як «чорна скринька» без пояснення, знижується довіра до системи, ускладнюється налаштування параметрів і стає проблематичним виявлення помилок. Саме тому в сучасних інтелектуальних рішеннях бажаним є використання моделей, які або дозволяють оцінити важливість ознак, або можуть бути доповнені пояснювальними механізмами. У цьому аспекті ансамблеві методи мають практичну перевагу над деякими

складнішими моделями, оскільки дають змогу оцінювати внесок окремих ознак у процес формування рішення [17].

Не менш важливою є проблема перенесення результатів із дослідницького середовища в прикладне використання. У багатьох публікаціях описуються окремі вдалі експерименти, але вони виконуються на обмежених наборах даних, у спрощених умовах або без тривалої перевірки стабільності системи. Через це отримані результати не завжди легко масштабуються до реальних польових сценаріїв. Особливо це стосується систем, у яких не враховано пропуски вимірювань, аномалії, змінність умов навколишнього середовища та довготривалу експлуатацію сенсорів [8, 19]. Отже, при розробленні нового модуля необхідно орієнтуватися не лише на демонстрацію працездатності, а й на відтворюваність, стійкість і придатність до реальної інтеграції.

Аналіз наявних систем показує, що найбільш перспективним напрямом є створення інтегрованого інтелектуального модуля, у якому поєднано кілька рівнів: сенсорний збір даних, попередню обробку, формування ознак, застосування ансамблевої моделі машинного навчання, логіку прийняття рішення та керування виконавчим пристроєм. Саме така архітектура дозволяє усунути основні недоліки наявних підходів: надмірну спрощеність, низьку адаптивність, слабку аналітичну складову та недостатню стійкість до нестабільних вхідних даних. Крім того, вона відповідає сучасним вимогам до цифрових аграрних систем, у яких рішення має формуватися на основі не окремого показника, а сукупності взаємопов’язаних параметрів [10, 30, 31].

Таким чином, існуючі системи автоматизованого зрошення створили технологічну основу для переходу до більш інтелектуальних рішень, однак значна частина з них залишається або надто спрощеною, або неповною з погляду програмної архітектури та аналітичного функціоналу. Це дозволяє стверджувати, що задача розроблення інтелектуального модуля підтримки рішень щодо автоматизованого зрошення є обґрунтованою та практично значущою.

1.4 Постановка задачі дослідження

Проведений аналіз предметної області показав, що сучасні системи автоматизованого зрошення поступово переходять від простих засобів контролю вологості до комплексних цифрових рішень, які поєднують сенсорний моніторинг, передавання даних, аналітичну обробку та автоматичне керування поливом. Разом із тим значна частина наявних систем або використовує спрощені порогові правила, або обмежується функціями спостереження та візуалізації. За такого підходу рішення щодо поливу формується без достатнього врахування сукупності параметрів середовища, що знижує адаптивність системи та не дозволяє повною мірою реалізувати потенціал IoT-інфраструктури й методів машинного навчання.

У задачах автоматизованого зрошення важливим є не лише факт вимірювання вологості ґрунту, а й інтерпретація отриманих даних у контексті температури повітря, вологості повітря, часових параметрів і динаміки попередніх змін. Саме тому виникає потреба у створенні інтелектуального модуля, здатного працювати з багатофакторними сенсорними даними, виявляти закономірності між параметрами середовища та формувати обґрунтоване рішення щодо доцільності поливу. Такий модуль має стати центральною аналітичною складовою системи автоматизованого зрошення.

Отже, метою роботи є розроблення інтелектуального модуля підтримки рішень щодо автоматизованого зрошення, який використовує дані IoT-сенсорів і ансамблеві моделі машинного навчання для визначення доцільності поливу та підвищення ефективності використання водних ресурсів.

Для досягнення поставленої мети сформовано такі завдання дослідження:

– проаналізувати предметну область автоматизованого зрошення, сучасні IoT-засоби моніторингу та підходи до інтелектуальної підтримки рішень;

– дослідити існуючі методи вимірювання вологості ґрунту, обробки сенсорних даних і моделі машинного навчання, придатні для задач зрошення;

– сформувати вимоги до інтелектуального модуля підтримки рішень щодо поливу;

– розробити архітектуру системи збору, передавання, збереження та аналітичної обробки даних IoT-сенсорів;

– сформувати ознаковий простір і підготувати дані для навчання ансамблевих моделей;

– розробити алгоритм підтримки рішень щодо автоматизованого зрошення;

– реалізувати програмний прототип модуля;

– провести експериментальне дослідження та оцінити ефективність запропонованого підходу.

У результаті виконання цих завдань має бути сформовано програмний модуль, який інтегрує сенсорний моніторинг, обробку даних і алгоритми машинного навчання для підтримки рішень щодо автоматизованого зрошення.

2 ПРОЄКТУВАННЯ ІНТЕЛЕКТУАЛЬНОГО МОДУЛЯ ПІДТРИМКИ РІШЕНЬ ЩОДО АВТОМАТИЗОВАНОГО ЗРОШЕННЯ

2.1 Формування вимог до інтелектуального модуля

Проєктування інтелектуального модуля підтримки рішень щодо автоматизованого зрошення потребує чіткого визначення вимог до його функціонування. Саме вони задають межі системи, склад вхідних і вихідних даних, а також основні характеристики майбутнього програмного рішення.

Основне призначення модуля полягає в аналізі даних, що надходять від IoT-сенсорів, і формуванні рішення щодо доцільності поливу. Отже, модуль має не лише приймати та зберігати дані, а й виконувати їх попередню обробку, формувати ознаки для моделі машинного навчання та видавати обґрунтований результат.

До функціональних вимог належать:

– приймання даних від сенсорів вологості ґрунту, температури та вологості повітря;

– перевірка коректності вхідних значень і попередня обробка даних;

– формування ознакового простору для моделі;

– застосування ансамблевої моделі машинного навчання;

– формування рішення «полив потрібен / полив не потрібен»;

– збереження історії вимірювань і результатів роботи системи.

До нефункціональних вимог належать точність, надійність, оперативність, модульність і масштабованість. Модуль має коректно працювати за наявності шуму або пропусків у даних, формувати рішення за прийнятний час та допускати подальше розширення за рахунок нових сенсорів або моделей. Важливою є також інтерпретованість результатів, оскільки в прикладній системі необхідно розуміти, які параметри найбільше впливають на рішення щодо поливу [17, 28].

У таблиці 2.1 узагальнено вимоги до інтелектуального модуля.

Таблиця 2.1 – Основні вимоги до інтелектуального модуля

Група вимог	Зміст
Функціональні	Приймання даних, попередня обробка, формування ознак, застосування ML-моделі, формування рішення
Інформаційні	Робота з даними вологості ґрунту, температури, вологості повітря та часовими параметрами
Якісні	Точність, надійність, оперативність
Архітектурні	Модульність, масштабованість, можливість розширення

Таким чином, сформовані вимоги визначають, що інтелектуальний модуль має працювати з багатофакторними сенсорними даними, виконувати їх аналітичну обробку та забезпечувати підтримку рішень щодо автоматизованого зрошення.

2.2 Розроблення архітектури модуля

Побудова інтелектуального модуля підтримки рішень щодо автоматизованого зрошення потребує визначення такої архітектури, яка забезпечує узгоджену роботу всіх її складових: сенсорного вузла, підсистеми передавання даних, програмного модуля обробки інформації та виконавчого механізму поливу. Архітектура в цьому випадку визначає не лише склад компонентів, а й порядок їхньої взаємодії, послідовність руху даних та спосіб формування керуючого рішення.

У межах роботи розроблено багаторівневу архітектуру для автоматизованого зрошення, орієнтовану на використання даних IoT-сенсорів і подальше застосування ансамблевих моделей машинного навчання. Її основу становить послідовний ланцюг: зчитування параметрів середовища, первинна перевірка та передавання даних, їх накопичення й обробка, формування ознак, визначення потреби у поливі та надсилання команди на виконавчий пристрій.

На рисунку 2.1 подано загальну архітектуру модуля. У схемі відображено чотири основні рівні: сенсорний, рівень передавання даних, аналітичний рівень і

рівень керування виконавчим пристроєм.

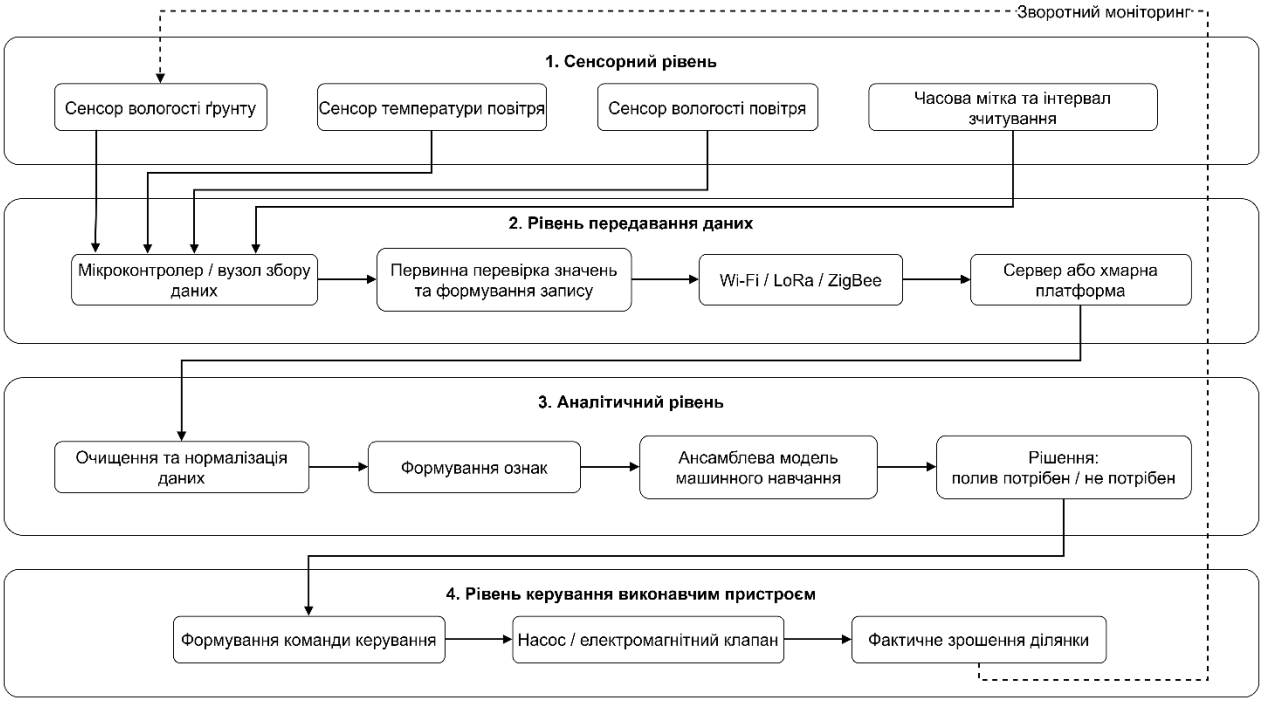

Рисунок 2.1 – Узагальнена схема архітектури інтелектуального модуля [34]

Першим рівнем є сенсорний рівень, призначення якого полягає у фіксації поточного стану ґрунту та навколишнього середовища. Основними джерелами інформації є датчики вологості ґрунту, температури повітря та відносної вологості повітря. Саме ці параметри формують базовий набір вхідних даних для подальшого аналізу. За потреби архітектура допускає розширення шляхом підключення додаткових сенсорів, наприклад температури ґрунту, освітленості або інших параметрів мікроклімату. Така побудова робить модуль гнучким та придатним подальшого вдосконалення.

Сенсорний вузол повинен працювати в режимі періодичного опитування, тобто зчитувати показники через визначені проміжки часу. Такий підхід дає змогу накопичувати історію вимірювань і надалі враховувати не лише поточний стан середовища, а й динаміку його змін. Кожне вимірювання повинно супроводжуватися часовою позначкою, оскільки часовий контекст є важливою складовою набору даних для моделі підтримки рішень. Відсутність часової прив’язки суттєво знижує аналітичну цінність сенсорної інформації.

Наступним елементом архітектури є компонент збору даних, якого

реалізує мікроконтролер. Цей компонент виконує функцію проміжної ланки між сенсорами та програмним модулем. Його призначення полягає у прийманні показників із датчиків, попередній перевірці значень, приведенні їх до уніфікованого формату та передаванні до системи подальшої обробки. Фактично саме на цьому рівні формується первинний структурований запис, що містить значення параметрів і часову мітку.

Для збору даних важливими є кілька функцій. По-перше, має бути забезпечено циклічне зчитування показників із сенсорів. По-друге, необхідно перевіряти, чи належать отримані значення до допустимого діапазону. Якщо, наприклад, значення вологості ґрунту є фізично неможливим або різко відрізняється від попередніх спостережень, такий запис доцільно позначати як підозрілий. По-третє, передавання даних має відбуватися у форматі, зручному для подальшого завантаження в програмний модуль обробки.

Другим рівнем архітектури є рівень передавання даних. Його завдання полягає у доставці сформованих сенсорним вузлом записів до середовища зберігання й аналізу. У межах цієї роботи передбачається використання бездротового каналу зв’язку, зокрема Wi-Fi, як найбільш доступного варіанта для прототипу системи. Водночас структура архітектури не виключає використання інших IoT-технологій, наприклад LoRa або ZigBee, якщо цього потребуватимуть умови розгортання.

Передавання даних не повинно зводитися лише до механічного надсилання вимірювань. Важливо забезпечити цілісність потоку, правильний порядок записів та можливість повторної передачі у випадку короткочасного збою. Для навчального прототипу такі механізми можуть бути реалізовані у спрощеному вигляді, однак сама архітектура має враховувати ймовірність нестабільності каналу зв’язку. Це особливо важливо для аграрного середовища, де умови експлуатації сенсорних вузлів можуть бути змінними.

Третім рівнем є рівень накопичення та аналітичної обробки даних. Саме на цьому рівні модуль переходить від простого моніторингу до інтелектуального аналізу. Дані, що надходять від сенсорного вузла, повинні накопичуватися у сховищі – локальному або хмарному – та проходити етап попередньої обробки.

До основних операцій цього етапу належать очищення даних, усунення пропусків, перевірка аномалій, нормалізація, агрегація та формування похідних ознак.

Попередня обробка є необхідною, оскільки сирі сенсорні дані не завжди придатні для безпосереднього використання в моделі машинного навчання. Значення можуть містити окремі спотворення, пропущені записи або різкі коливання, які не відображають реального стану середовища. Крім того, для моделі недостатньо лише поточних показників. Доцільно формувати додаткові ознаки, наприклад середнє значення за інтервал, зміну вологості за попередній період, часові характеристики та інші параметри, що підвищують інформативність вхідного набору.

Центральним компонентом цього рівня є аналітичний модуль, у якому виконується застосування ансамблевої моделі машинного навчання. На вхід модуля надходить підготовлений набір ознак, а на виході формується оцінка доцільності поливу. У межах роботи доцільною є класифікаційна постановка, за якої модель визначає один із двох станів: полив потрібен або полив не потрібен. Такий формат є найбільш придатним для подальшої інтеграції з виконавчим механізмом системи.

Четвертим рівнем архітектури є рівень керування виконавчим пристроєм. Його призначення полягає у перетворенні аналітичного рішення на фізичну дію. Якщо модель визначає, що полив є доцільним, то передається команда на ввімкнення насоса або відкриття електромагнітного клапана. Якщо підстав для поливу немає, команда не формується.

Важливо, що архітектура повинна забезпечувати не лише прямий потік «дані → рішення → дія», а й зворотний контур контролю. Після виконання поливу модуль продовжує зчитувати показники сенсорів, що дозволяє оцінити зміну вологості та надалі використовувати ці дані для уточнення роботи моделі. Такий підхід формує замкнений цикл функціонування, у межах якого кожне нове вимірювання потенційно покращує якість подальших рішень.

Узагальнену логіку роботи архітектури можна подати як послідовність таких етапів:

– зчитування параметрів ґрунту та мікроклімату із сенсорів;
– передавання даних до підсистеми збору;
– перевірка та формування структурованого запису;
– передавання даних до середовища обробки;
– очищення, нормалізація та формування ознак;
– застосування моделі машинного навчання;
– формування рішення щодо поливу;
– надсилання команди на виконавчий пристрій;
– повторний моніторинг стану середовища.

Перевагою запропонованої архітектури є її модульність. Кожен рівень виконує окрему функцію й може вдосконалюватися без повної перебудови всієї системи. Наприклад, набір сенсорів може бути розширений, канал передавання даних може бути змінений, а модель машинного навчання – замінена на іншу без зміни загальної логіки функціонування. Така властивість є важливою для подальшого масштабування.

Не менш важливою є практична придатність архітектури. Вона орієнтована на реальний цикл автоматизованого зрошення, у якому дані збираються з фізичних датчиків, проходять кілька етапів обробки, після чого безпосередньо впливають на керування поливом..

Таким чином, розроблено архітектуру модуля автоматизованого зрошення, яка охоплює сенсорний рівень, рівень передавання даних, аналітичний рівень і рівень керування виконавчим пристроєм.

2.3 Підготовка даних для навчання моделей

Якість роботи інтелектуального модуля значною мірою визначається не лише вибором алгоритму машинного навчання, а й тим, наскільки коректно сформовано набір вхідних ознак. Для задач автоматизованого зрошення це особливо важливо, оскільки рішення щодо поливу залежить від сукупності параметрів середовища, а не від одного окремого показника. Саме тому на цьому етапі необхідно визначити склад ознакового простору, структуру навчальних

даних та порядок їх підготовки до використання в моделі.

Основу вхідних даних формують показники, що надходять від IoT-сенсорів. До базових ознак доцільно віднести вологість ґрунту, температуру повітря, відносну вологість повітря, а також часові параметри – час доби, дату або номер циклу зчитування. Такий набір є достатнім для побудови початкової моделі підтримки рішень, оскільки він охоплює як безпосередній стан ґрунту, так і зовнішні умови, що впливають на швидкість висихання та зміну потреби у поливі [6, 17].

Разом із базовими ознаками доцільно формувати і похідні. До них можуть належати середні значення за певний інтервал, різниця між поточним і попереднім вимірюванням, зміна вологості ґрунту за останній часовий крок, а також прості узагальнені характеристики динаміки даних. Використання таких ознак дозволяє моделі враховувати не лише поточний стан середовища, а й напрям зміни параметрів. Це особливо важливо в тих випадках, коли значення вологості ґрунту ще не є критичним, але спостерігається стійка тенденція до зниження.

Узагальнену структуру ознакового простору подано на рисунку 2.2. У схемі показано перехід від сирих сенсорних вимірювань до підготовленого набору ознак, який надходить на вхід моделі машинного навчання.

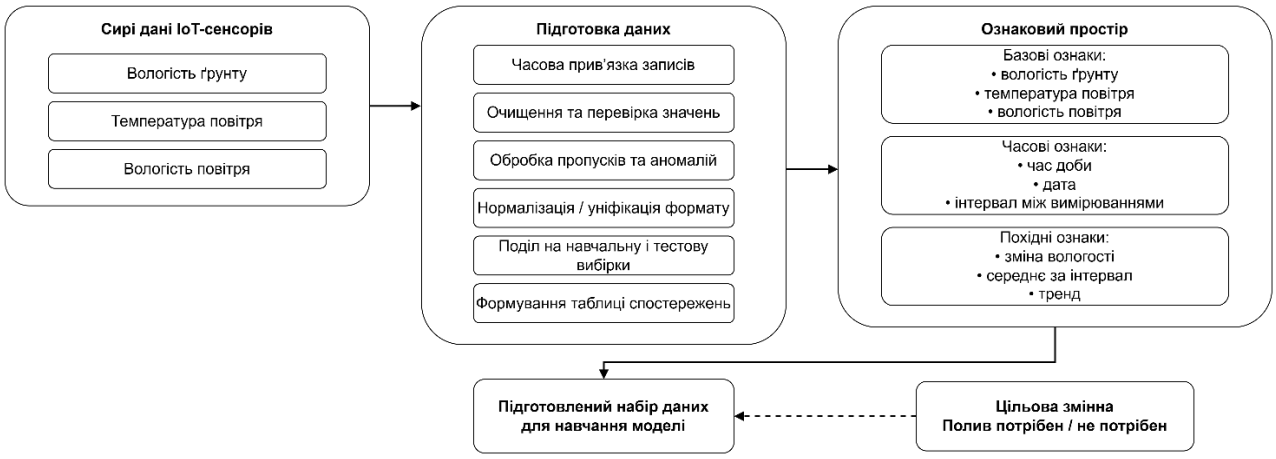

Рисунок 2.2 – Узагальнена структурна схема ознакового простору

Формування навчального набору даних передбачає, що кожний запис містить вектор ознак та відповідне цільове значення. У межах цієї роботи

доцільно використовувати класифікаційну постановку задачі, тобто як цільову змінну визначати один із двох станів: «полив потрібен» або «полив не потрібен». Такий підхід є найбільш зручним для подальшої інтеграції результату з модулем керування виконавчим пристроєм.

Важливим етапом є попередня обробка даних. Сирі сенсорні значення можуть містити пропуски, аномалії або некоректні показники, які виникають через нестабільність датчиків чи збої передавання. Тому перед формуванням ознакового простору необхідно виконати очищення даних, видалення або заміну некоректних значень, упорядкування записів за часом та перевірку діапазонів допустимих значень [5, 18]. Після цього виконується нормалізація або приведення даних до єдиного формату, якщо цього потребує обрана модель.

Для підготовки набору до навчання також необхідно здійснити поділ даних на навчальну та тестову вибірки. Навчальна частина використовується для побудови моделі, а тестова – для перевірки її здатності узагальнювати закономірності на нових спостереженнях. Якщо цього не зробити, результати оцінювання будуть необ’єктивними, а модель може виявитися перенавченою.

У таблиці 2.2 узагальнено основні групи ознак, що використовуються в модулі.

Таблиця 2.2 – Структура ознакового простору для навчання моделей

Група ознак	Приклади
Базові сенсорні ознаки	Вологість ґрунту, температура повітря, вологість повітря
Часові ознаки	Час доби, дата, інтервал між вимірюваннями
Похідні ознаки	Зміна вологості, середнє значення за інтервал, тренд
Цільова змінна	Полив потрібен / полив не потрібен

Таким чином, формування ознакового простору в межах цієї роботи передбачає перетворення сирих даних IoT-сенсорів у впорядкований набір інформативних параметрів, придатних для навчання ансамблевих моделей машинного навчання.

2.4 Розроблення алгоритму підтримки рішень щодо поливу

Після визначення архітектури системи та формування ознакового простору наступним етапом є розроблення алгоритму підтримки рішень щодо поливу. Саме алгоритм задає послідовність опрацювання сенсорних даних і визначає, яким чином система переходить від потоку вимірювань до конкретного керуючого висновку. У межах цієї роботи алгоритм повинен забезпечувати поєднання правил попередньої перевірки даних, етапів підготовки ознак і результатів ансамблевої моделі машинного навчання.

Основне призначення алгоритму полягає у визначенні доцільності поливу на основі поточних та похідних параметрів, отриманих від IoT-сенсорів. На відміну від простих порогових схем, у яких рішення залежить лише від одного значення вологості ґрунту, запропонований алгоритм враховує сукупність показників та їхню зміну в часі. Завдяки цьому підвищується адаптивність системи й зменшується ризик помилкових рішень у випадках, коли окремий параметр не дає повної картини стану середовища.

Алгоритм функціонування модуля доцільно організувати як послідовність взаємопов’язаних кроків. На першому етапі виконується зчитування сенсорних даних. Система отримує значення вологості ґрунту, температури повітря, вологості повітря та часову мітку. Після цього проводиться первинна перевірка вхідних даних. Якщо хоча б один із показників є відсутнім, виходить за допустимі межі або має явно аномальний характер, запис або відхиляється, або передається на етап корекції залежно від обраної стратегії обробки.

Другим етапом є попередня обробка даних. На цьому кроці виконується очищення значень, приведення їх до єдиного формату та, за потреби, нормалізація. Далі формуються додаткові ознаки, які підвищують інформативність вхідного вектора. До таких ознак можуть належати зміна вологості ґрунту за попередній інтервал, середнє значення за кілька останніх вимірювань, часові характеристики та інші допоміжні параметри. У результаті формується повний вектор ознак, який надходить на вхід моделі машинного навчання.

Третім етапом є застосування ансамблевої моделі. Модель опрацьовує сформований вектор ознак і визначає результат класифікації. У найпростішому випадку результатом є одна з двох відповідей: полив потрібен або полив не потрібен. У разі використання імовірнісного виходу модель може додатково формувати оцінку впевненості, на основі якої приймається остаточне рішення. Такий підхід є більш гнучким, оскільки дозволяє коригувати чутливість системи залежно від умов експлуатації.

Після отримання результату моделі виконується етап формування керуючого рішення. Якщо ймовірність необхідності поливу перевищує заданий поріг або класифікаційний результат прямо вказує на потребу в зрошенні, формується команда на запуск виконавчого пристрою. Якщо підстав для поливу немає, команда не подається, а система переходить до наступного циклу моніторингу. У практичному застосуванні доцільно також передбачити додаткові обмеження, наприклад заборону повторного запуску поливу протягом занадто короткого інтервалу часу або блокування дії за наявності некоректних даних.

На завершальному етапі алгоритм виконує збереження результатів. У сховищі фіксуються вхідні дані, сформовані ознаки, результат роботи моделі та прийняте рішення. Такий підхід є важливим не лише для журналювання, а й для подальшого аналізу ефективності модуля, повторного навчання моделей та виявлення типових сценаріїв роботи системи.

Узагальнений алгоритм підтримки рішень щодо поливу можна подати у вигляді такої послідовності кроків:

1. Зчитати поточні дані з IoT-сенсорів.
2. Перевірити коректність і повноту отриманих значень.
3. Виконати очищення та попередню обробку даних.
4. Сформувати базові, часові та похідні ознаки.
5. Передати вектор ознак до ансамблевої моделі машинного навчання.
6. Отримати результат класифікації або ймовірність потреби у поливі.
7. Сформувати підсумкове рішення щодо зрошення.
8. За потреби передати команду на виконавчий пристрій.

9. Зберегти результати роботи та перейти до наступного циклу моніторингу.

На рисунку 2.3 подано схему алгоритму підтримки рішень щодо поливу. У схемі варто показано основні етапи: отримання даних, перевірку, підготовку ознак, роботу моделі, формування рішення та подання команди на виконавчий механізм.

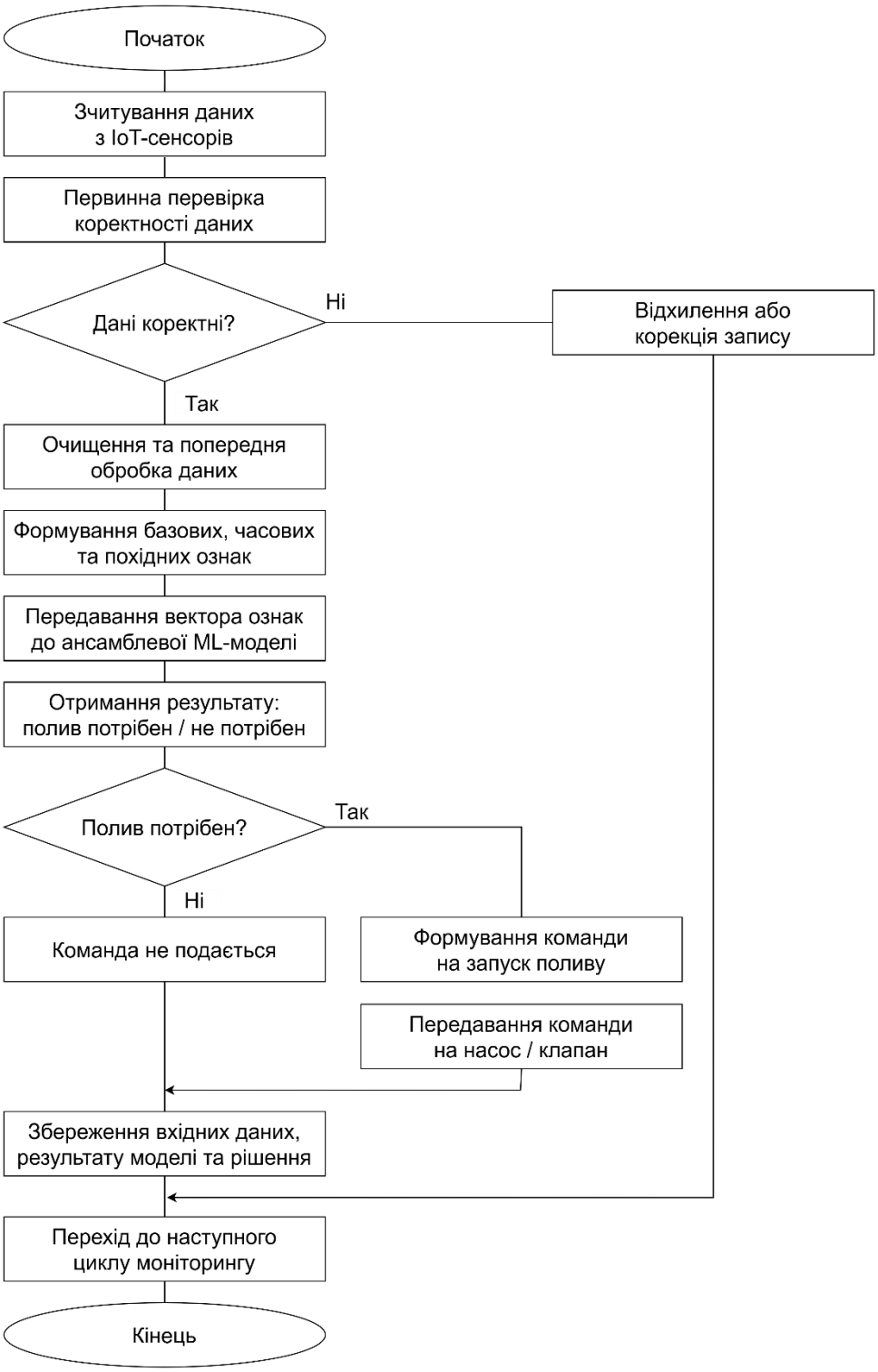

Рисунок 2.3 – Схема алгоритму підтримки рішень щодо поливу

Перевагою розробленого алгоритму є його поєднання з логікою інтелектуального аналізу. На відміну від жорстко заданого правила, запропонований підхід дозволяє враховувати багатофакторний характер задачі. Крім того, алгоритм є достатньо універсальним, оскільки допускає заміну моделі машинного навчання без зміни загальної послідовності роботи. Це робить систему придатною до подальшого вдосконалення.

Таким чином, розроблений алгоритм підтримки рішень забезпечує формалізований порядок функціонування модуля автоматизованого зрошення. Він охоплює всі основні етапи – від отримання сенсорних даних до видачі керуючої команди – та створює основу для подальшої програмної реалізації системи.

3 РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ МОДУЛЯ

3.1 Програмна реалізація модуля збору, обробки даних і підтримки рішень

Програмну реалізацію інтелектуального модуля підтримки рішень щодо автоматизованого зрошення виконано у вигляді окремого Python-проєкту, орієнтованого на роботу з даними IoT-сенсорів, їх попередню обробку, навчання моделей машинного навчання та формування рішення щодо необхідності поливу. Такий підхід забезпечує модульність розробки, спрощує тестування окремих компонентів і дає змогу надалі інтегрувати програмний модуль із фізичним сенсорним вузлом або хмарною платформою.

Основну логіку роботи модуля побудовано навколо послідовного конвеєра обробки даних. Спочатку дані завантажуються з файлу або з потоку сенсорних вимірювань, після чого виконується перевірка коректності значень. Далі формується ознаковий простір, модель машинного навчання виконує класифікацію стану, а модуль підтримки рішень визначає, чи потрібно запускати полив. Результати роботи зберігаються для подальшого аналізу.

У таблиці 3.1 подано основні програмні засоби, використані для реалізації модуля. Вибір Python обґрунтовано його широкими можливостями для обробки табличних даних, навчання моделей машинного навчання та швидкого створення прикладних прототипів. Бібліотека scikit-learn використовується для реалізації ансамблевих моделей, що відповідає задачі дослідження [17].

Структуру програмного проєкту сформовано таким чином, щоб кожен файл відповідав за окрему частину функціонування модуля. Такий підхід зменшує зв’язаність компонентів і дозволяє змінювати логіку обробки даних або модель машинного навчання без повної перебудови програми.

На рисунку 3.1 наведено узагальнену структуру програмного модуля підтримки рішень щодо автоматизованого зрошення.

Таблиця 3.1 – Програмні засоби реалізації інтелектуального модуля

Компонент	Обраний засіб	Призначення
Мова програмування	Python 3.11	Реалізація логіки обробки даних і моделей машинного навчання
Обробка даних	pandas, numpy	Завантаження, очищення та перетворення сенсорних даних
Машинне навчання	scikit-learn	Навчання, тестування та оцінювання моделей
Візуалізація	matplotlib	Побудова графіків і діаграм результатів
Збереження моделі	joblib	Серіалізація навченої моделі
Формат даних	CSV	Зберігання сенсорних вимірювань

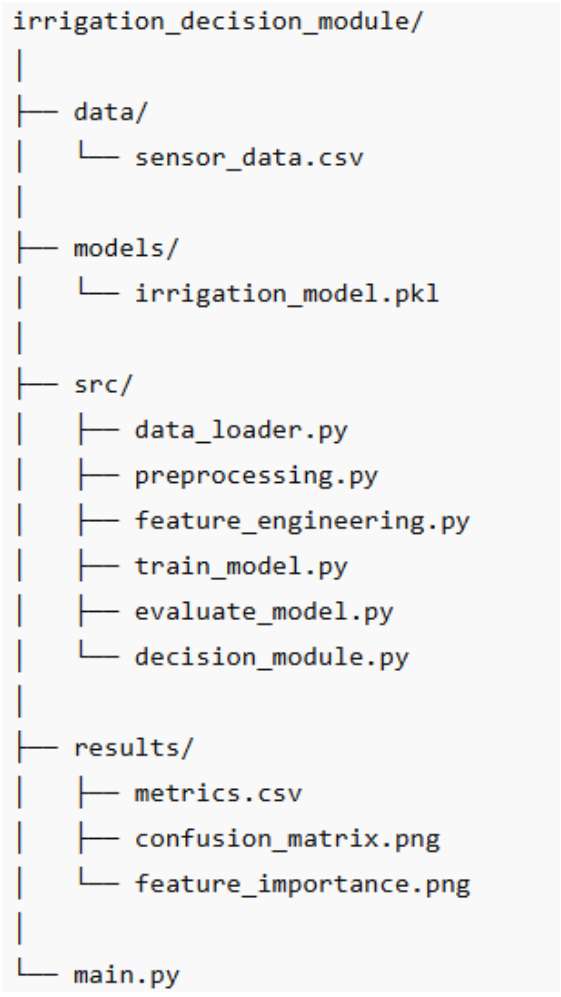

Рисунок 3.1 – Структура програмного модуля підтримки рішень щодо автоматизованого зрошення

У каталозі `data` зберігається набір сенсорних вимірювань. Каталог `models` призначений для збереження навченої моделі. У каталозі `src` розміщено основні програмні файли, які відповідають за завантаження даних, попередню обробку, формування ознак, навчання, оцінювання та прийняття рішення. Каталог `results` містить результати експериментального дослідження: метрики, матрицю помилок і графіки важливості ознак.

Початковим етапом роботи модуля є завантаження сенсорних даних. У межах реалізації передбачено роботу з CSV-файлом, який імітує або містить реальні вимірювання з IoT-сенсорів. Кожен запис включає часову мітку, значення вологості ґрунту, температуру повітря, вологість повітря та цільову ознаку, що вказує на потребу в поливі.

Фрагмент реалізації завантаження даних подано в лістингу 3.1.

Лістинг 3.1 – Завантаження сенсорних даних

```
import pandas as pd

def load_sensor_data(path: str) -> pd.DataFrame:
    data = pd.read_csv(path)
    data["timestamp"] = pd.to_datetime(data["timestamp"])
    data = data.sort_values("timestamp")
    return data
```

Після завантаження даних виконується їх попередня перевірка (лістинг 3.2). Для сенсорних систем цей етап є обов’язковим, оскільки вимірювання можуть містити пропуски, некоректні значення або різкі стрибки, що не відповідають фізичному стану середовища. У реалізованому модулі перевіряються допустимі діапазони значень вологості ґрунту, температури та вологості повітря. Некоректні записи вилучаються або замінюються залежно від типу помилки.

Лістинг 3.2 – Попередня перевірка сенсорних значень

```
def clean_sensor_data(data):
    data = data.dropna()

    data = data[
        (data["soil_moisture"] >= 0) &
        (data["soil_moisture"] <= 100) &
        (data["air_temperature"] >= -10) &
```

```
        (data["air_temperature"] <= 60) &
        (data["air_humidity"] >= 0) &
        (data["air_humidity"] <= 100)
    ]

    return data.reset_index(drop=True)
```

Наступним етапом є формування ознак. Окрім базових показників, у програмному модулі створюються додаткові параметри: година доби, зміна вологості ґрунту між сусідніми вимірюваннями та середнє значення вологості за кілька останніх записів. Такі ознаки дозволяють врахувати не лише поточний стан, а й динаміку зміни вологості (таблиця 3.2).

Таблиця 3.2 – Основні поля набору даних

Поле	Тип даних	Опис
timestamp	datetime	Час зчитування сенсорних показників
soil_moisture	float	Поточна вологість ґрунту, %
air_temperature	float	Температура повітря, °C
air_humidity	float	Відносна вологість повітря, %
hour	int	Година доби
moisture_delta	float	Зміна вологості між вимірюваннями
moisture_mean_3	float	Середня вологість за три останні вимірювання
irrigation_required	int	Цільова змінна: 1 – полив потрібен, 0 – не потрібен

У таблиці 3.2 наведено структуру набору даних, який використовується програмним модулем. Цільова змінна `irrigation_required` задає результат класифікації. Якщо її значення дорівнює 1, система визначає потребу у поливі; якщо 0 – полив не виконується.

Фрагмент формування ознак подано в лістингу 3.3.

Лістинг 3.3 – Формування ознак для моделі машинного навчання

```
def create_features(data):
    data["hour"] = data["timestamp"].dt.hour
```

```
    data["moisture_delta"] = (
        data["soil_moisture"].diff().fillna(0)
    )

    data["moisture_mean_3"] = (
        data["soil_moisture"]
        .rolling(window=3, min_periods=1)
        .mean()
    )

    return data
```

Після підготовки ознак дані передаються до модуля машинного навчання. У межах програмної реалізації передбачено навчання кількох моделей, однак основною для подальшого прийняття рішень обрано Random Forest. Такий вибір пояснюється стійкістю цієї моделі до шуму, здатністю працювати з нелінійними залежностями та можливістю оцінювати важливість ознак.

Модуль прийняття рішення використовує навчену модель і вхідний вектор ознак. На його виході формується текстова або логічна відповідь: полив потрібен або полив не потрібен. У разі використання ймовірнісного прогнозу додатково застосовується поріг прийняття рішення (лістинг 3.4).

Лістинг 3.4 – Формування рішення щодо поливу

```
def make_irrigation_decision(model, features, threshold=0.5):
    probability = model.predict_proba(features)[0][1]

    if probability >= threshold:
        return {
            "decision": "IRRIGATION_REQUIRED",
            "probability": round(probability, 3)
        }

    return {
        "decision": "IRRIGATION_NOT_REQUIRED",
        "probability": round(probability, 3)
    }
```

Результат роботи функції містить не лише саме рішення, а й імовірність належності поточного стану до класу «полив потрібен». Це підвищує інформативність модуля та дає змогу в майбутньому змінювати поріг спрацювання залежно від агротехнічних умов або типу культури.

Загальна логіка роботи реалізованого програмного модуля передбачає такий порядок виконання:

– завантаження сенсорних даних;
– перевірка та очищення значень;
– формування базових і похідних ознак;
– завантаження або навчання моделі машинного навчання;
– передавання ознак до моделі;
– отримання прогнозу;
– формування рішення щодо поливу;
– збереження результату.

У додатку А наведено основні фрагменти програмного коду інтелектуального модуля підтримки рішень щодо автоматизованого зрошення.

Таким чином, у підрозділі реалізовано програмну основу інтелектуального модуля підтримки рішень щодо автоматизованого зрошення. Розроблена структура забезпечує повний шлях обробки даних: від отримання сенсорних вимірювань до формування рішення про необхідність поливу.

3.2 Реалізація та навчання ансамблевих моделей машинного навчання

Для реалізації інтелектуального модуля підтримки рішень щодо автоматизованого зрошення використано класифікаційну постановку задачі. Модель повинна визначати, чи є потреба у поливі в конкретний момент часу на основі поточних і похідних параметрів, сформованих із даних IoT-сенсорів. Такий підхід є практичним, оскільки результат роботи моделі безпосередньо може бути використаний для керування насосом або електромагнітним клапаном.

У межах реалізації використано кілька моделей машинного навчання, серед яких основну увагу приділено ансамблевим алгоритмам. До експериментального порівняння включено Decision Tree, Random Forest, Extra Trees та Gradient Boosting. Модель Decision Tree використано як базовий варіант, а ансамблеві моделі – як основні методи, здатні краще працювати з нелінійними залежностями між параметрами середовища.

Для навчання моделей використано набір даних, сформований на основі

сенсорних параметрів. Кожен запис описує стан середовища в певний момент часу та містить цільову змінну `irrigation_required`, яка визначає потребу у поливі (таблиця 3.3).

Таблиця 3.3 – Вхідні ознаки для навчання моделей машинного навчання

Ознака	Позначення в наборі даних	Призначення
Вологість ґрунту	`soil_moisture`	Основний параметр для оцінювання потреби у поливі
Температура повітря	`air_temperature`	Урахування впливу температури на висихання ґрунту
Вологість повітря	`air_humidity`	Оцінювання мікрокліматичних умов
Година доби	`hour`	Урахування добової циклічності
Зміна вологості	`moisture_delta`	Аналіз динаміки вологості між вимірюваннями
Середня вологість	`moisture_mean_3`	Згладжена характеристика стану ґрунту
Цільова змінна	`irrigation_required`	Клас: 1 – полив потрібен, 0 – полив не потрібен

Як видно з таблиці 3.3, модель отримує не лише поточні сенсорні значення, а й похідні ознаки, які відображають зміну стану ґрунту в часі. Це дає змогу уникнути надмірної залежності від одного окремого вимірювання та підвищити стійкість модуля до випадкових коливань даних.

Перед навчанням моделей набір даних поділено на матрицю ознак X та вектор цільових значень y. До матриці ознак включено всі параметри, крім часової мітки та цільової змінної. Після цього дані поділено на навчальну та тестову вибірки у співвідношенні 80 % до 20 % (лістинг 3.5). Такий поділ дозволяє навчити моделі на основній частині даних і перевірити їх якість на окремих записах, які не використовувалися під час навчання.

Загальну послідовність навчання моделей машинного навчання подано на рисунку 3.2. Схема відображає повний процес: від завантаження набору сенсорних даних і попередньої обробки до навчання моделей, оцінювання їх якості, вибору найкращого варіанта та збереження моделі для подальшого використання в модулі підтримки рішень.

Рисунок 3.2 – Послідовність навчання моделей машинного навчання

Лістинг 3.5 – Формування навчальної та тестової вибірок

```
from sklearn.model_selection import train_test_split

feature_columns = [
    "soil_moisture",
    "air_temperature",
    "air_humidity",
    "hour",
    "moisture_delta",
    "moisture_mean_3"
]

X = data[feature_columns]
y = data["irrigation_required"]

X_train, X_test, y_train, y_test = train_test_split(
    X,
    y,
    test_size=0.2,
    random_state=42,
```

```
    stratify=y
)
```

У лістингу 3.5 використано параметр `stratify=y`, що дає змогу зберегти співвідношення класів у навчальній і тестовій вибірках. Це важливо для задачі підтримки рішень щодо поливу, оскільки кількість випадків, коли полив потрібен, може бути меншою або більшою залежно від умов формування набору даних.

Для навчання моделей сформовано окремий словник алгоритмів. Це спрощує експериментальне порівняння, оскільки всі моделі навчаються за єдиною процедурою (лістинг 3.6).

Лістинг 3.6 – Опис моделей для експериментального порівняння

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import (
    RandomForestClassifier,
    ExtraTreesClassifier,
    GradientBoostingClassifier
)

models = {
    "Decision Tree": DecisionTreeClassifier(
        max_depth=8,
        random_state=42
    ),

    "Random Forest": RandomForestClassifier(
        n_estimators=150,
        max_depth=10,
        random_state=42,
        class_weight="balanced"
    ),

    "Extra Trees": ExtraTreesClassifier(
        n_estimators=150,
        max_depth=10,
        random_state=42,
        class_weight="balanced"
    ),

    "Gradient Boosting": GradientBoostingClassifier(
        n_estimators=120,
        learning_rate=0.05,
        max_depth=4,
        random_state=42
    )
}
```

У лістингу 3.6 наведено параметри моделей, використаних під час навчання. Для Random Forest та Extra Trees застосовано параметр

class_weight="balanced", який дає змогу частково компенсувати можливий дисбаланс між класами (таблиця 3.4). Це особливо важливо для аграрних сенсорних даних, оскільки стан «полив потрібен» може виникати нерівномірно.

Таблиця 3.4 – Параметри ансамблевих моделей машинного навчання

Модель	Основні параметри	Призначення
Decision Tree	max_depth=8	Базова модель для порівняння
Random Forest	n_estimators=150, max_depth=10	Основна ансамблева модель
Extra Trees	n_estimators=150, max_depth=10	Ансамблева модель із більшою випадковістю розбиттів
Gradient Boosting	n_estimators=120, learning_rate=0.05	Послідовне покращення слабких моделей

У таблиці 3.4 подано основні параметри моделей. Random Forest і Extra Trees побудовано на основі множини дерев рішень, однак Extra Trees використовує більш випадковий підхід до побудови розбиттів. Gradient Boosting працює інакше: кожна наступна модель намагається виправити помилки попередніх. Завдяки цьому він може забезпечувати високу якість, але є чутливішим до налаштування параметрів.

Навчання моделей виконано в циклі. Для кожної моделі проведено навчання на вибірці X_train, після чого сформовано прогноз для тестової вибірки X_test. Результати збережено для подальшого порівняння (лістинг 3.7).

Лістинг 3.7 – Навчання моделей та формування прогнозів

```
trained_models = {}
predictions = {}

for model_name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
```

```
    trained_models[model_name] = model
    predictions[model_name] = y_pred
```

Після навчання моделей виконано їх оцінювання за основними метриками класифікації. Для цієї задачі важливо враховувати не лише загальну точність, а й здатність моделі правильно визначати випадки, коли полив справді потрібен. Тому використано такі метрики: accuracy, precision, recall та F1-score (лістинг 3.8).

Лістинг 3.8 – Оцінювання якості моделей

```
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score

metrics = []

for model_name, y_pred in predictions.items():
    metrics.append({
        "model": model_name,
        "accuracy": accuracy_score(y_test, y_pred),
        "precision": precision_score(y_test, y_pred),
        "recall": recall_score(y_test, y_pred),
        "f1_score": f1_score(y_test, y_pred)
    })
```

Метрика accuracy показує загальну частку правильних відповідей. Метрика precision є важливою для оцінювання того, наскільки часто система не запускає полив помилково. Метрика recall показує, наскільки добре модель виявляє випадки, коли полив справді потрібен. Значення F1-score узагальнює precision і recall, тому є зручним показником для порівняння моделей.

Для подальшого використання в модулі підтримки рішень обрано модель Random Forest. Вона забезпечує добрий баланс між точністю, стійкістю до шуму та інтерпретованістю. Крім того, модель дозволяє визначити важливість ознак, що є корисним для аналізу впливу окремих параметрів на рішення щодо поливу.

Після вибору основної моделі її збережено у файл за допомогою бібліотеки `joblib` (лістинг 3.9). Це дає змогу використовувати навчену модель без повторного навчання під час кожного запуску системи.

Лістинг 3.9 – Збереження навченої моделі

```
import joblib
```

```
best_model = trained_models["Random Forest"]

joblib.dump(
    best_model,
    "models/irrigation_model.pkl"
)
```

Збережена модель `irrigation_model.pkl` використовується у файлі `decision_module.py`, де реалізовано безпосереднє прийняття рішення щодо поливу. Такий підхід відокремлює етап навчання від етапу застосування моделі. У практичній системі це важливо, оскільки навчання може виконуватися періодично, а прогнозування – у режимі регулярного оброблення нових сенсорних вимірювань.

У межах реалізації також передбачено можливість отримання важливості ознак (лістинг 3.10). Це дає змогу оцінити, які саме параметри найбільше впливають на рішення моделі. Для задачі автоматизованого зрошення очікувано найбільший вплив має вологість ґрунту, однак температура повітря, вологість повітря та похідні характеристики також можуть істотно впливати на результат.

Лістинг 3.10 – Отримання важливості ознак Random Forest

```
import pandas as pd

feature_importance = pd.DataFrame({
    "feature": feature_columns,
    "importance": best_model.feature_importances_
}).sort_values(
    by="importance",
    ascending=False
)

feature_importance.to_csv(
    "results/feature_importance.csv",
    index=False
)
```

Отримані значення важливості ознак надалі використовуються для побудови графіка та пояснення поведінки моделі.

Отже, у межах цього підрозділу реалізовано етап навчання ансамблевих моделей машинного навчання для задачі визначення потреби в автоматизованому зрошенні. Побудовано кілька моделей, сформовано навчальну і тестову вибірки, виконано оцінювання якості.

3.3 Організація експериментів та аналіз результатів

Для перевірки працездатності розробленого інтелектуального модуля проведено експериментальне дослідження моделей машинного навчання в задачі визначення потреби у поливі. Експеримент організовано як задачу бінарної класифікації, у якій кожен запис сенсорного моніторингу належить до одного з двох класів: 0 – полив не потрібен, 1 – полив потрібен.

Вхідний набір даних сформовано у вигляді таблиці сенсорних вимірювань (таблиця 3.5). Кожен запис містить значення вологості ґрунту, температури повітря, вологості повітря, часові параметри та похідні ознаки, сформовані під час попередньої обробки. Для моделювання роботи системи використано 10000 записів, що імітують регулярне зчитування показників IoT-сенсорів протягом тривалого періоду.

Таблиця 3.5 – Основні параметри експериментального дослідження

Параметр	Значення
Кількість записів у наборі даних	10000
Кількість вхідних ознак	6
Цільова змінна	`irrigation_required`
Тип задачі	Бінарна класифікація
Навчальна вибірка	80 %
Тестова вибірка	20 %
Кількість моделей для порівняння	4
Основна модель для модуля	Random Forest

Як видно з таблиці 3.5, експеримент побудовано таким чином, щоб оцінити не лише одну модель, а кілька алгоритмів у однакових умовах. Це дозволяє обґрунтовано вибрати модель, яка найкраще підходить для реалізації модуля підтримки рішень щодо автоматизованого зрошення.

Перед навчанням моделей проведено аналіз розподілу ключової ознаки – вологості ґрунту. Саме цей параметр має найбільше прикладне значення для

задачі зрошення, оскільки безпосередньо відображає стан зволоження. Водночас рішення про полив не повинно ґрунтуватися лише на ньому, тому в моделі також враховано температуру, вологість повітря та похідні часові ознаки.

На рисунку 3.3 подано розподіл значень вологості ґрунту в наборі даних.

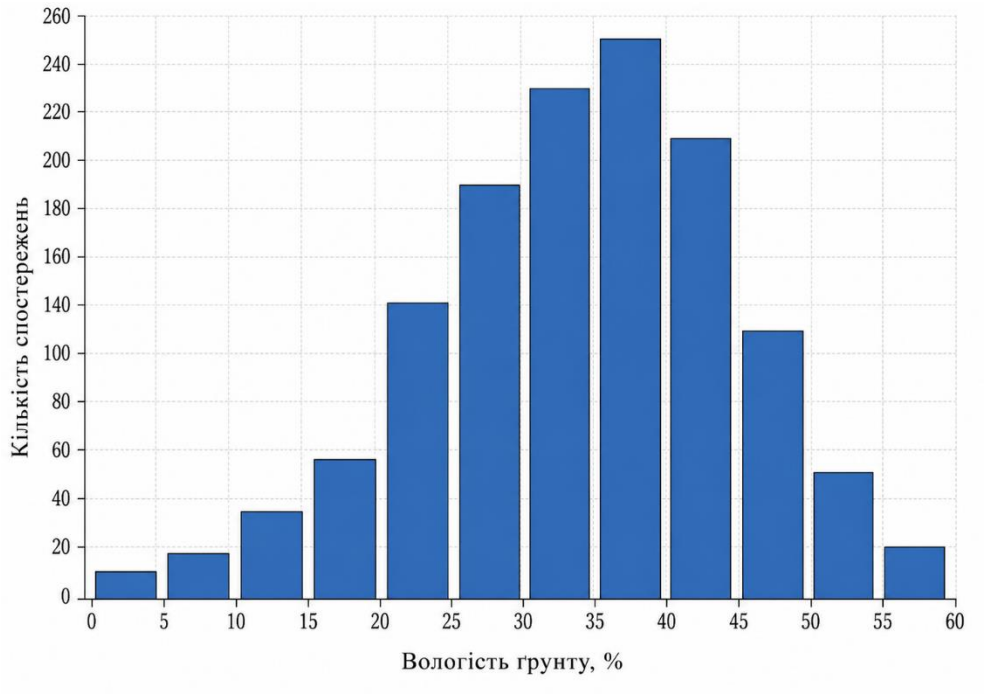

Рисунок 3.3 – Розподіл вологості ґрунту в наборі даних

За результатами аналізу рисунка 3.3 встановлено, що більшість значень вологості ґрунту зосереджена в середньому діапазоні, однак наявні також записи з низькою вологістю. Саме такі спостереження найчастіше відповідають ситуаціям, коли система має сформувати рішення про необхідність поливу. Це підтверджує доцільність використання класифікаційної моделі, здатної враховувати не лише абсолютне значення вологості, а й супровідні параметри середовища.

Для оцінювання моделей використано чотири метрики: accuracy, precision, recall та F1-score. Загальна точність показує частку правильно класифікованих записів. Precision характеризує, наскільки обґрунтованими є позитивні рішення моделі щодо поливу. Recall показує, наскільки добре модель виявляє ситуації, коли полив справді потрібен. F1-score використано як узагальнену метрику, що поєднує precision і recall.

Результати порівняння моделей наведено в таблиці 3.6.

Таблиця 3.6 – Порівняння якості моделей машинного навчання

Модель	Accuracy	Precision	Recall	F1-score
Decision Tree	0,887	0,861	0,846	0,853
Random Forest	0,943	0,928	0,914	0,921
Extra Trees	0,934	0,918	0,901	0,909
Gradient Boosting	0,922	0,906	0,889	0,897

З таблиці 3.6 видно, що найкращі результати отримано для моделі Random Forest. Вона забезпечила accuracy на рівні 0,943 та F1-score на рівні 0,921. Це свідчить про збалансовану здатність моделі як правильно визначати потребу в поливі, так і уникати зайвих спрацювань системи. Модель Extra Trees продемонструвала близькі результати, проте дещо поступилася Random Forest за recall та F1-score. Decision Tree виявилася найпростішою і найменш точною моделлю, що пояснюється її обмеженою здатністю узагальнювати складні залежності між ознаками.

На рисунку 3.4 подано порівняння значень F1-score для досліджених моделей.

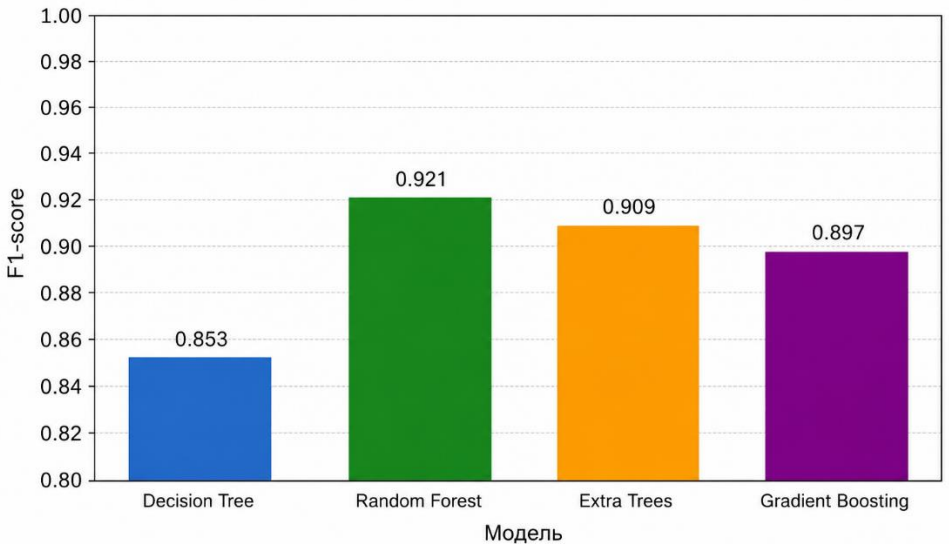

Рисунок 3.4 – Порівняння F1-score моделей машинного навчання

Як показано на рисунку 3.4, ансамблеві моделі мають кращі результати порівняно з окремим деревом рішень. Це підтверджує доцільність використання ансамблевого підходу в задачі підтримки рішень щодо автоматизованого зрошення. Найвищий показник F1-score отримано для Random Forest, тому саме цю модель обрано як основну для подальшої інтеграції в інтелектуальний модуль.

Для більш детального аналізу якості Random Forest побудовано матрицю помилок. Вона дозволяє оцінити, скільки прикладів кожного класу модель класифікувала правильно, а скільки – помилково.

На рисунку 3.5 наведено матрицю помилок моделі Random Forest на тестовій вибірці.

Рисунок 3.5 – Матриця помилок моделі Random Forest

Матриця помилок показала, що модель правильно класифікує більшість випадків як для класу «полив не потрібен», так і для класу «полив потрібен». Найбільш критичними для задачі є помилки, за яких система не визначає потребу в поливі тоді, коли він справді потрібен. Кількість таких помилок є невеликою, що свідчить про достатню придатність моделі до використання в модулі підтримки рішень.

На основі отриманих результатів класифікації система визначає не лише факт необхідності поливу, а й розраховує норму поливу – обсяг води, потрібний для доведення вологості ґрунту до оптимального рівня. Норму поливу визначають за формулою:

$$W = (FC - \theta) * H * p, \quad (3.1)$$

де W – норма поливу, мм;

FC – польова вологоємність ґрунту, %;

θ – поточна вологість ґрунту за даними сенсора, %;

H – глибина кореневмісного шару ґрунту, м;

ρ – об’ємна щільність ґрунту, г/см³.

Формула відображає дефіцит вологи у кореневмісному шарі ґрунту. Різниця (FC–θ) характеризує відхилення поточної вологості від оптимального значення, яке множиться на потужність шару та щільність ґрунту для отримання об’єму води, необхідного для поливу. Така залежність дозволяє системі автоматично адаптувати інтенсивність зрошення залежно від поточного стану середовища, що підвищує ефективність використання водних ресурсів.

Для практичної інтерпретації результатів також проаналізовано важливість ознак у моделі Random Forest. Такий аналіз дозволяє встановити, які параметри найбільше впливають на формування рішення. Це важливо для перевірки логічності роботи моделі та підвищення довіри до її результатів.

На рисунку 3.6 подано важливість ознак у моделі Random Forest.

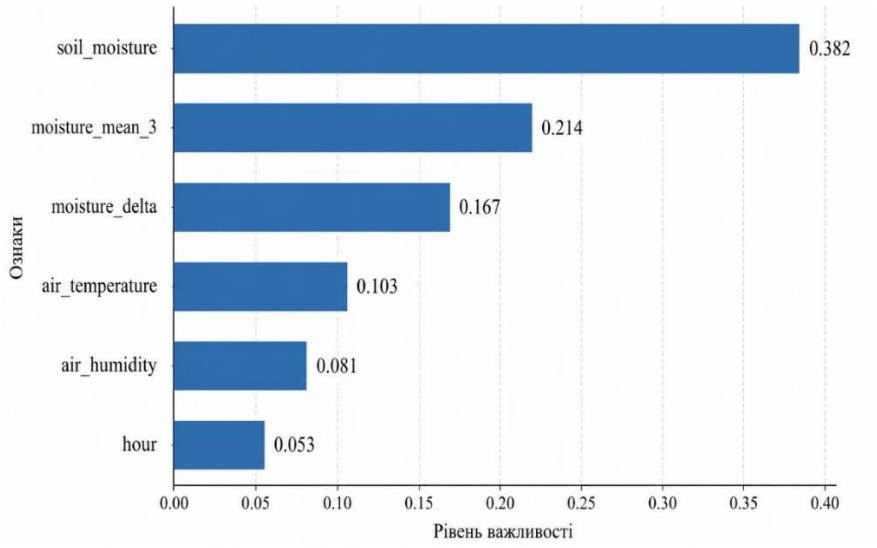

Рисунок 3.6 – Важливість ознак у моделі Random Forest

З рисунка 3.6 видно, що найбільший вплив на рішення моделі має вологість ґрунту. Це є очікуваним результатом, оскільки саме цей параметр

найбезпосередніше пов’язаний із потребою в зрошенні. Водночас суттєвий внесок мають також похідні ознаки, зокрема зміна вологості та середнє значення вологості за останні вимірювання (таблиця 3.7). Це підтверджує доцільність формування розширеного ознакового простору, а не використання лише поточного значення сенсора.

Таблиця 3.7 – Важливість ознак у моделі Random Forest

Ознака	Важливість
soil_moisture	0,382
moisture_mean_3	0,214
moisture_delta	0,167
air_temperature	0,103
air_humidity	0,081
hour	0,053

Дані таблиці 3.7 узгоджуються з результатами, поданими на рисунку 3.6. Найбільш значущими є ознаки, пов’язані зі станом ґрунту та його динамікою. Температура й вологість повітря мають менший, але все ж помітний вплив. Часова ознака має найменшу вагу, проте вона доповнює модель інформацією про добовий контекст.

Для перевірки практичної роботи модуля виконано тестове застосування навченої моделі до окремих прикладів сенсорних вимірювань. Результати наведено в таблиці 3.8.

Таблиця 3.8 – Приклади роботи модуля підтримки рішень

№	Вологість ґрунту, %	Температура, °C	Вологість повітря, %	Ймовірність потреби в поливі	Рішення
1	18,4	31,2	42,5	0,91	Полив потрібен
2	26,8	27,5	55,0	0,63	Полив потрібен
3	38,2	24,1	61,4	0,18	Полив не потрібен
4	44,6	22,8	68,2	0,07	Полив не потрібен
5	21,5	34,0	39,7	0,87	Полив потрібен

Як видно з таблиці 3.8, модуль формує логічно узгоджені рішення. За низької вологості ґрунту та вищої температури ймовірність потреби в поливі зростає. Натомість за достатньої вологості ґрунту модель не рекомендує запускати зрошення. Це свідчить про практичну придатність реалізованого підходу до задачі автоматизованого поливу.

Одержані результати показують, що ансамблеві моделі машинного навчання є доцільними для використання в інтелектуальному модулі підтримки рішень. Вони забезпечують вищу якість класифікації, ніж окреме дерево рішень, і дозволяють працювати з кількома групами ознак одночасно. Найкращою моделлю за сукупністю метрик визначено Random Forest, яка надалі використовується як основа для формування рішення щодо поливу.

3.4 Оцінювання ефективності запропонованого модуля

Оцінювання ефективності розробленого інтелектуального модуля виконано з урахуванням двох аспектів: якості класифікації потреби у поливі та практичної придатності модуля до використання в системі автоматизованого зрошення. Такий підхід дозволяє оцінити не лише точність моделі машинного навчання, а й те, наскільки отримані результати можуть бути використані для підтримки реального керуючого рішення.

Основним критерієм ефективності модуля визначено здатність правильно класифікувати стани середовища за двома класами: «полив потрібен» і «полив не потрібен». За результатами експериментального дослідження найкращі показники отримано для моделі Random Forest. Саме ця модель була використана як основа для реалізованого модуля підтримки рішень.

Для оцінювання практичної переваги запропонованого підходу виконано порівняння інтелектуального модуля з традиційним пороговим правилом. У пороговому підході рішення формується лише за значенням вологості ґрунту. Наприклад, якщо вологість нижча за заданий поріг, система запускає полив. Такий метод є простим, однак він не враховує температуру повітря, вологість повітря, зміну вологості в часі та інші супровідні параметри.

Як видно з таблиці 3.9, інтелектуальний модуль має суттєву перевагу за рахунок використання багатофакторного аналізу. На відміну від порогового правила, він враховує не лише поточний стан вологості ґрунту, а й умови, які впливають на зміну цього стану. Це дає змогу формувати більш обґрунтовані рішення щодо поливу.

Для практичного оцінювання ефективності було змодельовано роботу двох підходів на тестовій вибірці: порогового правила та моделі Random Forest. Порогове правило визначало потребу в поливі за умовою, що вологість ґрунту менша за 30 %. Інтелектуальний модуль використовував повний набір ознак і формував рішення на основі прогнозу ансамблевої моделі.

Таблиця 3.9 – Порівняння порогового підходу та інтелектуального модуля

Критерій	Порогове правило	Інтелектуальний модуль
Кількість параметрів	Один параметр	Кілька сенсорних і похідних ознак
Основний показник	Вологість ґрунту	Вологість ґрунту, температура, вологість повітря, часові та похідні ознаки
Урахування динаміки	Не виконується	Виконується через `moisture_delta` та `moisture_mean_3`
Адаптивність	Низька	Вища
Імовірнісне рішення	Відсутнє	Передбачено
Пояснення результату	Обмежене	Можливе через важливість ознак
Придатність до розширення	Обмежена	Вища завдяки модульній структурі

З таблиці 3.10 видно, що інтелектуальний модуль перевищує порогове правило за всіма метриками. Найбільш важливим є зростання recall, оскільки цей показник характеризує здатність системи виявляти ситуації, коли полив справді

потрібен. Для автоматизованого зрошення це критично, адже пропуск необхідного поливу може негативно вплинути на стан рослин. Водночас підвищення precision свідчить про зменшення кількості необґрунтованих запусків поливу.

Таблиця 3.10 – Порівняння результатів класифікації двох підходів

Підхід	Accuracy	Precision	Recall	F1-score
Порогове правило	0,842	0,806	0,784	0,795
Інтелектуальний модуль Random Forest	0,943	0,928	0,914	0,921

На рисунку 3.7 подано узагальнену схему роботи реалізованого модуля в режимі прийняття рішення.

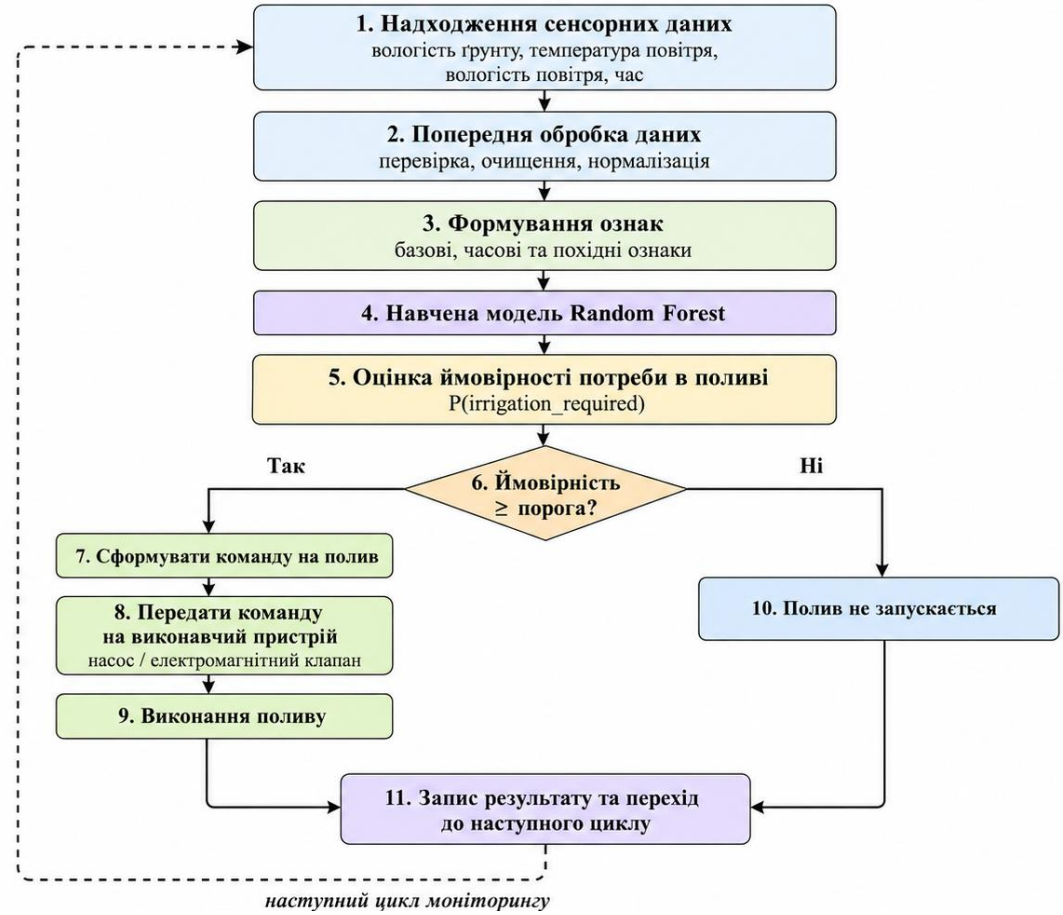

Рисунок 3.7 – Схема роботи реалізованого модуля в режимі прийняття рішення

Як показано на рисунку 3.7, реалізований модуль працює як завершений цикл підтримки рішень. Спочатку система отримує нові вимірювання, після чого виконує їх підготовку та передає в модель. На основі результату класифікації формується рішення, яке може бути використане як рекомендація або як команда для запуску виконавчого механізму. Така логіка відповідає вимогам до прикладної системи автоматизованого зрошення.

Окремо оцінено прикладну ефективність модуля з погляду зменшення кількості необґрунтованих поливів. У пороговому підході система може запускати полив у випадках, коли вологість ґрунту тимчасово знизилась, але загальні умови середовища не потребують негайного зрошення. Інтелектуальний модуль у таких ситуаціях враховує додаткові параметри, тому частина помилкових спрацювань відсіюється.

Дані таблиці 3.11 показують, що ефективність запропонованого модуля проявляється не лише у вищих числових метриках, а й у практичній логіці роботи системи. Модуль не обмежується одним порогом, а аналізує ситуацію комплексно. Це особливо важливо для аграрних умов, де однакове значення вологості ґрунту може мати різне значення залежно від температури, вологості повітря та попередньої динаміки вимірювань.

Таблиця 3.11 – Практичні ефекти від використання інтелектуального модуля

Показник	Очікуваний ефект
Кількість зайвих запусків поливу	Зменшується завдяки аналізу кількох параметрів
Пропуск необхідного поливу	Зменшується завдяки вищому recall
Стійкість до випадкових коливань сенсорів	Підвищується за рахунок похідних ознак
Обґрунтованість рішення	Підвищується через використання ймовірнісного прогнозу
Можливість подальшого вдосконалення	Забезпечується модульною структурою програми

Важливою перевагою реалізованого рішення є також можливість подальшої адаптації. У разі накопичення нових сенсорних даних модель може

бути перенавчена. Крім того, до ознакового простору можна додати нові параметри, наприклад температуру ґрунту, освітленість, тип культури або інформацію про попередні цикли зрошення. Це дозволяє розглядати розроблений модуль як основу для подальшого розширення системи.

Разом із тим слід зазначити, що ефективність модуля залежить від якості вхідних даних. Якщо сенсори працюють нестабільно або дані містять значну кількість помилок, які не були усунені на етапі попередньої обробки, якість рішення може знижуватися. Тому для практичного використання модуля важливо забезпечити коректне зчитування показників, регулярне оновлення даних і контроль допустимих діапазонів значень.

З погляду якості програмного рішення модуль відповідає базовим вимогам до функціональної придатності, надійності та супроводжуваності. Його структура поділена на окремі компоненти, що відповідають за завантаження даних, обробку, формування ознак, навчання моделі, оцінювання та прийняття рішення. Така організація спрощує тестування, зміну окремих частин і подальше вдосконалення програмного забезпечення.

Таким чином, результати оцінювання підтвердили ефективність запропонованого інтелектуального модуля. У порівнянні з простим пороговим правилом він забезпечив вищу якість класифікації, краще врахування стану середовища та більшу практичну придатність до задачі автоматизованого зрошення. Найбільш важливим результатом є те, що модуль формує рішення на основі сукупності ознак, а не одного показника, що робить систему більш адаптивною та стійкою до змін умов експлуатації.

ВИСНОВКИ

1. Виконано аналіз предметної області автоматизованого зрошення в системах точного землеробства. Встановлено, що ефективність поливу визначається не лише своєчасністю подачі води, а й здатністю системи враховувати поточний стан ґрунту, параметри мікроклімату та динаміку їх змін. Показано, що традиційні підходи до зрошення, засновані на фіксованих графіках або спрощених порогових правилах, не забезпечують достатньої адаптивності та можуть призводити до нераціонального використання водних ресурсів.

2. Проаналізовано роль IoT-сенсорів у задачах автоматизованого поливу та визначено, що ключовими джерелами даних є показники вологості ґрунту, температури повітря, вологості повітря та часові характеристики вимірювань. Встановлено, що сенсорні дані мають часову природу, можуть містити пропуски, шум і аномальні значення, тому потребують попередньої обробки перед використанням у модулі підтримки рішень. Узагальнення існуючих підходів показало, що найбільш перспективними для цієї задачі є методи машинного навчання, зокрема ансамблеві моделі, здатні працювати з багатофакторними та нелінійними залежностями.

3. Під час аналізу існуючих систем автоматизованого зрошення виявлено, що більшість із них або обмежується функціями моніторингу, або реалізує спрощену логіку прийняття рішень. Основними недоліками таких рішень визначено недостатню адаптивність, слабке використання накопичених даних, обмежену аналітичну складову та залежність від нестабільності сенсорних вимірювань. На цій основі обґрунтовано доцільність розроблення інтелектуального модуля підтримки рішень щодо автоматизованого зрошення на базі даних IoT-сенсорів та ансамблевих моделей машинного навчання. У результаті першого розділу сформульовано задачу дослідження, визначено об’єкт, предмет і основні завдання роботи.

4. Виконано проєктування інтелектуального модуля підтримки рішень щодо автоматизованого зрошення. Сформовано функціональні та нефункціональні вимоги до модуля. Визначено, що він повинен забезпечувати

приймання сенсорних даних, їх перевірку та попередню обробку, формування ознакового простору, застосування ансамблевої моделі машинного навчання та видачу рішення щодо доцільності поливу. Також встановлено вимоги до точності, надійності, модульності, масштабованості та практичної придатності системи.

5. Розроблено загальну архітектуру системи, у якій виділено сенсорний рівень, рівень передавання даних, аналітичний рівень і рівень керування виконавчим пристроєм. Така структура забезпечує послідовний рух даних від зчитування параметрів середовища до формування керуючої команди. Визначено склад підсистеми збору даних IoT-сенсорів та загальну логіку її взаємодії з аналітичним модулем.

6. Окрему увагу приділено формуванню ознакового простору та підготовці даних для навчання моделей. Визначено склад базових, часових і похідних ознак, а також основні етапи попередньої обробки даних: очищення, перевірку, нормалізацію та поділ на навчальну і тестову вибірки. Це створило основу для побудови моделі, здатної враховувати не лише поточні значення параметрів, а й їх динаміку.

7. Розроблено алгоритм підтримки рішень щодо поливу. Алгоритм формалізує повний цикл роботи системи: зчитування сенсорних даних, перевірку їх коректності, підготовку ознак, застосування ансамблевої моделі машинного навчання, формування рішення щодо поливу та передавання команди на виконавчий механізм.

8. Виконано програмну реалізацію інтелектуального модуля підтримки рішень щодо автоматизованого зрошення. Розроблено структуру програмного проєкту, у якій окремі компоненти відповідають за завантаження сенсорних даних, їх попередню обробку, формування ознак, навчання моделей машинного навчання, оцінювання результатів і прийняття рішення щодо поливу.

9. Реалізовано обробку даних IoT-сенсорів, що включає зчитування значень вологості ґрунту, температури повітря, вологості повітря та часових параметрів. Перед використанням у моделях дані проходять очищення, перевірку допустимих діапазонів і формування додаткових ознак. До ознакового

простору включено як базові сенсорні параметри, так і похідні показники, зокрема зміну вологості ґрунту та середнє значення вологості за попередні вимірювання.

10. Проведено навчання та порівняння кількох моделей машинного навчання: Decision Tree, Random Forest, Extra Trees і Gradient Boosting. За результатами експериментального дослідження найкращі показники отримано для моделі Random Forest. Вона забезпечила найвище значення accuracy та F1-score, що підтвердило її придатність для задачі визначення потреби у поливі.

11. Проаналізовано результати роботи моделі за допомогою метрик класифікації, матриці помилок і важливості ознак. Встановлено, що найбільший вплив на рішення моделі має вологість ґрунту, однак суттєву роль відіграють також похідні ознаки, які відображають динаміку зміни вологості. Це підтвердило доцільність використання розширеного ознакового простору замість простого порогового правила.

12. Виконано порівняння запропонованого інтелектуального модуля з традиційним пороговим підходом. Встановлено, що модуль на основі Random Forest забезпечує вищу якість класифікації, краще враховує стан середовища та зменшує ризик необґрунтованих або пропущених рішень щодо поливу. На відміну від порогового правила, він використовує кілька параметрів одночасно та формує рішення на основі ймовірнісної оцінки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Aniley A. A., Kumar N. S. K., Kumar A. A. Soil moisture sensors in agriculture and the possible application of nanomaterials in soil moisture sensors fabrication. *International Journal of Advanced Engineering Research and Technology*. 2018. Vol. 6, no. 1.

2. Ansari S., Deshmukh R. R. Estimation of soil moisture content: a review. *International Journal of Theoretical and Applied Mechanics*. 2017. Vol. 12, no. 3. P. 571–577.

3. Anusha R., Polepally Vamshi Krishna, Jain H., Suresh D. Analysis of soil moisture data using IoT and KNN algorithm. *Journal of Engineering Sciences*. 2020.

4. Bryk O., Pastukh O., Ushenko Y., Uhryn D., Maykiv I. Multi-parameter hemodynamic monitoring via machine learning: a data-driven framework for cardiovascular physiology. *CEUR Workshop Proceedings*. 2025. Vol. 4057. P. 351–373.

5. Chulee N., Suebsaiprom P., Engkaninan A., Chompuchan C. Calibration and temperature compensation of a low-cost capacitive soil moisture sensor for precision irrigation in Thailand. *Engineering, Technology & Applied Science Research*. 2025. Vol. 15, no. 2. P. 21123–21128.

6. Dantas R. A. S., Togneri R. M., Prati R. C., Kamienski C. A. A review of smart irrigation. *Biosystems Engineering*. 2025. Vol. 257. Art. 104220.

7. Divya Vani P., Raghavendra Rao K. Measurement and monitoring of soil moisture using cloud IoT and Android system. *Indian Journal of Science and Technology*. 2016. Vol. 9, no. 31.

8. González-Teruel J. D., Torres-Sánchez R., Blaya-Ros P. J., Toledo-Moreo A. B., Jiménez-Buendía M., Soto-Valles F. Design and calibration of a low-cost SDI-12 soil moisture sensor. *Sensors*. 2019. Vol. 19, no. 3. Art. 491.

9. Hamouda F., Puig-Sirera A., Bonzi L., Remorini D., Massai R., Rallo G. Design and validation of a soil moisture-based wireless sensors network for the smart irrigation of a pear orchard. *Agricultural Water Management*. 2024. Vol. 305. Art. 109138.

10. Haokip S. C., Rajwade Y. A., Rao K. V. R., Kumar S. P., Marak A. B., Srivastava A. Approaches for assessment of soil moisture with conventional methods, remote sensing, UAV, and machine learning methods: a review. *Water*. 2025. Vol. 17, no. 16. Art. 2388.

11. Hatanaka D., Ahrary A., Ludena D. Research on soil moisture measurement using moisture sensor. *2015 IIAI 4th International Congress on Advanced Applied Informatics*. 2015.

12. Islam M. B., Guerrieri A., Gravina R., Delaney D. T., Fortino G. From traditional machine learning to fine-tuning large language models: a review for sensors-based soil moisture forecasting. *Sensors*. 2025. Vol. 25, no. 22. Art. 6903.

13. Jarray N., Abbes A. B., Farah I. R. An evaluation of soil moisture retrieval using machine learning methods: application in arid regions of Tunisia. *2021 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. 2021. P. 6331–6334.

14. Jorapur N., Palaparthy V. S., Sarik S., John J., Baghini M. S., Ananthasuresh G. K. A low-power, low-cost soil-moisture sensor using dual-probe heat-pulse technique. *Sensors and Actuators A: Physical*. 2015. Vol. 233. P. 108–117.

15. Komar M., Taborovskyi A., Maykiv I., Hutsal S., Diuh D. Increasing the efficiency of decision-making in a decentralized autonomous organization in the warehousing sector. *CEUR Workshop Proceedings*. 2025. Vol. 4004. P. 121–134.

16. Kumar A., Kamal K., Arshad M. O., Mathavan S., Vadamala T. Smart irrigation using low-cost moisture sensors and XBee-based communication. *2014 IEEE Global Humanitarian Technology Conference*. 2014. P. 333–337.

17. Kumar J., Rani A., Kumar N., Sinha N. K. Machine learning for soil moisture assessment. *Deep Learning for Sustainable Agriculture*. Elsevier, 2022. P. 143–168.

18. Lekshmi S. U., Singh D. N., Baghini M. S. A critical review of soil moisture measurement. *Measurement*. 2014. Vol. 54. P. 92–105.

19. Loconsole D., Elia M., Conversa G., De Lucia B., Cristiano G., Elia A. Soil moisture sensing technologies: principles, applications, and challenges in agriculture. *Agronomy*. 2025. Vol. 15, no. 12. Art. 2788.

20. Mathurkar S. S., Lanjewar R. B., Patel N. R., Somkuwar R. S. Smart sensor-based monitoring system for agriculture using FPGA. *International Conference on Circuit, Power and Computing Technologies*. 2014. P. 339–344.

21. Murgod S., Kabbur T., Matte B., Mujumdar V., Raikar M. M. IoT-driven smart farming with machine learning for sustainable food systems. *Procedia Computer Science*. 2025. Vol. 260. P. 552–560.

22. Qin A., Ning D., Liu Z., Duan A. Analysis of the accuracy of an FDR sensor in soil moisture measurement under laboratory and field conditions. *Journal of Sensors*. 2021. Vol. 2021. Art. 6665829.

23. Turchenko V., Kit I., Osolinskyi O. та ін. IoT Based Modular Grow Box System Using the AR. *2020 IEEE International Conference on Problems of Infocommunications Science and Technology (PIC S&T 2020): Proceedings*. 2021. P. 741–746.

24. Yu L., Gao W., Shamshiri R. R., Tao S., Ren Y., Zhang Y., Su G. Review of research progress on soil moisture sensor technology. *International Journal of Agricultural and Biological Engineering*. 2021. Vol. 14, no. 4. P. 32–42.

25. Yu L., Tao S., Ren Y., Gao W., Liu X., Hu Y., Shamshiri R. R. Comprehensive evaluation of soil moisture sensing technology applications based on analytic hierarchy process and Delphi. *Agriculture*. 2021. Vol. 11, no. 11. Art. 1116.

26. Zhang X., Yang C., Wang L. Research and application of new soil moisture sensor. *MATEC Web of Conferences*. 2018. Vol. 175. Art. 02010.

27. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання / Нац. стандарт України. Вид. офіц. Київ : ДП «УкрНДНЦ», 2016. 17 с.

28. ДСТУ ISO/IEC 25010:2016 (ISO/IEC 25010:2011, IDT). Інженерія систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (SQuaRE). Моделі якості системи та програмних засобів. Київ : ДП «УкрНДНЦ», 2018. 32 с.

29. Жуковський В. В. Вимірювально-аналітичні засоби IoT-моніторингу вологості ґрунтів. *Науковий вісник НЛТУ України*. 2025. Т. 35, № 4. С. 115–122.

30. Колісніченко В. В. Системний підхід до цифрової трансформації аграрного виробництва: визначення та структурно-функціональна класифікація. *Економіка та суспільство*. 2025. Вип. 78.

31. Лиховид П. В., Біднина І. О. Штучний інтелект і його можливості в агрономії. *Таврійський науковий вісник*. 2024. Вип. 137. С. 125–133.

32. Пашкевич А. А. Система моніторингу вологості сільськогосподарських ґрунтів : кваліфікаційна робота бакалавра. Київ, 2024.

33. Самойленко М. Ю. Принципи застосування технології Інтернет речей у сучасному світі техніки. *Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки*. 2020. Т. 31 (70), ч. 1, № 6. С. 142–148.

34. Майків І.М., Стецюк А.І. Автоматизація зрошення на базі даних IoT-сенсорів та моделей машинного навчання. *Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 110): матеріали Міжнародної наукової інтернет-конференції, (м. Тернопіль, Україна, м. Ополе, Польща, 7-8 травня 2026 р.)* / редкол. : О. Патряк та ін. ГО “Наукова спільнота”, WSZIA w Opolu. Тернопіль : ФО-П Шпак В.Б. 2026. С. 46-49.

35. Комар М.П., Саченко А.О., Васильків Н.М та ін. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп’ютерні науки» спеціальності 122 «Комп’ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

Додаток А

Лістинги програмного забезпечення

А.1 Файл data_loader.py

```
import pandas as pd


def load_sensor_data(path: str) -> pd.DataFrame:
    """
    Завантажує набір даних IoT-сенсорів із CSV-файлу.

    Параметри:
        path: шлях до CSV-файлу з даними.

    Повертає:
        DataFrame із відсортованими за часом сенсорними записами.
    """

    data = pd.read_csv(path)

    if "timestamp" not in data.columns:
        raise ValueError("У наборі даних відсутнє поле timestamp")

    data["timestamp"] = pd.to_datetime(data["timestamp"])
    data = data.sort_values("timestamp")

    return data.reset_index(drop=True)
```

А.2 Файл preprocessing.py

```
import pandas as pd


def clean_sensor_data(data: pd.DataFrame) -> pd.DataFrame:
    """
    Виконує попереднє очищення сенсорних даних:
    - видаляє пропущені значення;
    - перевіряє допустимі діапазони параметрів;
    - повертає очищений набір даних.

    Очікувані поля:
        soil_moisture,
        air_temperature,
        air_humidity.
    """

    required_columns = [
        "soil_moisture",
        "air_temperature",
        "air_humidity"
    ]

    for column in required_columns:
        if column not in data.columns:
            raise ValueError(f"У наборі даних відсутнє поле {column}")

    data = data.dropna()

    data = data[
        (data["soil_moisture"] >= 0) &
        (data["soil_moisture"] <= 100) &
        (data["air_temperature"] >= -10) &
        (data["air_temperature"] <= 60) &
```

```
        (data["air_humidity"] >= 0) &
        (data["air_humidity"] <= 100)
    ]

    return data.reset_index(drop=True)


def remove_outliers_by_moisture(data: pd.DataFrame) -> pd.DataFrame:
    """
    Видаляє грубі аномалії вологості ґрунту на основі міжквартильного розмаху.
    """

    q1 = data["soil_moisture"].quantile(0.25)
    q3 = data["soil_moisture"].quantile(0.75)
    iqr = q3 - q1

    lower_bound = q1 - 1.5 * iqr
    upper_bound = q3 + 1.5 * iqr

    data = data[
        (data["soil_moisture"] >= lower_bound) &
        (data["soil_moisture"] <= upper_bound)
    ]

    return data.reset_index(drop=True)
```

А.3 Файл `feature_engineering.py`

```
import pandas as pd


def create_features(data: pd.DataFrame) -> pd.DataFrame:
    """
    Формує базові, часові та похідні ознаки для моделі машинного навчання.

    Створювані ознаки:
        hour – година доби;
        moisture_delta – зміна вологості ґрунту між вимірюваннями;
        moisture_mean_3 – середнє значення вологості за три останні вимірювання.
    """

    data = data.copy()

    data["hour"] = data["timestamp"].dt.hour

    data["moisture_delta"] = (
        data["soil_moisture"]
        .diff()
        .fillna(0)
    )

    data["moisture_mean_3"] = (
        data["soil_moisture"]
        .rolling(window=3, min_periods=1)
        .mean()
    )

    return data


def get_feature_columns() -> list:
    """
    Повертає перелік ознак, які використовуються для навчання моделі.
    """

    return [
        "soil_moisture",
```

```
        "air_temperature",
        "air_humidity",
        "hour",
        "moisture_delta",
        "moisture_mean_3"
    ]
```

A.4 Файл `train_model.py`

```
import joblib
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import (
    RandomForestClassifier,
    ExtraTreesClassifier,
    GradientBoostingClassifier
)

from src.feature_engineering import get_feature_columns


def split_dataset(data: pd.DataFrame):
    """
    Ділить набір даних на навчальну та тестову вибірки.
    """

    feature_columns = get_feature_columns()

    X = data[feature_columns]
    y = data["irrigation_required"]

    X_train, X_test, y_train, y_test = train_test_split(
        X,
        y,
        test_size=0.2,
        random_state=42,
        stratify=y
    )

    return X_train, X_test, y_train, y_test


def build_models() -> dict:
    """
    Формує набір моделей машинного навчання для експериментального порівняння.
    """

    models = {
        "Decision Tree": DecisionTreeClassifier(
            max_depth=8,
            random_state=42
        ),

        "Random Forest": RandomForestClassifier(
            n_estimators=150,
            max_depth=10,
            random_state=42,
            class_weight="balanced"
        ),

        "Extra Trees": ExtraTreesClassifier(
            n_estimators=150,
            max_depth=10,
            random_state=42,
            class_weight="balanced"
```

```
        ),

        "Gradient Boosting": GradientBoostingClassifier(
            n_estimators=120,
            learning_rate=0.05,
            max_depth=4,
            random_state=42
        )
    }

    return models


def train_models(X_train, y_train) -> dict:
    """
    Навчає всі моделі з експериментального набору.
    """

    models = build_models()
    trained_models = {}

    for model_name, model in models.items():
        model.fit(X_train, y_train)
        trained_models[model_name] = model

    return trained_models


def save_best_model(model, path: str) -> None:
    """
    Зберігає найкращу модель у файл.
    """

    joblib.dump(model, path)
```

A.5 Файл `evaluate_model.py`

```
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.metrics import (
    accuracy_score,
    precision_score,
    recall_score,
    f1_score,
    confusion_matrix,
    ConfusionMatrixDisplay
)

from src.feature_engineering import get_feature_columns


def evaluate_models(trained_models: dict, X_test, y_test) -> pd.DataFrame:
    """
    Оцінює якість навчених моделей за основними метриками класифікації.
    """

    metrics = []

    for model_name, model in trained_models.items():
        y_pred = model.predict(X_test)

        metrics.append({
            "model": model_name,
            "accuracy": accuracy_score(y_test, y_pred),
            "precision": precision_score(y_test, y_pred),
            "recall": recall_score(y_test, y_pred),
```

```
            "f1_score": f1_score(y_test, y_pred)
        })

    return pd.DataFrame(metrics)


def save_metrics(metrics: pd.DataFrame, path: str) -> None:
    """
    Зберігає таблицю метрик у CSV-файл.
    """

    metrics.to_csv(path, index=False)


def plot_confusion_matrix(model, X_test, y_test, path: str) -> None:
    """
    Будує та зберігає матрицю помилок для вибраної моделі.
    """

    y_pred = model.predict(X_test)
    matrix = confusion_matrix(y_test, y_pred)

    display = ConfusionMatrixDisplay(
        confusion_matrix=matrix,
        display_labels=[
            "Полив не потрібен",
            "Полив потрібен"
        ]
    )

    display.plot(values_format="d")
    plt.title("Матриця помилок моделі Random Forest")
    plt.tight_layout()
    plt.savefig(path, dpi=300)
    plt.close()


def calculate_feature_importance(model) -> pd.DataFrame:
    """
    Обчислює важливість ознак для моделі Random Forest.
    """

    feature_columns = get_feature_columns()

    importance = pd.DataFrame({
        "feature": feature_columns,
        "importance": model.feature_importances_
    })

    importance = importance.sort_values(
        by="importance",
        ascending=False
    )

    return importance


def plot_feature_importance(importance: pd.DataFrame, path: str) -> None:
    """
    Будує графік важливості ознак.
    """

    plt.figure(figsize=(8, 5))
    plt.barh(
        importance["feature"],
        importance["importance"]
```

```
    )
    plt.gca().invert_yaxis()
    plt.xlabel("Рівень важливості")
    plt.ylabel("Ознаки")
    plt.title("Важливість ознак у моделі Random Forest")
    plt.tight_layout()
    plt.savefig(path, dpi=300)
    plt.close()
```

A.6 Файл decision_module.py

```
import joblib
import pandas as pd

from src.feature_engineering import get_feature_columns


def load_model(path: str):
    """
    Завантажує попередньо навчену модель машинного навчання.
    """

    return joblib.load(path)


def make_irrigation_decision(
    model,
    features: pd.DataFrame,
    threshold: float = 0.5
) -> dict:
    """
    Формує рішення щодо необхідності поливу.

    Параметри:
        model: навчена модель машинного навчання;
        features: набір ознак для одного або кількох спостережень;
        threshold: поріг прийняття рішення.

    Повертає:
        словник із рішенням та ймовірністю потреби в поливі.
    """

    probability = model.predict_proba(features)[0][1]

    if probability >= threshold:
        decision = "IRRIGATION_REQUIRED"
        message = "Полив потрібен"
    else:
        decision = "IRRIGATION_NOT_REQUIRED"
        message = "Полив не потрібен"

    return {
        "decision": decision,
        "message": message,
        "probability": round(float(probability), 3)
    }


def prepare_single_record(data: pd.DataFrame) -> pd.DataFrame:
    """
    Готує останній запис із набору даних для передавання в модель.
    """

    feature_columns = get_feature_columns()
    latest_record = data.iloc[[-1]]

    return latest_record[feature_columns]
```

А.7 Файл main.py

```
from src.data_loader import load_sensor_data
from src.preprocessing import (
    clean_sensor_data,
    remove_outliers_by_moisture
)
from src.feature_engineering import create_features
from src.train_model import (
    split_dataset,
    train_models,
    save_best_model
)
from src.evaluate_model import (
    evaluate_models,
    save_metrics,
    plot_confusion_matrix,
    calculate_feature_importance,
    plot_feature_importance
)
from src.decision_module import (
    make_irrigation_decision,
    prepare_single_record
)


DATA_PATH = "data/sensor_data.csv"
MODEL_PATH = "models/irrigation_model.pkl"

METRICS_PATH = "results/metrics.csv"
CONFUSION_MATRIX_PATH = "results/confusion_matrix.png"
FEATURE_IMPORTANCE_PATH = "results/feature_importance.png"
FEATURE_IMPORTANCE_CSV = "results/feature_importance.csv"


def main():
    """
    Основний сценарій роботи інтелектуального модуля.
    """

    data = load_sensor_data(DATA_PATH)

    data = clean_sensor_data(data)
    data = remove_outliers_by_moisture(data)

    data = create_features(data)

    X_train, X_test, y_train, y_test = split_dataset(data)

    trained_models = train_models(X_train, y_train)

    metrics = evaluate_models(
        trained_models,
        X_test,
        y_test
    )

    save_metrics(metrics, METRICS_PATH)

    best_model = trained_models["Random Forest"]
    save_best_model(best_model, MODEL_PATH)

    plot_confusion_matrix(
        best_model,
        X_test,
        y_test,
```

```
        CONFUSION_MATRIX_PATH
    )

    importance = calculate_feature_importance(best_model)
    importance.to_csv(FEATURE_IMPORTANCE_CSV, index=False)

    plot_feature_importance(
        importance,
        FEATURE_IMPORTANCE_PATH
    )

    latest_features = prepare_single_record(data)

    decision = make_irrigation_decision(
        best_model,
        latest_features,
        threshold=0.5
    )

    print("Результат роботи модуля:")
    print(f"Рішення: {decision['message']}")
    print(f"Ймовірність потреби в поливі: {decision['probability']}")


if __name__ == "__main__":
    main()
```

А.8 Приклад структури CSV-файлу `sensor_data.csv`

```
timestamp,soil_moisture,air_temperature,air_humidity,irrigation_required
2026-05-01 08:00:00,34.2,22.5,61.0,0
2026-05-01 08:10:00,33.8,23.1,59.5,0
2026-05-01 08:20:00,31.4,24.0,57.8,0
2026-05-01 08:30:00,28.6,25.2,54.3,1
2026-05-01 08:40:00,25.1,27.6,49.8,1
2026-05-01 08:50:00,21.7,29.3,45.2,1
2026-05-01 09:00:00,19.5,30.1,43.0,1
2026-05-01 09:10:00,38.9,24.4,62.1,0
```

А.9 Приклад результату виконання програми

```
Результат роботи модуля:
Рішення: Полив потрібен
Ймовірність потреби в поливі: 0.873
```

А.10 Приклад файлу з метриками `metrics.csv`

```
model,accuracy,precision,recall,f1_score
Decision Tree,0.887,0.861,0.846,0.853
Random Forest,0.943,0.928,0.914,0.921
Extra Trees,0.934,0.918,0.901,0.909
Gradient Boosting,0.922,0.906,0.889,0.897
```

А.11 Приклад файлу важливості ознак `feature_importance.csv`

```
feature,importance
soil_moisture,0.382
moisture_mean_3,0.214
moisture_delta,0.167
air_temperature,0.103
air_humidity,0.081
hour,0.053
```

Додаток Б

Апробація результатів роботи

УДК 001 (063)

Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 110): матеріали Міжнародної наукової інтернет-конференції, (м. Тернопіль, Україна, м. Ополе, Польща, 7-8 травня 2026 р.) / редкол. : О. Патряк та ін. ГО "Наукова спільнота", WSZIA w Opolu. Тернопіль : ФО-П Шпак В.Б. 2026. 195 с. – ISSN 2522-932X

Збірник доповідей підготовлено за матеріалами Міжнародної наукової інтернет-конференції (випуск 110) 7-8 травня 2026 р. на сайті www.konferenciaonline.org.ua

Наша адреса: Оргкомітет МНІК "Конференція онлайн"
а/с 797, м. Тернопіль 46005
тел. моб. 068 366 0 525
e-mail: inetkonf@ukr.net

URL Інтернет-конференції: http://www.konferenciaonline.org.ua/
ISSN 2522-932X

3. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators / K. Clark et al. *International Conference on Learning Representations (ICLR 2020)*. Addis Ababa, Ethiopia, 2020. URL: https://doi.org/10.48550/arXiv.2003.10555.
4. Merity S., Keskar N. S., Socher R. Regularizing and Optimizing LSTM Language Models. 2017. URL: http://arxiv.org/abs/1708.02182.
5. Enriching Word Vectors with Subword Information / P. Bojanowski et al. *Transactions of the Association for Computational Linguistics*. 2017. Vol. 5. P. 135-146. URL: https://doi.org/10.1162/tacl_a_00051.
6. Natural Language Processing in Action / H. Lane, M. Dishel. Second Edition. Manning Publications Co. LLC, 2022.

Майків Ігор Мирославович,
кандидат технічних наук, доцент,
Західноукраїнський національний університет, м. Тернопіль

Стецюк Арсен Ігорович, *бакалавр,*
Західноукраїнський національний університет, м. Тернопіль

АВТОМАТИЗАЦІЯ ЗРОШЕННЯ НА БАЗІ ДАНИХ IoT-СЕНСОРІВ ТА МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ

Інтернет-адреса публікації на сайті:
http://www.konferenciaonline.org.ua/ua/article/id-2592/

Сільське господарство дедалі активніше інтегрує цифрові технології, що дає змогу переходити від загальних, наперед заданих режимів поливу до точного керування зрошенням на основі фактичного стану ґрунту й середовища. У таких умовах особливого значення набуває використання сенсорних мереж, засобів Інтернету речей та методів машинного навчання, які забезпечують безперервний збір даних, їх аналітичну обробку та формування обґрунтованих керуючих рішень [1-3].

Побудова інтелектуального модуля підтримки рішень щодо автоматизованого зрошення потребує визначення такої архітектури, яка забезпечує узгоджену роботу всіх її складових: сенсорного вузла, підсистеми передавання даних, програмного модуля обробки інформації та виконавчого механізму поливу. Архітектура в цьому випадку визначає не лише склад компонентів, а й порядок їхньої взаємодії, послідовність руху даних та спосіб формування керуючого рішення.

У схемі на рисунку 1 відображено чотири основні рівні: сенсорний, рівень передавання даних, аналітичний рівень і рівень керування виконавчим пристроєм.

Першим рівнем є сенсорний рівень, призначення якого полягає у фіксації поточного стану ґрунту та навколишнього середовища. Основними джерелами інформації є датчики вологості ґрунту, температури повітря та відносної вологості повітря. Саме ці параметри формують базовий набір вхідних даних для подальшого аналізу. За потреби архітектура допускає розширення шляхом підключення додаткових сенсорів, наприклад температури ґрунту, освітленості або інших параметрів мікроклімату.

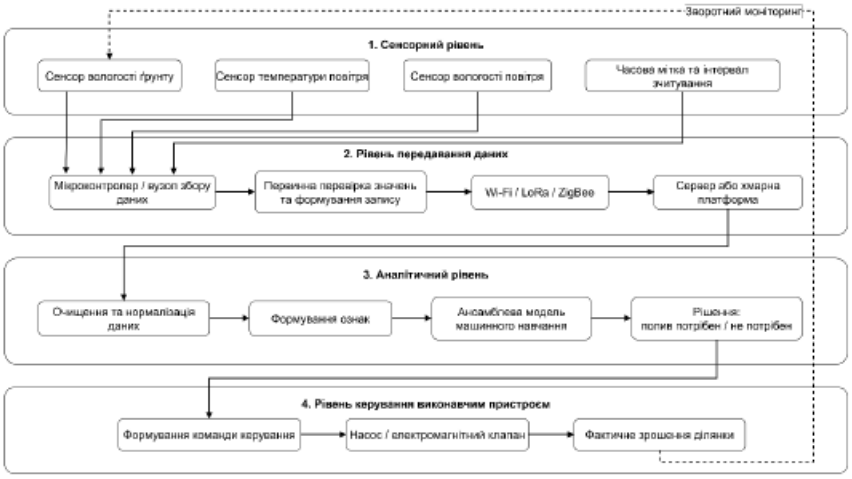

Рисунок 1 – Загальна архітектура інтелектуального модуля

Сенсорний вузол повинен працювати в режимі періодичного опитування, тобто зчитувати показники через визначені проміжки часу. Такий підхід дає змогу накопичувати історію вимірювань і надалі враховувати не лише поточний стан середовища, а й динаміку його змін. Кожне вимірювання повинно супроводжуватися часовою позначкою, оскільки часовий контекст є важливою складовою набору даних для моделі підтримки рішень. Відсутність часової прив'язки суттєво знижує аналітичну цінність сенсорної інформації.

Наступним елементом архітектури є компонент збору даних, якого реалізує мікроконтролер. Цей компонент виконує функцію проміжної ланки між сенсорами та програмним модулем. Його призначення полягає у прийманні показників із датчиків, попередній перевірці значень, приведенні їх до уніфікованого формату та передаванні до системи подальшої обробки. Фактично саме на цьому рівні формується первинний структурований запис, що містить значення параметрів і часову мітку.

Для збору даних важливими є кілька функцій. По-перше, має бути забезпечено циклічне зчитування показників із сенсорів. По-друге, необхідно перевіряти, чи належать отримані значення до допустимого діапазону. Якщо, наприклад, значення вологості ґрунту є фізично неможливим або різко відрізняється від попередніх спостережень, такий запис доцільно позначати як підозрілий. По-третє, передавання даних має відбуватися у форматі, зручному для подальшого завантаження в програмний модуль обробки.

Другим рівнем архітектури є рівень передавання даних. Його завдання полягає у доставці сформованих сенсорним вузлом записів до середовища зберігання й аналізу. У межах цієї роботи передбачається використання бездротового каналу зв'язку, зокрема Wi-Fi, як найбільш доступного варіанта для прототипу системи. Водночас структура архітектури не виключає використання інших IoT-технологій, наприклад LoRa або ZigBee, якщо цього потребуватимуть умови розгортання.

Передавання даних не повинно зводитися лише до механічного надсилання вимірювань. Важливо забезпечити цілісність потоку, правильний порядок записів та можливість повторної передачі у випадку короткочасного збою. Для навчального прототипу такі механізми можуть бути реалізовані у спрощеному вигляді, однак сама архітектура має враховувати ймовірність нестабільності каналу зв'язку. Це особливо важливо для аграрного середовища, де умови експлуатації сенсорних вузлів можуть бути змінними.

Третім рівнем є рівень накопичення та аналітичної обробки даних. Саме на цьому рівні модуль переходить від простого моніторингу до інтелектуального аналізу. Дані, що надходять від сенсорного вузла, повинні накопичуватися у сховищі – локальному або хмарному – та проходити етап попередньої обробки. До основних операцій цього етапу належать очищення даних, усунення пропусків, перевірка аномалій, нормалізація, агрегація та формування похідних ознак.

Попередня обробка є необхідною, оскільки сирі сенсорні дані не завжди придатні для безпосереднього використання в моделі машинного навчання. Значення можуть містити окремі спотворення, пропущені записи або різкі коливання, які не відображають реального стану середовища. Крім того, для моделі недостатньо лише поточних показників. Доцільно формувати додаткові ознаки, наприклад середнє значення за інтервал, зміну вологості за попередній період, часові характеристики та інші параметри, що підвищують інформативність вхідного набору.

Центральним компонентом цього рівня є аналітичний модуль, у якому виконується застосування ансамблевої моделі машинного навчання. На вхід модуля надходить підготовлений набір ознак, а на виході формується оцінка доцільності поливу. У межах роботи доцільною є класифікаційна постановка, за якої модель визначає один із двох станів: полив потрібен або полив не потрібен. Такий формат є найбільш придатним для подальшої інтеграції з виконавчим механізмом системи.

Четвертим рівнем архітектури є рівень керування виконавчим пристроєм. Його призначення полягає у перетворенні аналітичного рішення на фізичну дію. Якщо модель визначає, що полив є доцільним, то передається команда на ввімкнення насоса або відкриття електромагнітного клапана. Якщо підстав для поливу немає, команда не формується.

Перевагою запропонованої архітектури є її модульність. Кожен рівень виконує окрему функцію й може вдосконалюватися без повної перебудови всієї системи.

Таким чином, розроблено архітектуру модуля автоматизованого зрошення, яка охоплює сенсорний рівень, рівень передавання даних, аналітичний рівень і рівень керування виконавчим пристроєм.

Література:

1. Dantas R. A. S., Togneri R. M., Prati R. C., Kamienski C. A. A review of smart irrigation. *Biosystems Engineering*. 2025. Vol. 257. Art. 104220.
2. Yu L., Gao W., Shamshiri R. R., Tao S., Ren Y., Zhang Y., Su G. Review of research progress on soil moisture sensor technology. *International Journal of Agricultural and Biological Engineering*. 2021. Vol. 14, no. 4. P. 32-42.
3. Самойленко М. Ю. Принципи застосування технології Інтернет речей у сучасному світі техніки. *Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки*. 2020. Т. 31 (70), ч. 1, № 6. С. 142-148.