НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

"На правах рукопису"
УДК _____

«До захисту допущено»
Завідувач кафедри
_____ Наталія АУШЕВА
“___” _____2025 р

Магістерська дисертація

на здобуття ступеня другого (магістерського) рівня вищої освіти
за освітньою програмою “Цифрові технології в енергетиці”
зі спеціальності 122 “Комп’ютерні науки”

на тему Аналіз та передбачення погодних умов на основі машинного навчання з інтеграцією веб-технологій

Виконав студент 2 курсу, групи ТР-41мп
Постернак Антон Володимирович _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник д.т.н., професор Сергій ОТРОХ _____
(науковий ступінь, вчене звання, ім’я ПРІЗВИЩЕ) (підпис)

Резензент професор кафедри інженерії програмного забезпечення в енергетиці, д.т.н., професор Олег БАРАБАШ _____
(посада, науковий ступінь, вчене звання, ім’я ПРІЗВИЩЕ) (підпис)

Н.контроль старший викладач кафедри цифрових технологій в енергетиці Ольга БЕСПАЛА _____
(посада, ім’я ПРІЗВИЩЕ) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2025

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

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – другий (магістерський)
спеціальність 122 “Комп’ютерні науки”
Освітньо-наукова програма “Цифрові технології в енергетиці”

ЗАТВЕРДЖУЮ
Завідувач кафедри ЦТЕ
_____ Наталія АУШЕВА

«___»_____2025 р

ЗАВДАННЯ
на магістерську дисертацію студенту

Постернаку Антону Володимировичу
(прізвище, ім’я, по батькові)

1. Тема роботи Аналіз та передбачення погодних умов на основі машинного навчання з інтеграцією веб-технологій

керівник роботи Отрох Сергій Іванович, д.т.н., професор
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « 3 » листопада 2025 р. № 4779-с

2. Термін подання студентом дисертації 06.12.2025

3. Об’єкт дослідження: процеси опрацювання, аналізу та передбачення погодних даних у веб-орієнтованих інформаційних системах.

4. Вихідні дані: JavaScript, React, Node.js, Express.js, Mongoose, MongoDB, Python, Scikit-learn, FastAPI, VS Code, Git, GitHub, логістична регресія, градієнтний бустинг, багатошаровий персептрон.

5. Перелік завдань: аналіз предметної області; аналіз та порівняння моделей машинного навчання; розробка архітектури веб-застосунку; реалізація та тестування модулів програмної системи; оцінка ринкового потенціалу застосунку.

6. Орієнтовний перелік ілюстративного матеріалу: діаграма прецедентів, концептуальна діаграма архітектури, структура директорій серверної частини, концептуальна UML-модель БД, ілюстрації процесу взаємодії з системою.

7. Орієнтовний перелік публікацій: 1) Otrokh S., Posternak A. Web-based System for Weather Data Analysis and Prediction with Machine Learning // Scientific Research: Emerging Theories and Practical Breakthroughs. – Edinburgh, 2025.
2) Мельник Ю. В., Отрох С. І., Постернак А. В. Аналіз та передбачення погодних умов на основі машинного навчання з інтеграцією веб-технологій // Наукові записки ДУІКТ. 2025. № 2.

8. Дата видачі завдання «16» вересня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів підготовки магістерської дисертації	Термін виконання	Примітка
1	Затвердження теми роботи	16.09.2024	виконано
2	Вивчення та аналіз задачі	01.09 – 10.09.2025	виконано
3	Аналіз методів машинного навчання для передбачення погодних даних	11.09 – 16.09.2025	виконано
4	Розроблення архітектури та загальної структури системи	17.09 – 27.09.2025	виконано
6	Програмна реалізація модулів системи	28.09 – 16.10.2025	виконано
7	Тестування на налагодження розробленого програмного забезпечення	17.10 – 20.10.2025	виконано
8	Захист програмного продукту	21.10.2025	виконано
9	Оформлення магістерської дисертації	21.10 – 24.11.2025	виконано
9	Передзахист	25.11.2025	виконано
10	Захист		

Студент _____ **Антон ПОСТЕРНАК**
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник роботи _____ **Сергій ОТРОХ**
(підпис) (ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Актуальність теми. Зростання ролі погодної аналітики посилює потребу в доступних засобах базового дослідницького аналізу погодних даних і швидкого отримання передбачень, а також у простих освітньо-демонстраційних інструментах для ознайомлення користувачів із основними методами машинного навчання без необхідності спеціалізованих технічних знань.

Метою роботи є розроблення інтегрованого веб-орієнтованого застосунку, який реалізує повний цикл опрацювання погодних даних: імпорт табличних вибірок, базовий дослідницький аналіз та формування передбачень на основі моделей контрольованого машинного навчання з фіксованими гіперпараметрами.

Завдання дослідження:

– здійснити аналіз предметної області та наявних підходів до опрацювання погодних даних і побудови прогнозів;

– оцінити придатність та ефективність основних моделей машинного навчання з учителем для передбачення погодних параметрів і обґрунтувати вибір;

– розробити архітектуру веб-застосунку та визначити функціональну взаємодію його компонентів;

– реалізувати модулі обробки та організації даних, базового дослідницького аналізу та формування передбачень і провести тестування їх роботи у межах єдиного програмного середовища;

– оцінити ринкові перспективи впровадження розробленого застосунку.

Об’єкт дослідження – процеси опрацювання, аналізу та передбачення погодних даних у веб-орієнтованих інформаційних системах.

Предмет дослідження – методи та алгоритмічні підходи машинного навчання і засоби їх інтеграції у веб-орієнтоване програмне середовище для класифікації та передбачення параметрів погодних даних.

Практична цінність отриманих результатів полягає у створенні веб-орієнтованого застосунку, що забезпечує можливості завантаження, базовий

дослідницький аналіз і передбачення погодних даних у єдиному інтерфейсі без потреби в поглиблених технічних знаннях. Отриманий інструмент може використовуватися для швидкого отримання передбачень у прикладних задачах невисокої складності, а також як наочний засіб ознайомлення користувачів з основами машинного навчання на прикладі погодних даних.

Методи дослідження включають аналіз наукових джерел та наявних рішень з обробки та прогнозування погодних даних, методи передбачення на основі моделей машинного навчання, порівняльний аналіз та оцінювання якості моделей з використанням кількісних метрик, а також експериментальне тестування розробленого веб-орієнтованого застосунку.

Апробація результатів дисертації.

Основні результати роботи доповідалися на 2-й Міжнародній науково-практичній конференції «Scientific Research: Emerging Theories and Practical Breakthroughs» (17–19 листопада 2025 р., Единбург, Шотландія).

Дисертація складається з переліку умовних позначень, скорочень і термінів; вступу; шести розділів та висновків. Загальний обсяг роботи становить 113 сторінок, з них 91 сторінка основного тексту. Список використаних джерел має обсяг 5 сторінки та містить 43 найменування, додатки займають 17 сторінок.

Публікації.

Основні положення дисертації відображено в таких публікаціях:

– Otrokh S., Posternak A. Web-based System for Weather Data Analysis and Prediction with Machine Learning. Scientific Research: Emerging Theories and Practical Breakthroughs : тези доп. 2-ї Міжнар. наук.-практ. конф. (Единбург, 17–19 листоп. 2025 р.) / European Open Science Space. Единбург, 2025. С. 104–107;

– Мельник Ю. В., Отрох С. І., Постернак А. В. Аналіз та передбачення погодних умов на основі машинного навчання з інтеграцією веб-технологій // Наукові записки Державного університету інформаційно-комунікаційних технологій. 2025. № 2.

Ключові слова: *передбачення погодних умов, машинне навчання, AutoML, веб-застосунок, MERN, Scikit-learn*.

ABSTRACT

Relevance of the topic. The increasing role of weather analytics strengthens the need for accessible tools for basic exploratory analysis of weather data and rapid generation of predictions, as well as for simple educational and demonstration tools that familiarize users with the main methods of machine learning without the need for specialized technical knowledge.

The purpose of the work is to develop an integrated web-based application that implements the full cycle of weather data processing: import of tabular samples, basic exploratory analysis, and generation of predictions based on supervised machine learning models with fixed hyperparameters.

The tasks of the research are:

– carry out an analysis of the subject area and existing approaches to processing weather data and constructing forecasts;

– assess the suitability and effectiveness of the main supervised machine learning models for predicting weather parameters and substantiate the choice;

– design the architecture of the web application and define the functional interaction of its components;

– implement modules for data processing and organization, basic exploratory analysis and prediction generation, and test their operation within a single software environment;

– assess the market prospects for introducing the developed application.

The object of the research is the processes of handling, analysis and prediction of weather data in web-based information systems.

The subject of the research is machine learning methods, algorithmic approaches, and the means of their integration into a web-based software environment for classification and prediction of weather data parameters.

The practical value of the results obtained lies in the creation of a web-based application that provides capabilities for uploading, basic exploratory analysis and prediction of weather data within a single interface without the need for advanced

technical knowledge. The resulting tool can be used for quick prediction generation in applied tasks of low complexity, as well as an educational tool for familiarizing users with the basics of machine learning using weather data as an example.

The research methods include analysis of scientific sources and existing solutions for processing and forecasting weather data, prediction methods based on machine learning models, comparative analysis and evaluation of model quality using quantitative metrics, as well as experimental testing of the developed web-based application.

Approbation of the dissertation results.

The main results of the work were presented at the 2nd International Scientific and Practical Conference “Scientific Research: Emerging Theories and Practical Breakthroughs” (17–19 November 2025, Edinburgh, Scotland).

The dissertation consists of a list of notations, abbreviations and terms; an introduction; six chapters and conclusions. The total volume of the work is 113 pages, including 91 pages of main text. The list of references occupies 5 pages and contains 43 items, and the appendices comprise 17 pages.

Publications.

The main provisions of the dissertation are reflected in the following publications:

– Otrokh S., Posternak A. Web-based System for Weather Data Analysis and Prediction with Machine Learning. Scientific Research: Emerging Theories and Practical Breakthroughs: abstracts of the 2nd International Scientific and Practical Conference (Edinburgh, 17–19 November 2025) / European Open Science Space. Edinburgh, 2025. P. 104–107;

– Melnyk Y. V., Otrokh S. I., Posternak A. V. Analysis and prediction of weather conditions based on machine learning with integration of web technologies // Scientific Notes of the State University of Information and Communication Technologies. 2025. № 2.

Keywords: *weather prediction, machine learning, AutoML, web application, MERN, Scikit-learn*.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DOM – Document Object Model.

HTTP – Hypertext Transfer Protocol.

JSON – JavaScript Object Notation.

JSX – JavaScript XML.

JWT – JSON Web Token.

ODM – Object Document Mapper.

REST – Representational State Transfer.

UML – Unified Modeling Language.

ВСТУП

Сучасні тенденції розвитку цифрових технологій зумовлюють зростання інтересу до автоматизованих інструментів аналізу даних та алгоритмічного передбачення у прикладних сферах. Погодні процеси належать до категорії складних явищ, яким притаманна наявність багатовимірних та неоднорідних наборів даних, що відображають взаємодію великої кількості фізичних факторів. Традиційні методи прогнозування часто потребують значних обчислювальних ресурсів, складних математичних моделей і спеціальної підготовки користувача. Водночас розвиток методів машинного навчання відкрив можливість створення інструментів, здатних ефективно працювати з такими даними та виявляти приховані закономірності, які не завжди очевидні за використання класичних підходів.

Попри активне впровадження рішень на основі машинного навчання, їх практичне застосування й надалі передбачає наявність у користувача відповідної технічної підготовки. Робота з такими інструментами зазвичай охоплює програмну інтеграцію бібліотек, вибір та параметризацію алгоритмів, налаштування їх гіперпараметрів, побудову конвеєра обробки даних, а також виконання супутніх процедур, зокрема оцінювання ефективності моделей і підготовку вхідних вибірок. Окремим завданням є здійснення базового дослідницького аналізу даних – необхідного етапу перед розробленням моделі, що також потребує певного рівня компетенцій. У багатьох доступних програмних рішень такі процеси автоматизовані лише частково, а їх практична складність для користувача істотно залежить від рівня фахової кваліфікації.

Крім того, більшість інструментів для погодного аналізу орієнтовані або на професійні метеорологічні служби, або на задачі наукових досліджень, що ускладнює їх використання для локальних прикладних потреб – наприклад, у сфері агрономії, муніципального планування або в межах освітніх демонстраційних та практичних заходів. Водночас у багатьох ситуаціях достатньо застосувати фундаментальні моделі машинного навчання зі спрощеною

конфігурацією, які забезпечують допустимий рівень точності та зрозумілу інтерпретацію результатів. Саме тому актуальним є створення програмного рішення, яке поєднує доступність веб-технологій та можливості алгоритмів машинного навчання.

На основі аналізу наукових публікацій, присвячених методам обробки погодних даних, можна зробити висновок, що класичні моделі машинного навчання, такі як логістична регресія, градієнтний бустинг чи багатошаровий персептрон, залишаються придатними для роботи з невеликими та локальними наборами даних. Вони забезпечують збалансоване співвідношення між точністю прогнозів, швидкістю обчислень та відсутністю потреби у складному налаштуванні. Такі властивості роблять їх доцільними для використання у програмних інструментах, орієнтованих на широке коло користувачів – зокрема тих, хто не має поглибленої підготовки у сфері машинного навчання, програмування чи суміжних технічних дисциплін.

Разом з тим розвиток веб-технологій і мікросервісних архітектур створює передумови для побудови інтерактивних та масштабованих інформаційних систем. Інтеграція модуля машинного навчання у клієнтсько-серверну інфраструктуру забезпечує логічне розмежування між користувацьким інтерфейсом і обчислювальними процесами, що сприяє підвищенню швидкодії та надійності обміну даними.

Отже, на тлі зростання обсягів відкритих кліматичних даних, розширення спектра джерел їх надходження та підвищення потреби у регулярному моніторингу погодних умов актуальним стає використання інструментів, що забезпечують просту та доступну обробку таких даних. Одним із перспективних напрямів є розроблення інтегрованої веб-системи, яка поєднує завантаження наборів даних, виконання первинного аналізу, вибір моделі та формування передбачень у єдиному програмному середовищі. Такий підхід сприяє підвищенню доступності методів аналізу погодних параметрів для широкого кола користувачів і має як практичну, так і освітню цінність.

Метою роботи є розроблення інтегрованого веб-орієнтованого застосунку, який реалізує повний цикл опрацювання погодних даних: імпорт табличних вибірок, базовий дослідницький аналіз та формування передбачень на основі моделей контрольованого машинного навчання з фіксованими гіперпараметрами.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. Здійснити аналіз предметної області та існуючих підходів до опрацювання погодних даних і побудови прогнозів.

2. Оцінити придатність та ефективність основних моделей машинного навчання з учителем для передбачення погодних параметрів та обґрунтувати їх вибір.

3. Розробити архітектуру веб-застосунку і визначити функціональну взаємодію його компонентів.

4. Реалізувати модулі обробки та організації даних, базового дослідницького аналізу та формування передбачень і провести тестування їх роботи у межах єдиного програмного середовища.

5. Оцінити ринкові перспективи впровадження розробленого застосунку.

Об’єктом дослідження є процеси опрацювання, аналізу та передбачення погодних даних у веб-орієнтованих інформаційних системах, що забезпечують повний цикл роботи з табличними кліматичними вибірками.

Предметом дослідження є методи та алгоритмічні підходи машинного навчання і засоби їх інтеграції у веб-орієнтоване програмне середовище для класифікації та передбачення параметрів погодних даних.

З причини досягнення поставленої мети застосовується комплекс методів, що охоплює базовий статистичний аналіз, алгоритмічні підходи машинного навчання та інженерні практики програмної розробки.

Базові методи математичної статистики використовуються для попереднього дослідницького аналізу. Вони допомагають оцінити структуру даних, визначити придатність вибірки до моделювання, обрати ознаки для передбачення та підвищити коректність інтерпретації результатів.

Алгоритмічні методи машинного навчання, представлені логістичною регресією, градієнтним бустингом та багатошаровим персептроном, застосовуються для побудови моделей передбачення на обраному наборі даних. Оцінювання якості створених моделей здійснюється за допомогою відповідних метрик, а отримані результати дозволяють визначити їх ефективність у розв’язанні задач багатокритеріальної класифікації чи регресії.

Методи програмної інженерії та технології веб-розробки забезпечують узгоджену інтеграцію алгоритмічного модуля з веб-середовищем, стабільність взаємодії компонентів і відтворюваність результатів.

Розроблене програмне забезпечення дає змогу виконувати базовий аналіз погодних даних і формувати передбачення із допустимою для прикладних задач точністю. Система може використовуватися у простих практичних обчисленнях, локальному моніторингу та в освітньо-демонстраційних цілях.

Результати роботи були апробовані на 2-й міжнародній науково-практичній конференції «Scientific Research: Emerging Theories and Practical Breakthroughs», що відбувалася 17–19 листопада 2025 р. в Единбурзі, Шотландії.

Основні положення дисертації відображено в таких публікаціях:

– Otrokh S., Posternak A. Web-based System for Weather Data Analysis and Prediction with Machine Learning. Scientific Research: Emerging Theories and Practical Breakthroughs : тези доп. 2-ї Міжнар. наук.-практ. конф. (Единбург, 17–19 листоп. 2025 р.) / European Open Science Space. Единбург, 2025. С. 104–107;

– Мельник Ю. В., Отрох С. І., Постернак А. В. Аналіз та передбачення погодних умов на основі машинного навчання з інтеграцією веб-технологій // Наукові записки Державного університету інформаційно-комунікаційних технологій. 2025. № 2.

Структурно магістерська дисертація складається з переліку умовних позначень, скорочень і термінів; вступу; шести розділів та висновків. Загальний обсяг роботи становить 113 сторінок, з них 91 сторінка основного тексту. Список використаних джерел має обсяг 5 сторінок та містить 43 найменування, додатки займають 17 сторінок.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

У даному розділі проаналізовано особливості предметної області, пов’язаної з обробкою та прогнозуванням погодних даних, а також подано огляд сучасних підходів та інструментів, які застосовуються для моделювання атмосферних процесів. Окрему увагу приділено автоматизованим технологіям побудови моделей машинного навчання та їх придатності для задач передбачення погодних параметрів. На основі проведеного аналізу сформовано відповідні задачі до реалізації та обґрунтовано необхідність створення веб-орієнтованого застосунку для доступного й спрощеного передбачення погодних характеристик.

1.1 Особливості предметної області

Погодні умови є одними з найважливіших природних факторів, що впливають на безпеку, економічну діяльність, сільське господарство, транспортні системи та повсякденну поведінку людей. У сучасному суспільстві точність прогнозування погоди має критичне значення як для державних інституцій і бізнесу, так і для індивідуальних користувачів. Саме тому процеси збирання, агрегування, обробки та аналізу погодних даних становлять окрему частину прикладної інформатики та займають важливе місце в системах моніторингу довкілля [1-3].

Погодні дані характеризуються різноманіттям параметрів, що описують стан атмосфери у конкретний момент часу та в конкретній точці простору. До найбільш поширених метеорологічних характеристик належать температура повітря, атмосферний тиск, відносна вологість, швидкість і напрям вітру, кількість опадів, рівень хмарності, сонячна радіація та низка додаткових параметрів (точка роси, видимість, ультрафіолетовий індекс тощо) [3]. Кожна з таких величин має власну природу зміни в часі та залежить від комплексу зовнішніх факторів –

географічного розташування, пори року, добового циклу, особливостей рельєфу, циркуляції повітряних мас та інших атмосферних процесів.

Однією з ключових особливостей погодних даних є їх багатовимірність. Для повного опису погодного стану необхідно враховувати не один параметр, а сукупність величин, які змінюються з різною інтенсивністю та мають складні нелінійні взаємозв'язки [1, 3]. Так, температура корелює з кількістю сонячної радіації та хмарністю, вологість залежить від температури та атмосферного тиску, а вітер – від градієнтів тиску й особливостей місцевості. Це створює умови, за яких аналіз окремих параметрів є недостатнім, а для отримання коректних передбачень важливо враховувати багатофакторність та взаємозалежність ознак. У таких умовах виникає потреба в застосуванні методів багатовимірного аналізу, здатних виявляти приховані залежності між параметрами та зменшувати розмірність без втрати інформативності.

Важливою характеристикою кліматичних даних є сезонність – повторюваність погодних патернів у часових інтервалах (доба, місяць, рік) [2]. Наприклад, зміни температури мають яскраво виражений добовий цикл, а кількість опадів може відображати характерну для певного регіону сезонну поведінку. Сезонність ускладнює моделювання, оскільки потребує спеціальних методів обробки та нормалізації часових рядів або ж передбачає наявність моделей, стійких до періодичних коливань.

Ще однією властивістю є стохастичність. Погодні процеси підпорядковуються законам фізики, проте їх точна динаміка містить випадкову складову, зумовлену величезною кількістю мікропроцесів, які неможливо описати в детальному вигляді. Навіть у межах одного регіону погодні умови можуть відрізнятися через локальні ефекти: теплові острови в містах, різницю у висотах, наявність водойм чи лісових масивів. Така непередбачуваність призводить до того, що навіть великі обсяги даних не гарантують повної точності прогнозу, а моделі повинні враховувати можливість значної варіації параметрів [1, 4].

Кліматичні дані також характеризуються нестабільністю та нерівномірністю. Зміни можуть відбуватися різко: наприклад, фронт

атмосферного тиску призводить до швидкого охолодження, шквалистий вітер виникає за лічені хвилини, а локальні опади можуть випадати лише у частині території. Така мінливість створює високі вимоги до оперативного аналізу та обробки даних, оскільки інформація втрачає актуальність дуже швидко [4].

Сучасні метеорологічні системи використовують різноманітні джерела для збирання та оновлення погодної інформації. До найбільш поширених належать метеорологічні станції, які забезпечують наземні вимірювання за допомогою автоматизованих комплексів та сенсорних модулів. Значну роль відіграють супутникові спостереження, зокрема програми NASA Earth Observing System, що надають дані з великою територією охоплення та придатні для аналізу глобальних атмосферних процесів. Важливим джерелом кліматичних наборів виступають системи NOAA (National Oceanic and Atmospheric Administration), які містять багаторічні архіви вимірювань та використовуються для дослідження довгострокових закономірностей. Крім того, набувають поширення автоматичні локальні мережі сенсорів – міські метеостанції, приватні датчикові комплекси та інші низькорівневі системи моніторингу. Значну частину оперативних і ретроспективних даних надають відкриті онлайн-сервіси, такі як Open-Meteo, Meteostat, OpenWeatherMap тощо, що забезпечують доступ до табличних вибірок і API для інтеграції у програмні системи [3].

Кожне з таких джерел має власні особливості щодо просторового охоплення, точності та частоти оновлення. Супутникові дані надають можливість аналізувати великі регіони, проте характеризуються нижчою часовою роздільністю. Натомість локальні сенсори забезпечують високу деталізацію в часі та просторі, але охоплюють обмежені ділянки. Це визначає необхідність коректного поєднання джерел та врахування їх обмежень під час подальшого аналізу. Поєднання різних джерел формує неоднорідні набори даних, що потребують ретельної уніфікації та попередньої обробки перед використанням у прогнозних моделях. Відповідно, етап підготовки даних стає визначальним для досягнення високої точності прогнозування [1, 3].

У прикладних задачах погодна інформація найчастіше зберігається у табличному форматі, де кожен запис відповідає певному часовому кроку, а стовпці відображають різні параметри стану атмосфери. Такий формат є зручним для статистичного аналізу та застосування алгоритмів машинного навчання, однак має властивості, що ускладнюють подальшу обробку. Дискретність вимірювань залежить від джерела та може варіюватися від хвилинних до добових інтервалів, що впливає на рівень деталізації аналізу. Інколи трапляються пропуски, спричинені технічними збоями або нестабільністю сенсорів, а також наявні випадкові стрибки значень, які можуть відображати як реальні атмосферні зміни, так і похибки вимірювання [3]. Параметри різняться за масштабом та одиницями вимірювання, що потребує нормалізації або стандартизації даних. Важливим чинником залишається і географічна залежність: однакові параметри поводяться по-різному в різних регіонах через місцеві кліматичні особливості [2].

Проблематика аналітичної роботи з погодними даними пов’язана з низкою факторів, серед яких можна виділити такі особливості [1-4]:

– висока кореляція ознак;

– географічні відмінності;

– наявність шумів;

– різна частота вимірювань;

– нестабільність динаміки параметрів у часі.

Корельованість параметрів ускладнює побудову моделей, які повинні враховувати залежності між ознаками. Географічний вплив призводить до того, що моделі, треновані на даних одного регіону, не завжди відтворюють коректні результати в іншому. Шум і аномальні значення погіршують якість моделей і потребують додаткових методів очищення та фільтрації. Різна частота вимірювань у різних джерелах ускладнює інтеграцію вибірок, а локальна нестабільність атмосферних процесів формує нерегулярні коливання, які важко спрогнозувати на основі історичних даних.

У сукупності, дані фактори визначають складність аналізу погодних вибірок та створюють високі вимоги до методів їх обробки. Це обумовлює необхідність

використання систем, здатних проводити попередній аналіз, очищення та стандартизацію даних перед побудовою моделей прогнозування [1-4].

1.2 Методи та підходи прогнозування погодних умов

Прогнозування погодних умов є складною науковою задачею, що вимагає врахування великої кількості взаємодіючих атмосферних процесів, високої просторової та часової роздільності, а також значного рівня невизначеності. Історично основу систем прогнозування становили фізичні моделі чисельного прогнозу погоди NWP (Numerical Weather Prediction), які приблизно розв’язують рівняння динаміки атмосфери. Такі моделі і сьогодні залишаються ключовим інструментом метеорологічних служб завдяки своїй здатності враховувати фундаментальні фізичні закономірності, що визначають поведінку повітряних мас, температури, вологості, тиску та інших параметрів. Проте чисельні моделі прогнозування погоди потребують значних обчислювальних ресурсів, складної інфраструктури та високої точності вхідних даних, що обмежує їх практичну доступність для широкого кола користувачів [5].

Чисельні моделі прогнозування ґрунтуються на інтегруванні системи нелінійних диференціальних рівнянь, які описують рух атмосфери. Невизначеність у початкових умовах, що виникає внаслідок неповноти спостережень або похибок у вимірюваннях, призводить до швидкого наростання помилок прогнозу – ефекту, який добре відомий як чутливість до початкових умов. Для подолання даних обмежень метеорологічні служби застосовують ансамблеві прогнози, у яких незалежні симуляції виконуються з різними початковими умовами або параметризаціями атмосферних процесів. Ансамблеві системи, такі як ENS (Ensemble Prediction System), забезпечують оцінку ймовірнісного розподілу майбутніх сценаріїв погоди, що є важливим для попередження про небезпечні явища, керування енергетичними системами чи логістичне планування [5]. Проте навіть найкращі ансамблеві прогнози залишаються дорогими й

повільними, а їх точність зменшується зі збільшенням горизонту прогнозування [5, 7].

Паралельно з розвитком фізичних моделей набули поширення статистичні методи прогнозування. Вони включають класичні підходи часових рядів – зокрема авторегресійні моделі ARIMA (Autoregressive Integrated Moving Average), сезонні моделі SARIMA (Seasonal ARIMA), моделі сезонної декомпозиції та регресійні методи, які встановлюють залежності між параметрами атмосфери за історичними даними. Статистичні моделі менш ресурсоємні та добре працюють на локальних часових рядах, де важливі короткострокові тренди й сезонні складові. Утім, вони обмежені у здатності відтворювати складну просторову динаміку атмосфери й не враховують фізичної природи явищ. Тому статистичні підходи часто комбінують з іншими методами або застосовують у спеціалізованих задачах – наприклад, передбаченні температури на метеостанції чи короткостроковому прогнозуванні генерації вітрових електростанцій [5, 6].

На тлі швидкого розвитку обчислювальних методів особливе поширення отримали методи машинного навчання, що розглядають прогнозування погоди як задачу регресії або класифікації. Згідно з оглядом сучасних методів машинного навчання для прогнозування погоди [6], такі методи стали одним з найбільш перспективних напрямів завдяки здатності ефективно працювати з багатовимірними наборами даних, виявляти нелінійні залежності та адаптуватися до різних джерел інформації. У загальному вигляді алгоритми машинного навчання прогнозують параметри погоди, встановлюючи функціональні залежності між вхідними змінними (температура, вологість, вітер, історичні значення) та результатом (прогнозоване значення або ймовірність). Перевага таких моделей полягає у можливості навчатися без детального опису фізичних закономірностей [6, 7].

Машинне навчання стало особливо корисним у задачах роботи з табличними погодними даними, які містять часові ряди метеорологічних параметрів. У таких випадках моделі машинного навчання здатні виявляти кореляційні структури, що не завжди помітні при традиційному статистичному аналізі, і швидко

встановлюють залежності між різними атмосферними характеристиками. У прогнозуванні застосовують широкий спектр методів – від лінійних регресійних моделей до алгоритмів підсилення (бустингу) та нейронних мереж різних типів. Особливо поширені методи градієнтного бустингу, які показують високу точність при роботі з табличними даними, та багатошарові персептрони, що узагальнюють нелінійні залежності між ознаками [6].

Методи машинного навчання також відіграють важливу роль у задачах аналізу просторових і просторово-часових даних, які містяться у глобальних кліматичних архівах. У новітніх дослідженнях підкреслюється, що алгоритмічні методи можуть перевершувати класичні моделі чисельного прогнозування в окремих аспектах, забезпечуючи меншу похибку та швидший час прогнозування [7]. Одним із ключових недоліків ранніх алгоритмів машинного навчання була відсутність коректної оцінки невизначеності, оскільки багато моделей були спрямовані на мінімізацію середньоквадратичної помилки та створювали розмиті або усереднені прогнози. Проблема “розмитості” прогнозу виникала через те, що модель намагалася наблизити ймовірнісну динаміку атмосфери до середнього сценарію, який у реальності не спостерігається. Як зазначається у роботі [8], саме урахування невизначеності є ключовим для якісного прогнозування, особливо у випадку небезпечних атмосферних явищ, де важливо знати не лише одне можливе значення, а й діапазон імовірних сценаріїв [7, 8].

Сучасні підходи машинного навчання у прогнозуванні погоди прямують у напрямку створення як детерміністичних, так і ймовірнісних моделей. Детерміністичні моделі забезпечують одне прогнозоване значення або набір значень, тоді як імовірнісні моделі формують розподіл можливих майбутніх станів атмосфери [7]. Велику увагу привертають моделі глибинного навчання, зокрема рекурентні нейронні мережі RNN (Recurrent Neural Network), моделі довгої короткочасної пам’яті LSTM (Long Short-Term Memory network), згорткові нейронні мережі CNN (Convolutional Neural Network) для просторових даних і трансформерні архітектури – для складних просторово-часових залежностей [6-8]. Поява генеративних моделей, зокрема дифузійних підходів, відзначена як новий

етап розвитку алгоритмічного прогнозування погоди. Такі моделі, як GenCast, здатні не лише генерувати точні детерміністичні прогнози, а й створювати ансамблі можливих сценаріїв з високою точністю просторових структур [6, 7], що раніше було характерно лише для фізичних моделей чисельного прогнозування. За результатами досліджень, новітні алгоритмічні моделі не лише конкурують, а подекуди перевершують фізичні ансамблеві системи у передбаченні екстремальних подій, траєкторій циклонів або сценаріїв для енергетики [7].

Застосування методів машинного навчання у прогнозуванні погоди включає широкий спектр практичних задач. Такі моделі використовуються для прогнозування температури, швидкості та напрямку вітру, ймовірності опадів, інтенсивності сонячної радіації та інших параметрів, важливих для галузей енергетики, агрономії, транспорту та захисту населення [6, 7]. В окремих дослідженнях методи машинного навчання застосовуються для прогнозування небезпечних погодних явищ, зокрема шквалів, гроз, екстремальної спеки чи надзвичайно низької температури. Важливою перевагою алгоритмічних підходів є здатність швидко адаптуватися до локальних кліматичних характеристик, що робить їх придатними для моделювання на рівні міст, районів або окремих метеостанцій. За наявності достатньої кількості локальних даних такі моделі дають змогу суттєво підвищити точність порівняно з глобальними прогнозами чисельного моделювання.

Поширеним напрямом є комбінування фізичних моделей та алгоритмічних методів, що дає змогу використовувати фізичні закономірності атмосфери у поєднанні з гнучкістю й високою швидкістю обчислень алгоритмів машинного навчання. Такі гібридні системи можуть виступати як коректори фізичних моделей або використовуватися для постобробки результатів прогнозів чисельного моделювання, покращуючи точність і калібрування прогнозів [5-7]. Окремим напрямом розвитку є побудова алгоритмічних моделей, що працюють на сирих даних, включно з радарними або супутниковими знімками, без попередньої обробки – такі моделі здатні напряму виявляти атмосферні структури [7, 8].

Узагальнюючи, сучасні методи прогнозування погоди охоплюють широкий спектр підходів – від традиційних фізичних моделей чисельного прогнозування до статистичних методів, глибинних нейронних архітектур та гібридних систем. Розвиток машинного навчання суттєво розширив можливості інтерпретації та моделювання просторово-часових атмосферних процесів, забезпечивши підвищення точності, швидкості обчислень і здатності моделювати невизначеність [6-8]. Комбінування фізичних принципів із алгоритмічними підходами формує новий рівень як детерміністичних, так і ймовірнісних прогнозів, що робить сучасні моделі більш гнучкими та придатними для різних масштабів і сценаріїв прогнозування [5-8].

1.3 Автоматизовані підходи до побудови моделей машинного навчання

Стрімкий розвиток машинного навчання впродовж останніх років супроводжується поширенням інструментів, що значно спрощують створення, експериментування та розгортання моделей. Однією з ключових тенденцій є автоматизація етапів побудови моделей машинного навчання, що охоплює підбір алгоритму, налаштування гіперпараметрів, обробку даних і формування кінцевого прогнозу [9]. Сучасні автоматизовані системи зменшують поріг входження для користувачів без глибокого технічного досвіду, однак при цьому зберігають високу гнучкість та ефективність. Такі підходи формують нову екосистему інструментів для дослідницьких, освітніх та прикладних задач [9, 10].

Поширеним технологічним рішенням для роботи з моделями машинного навчання є інтерактивні обчислювальні середовища, зокрема Google Colab. Це хмарний сервіс, що надає можливість виконувати Python-код, використовувати потужні апаратні ресурси (CPU, GPU, TPU) та інтегрувати популярні бібліотеки машинного навчання без додаткових налаштувань локального середовища. Колаборaційні можливості Google Colab та доступність попередньо встановлених

пакетів зробили його одним із найвідоміших інструментів для проведення експериментів, навчання та прототипування моделей машинного навчання. Завдяки можливості безкоштовного доступу до базових ресурсів користувачі можуть швидко тестувати алгоритми на різних типах даних, аналізувати поведінку моделей і здійснювати візуалізацію результатів [9, 10].

Окремим інструментом для практичної роботи з даними є платформа Kaggle, яка поєднує репозиторій відкритих наборів даних та середовище інтерактивних блокнотів. Kaggle Notebooks надають готове оточення для машинного навчання з підтримкою PyTorch, TensorFlow та інших бібліотек, а також можливість використання GPU. На відміну від Google Colab, платформа орієнтована на змагальне моделювання та містить велику кількість готових наборів даних, у тому числі з реальними погодними параметрами. Завдяки навчальним матеріалам і відкритим конкурсам Kaggle є зручним інструментом для експериментів з алгоритмами аналізу та передбачення [9, 10].

Паралельно з розвитком інтерактивних середовищ сформувався окремий клас технологій – AutoML (Automated Machine Learning). Такі системи спрямовані на автоматизацію основних етапів побудови моделі: вибір алгоритму, налаштування гіперпараметрів, оптимізацію структури конвеєра машинного навчання, попередню обробку даних та, у багатьох випадках, формування ансамблів моделей. Основна мета AutoML полягає у забезпеченні можливості отримувати якісні результати без необхідності вручну перебирати десятки варіантів моделей і конфігурацій [9]. Такий підхід є особливо цінним у сфері аналізу погодних умов, оскільки відповідні дані часто мають складну структуру, а побудова оптимальної моделі потребує багаторазових експериментів і ретельного підбору параметрів.

Серед найбільш відомих та широко застосовуваних фреймворків AutoML можна виокремити кілька груп рішень, що відрізняються алгоритмічними підходами, масштабованістю та спектром застосувань. До найпоширеніших систем автоматизованого машинного навчання належать [9-11]:

– Auto-sklearn. Фреймворк, побудований на основі бібліотеки Scikit-learn, що автоматизує вибір алгоритму та оптимізацію гіперпараметрів, використовуючи байєсівську оптимізацію, метанавчання та механізми побудови ансамблів. Підтримує широкий спектр моделей і методів попередньої обробки даних, автоматично формуючи оптимальний конвеєр машинного навчання під конкретний набір даних.;

– H2O AutoML. Розподілена система автоматизованого машинного навчання, орієнтована на роботу з великими наборами даних. Автоматично тренує десятки моделей (GBM, Random Forest, GLM, нейронні мережі) та формує ансамбль найкращих рішень. Підтримує інструменти пояснюваності моделей та може функціонувати як локально, так і у кластерних середовищах;

– Google Cloud Vertex AI. Хмарна платформа з інструментами AutoML для структурованих і неструктурованих даних. Забезпечує автоматизацію циклу моделювання, від інженерії ознак до розгортання, гарантуючи високу продуктивність та масштабованість;

– AutoGluon. Сучасний фреймворк автоматизованого машинного навчання, орієнтований на багаторівневе стекування моделей та побудову вкладених ансамблів. Демонструє високу точність на табличних даних і в багатьох випадках перевершує класичні системи AutoML;

– FLAML. Ресурсоефективний інструмент автоматизованого машинного навчання, спрямований на мінімізацію обчислювальних витрат при збереженні високої точності;

– GAMA. Модульна система AutoML, орієнтована на дослідження в галузі автоматизованого моделювання. Дозволяє поєднувати різні методи пошуку, оптимізації та побудови ансамблів, забезпечуючи гнучкість для експериментальних сценаріїв;

– TPOT. Інструмент AutoML, що застосовує генетичні алгоритми для автоматичного проєктування конвеєрів машинного навчання. Оптимізує комбінації моделей, трансформацій та гіперпараметрів, формуючи ефективні рішення без ручного налаштування.

Наведені системи мають спільну мету – спростити процес побудови моделей машинного навчання, однак їх використання все одно потребує розуміння базових концепцій. Незважаючи на високий рівень автоматизації, користувач має розуміти принципи налаштування гіперпараметрів, інтерпретації результатів, обмеження алгоритмів і критерії оцінювання моделі [9].

Важливою проблемою систем AutoML є їхня орієнтованість на технічних фахівців. Переважна більшість інструментів потребує програмних навичок: роботи з Python, налаштування середовища, інсталяції залежностей та розуміння того, які параметри можна змінювати [10].

Ще однією проблемою є те, що підходи AutoML зазвичай оптимізують моделі за єдиною метою – максимізацією точності. У прикладних задачах передбачення важливий баланс між точністю, швидкістю роботи та простотою інтерпретації результатів. Складні моделі, створені фреймворками AutoML, не завжди зручні для навчальних і демонстраційних цілей, де потрібна прозорість та зрозумілість обчислювального процесу [10].

У контексті аналізу погодних даних і побудови прикладних систем AutoML забезпечує значні переваги, але не вирішує ключової проблеми – створення доступного інструменту для користувачів, які не володіють знаннями у сфері програмування або машинного навчання. Системи орієнтовані переважно на інженерів або дослідників і не пропонують простого способу виконання повного циклу роботи з даними через зручний графічний інтерфейс. Це створює потребу в системах, які поєднують можливості автоматизації з максимальною доступністю та мінімальними вимогами до технічної підготовки користувача [10].

Таким чином, сучасні автоматизовані підходи до побудови моделей машинного навчання формують потужну екосистему засобів, однак їх використання пов’язане із певним порогом входження. Це визначає актуальність розроблення інтерактивного інструменту, що інтегрує можливості аналізу, обробки та прогнозування даних у єдиному середовищі, забезпечуючи простоту використання та доступність для широкої аудиторії [9-11].

1.4 Постановка задачі

Сучасні інструменти передбачення погодних умов на основі методів машинного навчання переважно орієнтовані на технічних фахівців, потребують налаштування програмного середовища та роботи з бібліотеками, що ускладнює їх використання користувачами без відповідних навичок. У межах освітньо-демонстраційних сценаріїв та простих прикладних задач достатньо доступного веб-застосунку, який дозволяє завантажити невеликий набір даних, виконати базовий аналіз і отримати передбачення обраного параметра. Тому постає задача створення веб-орієнтованої системи, що забезпечує повний цикл роботи з погодними даними та автоматизує ключові етапи побудови прогнозу.

У межах постановки задачі визначаються такі завдання, необхідні для реалізації програмної системи:

– розробка модуля обробки даних, який забезпечує завантаження користувацького набору даних у форматі CSV, імпорт інформації до бази даних, автоматичне визначення типів атрибутів, базову валідацію значень та попередній перегляд структури вибірки;

– реалізація модуля базового дослідницького аналізу, що забезпечує формування описової статистики, побудову стовпчикових діаграм розподілу, візуалізацію часових рядів та розрахунок кореляційної матриці;

– інтеграція модуля машинного навчання, який дозволяє автоматизовано застосовувати фундаментальні моделі (логістична регресія, градієнтний бустинг, багатошаровий персептрон) із фіксованими гіперпараметрами;

– розробка підсистеми прогнозування та оцінки, що надає можливість вибору цільової змінної та набору ознак, генерування передбачення та відображення метрик якості створеної моделі;

– створення інтуїтивного веб-інтерфейсу, який об’єднує всі етапи роботи в єдиний сценарій та забезпечує доступність системи для користувачів без спеціальних технічних навичок.

Виконання зазначених задач передбачає реалізацію веб-орієнтованого застосунку, який поєднує базові засоби аналізу погодних даних та автоматизоване застосування фундаментальних алгоритмів машинного навчання, а також може використовуватися для освітньо-демонстраційних потреб і прикладних задач невисокої складності.

1.5 Висновки до розділу 1

У даному розділі розглянуто ключові властивості погодних даних, серед яких багатовимірність, сезонність, стохастичність, наявність шумів і пропусків, що ускладнює їх підготовку та впливає на точність подальшого прогнозування. Окреслено основні підходи до моделювання погодних умов, включаючи фізичні моделі, статистичні методи та алгоритми машинного навчання, які демонструють високу ефективність для табличних часових рядів, проте потребують технічної кваліфікації користувача.

Проаналізовано можливості автоматизованих інструментів і середовищ для побудови моделей, які спрощують процес, однак не забезпечують достатньої доступності для користувачів без відповідних навичок. На основі отриманих висновків сформульовано постановку задачі та визначено функціональні вимоги до інтегрованого веб-орієнтованого застосунку, що повинен забезпечувати завантаження, аналіз і передбачення погодних даних.

Розділ задає методологічні передумови для подальшої розробки системи, орієнтованої на спрощене застосування методів машинного навчання в задачах передбачення погодних параметрів для освітньо-демонстраційних цілей та практичних задач низької складності.

2 МЕТОДИ МАШИННОГО НАВЧАННЯ ДЛЯ ПЕРЕДБАЧЕННЯ ПОГОДНИХ ПАРАМЕТРІВ

У даному розділі описуються основні методи машинного навчання, що застосовуються для передбачення на основі структурованих табличних даних. Проаналізовано концептуальні підходи до навчання моделей, їх алгоритмічні принципи та особливості використання у прогнозних задачах. Значну увагу приділено етапам попередньої підготовки даних, які визначають якість моделювання та впливають на здатність моделей до узагальнення. Також подано характеристику поширених алгоритмів, зокрема логістичної регресії, градієнтного бустингу та багатошарового персептрона, а також огляд метрик оцінювання, що дозволяють кількісно оцінити точність і практичну ефективність моделей.

2.1 Методи та підходи машинного навчання

Машинне навчання охоплює сукупність математичних та алгоритмічних методів, спрямованих на побудову моделей, здатних ідентифікувати закономірності в даних та формувати узагальнення для підтримки прийняття рішень або прогнозування майбутніх значень параметрів. У межах даної галузі сформувалося кілька концептуальних підходів, що відрізняються організацією процесу навчання, характером доступної інформації та класом задач, які вони розв’язують [12].

Однією з базових груп методів є навчання з учителем (supervised learning), що передбачає наявність навчальної вибірки пар виду “вектор ознак – цільова змінна”. Метою є побудова відображення між простором ознак та цільовим параметром, яке добре апроксимує невідому залежність і забезпечує коректні передбачення на нових, раніше не спостережуваних даних. До поширених методів навчання з учителем належать лінійна та логістична регресія, метод опорних векторів (SVM), дерева рішень, а також ансамблеві алгоритми, зокрема

випадковий ліс (Random Forest) та градієнтний бустинг. У задачах на табличних вибірках такі методи дають змогу досягати прийнятного компромісу між точністю, інтерпретованістю моделі та обчислювальною складністю [12, 13].

У межах навчання з учителем традиційно розрізняють два основні типи завдань. Задачі класифікації спрямовані на віднесення об’єкта до одного з наперед визначених класів на основі значень його ознак; прикладами є діагностика станів об’єкта, виявлення аномалій або віднесення спостережень до дискретних категорій. Задачі регресії, у свою чергу, орієнтовані на прогнозування безперервного числового параметра, наприклад, температури, рівня навантаження чи обсягів споживання ресурсу. Обидва типи завдань є природними для структурованих табличних даних, де між ознаками існують стійкі кореляційні зв’язки, а їх поєднання дає змогу відтворити поведінку цільової змінної [12, 13].

Другий концептуальний підхід становить навчання без учителя (unsupervised learning), яке використовується для вибірок, що не містять явно заданої цільової змінної. У даному випадку алгоритм самостійно виявляє внутрішню структуру даних, приховані групи або імпліцитні компоненти. Найпоширенішими є методи кластеризації (k-середніх, ієрархічна кластеризація, DBSCAN), які групують об’єкти за мірою подібності ознак, а також методи зменшення розмірності (метод головних компонент PCA, багатовимірне шкалювання, t-SNE), що дозволяють спростити представлення багатовимірних даних із мінімальними втратами інформації. Такі підходи застосовуються для аналізу прихованих патернів, попередньої обробки даних та побудови інформативних ознак для подальших моделей [12, 13].

Окрему групу становить навчання з підкріпленням (reinforcement learning), у якому модель (агент) взаємодіє з динамічним середовищем і отримує за свої дії сигнали винагороди або штрафу. На основі багаторазових епізодів взаємодії формується стратегія (політика) поведінки, що максимізує сумарну очікувану винагороду у часі. До характерних методів даного класу належать табличні алгоритми Q-learning і SARSA, алгоритми пошуку оптимальної політики керування у марківських процесах прийняття рішень, а також підходи з

апроксимацією функцій цінності або політики за допомогою нейронних мереж (Deep Q-Networks). Такі методи застосовуються в робототехніці, задачах оптимального керування, ігрових середовищах та адаптивних системах підтримки рішень; водночас, вони менш придатні для опрацювання статичних табличних вибірок, де структура зв’язків не змінюється у часі [12-14].

Суттєву роль у сучасних дослідженнях відіграє глибоке навчання (deep learning), що базується на багатошарових нейронних мережах і дає змогу автоматично формувати ієрархію ознак з великих масивів даних. До ключових архітектур належать згорткові нейронні мережі CNN (Convolutional Neural Networks) для опрацювання просторово структурованих даних (зображень), рекурентні мережі RNN (Recurrent Neural Networks) та їх модифікації як LSTM чи GRU для послідовностей і часових рядів, а також трансформери для моделювання складних залежностей у послідовностях довільної довжини. Такі моделі демонструють високу точність у складних задачах, однак потребують значних обсягів даних, суттєвих обчислювальних ресурсів та складних процедур налаштування [14].

Попри різноманіття підходів, для роботи зі структурованими табличними даними, де метою є передбачення числових або категоріальних показників за сукупністю виміряних ознак, природним і найбільш поширеним залишається саме підхід навчання з учителем. Він забезпечує формалізований опис залежностей між параметрами, дає змогу здійснювати строгий контроль якості моделі, застосовувати процедури валідації та оцінювати точність на тестових вибірках.

2.2. Етапи попередньої підготовки даних

Попередня підготовка даних є ключовою складовою процесу побудови моделей машинного навчання, оскільки якість та структурна цілісність вхідної вибірки безпосередньо впливають на точність, стабільність і здатність моделі до узагальнення. Реальні табличні дані зазвичай містять пропуски, аномальні або

некоректні значення, різнорідні масштаби ознак та неоднорідні формати представлення інформації. Тому перед застосуванням алгоритмів машинного навчання необхідно виконати комплекс процедур, спрямованих на очищення, уніфікацію та статистичне впорядкування даних [15, 16].

Одним із базових етапів є обробка пропусків, оскільки відсутні значення можуть призводити до некоректної роботи алгоритмів або спричиняти статистичні викривлення. Пропуски виникають через технічні збої, нерівномірну частоту вимірювань, помилки введення або об’єктивну недоступність окремих параметрів. Для їх корекції застосовують методи заповнення пропусків середніми та медіанними значеннями, інтерполяційні підходи або видалення записів із надмірною кількістю порожніх полів. Коректне опрацювання пропусків забезпечує однорідність вибірки та зменшує ризик зміщення статистичних оцінок.

Важливою процедурою є виявлення та обробка аномалій. Аномальні значення можуть бути результатом некоректних вимірювань, збоїв обладнання, людських помилок або рідкісних подій у реальному середовищі. Їх наявність спотворює розподіли ознак, впливає на параметри моделі та знижує її стійкість. Для ідентифікації аномалій застосовують аналіз статистичних розподілів, інтерквартильний діапазон, критерії відстаней та кластеризаційні підходи. Усунення або коригування аномалій підвищує якість навчання та зменшує чутливість моделі до шумів [15, 16].

Для вибірок, що містять категоріальні ознаки, необхідним етапом є їх перетворення у числові представлення. Більшість алгоритмів машинного навчання працює з числовими даними, тому категорії повинні бути закодовані у формі бінарних індикаторів або впорядкованих значень у випадку ординальних ознак. Кодування забезпечує коректне включення таких змінних до простору ознак та збереження інформаційної структури категоріальних параметрів [15, 16].

Значну роль у забезпеченні стабільності моделі відіграють масштабування та нормалізація ознак. Оскільки окремі параметри можуть мати різні одиниці вимірювання або значно відмінні діапазони, ознаки з великими значеннями можуть непропорційно домінувати під час оптимізації. Масштабування приводить

значення до фіксованого діапазону, тоді як нормалізація у вигляді стандартизації забезпечує нульове середнє та одиничне стандартне відхилення ознаки. Дані процедури гарантують збалансований внесок усіх змінних у процес навчання та покращують збіжність алгоритмів [15, 16].

Необхідним етапом є розподіл вибірки на навчальну та тестову частини. Навчальна вибірка використовується для побудови моделі, тоді як тестова дозволяє оцінити її здатність узагальнювати закономірності на нових даних. Такий поділ запобігає перенавчанню та забезпечує об’єктивність оцінювання результатів. У складніших сценаріях, коли необхідно здійснити добір гіперпараметрів або провести додаткову оцінку проміжних результатів навчання, застосовують окрему валідаційну вибірку. У таких випадках також використовують методи перехресної перевірки, які забезпечують більш надійну та стабільну оцінку якості моделі за рахунок багаторазового розбиття даних на навчальні та валідаційні підмножини.

Комплексне виконання етапів попередньої підготовки забезпечує статистично однорідну, очищену та структурно узгоджену вибірку, яка є необхідною передумовою для побудови надійних і високоточних моделей машинного навчання незалежно від алгоритмічної парадигми [15, 16].

2.3 Логістична регресія

Логістична регресія є одним із базових та найпоширеніших методів бінарної класифікації у машинному навчанні. Основна мета даного методу полягає в моделюванні ймовірності віднесення об’єкта до певного класу на основі значень його вхідних ознак. На відміну від лінійної регресії, яка прогнозує неперервні значення, логістична регресія формує ймовірнісну оцінку у межах від нуля до одного, що дозволяє застосування для вирішення задач класифікації [17-19].

Математична модель логістичної регресії ґрунтується на на обчисленні лінійної комбінації ознак, значення якої перетворюється на ймовірнісну оцінку за допомогою логістичної (сигмоїдної) функції активації. Спочатку для об’єкта з

вектором ознак x формується лінійна комбінація ознак s, яка визначається наступним чином [17]:

$$s = w^T \cdot x + b, \tag{1}$$

де w – вектор вагових коефіцієнтів моделі,

x – вектор вхідних ознак,

b – зміщення (вільний член).

Отримана лінійна комбінація ознак s перетворюється на ймовірність належності об’єкта до позитивного класу шляхом застосування логістичної (сигмоїдної) функції. Прогнозована ймовірність p(x) визначається наступною формулою [17]:

$$p(x) = \frac{1}{(1 + e^{-s})}, \tag{2}$$

де p(x) – прогнозована ймовірність позитивного класу,

s – лінійна комбінація ознак.

Навчання моделі полягає у пошуку таких параметрів w та b, які максимізують правдоподібність спостережуваної вибірки. На практиці це реалізується шляхом мінімізації логістичної функції втрат у вигляді негативного логарифма правдоподібності (Log-Loss, або бінарної крос-ентропії), що має наступний вигляд [17]:

$$L(w, b) = -\frac{1}{N} \sum_{i=1}^{N} [y_i \cdot \ln p_i + (1 - y_i) \cdot \ln (1 - p_i)], \tag{3}$$

де N – кількість спостережень у вибірці,

y_i – фактична мітка класу для і-го об’єкта,

p_i – прогнозована ймовірність позитивного класу для і-го об’єкта.

Логістична регресія характеризується високою інтерпретованістю, оскільки знаки та величини вагових коефіцієнтів безпосередньо відображають напрямок та інтенсивність впливу відповідних ознак на логарифм відношення шансів (log-odds). Модель має відносно низьку обчислювальну складність як на етапі навчання, так і на етапі передбачення, що надає можливість використання для роботи з великими масивами даних [17-19].

До важливих передумов ефективного застосування логістичної регресії належать наближена лінійна роздільність класів у просторі ознак або можливість її досягнення за рахунок розширення ознакового простору, відсутність сильної мультиколінеарності між ознаками, а також бажане масштабування ознак, яке забезпечує стабільність та швидку збіжність градієнтних методів оптимізації [19].

До переваг логістичної регресії належать її простота, інтерпретованість та природна ймовірнісна форма вихідних значень. Водночас модель без додаткових нелінійних перетворень простору ознак здатна адекватно описувати переважно лінійно роздільні залежності, що звужує область її застосування до задач з відносно простою геометрією розподілу класів[17-19].

У випадках, коли задача класифікації містить більше двох класів, логістична регресія узагальнюється до багатокласового підходу. Найпоширенішим методом є мультикласова логістична регресія (Softmax регресія), яка моделює ймовірність належності об’єкта до кожного з K можливих класів. Для цього спочатку обчислюються окремі лінійні значення s_k для кожного класу, після чого застосовується Softmax функція, що нормалізує дані значення у вигляді ймовірностей. Прогнозована ймовірність класу k визначається за наступною формулою [17, 19]:

$$p_k = \frac{\exp(s_k)}{\sum_{j=1}^{K} \exp(s_j)}, \tag{4}$$

де p_k – прогнозована ймовірність належності класу k,

s_k – лінійне значення (оцінка) для класу k,

s_j – лінійне значення для j-го класу.

Такий підхід забезпечує коректне моделювання багатокласових залежностей та гарантує, що всі вихідні значення формуватимуть узгоджений вектор ймовірностей. Навчання моделі здійснюється шляхом мінімізації узагальненої крос-ентропійної втрати, яка є продовженням бінарної логістичної функції втрат. Мультикласова логістична регресія зберігає інтерпретованість параметрів моделі та ефективність оптимізації, хоча потребує більшої кількості вагових коефіцієнтів і залишається чутливою до мультиколінеарності та масштабування ознак [17, 19].

2.4 Градієнтний бустинг

Метод градієнтного бустингу належить до ансамблевих підходів у машинному навчанні, в яких підсумковий прогноз отримують шляхом поєднання множини слабких моделей. На відміну від методів, що будують незалежні моделі та усереднюють їхні передбачення, бустинг реалізує послідовне навчання: кожна наступна модель спрямована на корекцію помилок, допущених попередніми. Завдяки цьому ансамбль поступово уточнює апроксимацію цільової залежності, зменшуючи похибку на складних або недостатньо добре пояснених спостереженнях [17, 18, 20].

У класичній схемі градієнтного бустингу базовими моделями зазвичай виступають дерева рішень малої глибини. Такі дерева мають відносно невисоку складність і добру здатність відтворювати нелінійні залежності між ознаками, при цьому їх варіативність значно нижча порівняно з глибокими деревами. Нехай на стартовому кроці формується початкове наближення $F_0(x)$. На кожній ітерації t будується нове дерево $h_t(x)$, яке апроксимує негативний градієнт функції втрат відносно поточного ансамблю. У загальному вигляді оновлення ансамблю можна подати наступним чином [17, 20]:

$$F_t(x) = F_{t-1}(x) + \eta \cdot h_t(x), \quad (5)$$

де $F_t(x)$ – оновлена модель після додавання нового дерева,

$F_{t-1}(x)$ – попереднє наближення ансамблю,

η – коефіцієнт навчання, що визначає вагу внеску нового дерева,

$h_t(x)$ – базова модель, навчена на значеннях негативного градієнта функції втрат.

Концептуально метод градієнтного бустингу розглядає задачу навчання як процес мінімізації функціоналу похибки за допомогою градієнтного спуску в просторі моделей (функцій). Кожне дерево, що додається до ансамблю, виконує роль корекційного доданка, напрямок якого визначається локальним негативним градієнтом функції втрат. Це забезпечує уточнення поточного наближення та поступове підвищення якості передбачення. У класифікаційних задачах

найчастіше використовують логістичну або експоненційну функцію втрат, а в регресійних – середньоквадратичну або абсолютну похибку [17-20].

Серед ключових переваг градієнтного бустингу варто відзначити здатність моделювати складні нелінійні структури та взаємодії між ознаками, які важко адекватно відтворити лінійними методами. Оскільки дерева рішень не потребують масштабування числових ознак, ансамбль коректно працює з різнорідними параметрами, включно з числовими та категоріальними змінними (за умови їх належного кодування або застосування спеціалізованих реалізацій). На табличних наборах даних метод зазвичай демонструє дуже високу точність, що є однією з причин його широкої популярності [17, 18, 20].

Значне поширення градієнтного бустингу стало можливим завдяки появі ефективних реалізацій, які пропонують різні оптимізації класичного алгоритму, серед яких найбільш відомими є наступні [17, 20]:

– XGBoost, що використовує регуляризацію параметрів дерев, оптимізовані за пам’яттю й обчислювальною складністю алгоритми побудови дерев, а також підтримку паралельних і розподілених обчислень;

– LightGBM, який застосовує гістограмний підхід до розбиття ознак, листовий (leaf-wise) ріст дерев з обмеженням глибини, а також техніки GOSS (Gradient-based One-Side Sampling) та EFB (Exclusive Feature Bundling), які істотно прискорюють навчання на великих вибірках;

– CatBoost, який забезпечує ефективну роботу з категоріальними ознаками завдяки адаптивним порядковим перетворенням та спеціальним схемам статистичного кодування, що підвищують стабільність оцінок і знижують ризик перенавчання.

Попри високу потенційну точність, градієнтний бустинг є чутливим до вибору гіперпараметрів, насамперед глибини дерев, коефіцієнта навчання та кількості ітерацій. Надмірно складний або занадто довго навчений ансамбль схильний до перенавчання, тому важливо застосовувати регуляризаційні механізми, включаючи обмеження складності дерев, випадкове відбирання частини спостережень і ознак (subsampling), перехресну перевірку та процедури

ранньої зупинки. Такі підходи забезпечують компроміс між гнучкістю моделі та її здатністю до узагальнення. Варто зазначити, що класичний градієнтний бустинг має обмежені можливості паралелізації, оскільки кожне дерево залежить від результатів попереднього кроку; сучасні реалізації частково долають дані обмеження [17, 18, 20].

Унаслідок поєднання високої точності, гнучкості та ефективності на структурованих даних градієнтний бустинг став одним із ключових інструментів машинного навчання. За умови коректної регуляризації та налаштування гіперпараметрів метод здатен забезпечувати високу якість моделювання, хоча його чутливість до викидів та шуму в даних вимагає обережного підходу до вибору функції втрат і параметрів ансамблю [17, 20].

2.5 Багатошаровий персептрон

Багатошаровий персептрон (Multilayer Perceptron, MLP) належить до класу штучних нейронних мереж, які моделюють роботу біологічних нейронних систем. Модель складається з множини взаємопов’язаних обчислювальних вузлів – штучних нейронів, організованих у послідовні шари. Кожен нейрон виконує просте перетворення вхідного сигналу, проте їх узгоджена робота дозволяє мережі моделювати складні залежності між вхідними ознаками та цільовими змінними. Типова архітектура MLP містить вхідний шар, один або кілька прихованих шарів та вихідний шар, через які сигнал поширюється у процесі прямого поширення.

Вхідний шар задає числове представлення початкових ознак, що надходять на приховані рівні. Приховані шари виконують послідовні нелінійні перетворення, комбінуючи сигнали через вагові коефіцієнти та активаційні функції. Нейрони прихованих і вихідних шарів обчислюють зважену суму вхідних сигналів і формують вихід, застосовуючи відповідну активаційну функцію. Вихідний шар генерує кінцевий прогноз – числовий або категоріальний, залежно від типу задачі (регресія чи класифікація) [17, 21, 22].

Ключову роль у здатності MLP моделювати складні залежності відіграють нелінійні функції активації. За їх відсутності навіть кілька послідовних шарів були б еквівалентні одному лінійному перетворенню, що робило б мережу неспроможною апроксимувати складні закономірності. Нелінійні активаційні функції забезпечують можливість формувати багатовимірні нелінійні розбиття простору ознак та узгоджувати модель із геометрією даних. Вибір активаційної функції визначає швидкість і збіжність навчання, стабільність оптимізації та ефективність глибших архітектур. До основних функцій активації належать [23]:

– сигмоїдна (Sigmoid). Забезпечує гладку нелінійність із виходом у діапазоні від нуля до одиниці. Добре підходить для бінарної класифікації, однак схильна до насичення та зникнення градієнта;

– гіперболічний тангенс (Tanh). Має симетричний діапазон вихідних значень від мінус одиниці до одиниці, що забезпечує стабільшіне навчання порівняно з сигмоїдною функцією. Проте також страждає від проблеми зникнення градієнтів у ділянках насичення;

– ReLU (Rectified Linear Unit). Обчислювально проста й ефективна функція, яка пропускає додатні значення без змін, а від’ємні зануляє. Забезпечує швидке навчання та стабільну роботу глибоких мереж, але може спричиняти “вимирання нейронів”, коли ваги фіксуються у ділянці нульового градієнта;

– Leaky ReLU. Модифікація ReLU, що зберігає малий негативний градієнт у від’ємній області. Зменшує ризик “вимирання нейронів” і покращує поширення градієнта під час навчання;

– ELU (Exponential Linear Unit). Має плавну негативну частину завдяки експоненційному члену, що сприяє швидшій збіжності. Додає зміщення, яке наближує середнє значення активацій до нуля;

– SELU (Scaled Exponential Linear Unit). Забезпечує самонормалізацію активацій (self-normalizing), підтримуючи стабільні розподіли значень у глибоких мережах. Працює коректно за умови фіксованих параметрів функції та ініціалізації LeCun normal. Ефективна для дуже глибоких MLP;

– Softmax. Активаційна функція для вихідного шару багатокласової класифікації. Перетворює набір значень на ймовірнісний розподіл, що сумується до одиниці. Не використовується в прихованих шарах;

– Linear (Identity). Використовується у вихідному шарі задач регресії. Не вносить нелінійності, але дозволяє прогнозувати неперервні значення у довільному діапазоні.

Навчання MLP здійснюється методом зворотного поширення помилки. На прямому проході формується прогноз, після чого обчислюється функція втрат, що характеризує розбіжність між передбаченими й фактичними значеннями цільової змінної. Під час зворотного проходу градієнт помилки, отриманий на вихідному шарі, передається назад через усі шари. Для кожного шару визначається внесок його ваг у загальну помилку, після чого ваги коригуються у напрямку, що зменшує значення функції втрат. Разом зі стохастичними методами оптимізації це забезпечує адаптацію параметрів моделі до статистичної структури даних [17, 21].

Завдяки властивості універсальної апроксимації багатошарова нейронна мережа здатна наближати широкий клас складних, нелінійних залежностей у даних. За умови достатньої кількості нейронів така мережа може моделювати взаємодії, які лінійним моделям недоступні без додаткового ручного конструювання ознак. Це робить багатошаровий персептрон значно гнучким інструментом для задач, у яких прості моделі не забезпечують належної точності.

Серед характерних особливостей багатошарового персептрона можна виокремити наступні [17, 21, 22]:

– здатність автоматично формувати проміжні представлення даних у прихованих шарах без явного проєктування ознак;

– можливість уніфіковано працювати як із задачами класифікації, так і з задачами регресії, змінюючи лише структуру вихідного шару та функцію втрат;

– високу гнучкість архітектури завдяки зміні кількості шарів та нейронів, що дозволяє регулювати модельну складність;

– чутливість до якості ініціалізації ваг, масштабування ознак і налаштування гіперпараметрів навчання (швидкість навчання, розмір пакета, кількість епох), що вимагає ретельної процедури валідації.

Багатошаровий персептрон є ефективним інструментом для моделювання складних залежностей між вхідними ознаками та цільовими змінними у задачах передбачення. Завдяки здатності враховувати нелінійні взаємодії, поєднувати різні типи даних та адаптуватися до специфіки предметної області, він дає змогу формувати точні моделі для структурованих даних за умови належної підготовки даних і достатньої репрезентативності вибірки [17, 21].

2.6 Метрики оцінювання моделей

Оцінювання якості моделей машинного навчання є завершальним, але водночас невід’ємним етапом розроблення систем прогнозування. За допомогою метрик якості можна кількісно оцінити, наскільки добре модель відтворює закономірності, виявлені в даних, чи здатна вона узагальнювати їх на нові спостереження та які типи помилок домінують у її роботі. Вибір конкретних метрик залежить від постановки задачі: для класифікації застосовують одні показники, а для регресії – інші. Кожна метрика відображає окремий аспект функціонування моделі й не забезпечує універсальної оцінки її якості [24, 25].

У задачах класифікації, результати роботи моделі зручно аналізувати за допомогою матриці плутанини, у якій підраховуються чотири типи рішень: кількість об’єктів позитивного класу, правильно віднесених до нього (True Positive, TP); кількість об’єктів негативного класу, коректно класифікованих як негативні (True Negative, TN); кількість хибнопозитивних спрацьовувань, коли об’єкти негативного класу помилково позначаються як позитивні (False Positive, FP); та кількість хибнонегативних рішень, коли об’єкти позитивного класу класифікуються як негативні (False Negative, FN). На основі цих чотирьох величин обчислюються основні класифікаційні метрики [24, 25].

Точність класифікації (accuracy) описує загальну частку правильно класифікованих об’єктів у вибірці та визначається як відношення суми кількості правильно класифікованих позитивних та негативних об’єктів до загальної кількості спостережень. Хоча дана метрика є простою для інтерпретації, у задачах зі значним дисбалансом класів вона може бути оманливою: модель, що майже завжди вибирає домінувальний клас, здатна демонструвати високу точність, водночас систематично помиляючись щодо рідкісного, але важливого класу [24].

Прецизійність (precision) характеризує частку об’єктів, позначених моделлю як позитивні, що дійсно належать до позитивного класу. Якщо модель часто видає хибнопозитивні спрацьовування, тобто помилково маркує негативні об’єкти як позитивні, значення precision знижується [24, 25].

Чутливість (recall) відображає інший аспект поведінки: яку частку реальних позитивних об’єктів модель змогла правильно виявити серед усіх наявних позитивних прикладів. У ситуаціях, коли критично важливо не пропустити жодного позитивного випадку, наприклад під час виявлення небезпечних станів або аномалій, саме чутливість набуває пріоритетного значення [24, 25].

Оскільки прецизійність і чутливість зазвичай демонструють протилежну динаміку (покращення однієї часто призводить до погіршення іншої), для їх узгодженого врахування застосовують F1-міру. Вона обчислюється як гармонійне середнє між precision та recall і набуває високих значень лише за умови, що обидва показники є водночас достатньо великими. Завдяки цьому F1-міра виступає інтегральним показником ефективності моделі в ситуаціях, коли небажаними є як помилки пропуску, так і помилки хибного спрацювання [24, 25].

Для задач регресії, у яких цільова змінна є неперервною, якість моделі описують за допомогою метрик, що відображають середній масштаб помилок прогнозування та чутливість до великих відхилень [24, 25].

Так, середня абсолютна похибка (Mean Absolute Error, MAE) належить до базових показників регресійної якості, оскільки дає узагальнену оцінку типової величини помилки моделі в тих самих одиницях, що й цільова змінна [24].

Середня абсолютна похибка визначається наступною формулою [24]:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - f(x_i)|, \quad (6)$$

де MAE – середня абсолютна похибка,

N – кількість спостережень у вибірці,

y_i – фактичне значення цільової змінної,

$f(x_i)$ – передбачене значення цільової змінної для і-го спостереження.

Значення середньої абсолютної похибки відображає середній рівень відхилення прогнозів від реальних значень і є відносно стійким до поодиноких великих викидів, оскільки всі помилки враховуються лінійно [24].

Середньоквадратична похибка (Mean Squared Error, MSE) та кореневе середньоквадратичне відхилення (Root Mean Square Error, RMSE) ґрунтуються на квадратичному штрафуванні відхилень між прогнозованими та фактичними значеннями. Такий підхід підсилює вплив великих помилок на загальну оцінку, що робить дані метрики особливо чутливими до грубих відхилень у моделі [24].

RMSE, на відміну від MSE, виражається в тих самих одиницях вимірювання, що й цільовий параметр. Завдяки цьому його часто застосовують як інтерпретовану характеристику середнього масштабу помилки, яка водночас зберігає підвищену чутливість до великих відхилень. Унаслідок цього RMSE вважається більш інформативним показником для практичної інтерпретації відхилень, особливо в задачах, де важливо оцінити реальний масштаб помилки передбачення [24, 25].

Коефіцієнт детермінації застосовують для оцінювання того, наскільки добре побудована модель пояснює змінність цільової змінної порівняно з наївним прогнозом, що завжди повертає її середнє значення. Значення коефіцієнта детермінації визначають за такою формулою [24, 25]:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - f(x_i))^2}{\sum_{i=1}^{N}(y_i - \mu_y)^2}, \quad (7)$$

де R^2 – коефіцієнт детермінації,

N – кількість спостережень у вибірці,

y_i – фактичне значення цільової змінної,

$f(x_i)$ – передбачене значення цільової змінної і-го спостереження,

μ_y – середнє значення цільового параметра у вибірці.

Коефіцієнт детермінації, значення якого наближається до одиниці, свідчить про високу здатність моделі відтворювати варіацію цільової змінної. Низькі значення, що тяжіють до нуля, означають відсутність суттєвої переваги моделі над базовим прогнозом, який повертає середнє значення. Водночас від'ємні значення даного показника вказують на те, що сумарна квадратична помилка моделі перевищує помилку наївного середнього прогнозу, що свідчить про незадовільну якість моделювання [24, 25].

У ширшому контексті оцінювання якості моделей машинного навчання важливо враховувати, що класифікаційні та регресійні метрики характеризують різні аспекти моделювання: збалансованість типів помилок, середній масштаб відхилень, чутливість до великих похибок або здатність моделі пояснювати змінність даних. Оскільки жодна окрема метрика не забезпечує вичерпної оцінки, на практиці застосовують комплексний аналіз кількох показників, узгоджуючи їх інтерпретацію з вимогами предметної області та специфікою наявних даних. Остаточний вибір метрик для порівняння моделей і перевірки їх ефективності визначається на етапах реалізації та експериментального тестування [24, 25].

2.7 Висновок до розділу 2

У даному розділі розглянуто основні методи машинного навчання, що застосовуються для передбачення параметрів на основі табличних даних, а також охарактеризовано алгоритмічні особливості їх роботи. Наведено, що якість моделей значною мірою залежить від коректної підготовки вибірки: усунення пропусків і аномалій, кодування категоріальних змінних, масштабування ознак та розподілу даних на навчальні й тестові підмножини. Дані процедури забезпечують узгодженість даних і підвищують стабільність моделювання.

Розглянуті алгоритми, до яких належать логістична регресія, градієнтний бустинг і багатошаровий персептрон, відрізняються гнучкістю, інтерпретованістю та здатністю відтворювати складні залежності. Логістична регресія є придатною для задач з приблизно лінійною структурою даних; бустингові моделі демонструють високу точність і добре працюють з нелінійними зв’язками; нейронні мережі забезпечують найбільшу гнучкість, але чутливі до параметрів навчання та масштабу ознак.

Оцінювання ефективності моделей здійснюється за допомогою метрик, що відображають різні аспекти точності: точність класифікації, прецизійність, чутливість, F1-міра, середня абсолютна похибка, середньоквадратична похибка, кореневе середньоквадратичне відхилення та коефіцієнт детермінації. Вони дозволяють комплексно проаналізувати помилки моделі та визначити її практичну придатність.

3 ЗАСОБИ І ТЕХНОЛОГІЇ РОЗРОБКИ ВЕБ-ОРІЄНТОВАНОГО ЗАСТОСУНКУ

У даному розділі обґрунтовано вибір мов програмування та технологічного стеку для реалізації веб-орієнтованого застосунку з інтегрованим модулем машинного навчання. Проаналізовано інструментарій розробки клієнтської частини на базі бібліотеки React та серверної складової з використанням платформи Node.js. Окрему увагу приділено засобам реалізації модуля передбачення мовою Python, зокрема бібліотекам Scikit-learn та FastAPI, а також обґрунтуванню використання документоорієнтованої бази даних MongoDB. Також наведено огляд інтегрованого середовища розробки та системи контролю версій, що забезпечують ефективність процесу створення, тестування та супроводу програмного продукту

3.1 Обґрунтування вибору мов програмування та платформ розробки

Поставлені вимоги до веб-орієнтованого застосунку визначають необхідність наявності клієнтської частини, що виконується безпосередньо в браузері користувача, серверної частини для обробки запитів і взаємодії з базою даних, а також окремого модуля, відповідального за інтеграцію за використання алгоритмів машинного навчання. Така структурна організація дозволяє використання узгодженого набору мов програмування та платформ, які забезпечують кросплатформність, масштабованість і мінімальні витрати на інтеграцію між компонентами системи.

Для реалізації клієнтської частини застосовано JavaScript – високорівневу інтерпретовану мову програмування з динамічною типізацією, яка підтримується всіма сучасними веб-браузерами. Дана мова стандартизована специфікацією ECMAScript і дозволяє без додаткових плагінів виконувати код безпосередньо в

браузері користувача. JavaScript забезпечує подійно-орієнтовану модель програмування, інтеграцію з об’єктною моделлю документа DOM для динамічних змін контенту веб-сторінки, а також підтримує асинхронні операції для взаємодії з сервером без перезавантаження сторінки. Розвинена екосистема інструментів для розробки, тестування та супроводу коду робить JavaScript ефективним інструментом для створення інтерактивних і масштабованих веб-інтерфейсів [26].

Серверну частину системи реалізовано з використанням платформи Node.js – серверного середовища виконання JavaScript, побудованого на високопродуктивному рушії V8 з подійно-орієнтованою архітектурою та асинхронним введенням-виведенням, що дає змогу ефективно обробляти велику кількість конкурентних мережевих з’єднань і запитів у умовах високого навантаження. Використання однієї мови програмування на клієнтській і серверній сторонах уніфікує мовне середовище, дозволяє застосовувати єдині структури даних на всіх рівнях програмного комплексу, спрощує обмін даними між компонентами, зменшує ризик помилок під час їх трансформації та полегшує розробку, тестування й супровід програмного продукту [27].

Програмну реалізацію модуля машинного навчання виконано із застосуванням Python – високорівневої об’єктно-орієнтованої інтерпретованої мови програмування, з суворою динамічною типізацією, яка є галузевим стандартом у сфері штучного інтелекту завдяки розвиненій екосистемі спеціалізованих бібліотек, орієнтованих на високопродуктивну обробку даних. Застосування даного інструментарію прискорює впровадження алгоритмів машинного навчання, а підтримка засобів міжсервісної взаємодії дозволяє інтегрувати модуль передбачення у загальну архітектуру застосунку, забезпечуючи можливість подальшого масштабування системи [28].

У складі засобів розробки для екосистеми JavaScript як базовий інструмент керування залежностями використано менеджер пакетів npm. Його застосування забезпечує централізоване отримання, оновлення та розповсюдження модулів, підтримує механізми семантичного версіонування та дає змогу однозначно фіксувати використовувані версії пакетів у конфігураційних файлах проєкту. Це

спрощує відтворення однакового робочого середовища на різних апаратно-програмних платформах, знижує ризик конфліктів версій і підвищує керованість життєвого циклу застосунку. Крім того, npm надає просунуті засоби опису та виконання типових операцій розробки, зокрема збирання, тестування й запуску допоміжних утиліт у вигляді сценаріїв, що сприяє формалізації, уніфікації та автоматизації процесів розробки, супроводу та розгортання програмного забезпечення [29].

3.2 Інструменти розробки клієнтської частини

Клієнтська частина веб-орієнтованого застосунку реалізована з використанням бібліотеки React, яка забезпечує компонентний підхід до проєктування інтерфейсу. Користувацький інтерфейс подається у вигляді сукупності незалежних багаторазово використовуваних компонентів із власним станом та інкапсульованою логікою, що підвищує модульність програмного коду, полегшує його розширення та супровід, а також спрощує організацію тестування. Застосування віртуальної моделі DOM дає змогу зменшити кількість безпосередніх операцій над деревом DOM браузера, що позитивно впливає на продуктивність та скорочує час відгуку інтерфейсу за умов частих оновлень даних. Додатковою перевагою React є сформована екосистема та широка підтримка з боку спільноти розробників, що забезпечує наявність перевірених підходів до реалізації типових задач розробки клієнтської частини [30].

Для стилізації інтерфейсу використано Tailwind CSS та бібліотеку компонентів DaisyUI. Tailwind CSS реалізує підхід utility-first, за якого розробник оперує набором елементарних класів, кожен з яких відповідає окремій властивості стилю, зокрема параметрам відступів, колірних схем, типографіки та просторового розташування елементів. За такого підходу немає потреби формувати розгалужені користувацькі таблиці стилів, а оформлення здійснюється безпосередньо на рівні розмітки, що прискорює процес верстки, спрощує

підтримку адаптивного дизайну та дає змогу зменшити обсяг фінального коду CSS за рахунок видалення невикористаних класів на етапі збирання. DaisyUI доповнює Tailwind готовим набором уніфікованих компонентів інтерфейсу, таких як кнопки, елементи форм, модальні вікна та навігаційні панелі, які узгоджені зі стилістичними правилами Tailwind, дозволяють швидко формувати цілісні інтерфейсні блоки з мінімальними витратами на ручну розробку стилів [31, 32].

Як інструмент збирання та середовище розробки фронтенду застосовано Vite. У режимі розробки Vite використовує нативну підтримку модулів ECMAScript у сучасних браузерах і забезпечує високошвидкісний запуск сервера розробки разом із механізмом гарячого оновлення модулів HMR без повного перезбирання проєкту. Це істотно скорочує час очікування результатів змін у коді та підвищує ефективність ітеративної розробки, що є важливим для складних застосунків із розгалуженою структурою інтерфейсу. Під час формування фінальної версії, Vite створює оптимізований і мінімізований код, який розподіляється на окремі частини, що зменшує час завантаження веб-сторінок і покращує чутливість інтерфейсу до дій користувача [33].

Для побудови графіків та діаграм застосовано бібліотеку Recharts, яка інтегрується з React і забезпечує можливість декларативного опису діаграм у форматі JSX. Бібліотека підтримує широкий спектр типів діаграм, зокрема лінійні, стовпчикові та кругові, а також надає засоби реалізації адаптивної верстки, налаштовуваних підказок і легенд. Такі можливості дозволяють швидко створювати інформативні візуалізації даних без необхідності розробки низькорівневої логіки рендерингу. Взаємодію клієнтської частини із серверною підсистемою та модулем машинного навчання організовано за допомогою бібліотеки Axios, яка надає уніфікований інтерфейс для виконання HTTP-запитів, підтримує роботу з промісами, використання інтерсепторів, централізовану обробку помилок, а також гнучке налаштування заголовків і параметрів запитів, що підвищує надійність та керованість мережевої взаємодії [34].

3.3 Технологічні компоненти реалізації серверної частини

Серверну частину застосунку реалізовано з використанням фреймворку Express для платформи Node.js. Даний компонент є мінімалістичним веб-фреймворком, що забезпечує засоби для розгортання сервера HTTP, конфігурації маршрутів та підключення проміжних обробників запитів. Архітектура фреймворку не нав’язує жорстко фіксованої структури проєкту, що дає змогу адаптувати організацію коду до функціональних вимог конкретного застосунку. Широке промислове використання Express супроводжується розвиненою екосистемою модулів і типових рішень, що підвищує надійність реалізації та зменшує витрати на супровід [35].

Для доступу до СКБД MongoDB застосовано бібліотеку Mongoose, яка реалізує підхід об’єктно-документного відображення даних. Схеми, що визначаються в Mongoose, дають змогу формально задати структуру документів, типи полів, значення за замовчуванням і правила валідації. Підтримка моделей і проміжних обробників забезпечує централізовану обробку даних до та після основних операцій, що сприяє збереженню їхньої цілісності та узгодженості. Завдяки додатковому рівню абстракції, бібліотека ізолює розробника від низькорівневих деталей драйвера MongoDB, а також знижує ймовірність помилок, пов’язаних із некоректною структурою документів [36, 37].

Підсистема автентифікації реалізована з використанням бібліотек bcrypt та JWT. Механізм bcrypt виконує криптографічно стійке хешування паролів із застосуванням випадкового параметра хешування та налаштовуваного параметра складності, що суттєво підвищує витрати на перебір значень у разі компрометації сховища облікових даних. JWT застосовується для генерування компактних маркерів доступу, які передаються між клієнтом і сервером та інкапсулюють мінімально необхідну інформацію про користувача. Використання токенів дозволяє організувати автентифікацію без серверних сесій, зменшити кількість серверних операцій з зберігання сесій і спростити горизонтальне масштабування системи.

Обробка завантажуваних файлів реалізована за допомогою бібліотеки Multer, орієнтованої на прийом даних у багаточастинному форматі передавання даних форм, зокрема файлів формату CSV, які надсилає користувач. Даний інструмент підтримує режими зберігання даних в оперативній пам’яті або на диску, а також реалізує обмеження на розмір і кількість файлів, що знижує ризики перевантаження та підвищує загальний рівень захищеності. Для аналізу вмісту файлів формату CSV використовується бібліотека потокового аналізу csv-parser, яка функціонує в потоковому режимі та забезпечує поетапну обробку великих наборів даних без суттєвого зростання споживання оперативної пам’яті.

Керування конфігураційними параметрами виконується за допомогою бібліотеки dotenv, що забезпечує завантаження змінних оточення із зовнішнього конфігураційного файлу. До таких параметрів належать, зокрема, ключі доступу, адреси підключення до бази даних та мережеві налаштування. Винесення конфігурації за межі вихідного коду спрощує розгортання програмного забезпечення в різних середовищах: від етапу розробки й тестування до промислової експлуатації, а також зменшує імовірність витоку чутливих даних через систему контролю версій.

3.4 Технології розробки модуля машинного навчання

Модуль машинного навчання реалізовано як окремий веб-сервіс на основі фреймворку FastAPI. Даний компонент враховує особливості сучасного Python, зокрема активно використовує анотації типів, автоматично виконує валідацію вхідних даних на основі формальних моделей та генерує опис інтерфейсу у форматі OpenAPI. На основі такого опису автоматично створюється інтерактивна документація, що спрощує тестування сервісу й інтеграцію з іншими компонентами системи. Використання асинхронного стеку ASGI забезпечує високу продуктивність і дає змогу обробляти значну кількість паралельних запитів до моделі.

Для реалізації алгоритмів машинного навчання застосовується бібліотека Scikit-learn. Вона містить широкий спектр класичних методів класифікації, регресії, кластеризації та засобів попередньої обробки даних. Уніфікований інтерфейс алгоритмів, заснований на використанні методів як fit, predict чи transform спрощує заміну, комбінування, а також порівняння моделей у межах одного програмного проєкту. Також, бібліотека надає інструменти для налаштування гіперпараметрів, побудови конвеєрів обробки даних та оцінювання якості моделей за різними метриками, що є необхідною умовою коректного аналізу результатів [38].

Бібліотека NumPy використовується як основний засіб для роботи з багатовимірними масивами та векторизованими операціями. Застосування даного інструменту надає змогу ефективно виконувати операції лінійної алгебри, обробляти великі обсяги числових даних та готувати їх до подальшого навчання моделей. Більшість алгоритмів, реалізованих у Scikit-learn, працюють безпосередньо з масивами NumPy, що забезпечує узгодженість між етапами підготовки даних і навчання та зменшує накладні витрати на перетворення структур даних [38, 39].

Для збереження та відновлення створених моделей використовується бібліотека joblib. Вона оптимізована для серіалізації об’єктів, які містять великі масиви даних, з мінімальними витратами часу та пам’яті. Такий підхід дозволяє один раз навчити модель, а далі багаторазово використовувати її для передбачення без повторного навчання, що знижує обчислювальне навантаження на систему.

3.5 Обґрунтування вибору СКБД MongoDB

Для зберігання даних використано документоорієнтовану СКБД MongoDB, ключовою перевагою якої є гнучка, еволюційна схема даних: структура документів може змінюватися з часом, а записи в межах однієї колекції можуть мати різний набір полів. Такий підхід зручний для застосунків, що працюють з

даними з різних джерел і форматів, де склад атрибутів та кількість показників відрізняються залежно від джерела. У разі використання реляційної СКБД подібна варіативність вимагала б регулярних змін схеми, міграцій та перебудови таблиць, тоді як MongoDB дозволяє безпосередньо зберігати структурно неоднорідні дані без додаткових складних перетворень [36, 37].

MongoDB використовує формат документів BSON, структурно сумісний із JSON, що природно інтегрується з об’єктною моделлю JavaScript у Node.js. Це спрощує серіалізацію та десеріалізацію даних під час обміну між сервером і клієнтом, зменшує обсяг допоміжного коду для перетворення структур і підвищує читабельність програмного коду. Підтримка вкладених документів і масивів дає змогу зберігати пов’язані дані в одному записі, зменшуючи потребу у з’єданнні даних з інших колекцій та скорочуючи час вибірки [36, 37].

Також, дана СКБД має вбудовані механізми горизонтального масштабування та реплікації. Шардінг забезпечує розподіл колекцій між кількома вузлами, що дає змогу масштабувати систему за обсягом даних та інтенсивністю запитів. Реплікація підвищує відмовостійкість, забезпечує актуальні резервні копії та безперервний доступ до даних. Механізм агрегування дозволяє виконувати складні аналітичні запити, включно з групуванням, фільтрацією та обчисленням агрегованих показників, безпосередньо на рівні СКБД, що є важливим для обробки великих масивів вимірювань [36, 37].

MongoDB інтегрується з Node.js через офіційний драйвер і бібліотеку Mongoose. Зокрема, Mongoose надає засоби для опису схем на рівні застосунку, валідації даних та виконання типових операцій читання й модифікації [36].

3.6 Інтегроване середовище розробки та система контролю версій

Для процесу реалізації застосунку обрано інтегроване середовище Visual Studio Code. Даний редактор є кросплатформним, містить вбудований термінал і

засоби налагодження, які дають змогу запускати сервер на Node.js, клієнтську частину та сервіс машинного навчання на Python безпосередньо з інтегрованого середовища розробки, встановлювати точки зупину, аналізувати стек викликів і значення змінних. Такі можливості зменшують потребу в перемиканні між різними інструментами та підвищують продуктивність роботи розробника.

Важливою перевагою Visual Studio Code є розвинена система розширень. За допомогою відповідних модулів забезпечено автоматичне форматування та статичний аналіз коду: для JavaScript і TypeScript застосовуються інструменти ESLint і Prettier, для Python використовується офіційне розширення з підтримкою засобів аналізу якості коду та форматерів. Це дає змогу підтримувати єдиний стиль написання коду, своєчасно виявляти типові помилки на етапі розробки та підвищувати загальну якість програмного продукту.

Як систему керування версіями у проєкті використано Git, а для віддаленого зберігання репозиторію – платформу GitHub. Git реалізує розподілену модель, за якої повна історія змін зберігається локально на кожному робочому місці; це дає змогу створювати гілки для нових функцій, виправлень і експериментів, а також за потреби повертатися до попередніх версій коду. Докладна історія фіксацій полегшує аналіз еволюції програмного забезпечення й пошук джерел помилок. Платформа GitHub виконує роль віддаленого сховища та середовища колективної розробки, надаючи засоби для створення запитів на злиття, організації експертної перевірки коду, постановки та відстеження завдань, ведення супровідної документації, а також налаштування процесів неперервної інтеграції та постачання [40].

Інтеграція з системою керування версіями Git реалізована безпосередньо в інтерфейсі редактора Visual Studio Code, що забезпечує виконання фіксації змін, перегляд історії модифікацій та розв’язання конфліктів об’єднання гілок без використання додаткових програмних засобів.

3.7 Висновки до розділу 3

У даному розділі обґрунтовано вибір технологічного стеку для реалізації веб-орієнтованого застосунку з модулем машинного навчання. Спільне використання React, Node.js та Python забезпечує кросплатформеність, узгоджене подання даних і спрощує інтеграцію між клієнтською, серверною частинами, а також сервісом передбачення. Менеджер пакунків npm, збірник Vite, а також бібліотеки Recharts та Axios формують основу для побудови інтерактивного інтерфейсу, візуалізації результатів і надійної мережевої взаємодії.

Серверна підсистема на базі Express, Mongoose, Multer, csv-parser, bcrypt, JWT і dotenv забезпечує гнучку організацію API, формалізований опис структури даних, безпечну автентифікацію, обробку завантажуваних файлів та централізоване керування конфігурацією. Документоорієнтована система керування базами даних MongoDB обрана завдяки гнучкій схемі даних, зручній інтеграції з Node.js та вбудованим засобам масштабування та реплікації, що є важливим для роботи з даними різної структури.

Модуль машинного навчання реалізовано із використанням FastAPI, Scikit-learn, NumPy та joblib, що забезпечує уніфікований інтерфейс навчання й оцінювання моделей, ефективну обробку числових даних і повторне використання створених моделей без додаткових витрат на повторне навчання. Застосування Visual Studio Code, Git і GitHub підтримує структурований процес розробки і налагодження.

Обраний технологічний стек забезпечує узгоджену взаємодію між клієнтською частиною, серверним API та модулем машинного навчання, спрощує інтеграцію нових компонентів, а також створює умови для масштабування та подальшого розширення веб-орієнтованого застосунку.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

У даному розділі подано узагальнений опис програмної реалізації веб-орієнтованої системи для обробки, аналізу та передбачення погодних даних. Наведено функціональні вимоги та архітектуру застосунку, реалізовану за багаторівневим клієнт-серверним підходом із окремим модулем машинного навчання. Розглянуто реалізацію серверної частини на Node.js та Express.js, модуля машинного навчання на основі FastAPI і Scikit-learn, а також клієнтського інтерфейсу, створеного за допомогою React. Подано структуру бази даних MongoDB, що забезпечує зберігання користувацьких даних і погодних записів.

4.1 Визначення функціональних вимог до системи

Сформуємо вимоги користувача до системи, тобто перерахуємо основні можливості, які повинен надавати програмний продукт. На рисунку 4.1 наведено повний перелік функціональних вимог до програмної системи у вигляді діаграми прецедентів. До складу даних вимог входять наступні групи прецедентів:

– керування погодними записами, яке передбачає можливість створення, редагування, видалення та перегляду погодних записів. Даний функціонал може бути розширений опціональними можливостями імпорту набору даних з бази даних у форматі CSV-файлу та завантаження первинного CSV-файлу;

– аналіз погодних даних, у межах якого користувач може переглядати перші п’ять рядків набору даних, основні статистичні описові характеристики та розподіл значень параметрів. Додатково можуть бути доступні перегляд кореляційної матриці та графіків залежності параметрів від дати;

– передбачення значень, яке включає вибір цільової змінної, моделі прогнозування, налаштування параметрів моделі та перегляд метрик якості створеної моделі. Виконання передбачення можливе після попереднього вибору погодного запису;

– робота з обліковими записами, яка охоплює створення нового облікового запису та авторизацію користувача шляхом входу до системи;

– вибір погодного запису, що є обов'язковим кроком для подальшого аналізу або передбачення даних конкретного набору.

Зазначені прецеденти з відповідними зв'язками відображають функціональні можливості системи, що дає змогу користувачу працювати з погодними записами, здійснювати їх аналіз, виконувати передбачення обраних параметрів, а також ефективно взаємодіяти з наборами даних у форматі CSV.

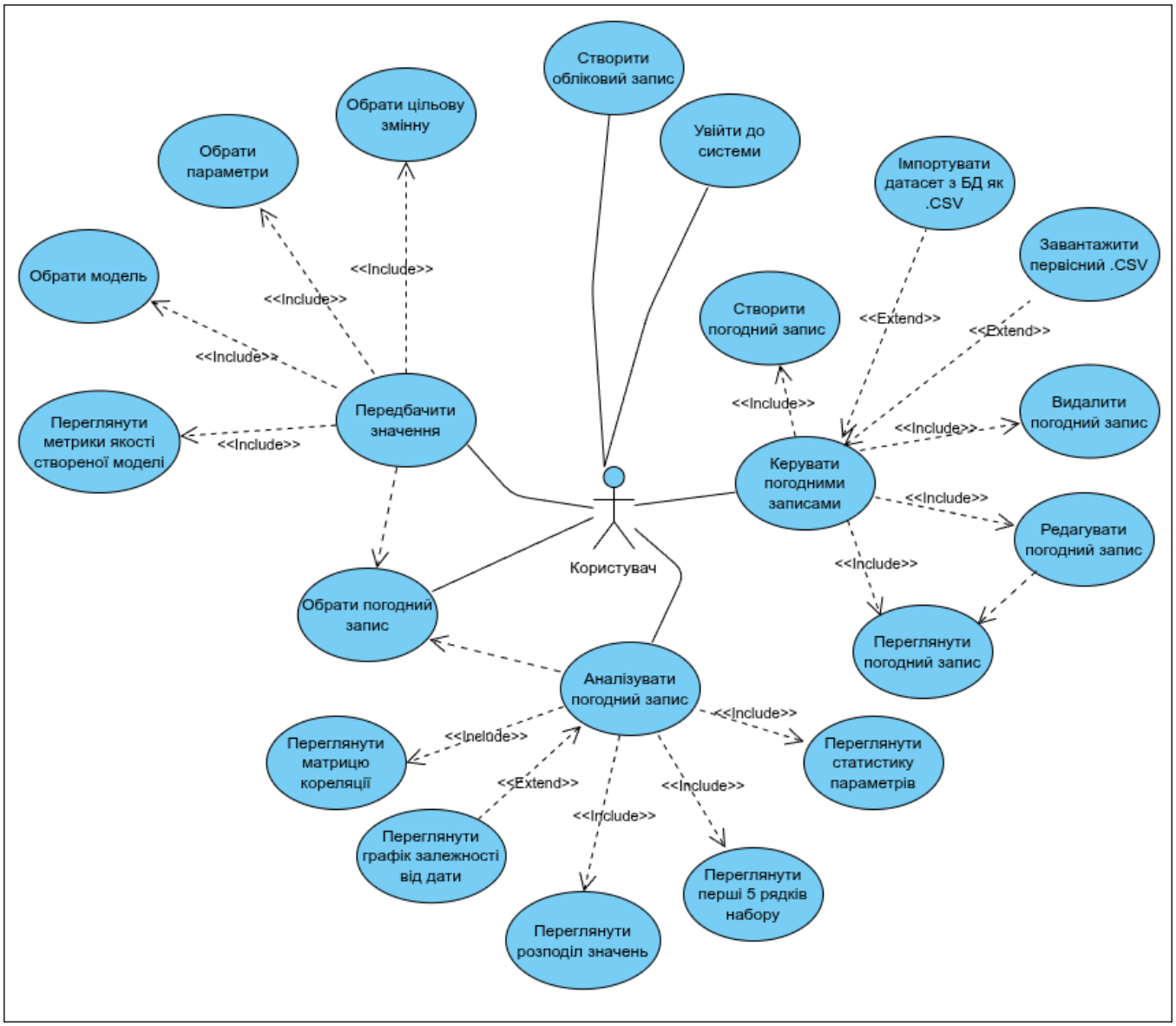


Рисунок 4.1 – Діаграма прецедентів

4.2 Загальна архітектура програмної системи

Програмна система реалізована за принципами багаторівневої клієнт-серверної архітектури з виокремленим модулем машинного навчання, що функціонує як автономний сервіс. Архітектура включає три основні компоненти: клієнтський веб-інтерфейс, сервер прикладної логіки та окремий обчислювальний модуль машинного навчання.

Клієнтська частина представлена односторінковим веб-застосунком, розробленим на основі бібліотеки React. Вона забезпечує відображення користувацького інтерфейсу, формування запитів до серверної частини, інтерактивний перегляд статистичних характеристик даних, а також візуалізацію результатів передбачення. Взаємодія між клієнтом та сервером здійснюється через REST-інтерфейс із використанням формату JSON та протоколу HTTP.

Серверна частина реалізована на платформі Node.js із застосуванням фреймворку Express.js. Вона виконує функції маршрутизації запитів, забезпечує автентифікацію та авторизацію користувачів за допомогою JWT-токенів, здійснює обробку CSV-файлів, управління користувацькими даними та надання аналітичних сервісів: обчислення статистичних показників, розподілів, кореляційних матриць, часових рядів тощо. Доступ до сховища даних реалізовано через ODM-бібліотеку Mongoose, що інкапсулює взаємодію з документно-орієнтованою СКБД MongoDB та дозволяє визначати структури сутностей у вигляді схем.

Модуль машинного навчання реалізований як окремий веб-сервіс, розгорнутий на основі фреймворку FastAPI та мови програмування Python. Він виконує попередню обробку даних, навчання моделей машинного навчання, зокрема логістичної регресії, градієнтного бустингу та багатошарового персептрона, з використанням бібліотеки Scikit-learn. Також, сервіс забезпечує збереження навчених моделей і формування передбачень. Взаємодія між сервером прикладної логіки та модулем машинного навчання здійснюється через REST-API

та забезпечує можливість фізичного відокремлення даного модуля, його незалежного масштабування й розгортання на окремих обчислювальних вузлах.

Концептуальну схему взаємодії основних компонентів програмної системи наведено на рисунку 4.2.

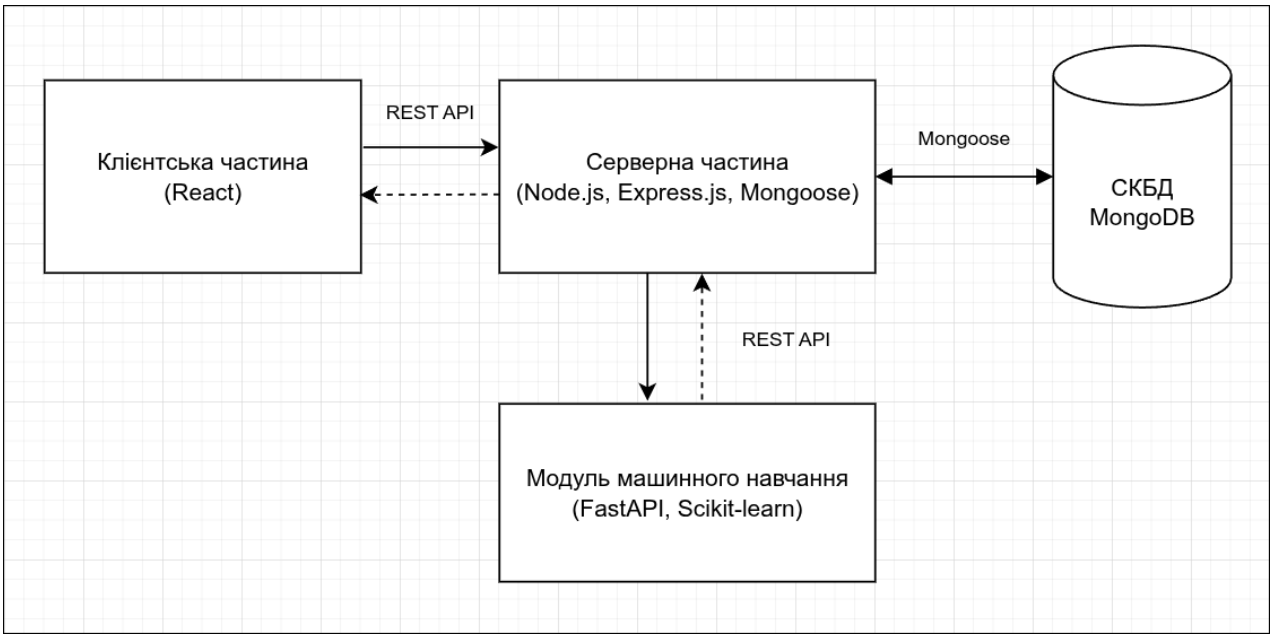


Рисунок 4.2 – Концептуальна діаграма архітектури програмної системи

Також, система використовує СКБД MongoDB для довгострокового зберігання облікових даних користувачів, метаданих завантажених CSV-файлів та самих табличних записів, що імпортуються з цих файлів. Документно-орієнтована модель СКБД забезпечує можливість гнучкого опису як структур користувачів, так і динамічних таблиць із супровідними параметрами.

4.3 Реалізації серверної частини

Серверна частина програмної системи реалізована на платформі Node.js із використанням веб-фреймворку Express.js та бібліотеки Mongoose для взаємодії з СКБД MongoDB. Логіка серверної компоненти впорядкована за функціональними

модулями, серед яких: конфігураційні файли, контролери, проміжне програмне забезпечення (middleware), моделі опису схем даних, допоміжні бібліотеки та маршрутизатори. Подібна модульна структура, організацію якої наведено на рисунку 4.3, забезпечує керованість, масштабованість та полегшує подальший розвиток системи.

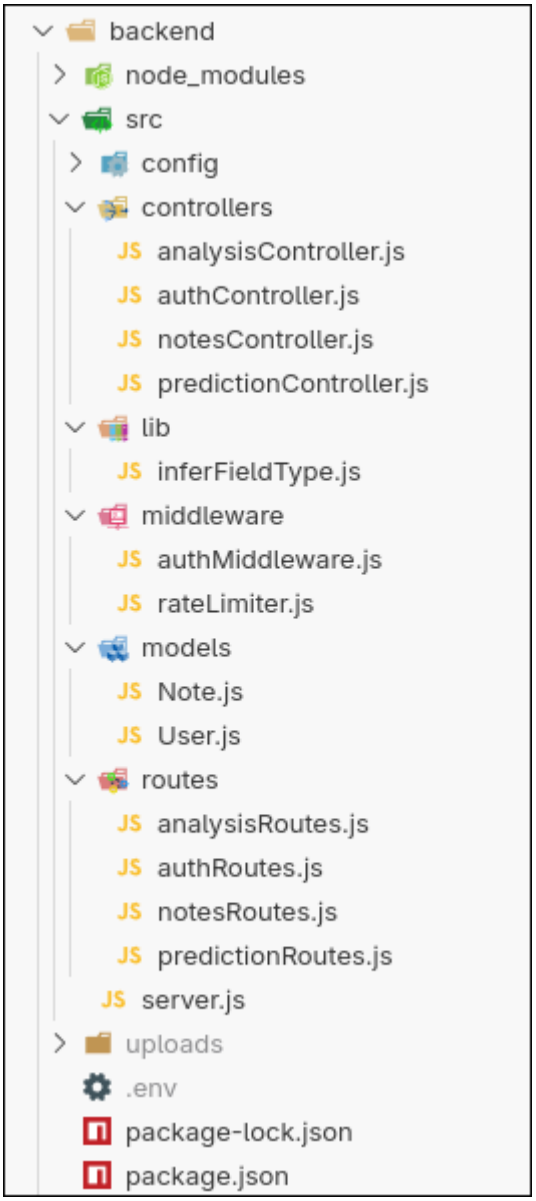


Рисунок 4.3 – Структура директорій серверної частини програмної системи

Конфігураційний каталог містить модуль підключення до MongoDB, який відповідає за встановлення з’єднання зі СКБД на основі параметрів середовища. У

головному модулі серверної частини зчитуються дані параметри, викликається функція підключення до бази даних, а після успішного встановлення з’єднання створюється екземпляр застосунку Express, підключається проміжне програмне забезпечення для обробки повідомлень JSON та налаштовується політика кросдоменного доступу CORS, що забезпечує взаємодію клієнтської частини з API.

Каталог авторизації містить middleware, яке забезпечує перевірку доступу до захищених ресурсів. Він аналізує коректність та валідність маркера доступу JWT, переданого в заголовку авторизації або у параметрах запиту. У разі успішної верифікації з токена береться ідентифікатор користувача, що надалі використовується контролерами для доступу до відповідних ресурсів. Якщо маркер відсутній або недійсний, сервер повертає помилку авторизації.

Функціональна логіка обробки запитів HTTP реалізована у спеціалізованих контролерах. Контролер управління обліковими записами забезпечує виконання операцій реєстрації та аутентифікації користувачів. У процесі реєстрації здійснюється перевірка повноти переданих даних, відсутності користувача з таким самим логіном або електронною поштою, а також хешування пароля. Під час автентифікації валідність облікових даних перевіряється шляхом порівняння хешованого пароля, після чого генерується обмежений у часі маркер доступу, який використовується клієнтом для запитів до захищених кінцевих точок.

Окремий контролер відповідає за керування погодними записами а підтримує наступні операції:

– отримання списку записів користувача без завантаження повного вмісту набору даних;

– отримання окремого запису з повним вмістом набору даних;

– створення нового запису на основі завантаженого файлу формату CSV;

– редагування наявних даних;

– видалення записів.

Крім базових CRUD-операцій, контролер також надає функції отримання списків унікальних текстових значень та діапазонів числових параметрів для

обраного набору даних. Згадані сервіси використовуються під час налаштування задач передбачення.

Під час створення або оновлення запису завантажений CSV-файл розбирається на табличні рядки, для кожного стовпця визначаються типи полів, після чого значення автоматично перетворюються у відповідні формати. За потреби вихідний файл зберігається у спеціальний директорії, а під час редагування чи видалення запису пов’язані з ним файли вилучаються, аби запобігти накопиченню застарілих даних. Також, контролер забезпечує можливість передавання як початкових, так і модифікованих CSV-файлів у відповідь на запит користувача.

Підсистема аналітичної обробки даних реалізована у спеціалізованому контролері, який забезпечує обчислення основних описових статистичних характеристик для вибраного погодного запису. Підтримуються наступні операції:

– отримання базових описових статистик (мінімум, максимум, середнє, стандартне відхилення, квартилі, кількість пропусків для числових полів);

– визначення кількості унікальних значень, моди та пропусків для текстових полів;

– визначення часових діапазонів значень дати;

– обчислення параметрів гістограм і частотних розподілів значень;

– формування кореляційної матриці між числовими параметрами;

– формування даних для побудови графіків часових рядів;

– отримання вибірки перших рядків таблиці для попереднього аналізу структури даних.

Для взаємодії з модулем машинного навчання передбачено окремий контролер, який приймає від клієнта параметри задачі передбачення: цільову змінну, набір ознак, значення для прогнозу. На основі ідентифікатора запису завантажуються дані відповідного набору, формується запит до зовнішнього сервісу машинного навчання та передаються дані разом з параметрами моделі. У відповідь сервер отримує передбачення та супутню інформацію, зокрема метрики

якості моделі, і без додаткової обробки передає їх клієнтській частині для подальшої візуалізації.

REST-інтерфейс системи структуровано у групи маршрутів відповідно до функціональних підсистем: управління обліковими записами, роботи з погодними даними та файлами, аналітичної обробки даних, а також ініціювання задач передбачення. Маршрути, що працюють з користувацькими даними, підключаються з використанням авторизаційного middleware, а контролери додатково перевіряють належність записів поточному користувачу.

Реалізована архітектура забезпечує чітке розмежування обов’язків між рівнями зберігання даних, контролерами, проміжним програмним забезпеченням та маршрутизаторами, що дозволяє досягти масштабованої й гнучкої обробки погодних даних та ефективної інтеграції з модулем машинного навчання.

4.4 Модуль машинного навчання

Модуль машинного навчання реалізований як окремий веб-сервіс на основі фреймворку FastAPI та мови програмування Python. Він розгортається окремо від серверної частини на Node.js та за потреби взаємодіє з нею через REST-інтерфейс. У застосунку FastAPI оголошено три кінцеві точки для виконання передбачення з використанням різних моделей машинного навчання: логістичної регресії, градієнтного бустингу та багатошарового персептрона. Формат вхідних даних описано за допомогою моделі Pydantic, яка визначає структуру запиту: ідентифікатор погодного запису, назву цільової змінної, перелік ознак, словник вхідних значень для передбачення та масив даних набору.

Після отримання запиту модуль завантажує дані погодного запису з тіла запиту та виконує попередню обробку: відбір вказаних ознак, розподіл полів на числові та категоріальні, приведення значень до числового представлення та формування навчальної вибірки. Для логістичної регресії та багатошарового персептрона категоріальні ознаки кодуються у форматі one-hot. У моделях

градієнтного бустингу категорії кодуються цілочисельними індексами. У класифікаційних задачах цільова змінна кодується мітками.

Реалізація логістичної регресії орієнтована на розв’язання задач класифікації: після нормалізації числових ознак і one-hot кодування категоріальних формується матриця ознак, на основі якої навчається модель та оцінюється за стандартними метриками. Для градієнтного бустингу та багатошарового персептрона модуль автоматично обирає режим класифікації або регресії залежно від типу цільової змінної. Нормалізація числових ознак за допомогою стандартного масштабування є спільною для всіх алгоритмів, тоді як кодування категоріальних ознак виконується відповідно до вимог конкретної моделі. Усі методи використовуються з фіксованими, наперед визначеними гіперпараметрами для кожного випадку передбачення.

Для зменшення витрат на повторне навчання реалізовано механізм збереження моделей. Для кожної задачі передбачення формується унікальний ідентифікатор, що враховує позначення набору даних, назву цільової змінної, перелік ознак та, за потреби, тип цільової змінної. На основі цього ідентифікатора у файловій системі створюється запис, у якому з використанням бібліотеки Joblib зберігається навчена модель разом із параметрами нормалізації, схемами кодування ознак і відображенням між початковими значеннями цільової змінної та їхніми індексами. У разі повторних звернень із тими самими параметрами модуль завантажує модель з диска без повторного навчання.

Формування прогнозу для заданого набору вхідних значень виконується за узгодженою схемою. На основі переліку ознак з тіла запиту беруться відповідні значення, до них застосовуються ті самі перетворення, що й під час навчання: масштабування числових полів, кодування категоріальних, після чого формується вектор ознак для одного об’єкта. Даний вектор подається на вхід навченої моделі, яка повертає числове значення передбачення або індекс класу. У випадку класифікації, індекс переводиться назад у відповідне категоріальне значення за допомогою зворотного відображення. У відповіді повертаються передбачене значення та метрики якості моделі.

4.5 Реалізація клієнтської частини

Клієнтська частина програмної системи реалізована як односторінковий веб-застосунок, розроблений на основі бібліотеки React. Основним призначенням клієнта є забезпечення користувацького інтерфейсу, керування навігацією між функціональними модулями системи та здійснення взаємодії з серверною частиною через REST-API. Обмін даними реалізовано у форматі JSON, що забезпечує уніфіковане представлення переданих структур.

Архітектура клієнтського застосунку реалізована за компонентним підходом. Окремі компоненти відповідають за виконання таких функцій, як автентифікація та реєстрація користувачів, перегляд і керування погодними записами, завантаження та редагування CSV-файлів, відображення статистичних характеристик, кореляційних матриць та часових рядів, а також налаштування параметрів моделей машинного навчання. Навігація між модулями інтерфейсу здійснюється за допомогою механізму маршрутизації на стороні клієнта, що забезпечує зміну вмісту без перезавантаження сторінки.

Для доступу до захищених ресурсів застосунок використовує механізм автентифікації на основі JWT-токенів. Після успішного входу маркер зберігається на клієнті й автоматично додається до HTTP-запитів, що надсилаються до серверної частини. Клієнтська логіка обробляє типові помилки авторизації, зокрема випадки завершення строку дії токена.

Функціональні компоненти інтерфейсу забезпечують завантаження даних на сервер, перегляд структури наборів, відображення ключових статистичних показників у табличному та графічному вигляді, а також формування параметрів задач передбачення. Отримані від серверної частини та модуля машинного навчання результати прогнозування та відповідні метрики якості візуалізуються відповідним чином.

Клієнтська частина виконує роль інтерфейсного рівня системи, забезпечуючи засоби для завантаження та перегляду даних, а також отримання результатів аналізу і передбачення, сформованих серверною компонентою.

4.6 Опис структури бази даних

Сховище даних програмної системи реалізовано на основі документоорієнтованої СКБД MongoDB. Структуру бази даних визначено за допомогою схем Mongoose, які задають формат документів для основних сутностей системи, тобто користувачів та погодних записів. На рисунку 4.4 наведено концептуальну UML-модель бази даних, що демонструє взаємозв'язок між сутностями User та Note, їхні основні атрибути, а також використання вкладених структур для зберігання табличних даних і метаданих файлів у документах MongoDB.

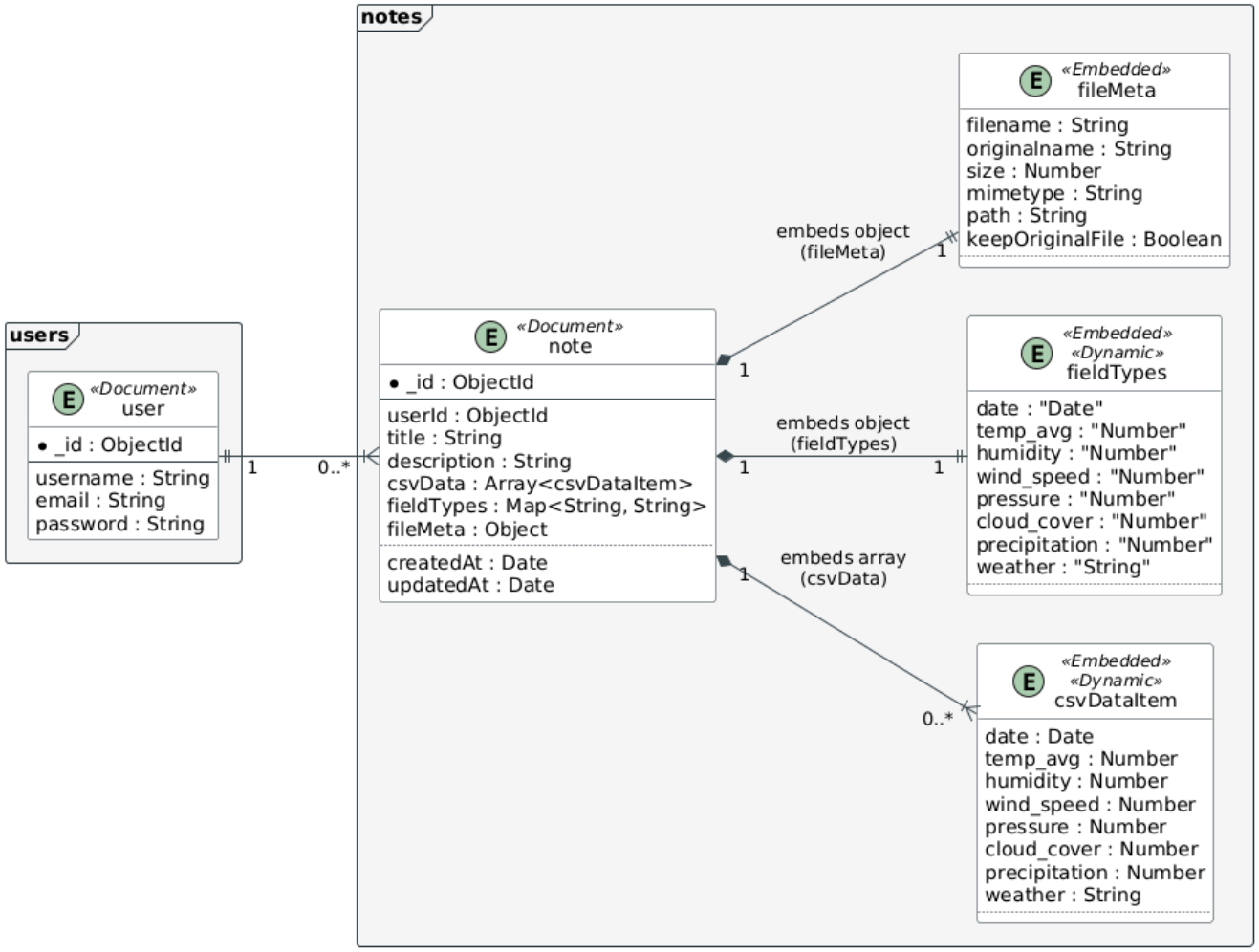


Рисунок 4.4 – Концептуальна UML-модель бази даних

Сутність User містить облікові дані, необхідні для автентифікації та ідентифікації в системі: унікальне ім’я користувача (username), адресу електронної пошти (email) та хешований пароль (password). Для полів username та email встановлено обмеження унікальності, що унеможливлює створення дублікованих облікових записів [41, 42].

Погодні дані представлені сутністю Note, яка пов’язана з користувачем через поле userId типу ObjectId з посиланням на модель User. Це забезпечує логічне групування погодних записів за власниками. До основних атрибутів Note належать: назва запису (title), текстовий опис (description), дані з набору погодного запису (csvData), а також метадані файлу [42].

Поле csvData реалізовано як масив змішаних типів, що дає змогу зберігати рядки таблиці довільної структури у вигляді набору пар “назва поля – значення”. Типи стовпців фіксуються окремо у властивості fieldTypes, представленій асоціативним словником “назва атрибута – тип даних”. Ця інформація використовується під час аналітичної обробки даних та налаштування задач передбачення [41, 42].

Для зберігання інформації про вихідний CSV-файл використовується вкладений об’єкт fileMeta, який містить системне ім’я файла (filename), початкову назву (originalname), шлях до файла на диску (path), розмір (size), MIME-тип вмісту (mimetype), а також логічний прапорець keepOriginalFile, що визначає стан опції довгострокового зберігання оригінального CSV-файлу [42].

Для сутності Note активовано автоматичне ведення службових полів createdAt та updatedAt, які фіксують час створення та останнього оновлення документа.

4.7 Висновки до розділу 4

У розділі наведено узагальнений опис програмної реалізації веб-орієнтованої системи для аналізу та передбачення погодних даних.

Сформульовано функціональні вимоги, що охоплюють керування погодними записами, аналітичне опрацювання даних, підтримку імпорту та експорту CSV-файлів, авторизацію користувачів, а також інтеграцію з модулем машинного навчання.

Архітектурна модель системи побудована за багаторівневим клієнт-серверним принципом з окремим модулем машинного навчання. Серверна частина на базі Node.js забезпечує маршрутизацію запитів, аутентифікацію користувачів, обробку структурованих даних і взаємодію з обчислювальним сервісом машинного навчання. Модуль прогнозування, реалізований на FastAPI, виконує підготовку вибірки, навчання моделей та формування передбачень із можливістю повторного використання збережених моделей, що оптимізує обчислювальні витрати. Усі алгоритми машинного навчання в модулі застосовуються з фіксованими, наперед визначеними гіперпараметрами. Клієнтська частина на основі React забезпечує відображення інтерфейсу, візуалізацію статистичних характеристик та надає доступ до параметризації задач передбачення.

Структура бази даних реалізована з використанням MongoDB та схем Mongoose, що забезпечує гнучке зберігання наборів структурованих даних. Логічні зв’язки між сутностями користувачів і погодних записів сприяють впорядкованій організації та цілісності даних.

Таким чином, система реалізована як модульне та масштабоване програмне рішення, що підтримує повний цикл роботи з погодними даними: від завантаження та базового дослідницького аналізу до формування передбачень у єдиному інтегрованому середовищі.

5 ВЗАЄМОДІЯ КОРИСТУВАЧА З ПРОГРАМНИМ ПРОДУКТОМ

Даний розділ містить перелік рекомендованих системних вимог до комп’ютера користувача, на якому передбачається виконання розробленого програмного продукту. Також розглядаються правила та поради, щодо використання програмного забезпечення, яких варто дотримуватися, з метою коректної роботи застосунку.

5.1 Рекомендовані системні вимоги

З метою забезпечення оптимальної роботи розробленого програмного продукту, рекомендується використовувати пристрій із технічними характеристикам не нижчими за наступні:

– операційна система Linux (наприклад, Ubuntu 20.04) або Windows 10 / 11;

– центральний процесор 6-ядерний x64 2,5 ГГц (Intel Core i5 / Ryzen 5);

– 8 Гб ОЗП;

– 10 Гб вільного місця на диску.

Також, для коректної роботи застосунку, необхідно попередньо встановити такі компоненти:

– Node.js (версія 20 або новіша) та npm;

– MongoDB (версія 6.0.26 або новіша);

– Python (версія 3.10 або новіша).

Вимоги до встановлених бібліотек та фреймворків:

– бекенд (Node.js): express, mongoose, jsonwebtoken, bcryptjs, axios, dotenv, cors, multer, csv-parser, csv-writer, date-fns, simple-statistics, uuid;

– фронтенд (React): vite, react, react-dom, axios, react-router, recharts, tailwindcss, lucide-react, react-hot-toast, daisyui;

– Python-сервіс: fastapi, uvicorn, pydantic, requests, scikit-learn, numpy, joblib.

5.2 Опис функціональних можливостей та варіантів взаємодії

Для початку роботи з програмним застосунком користувачу слід, за потреби, створити обліковий запис та увійти до системи. Після успішного входу в обліковий запис, користувач може взаємодіяти з навігаційним меню, розміщеним у верхній частині екрану, та елементами головної сторінки, що продемонстровано на рисунку 5.1.

Панель навігації дозволяє дізнатися логін поточного облікового запису та вийти з нього, а також переходити між сторінками: головною, аналітики та передбачення.

Елементи головної сторінки включають панель інструментів для сортування та пошуку погодних записів, сітку карток із такими записами (по шість на одну сторінку) та систему пагінації для переходу між сторінками погодних записів.

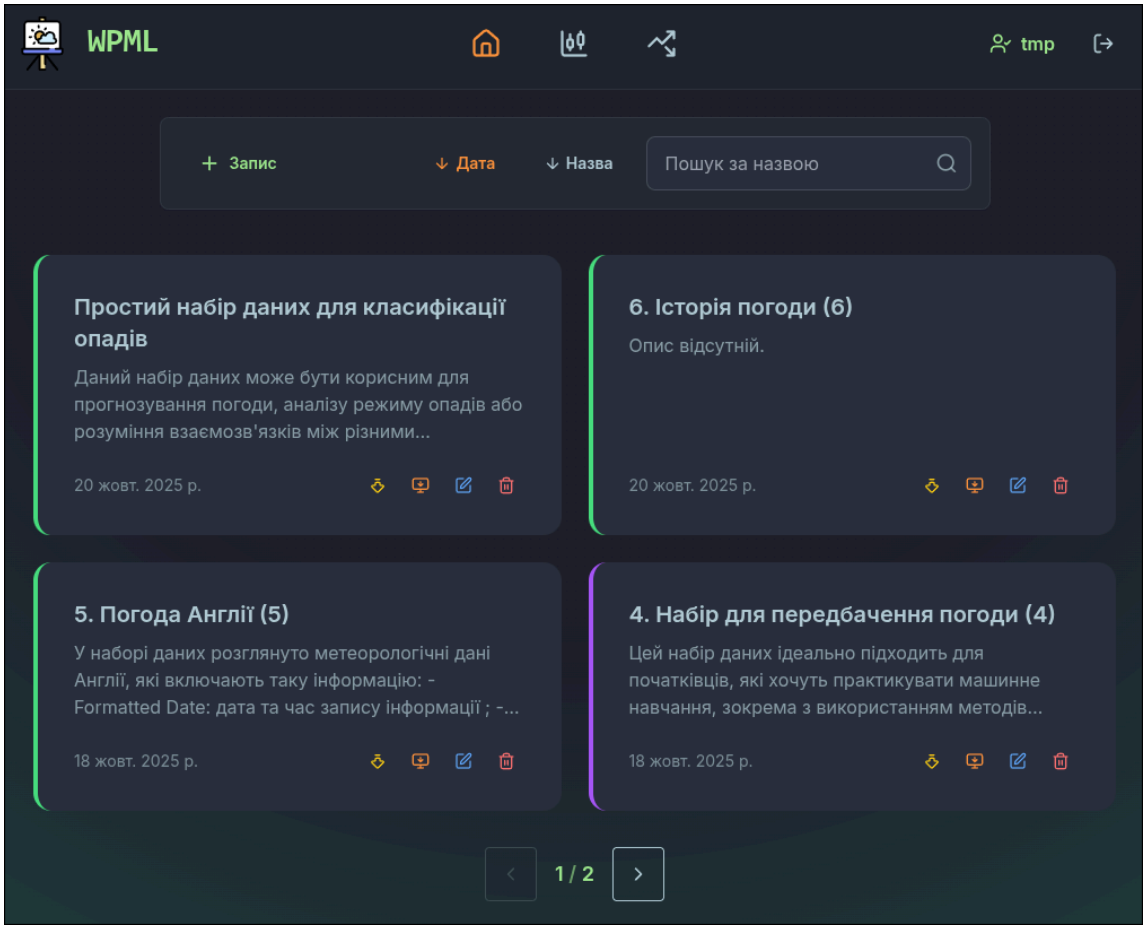


Рисунок 5.1 – Головна сторінка застосунку

Кожна картка погодного запису містить назву та короткий опис запису, дату його створення, а також кнопки для редагування, видалення та завантаження файлів. Кнопка завантаження оригінального документу дозволяє зберегти файл CSV у його первісному вигляді, тоді як кнопка завантаження опрацьованого файлу надає можливість збереження даних після обробки та внесення до бази даних. Картка реагує на виділення користувачем, змінюючи стиль рамки з зеленого на фіолетовий, і дозволяє вибрати запис для подальших дій.

При виборі опції редагування погодного запису (рисунок 5.2) користувач може змінити його назву та опис, а також додати новий файл формату CSV для заміни існуючого.

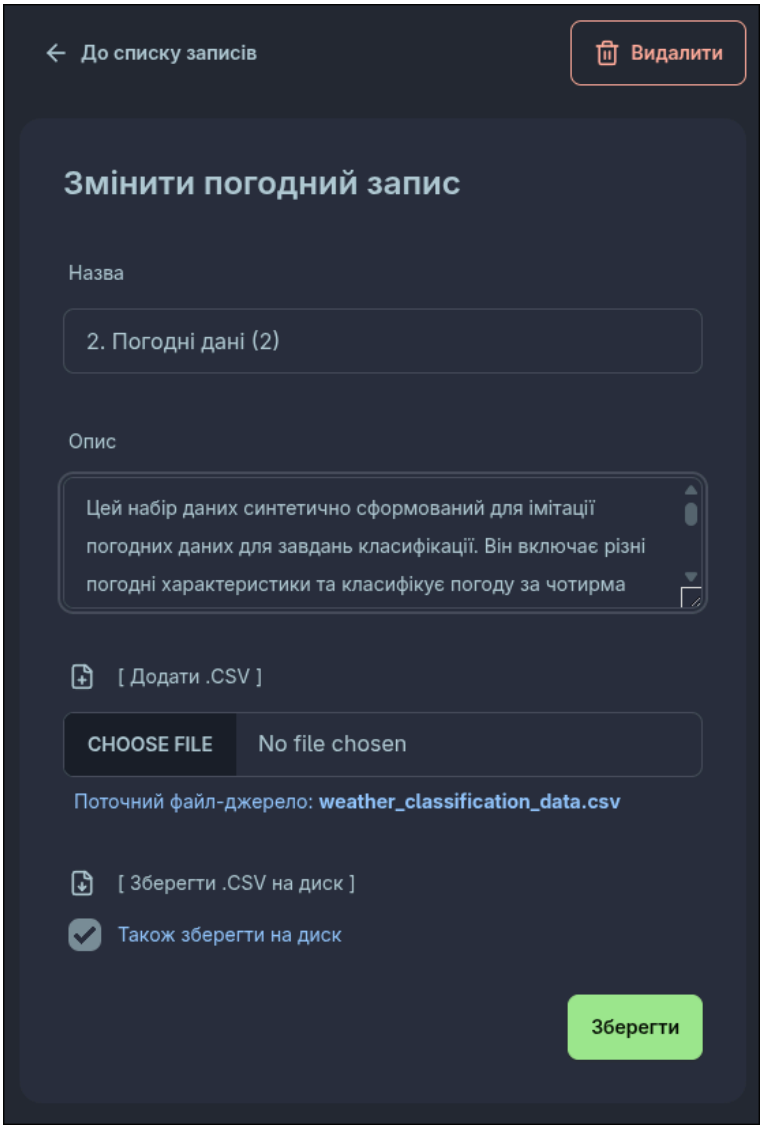


Рисунок 5.2 – Приклад редагування погодного запису

При додаванні нового файлу, доступна опція його збереження на диск: документ можна зберегти як окремий оригінал на сервері або лише імпортувати дані з нього у базу даних. Інтерфейс відображає поточний файл-джерело та обраний новий файл, а кнопки дозволяють зберегти зміни або видалити запис.

При створенні нового запису використовуються подібний інтерфейс, але кнопка видалення відсутня, оскільки запис ще не існує в системі.

Для сортування та пошуку погодних записів користувач може скористатися панеллю інструментів, розташованою над сіткою карток. За її допомогою доступне сортування записів за датою (від найновіших до найстаріших та навпаки) і за назвою (від А до Я та від Я до А), а також пошук записів за ключовими словами у назві. Кнопки сортування відображають поточний напрямок сортування, а поле пошуку дозволяє швидко відфільтрувати записи за введеним текстом.

На сторінці аналізу (рисунок 5.3) користувач може здійснити огляд даних обраного погодного запису. Автоматично завантажуються метадані запису, включно з назвами полів та їх типами. Для отримання візуалізованих даних необхідно вибрати відповідну опцію перегляду; поточна опція підсвічується оранжевим кольором. Наприклад, користувач може відобразити перші п’ять рядків у табличному форматі. Кількість видимих колонок залежить від ширини екрану: на великих екранах відображаються одночасно три колонки, на менших – дві або одна. Для перегляду всіх записів використовується пагінація. Поточна сторінка пагінації відображається між кнопками навігації, що дозволяє зручно рухатися вперед або назад по всіх записах. Крім того, користувач може переглянути перші п’ять рядків у форматі JSON.

Доступне відображення матриці кореляції для числових параметрів (рисунок 5.3), яке демонструє взаємозв’язки між усіма числовими змінними. Це дозволяє швидко оцінити наявність сильних кореляцій або аномалій і попередньо визначитися з методом машинного навчання для передбачення.

Для числових змінних доступна діаграма розподілу (рисунок 5.4), яка відображає частоту значень у вигляді гістограми з десятьма рівними інтервалами.

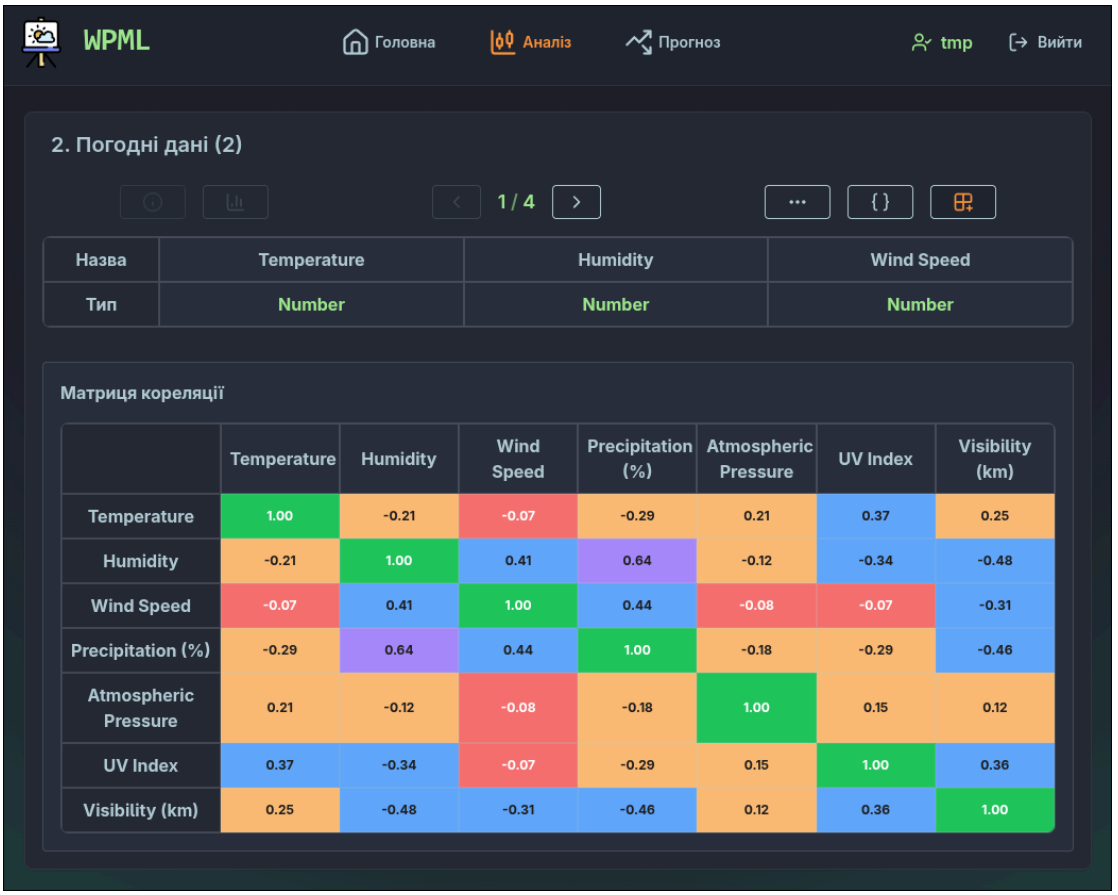


Рисунок 5.3 – Кореляційна матриця обраного набору даних

Рисунок 5.4 – Гістограма розподілу значень обраного числового параметра

Для рядкових змінних також доступна діаграма розподілу значень (рисунок 5.5), яка відображає частоту повторень кожної категорії.

Рисунок 5.5 – Діаграма розподілу значень рядкового параметра

Рисунок 5.6 – Коротка статистика обраного числового параметра

Для числових, рядкових та змінних типу дати доступна коротка статистика (рисунок 5.6). Для числових змінних статистика включає мінімум, максимум, середнє, стандартне відхилення, квантилі, кількість спостережень та пропусків; для рядкових – кількість категорій, моду з частотою, кількість спостережень та пропусків; для змінних типу дати – мінімальну та максимальну дату, кількість спостережень та пропусків.

Якщо у наборі даних присутнє поле типу дати, користувач може відобразити графік залежності обраної числової змінної від дати (рисунок 5.7), що дозволяє оцінити динаміку змін значень у часі.

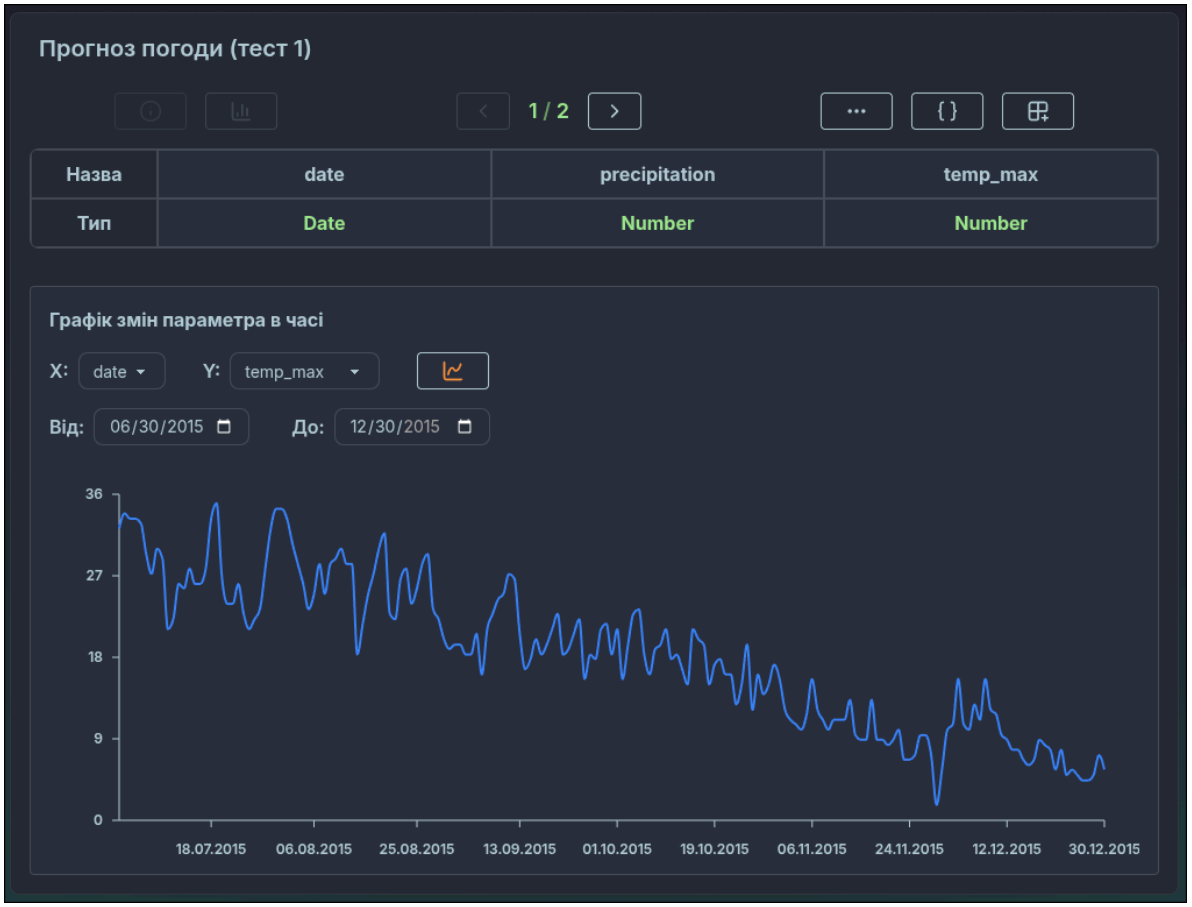


Рисунок 5.7 – Відображення графіка залежності

Сторінка передбачення (рисунок 5.8) забезпечує виконання прогнозування на основі обраного погодного запису. Якщо запис не обрано, користувач отримує

повідомлення про необхідність вибору даних на головній сторінці. Після вибору запису автоматично завантажується його ідентифікатор для подальших операцій.

Інтерфейс містить форму для вибору цільової змінної, вхідних параметрів та типу моделі машинного навчання. Користувач може обрати одну з моделей: лінійну регресію, градієнтний бустинг або MLP. Система застосовує оптимальні параметри моделі за замовчуванням, що спрощує процес передбачення.

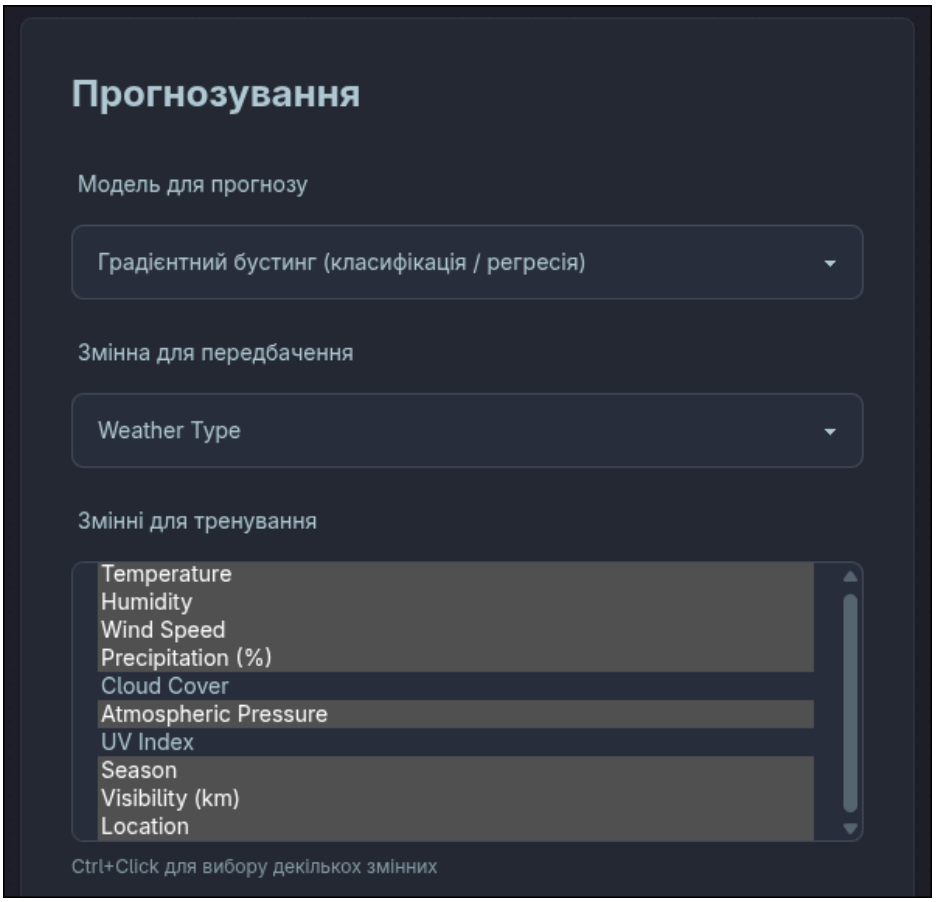


Рисунок 5.8 – Вибір параметрів моделі

При заповненні форми передбачення (рисунок 5.9) для числових змінних відображаються межі допустимих значень, а для рядкових – вибір із випадаючого списку на основі даних набору. Після натискання кнопки “Отримати прогноз” запускається обчислення на сервері, де система враховує вибрані дані, навчає модель і виконує передбачення.

Отримані результати (рисунок 5.10) повертаються у вигляді числових або категоріальних значень і відображаються в інтерфейсі разом із показниками якості моделі, що дозволяє оцінити точність прогнозу.

Рисунок 5.9 – Вибір значень ознак для передбачення

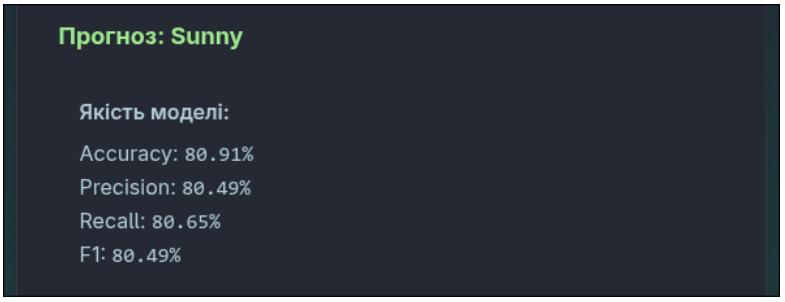


Рисунок 5.10 – Результат вирішення задачі класифікації

5.3 Висновки до розділу 5

У даному розділі розглянуто принципи взаємодії користувача з веб-орієнтованим застосунком для аналізу та передбачення погодних даних. Окреслено системні вимоги до апаратного та програмного забезпечення, зокрема сучасні версії Node.js, MongoDB, Python та набори бібліотек, необхідні для стабільної роботи всіх модулів системи – від обробки даних до формування статистики і передбачень.

Описано логіку використання інтерфейсу: створення облікового запису, роботу з головною сторінкою, де реалізовано сортування, пошук, редагування та завантаження погодних записів. Динамічні картки дозволяють швидко виконувати базові операції та переглядати ключову інформацію.

Сторінка аналізу надає можливості для базового дослідження набору даних: попередній перегляд набору у табличному та JSON форматах, кореляційні матриці, гістограми, діаграми розподілу та коротку статистику для числових, рядкових і датованих параметрів. Це забезпечує проведення базового дослідницького аналізу без додаткових інструментів.

Сторінка прогнозування дозволяє обрати модель машинного навчання, цільову змінну та вхідні параметри. Система автоматично навчає модель і повертає передбачене значення разом із відповідними метриками якості, що дає користувачу можливість оцінити точність отриманих результатів.

Програмна система забезпечує послідовний процес роботи: від завантаження даних та їх початкового аналізу до отримання передбачень. Програмне забезпечення дозволяє користувачам без спеціальних технічних навичок застосовувати базові методи машинного навчання для передбачення погодних параметрів.

6 РОЗРОБКА СТАРТАП-ПРОЄКТУ

У даному розділі представлено обґрунтування можливостей практичного використання розробленого веб-орієнтованого застосунку для аналізу та передбачення погодних даних, оцінено його потенційну цінність у ринковому середовищі, проаналізовано конкурентні позиції, визначено ринкову та маркетингову стратегії. Розроблений програмний продукт розглядається як стартап-рішення у сфері програмних засобів для базового аналізу та передбачення погодних даних, що поєднує методи машинного навчання та простоту використання через веб-інтерфейс.

У підготовці стартап-проєкту, зокрема для оформлення таблиць, частково використано окремі приклади та рекомендації з методичного посібника [43].

6.1 Опис ідеї проєкту

Зростання ролі погодної аналітики посилює потребу в доступних інструментах для роботи з кліматичними даними та швидкого отримання передбачень, а також в освітньо-демонстраційних засобах, що знайомлять із базовими методами машинного навчання, зокрема на основі погодних процесів.

Попри наявність інтерактивних обчислювальних середовищ як Google Colab та інструментів автоматизованого машинного навчання, зокрема Auto-sklearn і H2O AutoML, їх ефективне використання й надалі потребує певного рівня технічних знань.

Стартап-проєкт EasyWeatherML може частково закрити дану потребу, пропонуючи просту та інтуїтивно зрозумілу програмну систему для базового дослідницького аналізу погодних даних та побудови передбачень без необхідності спеціалізованих технічних знань.

У таблиці 6.1 наведено стислий опис ідеї стартап-проєкту EasyWeatherML, що включає зміст запропонованого рішення, можливі напрями його застосування та основні вигоди для користувачів.

Таблиця 6.1 – Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Веб-орієнтований застосунок для базового дослідницького аналізу та передбачення погодних даних з використанням методів машинного навчання.	Освітньо-демонстраційний інструмент для ознайомлення з концепцією базового дослідницького аналізу та основними методами навчання з учителем на прикладі погодних даних.	Простий інтуїтивний інтерфейс, відсутність потреби у спеціалізованих технічних компетенціях з аналізу даних та машинного навчання.
	Практичний інструмент для побудови простих прогнозів погодних параметрів на невеликих локальних наборах даних.	Можливість швидкого формування передбачень на невеликих наборах погодних даних та аналізу впливу окремих факторів на цільову змінну без потреби у програмуванні та налаштуванні системи.

У таблиці 6.2 подано перелік техніко-економічних характеристик ідеї стартап-проєкту EasyWeatherML та їх попередню оцінку із виокремленням сильних, слабких і нейтральних сторін порівняно з конкурентами. Це дозволяє систематизовано оцінити конкурентні переваги й обмеження проєкту.

Таблиця 6.2 – Визначення характеристик ідеї проєкту

№ п/п	Техніко-економічні характеристики ідеї	Потенційні товари/концепції конкурентів			W	N	S
		EasyWeather ML	Google Colab	H2O AutoML			

Продовження таблиці 6.2

1	Простота та інтуїтивність інтерфейсу	Інтуїтивний веб-інтерфейс для аналізу та передбачення погодних даних	Зручний веб-інтерфейс середовища програмування	Веб-інтерфейс та API з гнучкими налаштуваннями		+	
2	Рівень кваліфікації користувача, необхідний для ефективної роботи	Базові знання з ML будуть корисними	Навички програмування та роботи з бібліотеками машинного навчання (Python)	Базові концепції ML та навички програмування будуть корисними			+
3	Потреба у програмуванні	Не потребує програмування; взаємодія повністю через веб-інтерфейс	Вимагає написання коду для завантаження даних, побудови моделей та візуалізації результатів	Базова робота через веб-інтерфейс; для гнучких налаштувань – використання API			+
4	Максимальний обсяг даних для зручної роботи	Орієнтований на невеликі локальні набори даних з обмеженою кількість ознак	Може опрацьовувати середні та великі набори даних, обмежений доступними ресурсами середовища виконання	Оптимізований для опрацювання великих наборів даних, у тому числі на кластерних конфігураціях	+		
5	Гнучкість налаштування моделей	Обмежений набір методів ML та фіксовані гіперпараметри для моделей	Максимальна гнучкість: повний контроль у коді над вибором моделей, гіперпараметрів та процедур навчання	Широкі можливості налаштування параметрів AutoML, простору гіперпараметрів та окремих моделей	+		

Кінець таблиці 6.2

6	Автоматизація вибору моделі	Обмежений перелік моделей ML	Автоматизація залежить від реалізованого користувачем коду; вбудованих засобів AutoML немає (можна використовувати сторонні бібліотеки)	Передбачено автоматичний підбір, оцінювання та ансамблювання моделей згідно з концепцією AutoML	+		
7	Предметна спеціалізація інструменту	Орієнтований на аналіз та передбачення погодних даних	Універсальне середовище	Загальний інструмент AutoML		+	
8	Вартість використання і впровадження	Низькі інфраструктурні вимоги, можливе безоплатне використання	Базовий доступ безоплатний; розширені ресурси – за підпискою	Наявні рішення з відкритим кодом, а також ліцензійні корпоративні версії			+

6.2 Технологічний аудит ідеї проєкту

Для оцінки технологічної здійсненності ідеї стартап-проєкту EasyWeatherML виконано аудит основних засобів розробки веб-орієнтованого застосунку, результати якого наведено у таблиці 6.3.

Таблиця 6.3 – Аудит засобів розробки проєкту

№ п/п	Компонент програмної системи	Технологія	Наявність	Доступність

Продовження таблиці 6.3

1	Веб-інтерфейс	Бібліотека React	Наявна як open-source бібліотека	Розповсюджується безоплатно, доступна як npm-пакет
		CSS-фреймворк TailwindCSS	Наявний як open-source CSS-фреймворк	Розповсюджується безоплатно, доступний як npm-пакет
2	База даних	СКБД MongoDB	Наявна як готова система керування базами даних	Має безоплатну версію для використання у проєкті
3	Серверна частина	Платформа Node.js	Наявна як програмна платформа виконання JavaScript-коду на стороні сервера	Розповсюджується безоплатно, доступна для основних операційних систем
		Веб-фреймворк Express.js	Наявний open-source веб-фреймворк для Node.js	Доступний у вигляді безоплатного npm-пакета
		Бібліотека ODM Mongoose	Наявна open-source ODM-бібліотека для роботи з MongoDB	Доступна у вигляді безоплатного npm-пакета
4	Керування доступом користувачів	Бібліотека jsonwebtoken	Наявна бібліотека, що реалізує стандарт JWT	Доступна у вигляді безоплатного npm-пакета
		Бібліотека bcrypt	Наявна, реалізує хешування паролів	Доступна у вигляді безоплатного npm-пакета

Кінець таблиці 6.3

5	Модуль машинного навчання	Веб-фреймворк FastAPI	Наявний open-source веб-фреймворк для побудови REST-API сервісів	Доступний у вигляді безоплатного pip-пакета
		Бібліотека ML Scikit-learn	Наявна для мови Python, open-source	Доступна у вигляді безоплатного pip-пакета

Оскільки всі основні необхідні технології є наявними на ринку та безоплатно доступними, технологічна реалізація стартап-проєкту EasyWeatherML є здійсненною.

6.3 Аналіз ринкових можливостей запуску стартап-проєкту

У межах аналізу ринкових можливостей даного стартап-проєкту визначено потенційні групи клієнтів та їх основні вимоги до програмного продукту, що відображено в таблиці 6.4.

Таблиця 6.4 – Характеристика потенційних клієнтів стартап-проєкту

№ п/п	Потреба, сформована ринком	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Доступний інструмент для освітньо-демонстраційних цілей щодо базового аналізу та	Студенти технічних спеціальностей, слухачі відповідних курсів, викладачі	Студенти орієнтовані на простоту та підказки, викладачі – на швидку	Доступ з браузера, інтуїтивний інтерфейс, підтримка невеликих

Кінець таблиці 6.4

	методів машинного навчання	профільних дисциплін	демонстрацію типових прикладів	наборів даних, змістовна демонстрація результатів
2	Інструмент для простих передбачень параметрів погодних даних та оцінки впливу окремих факторів без необхідності програмування	Користувачі, зацікавлені в простому аналізі та передбаченні погодних даних без заглиблення в програмування та складні інструменти	Частина користувачів працює з власними невеликими наборами даних, інші очікують швидких результатів з простих задач за мінімальної кількості дій в інтерфейсі	Підтримка невеликих наборів даних, мінімальні налаштування, швидке передбачення та змістовна демонстрація результатів без необхідності програмування

Для оцінки впливу ринкового середовища на стартап-проєкт EasyWeatherML виокремлено та згруповано ключові фактори загроз (таблиця 6.5) і можливостей (таблиця 6.6).

Таблиця 6.5 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Конкуренція з наявними аналітичними та освітніми платформами	Потенційні користувачі можуть віддати перевагу вже відомим сервісам	Підкреслення простоти використання інструменту для навчання базовому аналізу та методам машинного навчання з учителем
2	Обмеженість інструментів базового аналізу та методів машинного навчання	Функціонал обмежений невеликою кількістю інструментів базового аналізу та методів машинного навчання	Поступове розширення інструментів базового аналізу та переліку методів машинного навчання

Кінець таблиці 6.5

3	Використання моделей машинного навчання з фіксованими гіперпараметрами	Обмежена можливість підвищення точності та адаптації моделей до специфіки конкретних задач і наборів даних	Впровадження можливості налаштування важливих гіперпараметрів ML моделей

Таблиця 6.6 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Зростання інтересу до аналізу даних та машинного навчання	Збільшується кількість користувачів, які шукають освітньо-демонстраційні інструменти	Розробка навчальних прикладів і матеріалів, орієнтованих на студентів і викладачів
2	Зростання попиту на рішення базового аналізу та передбачень погодних даних без необхідності програмування	Користувачі виявляють бажання працювати з погодними даними без необхідності написання коду та складних налаштувань	Акцент на можливості роботи без навичок програмування
3	Покращення доступності відкритих погодних даних технологій	Можливість знизити витрати на інфраструктуру та покращувати функціональність	Використання відкритих даних і вільного ПЗ як основи технічної реалізації
4	Потенціал партнерств з освітніми та тренінговими програмами	Інструмент може стати частиною навчальних курсів з аналізу даних і ML	Пропозиція спеціальних освітніх тарифів та інтеграцій із навчальними платформами

Попри наявні технологічні та конкурентні загрози, ринкове середовище загалом є сприятливим для подальшого розвитку стартап-проєкту. Важливою умовою є поступове розширення функціональності із збереженням простоти використання.

6.4 Розроблення ринкової стратегії проєкту

Ринкову стратегію стартап-проєкту деталізовано через базову стратегію конкурентної поведінки (таблиця 6.7) та стратегію позиціонування (таблиця 6.8).

Таблиця 6.7 – Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект “першопрохідцем” на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Ні	Шукати нових споживачів	Можливе впровадження налаштування гіперпараметрів ML моделей, або навіть AutoML рішень	Стратегія зайняття конкурентної ніші

Таблиця 6.8 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап‑проєкту	Асоціації, що формують комплексну позицію власного проєкту
1	Простота використання, інтуїтивний інтерфейс, відсутність потреби програмування, інформативність результатів	Стратегія зайняття конкурентної ніші	Базовий аналіз та передбачення погодних даних, зрозумілий веб-інтерфейс, можливість роботи без програмування	Інтуїтивність, простота, інформативність

6.5 Розроблення маркетингової програми стартап-проєкту

Ключові потреби цільової аудиторії та відповідні переваги концепції потенційного товару стартап-проєкту наведено в таблиці 6.9.

Таблиця 6.9 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі, або такі що слід створити)
1	Освітній інструмент для демонстрації підходів базового аналізу та методів машинного навчання на погодних даних	Швидка демонстрація реальних задач аналізу та передбачення погодних даних у навчальному процесі	Орієнтація на погодні дані, інтуїтивний веб-інтерфейс, застосування моделей машинного навчання без потреби налаштування та програмування
2	Простий інструмент для передбачень погодних параметрів і оцінки впливу факторів без необхідності програмування	Швидке отримання результатів базового аналізу та передбачень, щодо завантажених даних без написання коду, безпосередньо у браузері	Необхідність в мінімальній кількості дій в інтерфейсі, інформативна візуалізація результатів аналізу та передбачень

6.6 Висновки до розділу 6

У даному розділі проведено комплексний маркетинговий аналіз стартап-проєкту EasyWeatherML. Розглянуто ідею програмного продукту, визначено його техніко-економічні характеристики та виконано порівняльну оцінку переваг і обмежень відносно існуючих аналогів. Встановлено наявність практичної цінності проєкту для користувачів, яким потрібні прості засоби базового аналізу й передбачення погодних параметрів, що не вимагають від

користувача спеціальних навичок, зокрема стосовно програмування чи машинного навчання.

Технологічний аудит підтвердив здійсненність реалізації EasyWeatherML на основі доступних відкритих технологій та класичних моделей машинного навчання. Усі необхідні фреймворки, бібліотеки та інструменти є наявними, відкритими та такими, що не створюють технологічних бар'єрів для впровадження.

Аналіз ринкових умов показав наявність стійкого попиту на інструменти для освітньо-демонстраційних задач та побудови простих передбачень на основі локальних наборів погодних даних. Визначено ключові загрози й можливості, включно з конкуренцією універсальних платформ та зростанням потреби в рішеннях, що не вимагають спеціалізованих навичок. Це дало підстави сформувати стратегію зайняття конкурентної ніші та позиціонувати продукт як інтуїтивний, доступний і зосереджений на погодних даних.

Оптимальним варіантом ринкового запуску є веб-орієнтований формат SaaS з фріміум-моделлю доступу, що мінімізує бар'єри входження та забезпечує охоплення основних цільових груп.

Результати аналізу показують, що EasyWeatherML має ринковий потенціал і може бути комерціалізований за умови поступового розширення функціональності, збереження простоти використання та адаптації стратегії розвитку до конкурентного середовища.

ВИСНОВКИ

У процесі підготовки магістерської дисертації виконано комплекс поставлених завдань, спрямованих на реалізацію інтегрованої веб-орієнтованої системи для базового дослідницького аналізу та передбачення погодних умов із застосуванням алгоритмів машинного навчання. Отримані результати відображені нижче.

1. На основі аналізу предметної області встановлено, що погодні дані характеризуються складною багатофакторною природою, включаючи сезонність, стохастичність, корельованість параметрів та наявність пропусків. Досліджено сучасні підходи до прогнозування атмосферних процесів, охоплюючи фізичні, статистичні та алгоритмічні методи, що підтвердило доцільність використання моделей машинного навчання в задачах аналізу невеликих кліматичних вибірок.

2. Оцінено ефективність основних методів машинного навчання з учителем для розв’язання задач класифікації та регресії на невеликих погодних вибірках: логістичної регресії, градієнтного бустингу та багатошарового персептрону. Обґрунтовано їх вибір як моделей, здатних забезпечити збалансоване співвідношення точності, швидкодії та простоти використання за умови фіксованих гіперпараметрів, що робить дані методи придатними для інтеграції у веб-орієнтоване програмне середовище.

3. Спроєктовано архітектуру веб-орієнтованого застосунку, яка передбачає чіткий розподіл функцій між клієнтською частиною (React), серверною логікою (Node.js, Express.js, Mongoose) та модулем машинного навчання (Python, FastAPI). Забезпечено узгоджену взаємодію між компонентами, обмін даними реалізовано відповідно до принципів REST, а також незалежність обчислювального модуля від інтерфейсної частини, що підвищує масштабованість і гнучкість системи.

4. Реалізовано комплекс програмних модулів для обробки даних, базового дослідницького аналізу та формування передбачень. Забезпечено функції завантаження файлів у форматі CSV, автоматичної валідації структури вибірки, побудови описової статистики, розрахунку кореляційних залежностей та графічної

візуалізації. Інтегрований модуль машинного навчання дозволяє формувати передбачення та виводити метрики якості моделей у межах єдиного програмного середовища. Проведено тестування реалізованого програмного продукту, у ході якого підтверджено стабільність роботи компонентів, коректність обміну даними між сервісами та відтворюваність результатів моделювання. Система дозволяє працювати з невеликими локальними вибірками погодних даних без додаткового налаштування з боку користувача.

5. Проаналізовано ринкові можливості застосування розробленого рішення, що показало його придатність для використання як доступного інструменту роботи з невеликими вибірками локальних погодних даних. Система може застосовуватися в освітньо-демонстраційних та прикладних задачах невисокої складності без потреби у спеціальній технічній підготовці. Отримані результати свідчать, що створене рішення має подальший потенціал розвитку і може бути масштабоване та комерціалізоване за умови поступового розширення функціональності, збереження простоти використання та адаптації до конкурентного ринкового середовища. Перспективним шляхом удосконалення системи є, зокрема, додання моделей часових рядів, наприклад ARIMA та SARIMA, що дозволить краще моделювати сезонні та автокореляційні характеристики погодних параметрів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Multivariate stochastic generation of meteorological data for building simulation through interdependent meteorological processes / Z. Jiao та ін. S*cientific Reports*. 2024. Т. 14, № 1. URL: https://doi.org/10.1038/s41598-024-75498-8 (дата звернення: 24.11.2025);

2. Torro H., Meneu V., Valor E. Single Factor Stochastic Models with Seasonality Applied to Underlying Weather Derivatives Variables. *SSRN Electronic Journal*. 2001. URL: https://doi.org/10.2139/ssrn.264178 (дата звернення: 24.11.2025);

3. Big Data Analytics in Weather Forecasting: A Systematic Review / M. Fathi та ін. *Archives of Computational Methods in Engineering*. 2021. URL: https://doi.org/10.1007/s11831-021-09616-4 (дата звернення: 24.11.2025);

4. Stochastic Parameterization: Toward a New View of Weather and Climate Models / J. Berner та ін. *Bulletin of the American Meteorological Society*. 2017. Т. 98, № 3. С. 565–588. URL: https://doi.org/10.1175/bams-d-15-00268.1 (дата звернення: 24.11.2025);

5. The ensemble prediction system – recent and ongoing developments / T. N. Palmer та ін. ECMWF Technical Memoranda. 2007. № 540. 53 с.;

6. Machine Learning Methods for Weather Forecasting: A Survey / H. Zhang та ін. *Atmosphere*. 2025. Т. 16, № 1. С. 82. URL: https://doi.org/10.3390/atmos16010082 (дата звернення: 24.11.2025);

7. Probabilistic weather forecasting with machine learning / I. Price та ін. *Nature*. 2024. URL: https://doi.org/10.1038/s41586-024-08252-9 (дата звернення: 24.11.2025);

8. Interpretable machine learning for weather and climate prediction: A review / R. Yang та ін. *Atmospheric Environment*. 2024. С. 120797. URL: https://doi.org/10.1016/j.atmosenv.2024.120797 (дата звернення: 24.11.2025);

9. AMLB An AutoML Benchmark. *AutoML Frameworks*. URL: https://openml.github.io/automlbenchmark/frameworks.html?utm_source=chatgpt.com (дата звернення: 24.11.2025);

10. AutoML | Tabular Data. AutoML.org. URL: https://www.automl.org/automl-for-x/tabular-data/ (дата звернення: 24.11.2025).;

11. What is AutoML? Understanding automated machine learning. *Nebius*. *The ultimate cloud for AI explorers. URL*: https://nebius.com/blog/posts/what-is-automl (дата звернення: 24.11.2025);

12. Types of Machine Learning: Supervised, Unsupervised and More | DigitalOcean. DigitalOcean | The Unified Agentic Cloud. URL: https://www.digitalocean.com/resources/articles/types-of-machine-learning (дата звернення: 24.11.2025);

13. MathWorks. Types of Machine Learning Models Explained. MathWorks - Maker of MATLAB and Simulink. URL: https://www.mathworks.com/discovery/machine-learning-models.html (дата звернення: 24.11.2025);

14. A Comprehensive Overview and Comparative Analysis on Deep Learning Models / F. M. Shiri та ін. Journal on Artificial Intelligence. 2024. Т. 6, № 1. С. 301–360. URL: https://doi.org/10.32604/jai.2024.054314 (дата звернення: 24.11.2025);

15. Data Preprocessing in Machine Learning: Steps & Best Practices. *lakeFS*. URL: https://lakefs.io/blog/data-preprocessing-in-machine-learning (дата звернення: 24.11.2025);

16. Data Preprocessing in Python. *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/machine-learning/data-preprocessing-machine-learning-python (дата звернення: 24.11.2025);

17. Geron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. – 2nd ed. – Sebastopol (CA) : O'Reilly Media, Inc., 2019. – 851 с.;

18. Raschka S., Liu Y., Mirjalili V. Machine Learning with PyTorch and Scikit-Learn. – Birmingham : Packt Publishing Ltd., 2022. – 771 с.;

19. Logistic Regression. *numiqo*. URL: https://numiqo.com/tutorial/logistic-regression (дата звернення: 24.11.2025);

20. Gradient Boosting in ML. *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/machine-learning/ml-gradient-boosting (дата звернення: 24.11.2025);

21. Multi-Layer Perceptron Explained: A Beginner's Guide. Quark Machine Learning. URL: https://www.quarkml.com/2023/01/multi-layer-perceptron-a-complete-overview.html (дата звернення: 24.11.2025);

22. What is a Multilayer Perceptron (MLP) or a Feedforward Neural Network (FNN)?. *AIML.com*. URL: https://aiml.com/what-is-a-multilayer-perceptron-mlp/ (дата звернення: 24.11.2025);

23. Activation Functions. *Machine Learning Glossary – ML Glossary documentation*. URL: https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html (дата звернення: 24.11.2025);

24. Evaluation Metrics in Machine Learning. *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/machine-learning/metrics-for-machine-learning-model (дата звернення: 24.11.2025);

25. Caballar R., Stryker C. What Is Model Performance in Machine Learning? | IBM. IBM. URL: https://www.ibm.com/think/topics/model-performance (дата звернення: 24.11.2025);

26. Introduction - JavaScript | MDN. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction (дата звернення: 24.11.2025);

27. Node.js – About Node.js. Node.js – Run JavaScript Everywhere. URL: https://nodejs.org/en/about (дата звернення: 24.11.2025);

28. What is Python? Executive Summary. *Python.org*. URL: https://www.python.org/doc/essays/blurb/ (дата звернення: 24.11.2025);

29. Node.js – An introduction to the npm package manager. *Node.js – Run JavaScript Everywhere*. URL: https://nodejs.org/en/learn/getting-started/an-introduction-to-the-npm-package-manager (дата звернення: 24.11.2025);

30. React Reference Overview – React. *React*. URL: https://react.dev/reference/react (дата звернення: 24.11.2025);

31. Styling with utility classes - Core concepts. Tailwind CSS – *Rapidly build modern websites without ever leaving your HTML*. URL: https://tailwindcss.com/docs/styling-with-utility-classes (дата звернення: 24.11.2025);

32. What is daisyUI? daisyUI – Tailwind CSS Component UI Library. Tailwind CSS Component Library – daisyUI. URL: https://daisyui.com/blog/what-is-daisyui/ (дата звернення: 24.11.2025);

33. Getting Started. *vitejs*. URL: https://vite.dev/guide (дата звернення: 24.11.2025);

34. Getting Started | Axios Docs. *Axios*. URL: https://axios-http.com/docs/intro (дата звернення: 24.11.2025);

35. Express – Node.js web application framework. *Express – Node.js web application framework*. URL: https://expressjs.com/ (дата звернення: 24.11.2025).;

36. MERN Stack Explained. *MongoDB*. URL: https://www.mongodb.com/resources/languages/mern-stack (дата звернення: 24.11.2025);

37. MongoDB – Working and Features. *GeeksforGeeks*. URL: https://www.geeksforgeeks.org/mongodb/what-is-mongodb-working-and-features/ (дата звернення: 24.11.2025);

38. Clark B. What is Scikit-Learn (Sklearn)? IBM. URL: https://www.ibm.com/think/topics/scikit-learn (дата звернення: 24.11.2025);

39. NumPy: the absolute basics for beginners – *NumPy* v2.3 Manual. NumPy. URL: https://numpy.org/doc/stable/user/absolute_beginners.html (дата звернення: 24.11.2025);

40. About Git – GitHub Docs. *GitHub Docs*. URL: https://docs.github.com/en/get-started/using-git/about-git (дата звернення: 24.11.2025);

41. Otrokh S., Posternak A. Web-based System for Weather Data Analysis and Prediction with Machine Learning. Scientific Research: Emerging Theories and Practical Breakthroughs : тези доп. 2-ї Міжнар. наук.-практ. конф. (Единбург, 17–19 листоп. 2025 р.) / European Open Science Space. Единбург, 2025. С. 104–107;

42. Мельник Ю. В., Отрох С. І., Постернак А. В. Аналіз та передбачення погодних умов на основі машинного навчання з інтеграцією веб-технологій //

Наукові записки Державного університету інформаційно-комунікаційних технологій. 2025. № 2;

43. Розроблення стартап-проекту : методичні рекомендації до виконання розділу магістерських дисертацій для студентів інженерних спеціальностей / уклад. О. А. Гавриш, С. О. Солнцев, В. В. Дергачова, О. В. Зозульов [та ін.] ; КПІ ім. Ігоря Сікорського. – Київ : НТУУ «КПІ», 2016. – 28 с. – URL: https://ela.kpi.ua/items/53c140af-585e-4e63-b6e2-e43460684209 (дата звернення: 24.11.2025).

ДОДАТОК А

Тези 2 міжнародної науково-практичної конференції
“Scientific Research: Emerging Theories and Practical Breakthroughs”

Web-based System for Weather Data Analysis and Prediction with Machine Learning

УКР.НТУУ“КПІ ім. Ігоря Сікорського” _ІАТЕ_ЦТЕ_ТР-41мп

Аркушів 6

Київ – 2025

eoss-conf.com

ISSUE
№62

EUROPEAN OPEN SCIENCE SPACE

COLLECTION OF SCIENTIFIC PAPERS

2ND INTERNATIONAL SCIENTIFIC AND PRACTICAL CONFERENCE

SCIENTIFIC RESEARCH: EMERGING THEORIES AND PRACTICAL BREAKTHROUGHS

NOVEMBER 17-19, 2025, EDINBURGH, SCOTLAND

Proceedings of the 2nd International Scientific and Practical Conference
"Scientific Research: Emerging Theories and Practical Breakthroughs"
November 17-19, 2025
Edinburgh, Scotland
EUROPEAN OPEN SCIENCE SPACE

WEB-BASED SYSTEM FOR WEATHER DATA ANALYSIS AND PREDICTION WITH MACHINE LEARNING

Otrokh Serhii
Doctor of Sciences (technical), Professor
Posternak Anton
Master's Student
Department of Digital Technologies in Energy
NTUU "Igor Sikorsky KPI", Ukraine

Machine learning plays a crucial role in modern weather forecasting by enabling the analysis of vast, multidimensional atmospheric datasets to identify complex, nonlinear relationships that traditional numerical models often overlook. Through advanced deep learning architectures such as convolutional and recurrent neural networks, machine learning enhances the precision of precipitation nowcasting, climate downscaling, and extreme weather detection. Its integration with physical modeling principles allows for more adaptive and efficient systems that improve both the temporal and spatial accuracy of forecasts. As computational power and data availability expand, machine learning continues to transform meteorology into a more data-driven and predictive science, offering a pathway toward faster, more reliable, and resource-efficient forecasting solutions [1].

Supervised machine learning remains an essential component of contemporary computational modeling, complementing deep learning rather than being replaced by it. Classical supervised algorithms such as logistic regression, gradient boosting, and multilayer perceptron continue to demonstrate strong performance on structured and tabular datasets, offering advantages in interpretability, training efficiency, and resource utilization compared to deep neural networks [2, 3].

Nevertheless, their practical application requires technical proficiency in data preprocessing, feature engineering, and hyperparameter optimization. Platforms such as Google Colab provide an accessible environment for model development and experimentation, yet still demand a fundamental understanding of programming, data handling, and algorithmic principles to be used effectively. Modern AutoML frameworks such as Auto-Sklearn, AutoGluon, FLAML, and H2O AutoML address these challenges by automating model selection and tuning, thereby reducing the need for manual configuration while maintaining high predictive performance and scalability. However, even these systems require at least basic programming and analytical skills, which establishes a minimal but necessary entry threshold for their effective use [4].

The purpose of the work is to develop an integrated web application that automates the workflow of weather data processing from dataset uploading and basic exploratory analysis to parameter prediction using selected models and features. The application is designed to make machine learning methods accessible to users without

Proceedings of the 2nd International Scientific and Practical Conference
"Scientific Research: Emerging Theories and Practical Breakthroughs"
November 17-19, 2025
Edinburgh, Scotland
EUROPEAN OPEN SCIENCE SPACE

specialized expertise by utilizing simple models with fixed hyperparameters, serving both educational and demonstrational purposes.

For prediction, logistic regression, gradient boosting, and a multilayer perceptron were selected, allowing the user to choose the most suitable model for a given task. Logistic regression serves as an interpretable and computationally efficient baseline for categorical prediction. Gradient boosting tends to provide higher predictive accuracy and robustness by combining multiple weak learners to model complex dependencies. The multilayer perceptron, as a neural network, can capture nonlinear patterns and interactions between features. This selection provides a flexible balance between interpretability, computational efficiency, and predictive performance in basic weather data forecasting [2, 3].

After uploading a dataset, a basic exploratory data analysis can be performed to examine key characteristics, detect patterns, and visualize relationships within the data. The module computes descriptive statistics, including measures of central tendency and dispersion, to assess data distribution and potential asymmetry. Histograms and correlation matrices help identify skewness and multicollinearity, supporting proper feature selection and preprocessing. For weather data, time-series plots reveal trends and seasonal variations, providing a foundation for selecting suitable predictive models [5].

The developed web-based system is designed using a modular, multilayer architecture consisting of three core components: the client interface, the application server, and the machine learning microservice. This architectural paradigm ensures horizontal scalability, maintainability, and strict separation of concerns [6, 7].

The client layer, implemented with React, provides an interactive user interface for seamless interaction with the system. User requests are transmitted to the application server, developed with Node.js and Express.js, which exposes a RESTful API, enforces JWT-based authentication, and performs persistent data operations through the Mongoose ODM for MongoDB integration [6].

The machine learning layer is implemented as a standalone Python FastAPI microservice dedicated to data preprocessing, model training, and prediction. It employs Scikit-learn and NumPy for computational routines, applying one-hot encoding for categorical variables, label encoding for target features, and standardization for numerical attributes. The microservice exposes dedicated endpoints for logistic regression, gradient boosting, and multilayer perceptron models. This microservice-based design allows asynchronous communication between the frontend, backend, and machine learning components, making module updates, error isolation, and data exchange more efficient across the system [6, 7].

The data persistence layer utilizes a document-oriented MongoDB database to store user profiles and uploaded datasets. Two principal document types are defined: User and Note. The User document encapsulates authentication credentials and user metadata, whereas the Note document manages dataset-related information. Uploaded tabular datasets are stored in a csvData array of nested objects representing table rows,

allowing flexible column structures. Both fieldTypes and fileMeta are defined as embedded objects. The fieldTypes object automatically records inferred data types for each column, while the fileMeta object stores metadata such as file name, size, storage path, MIME type, and storage status indicating whether the file is currently saved on disk. This schema design ensures data integrity, supports extensibility, and eliminates the need for manual schema migration. The conceptual UML data model is illustrated in Figure 1.

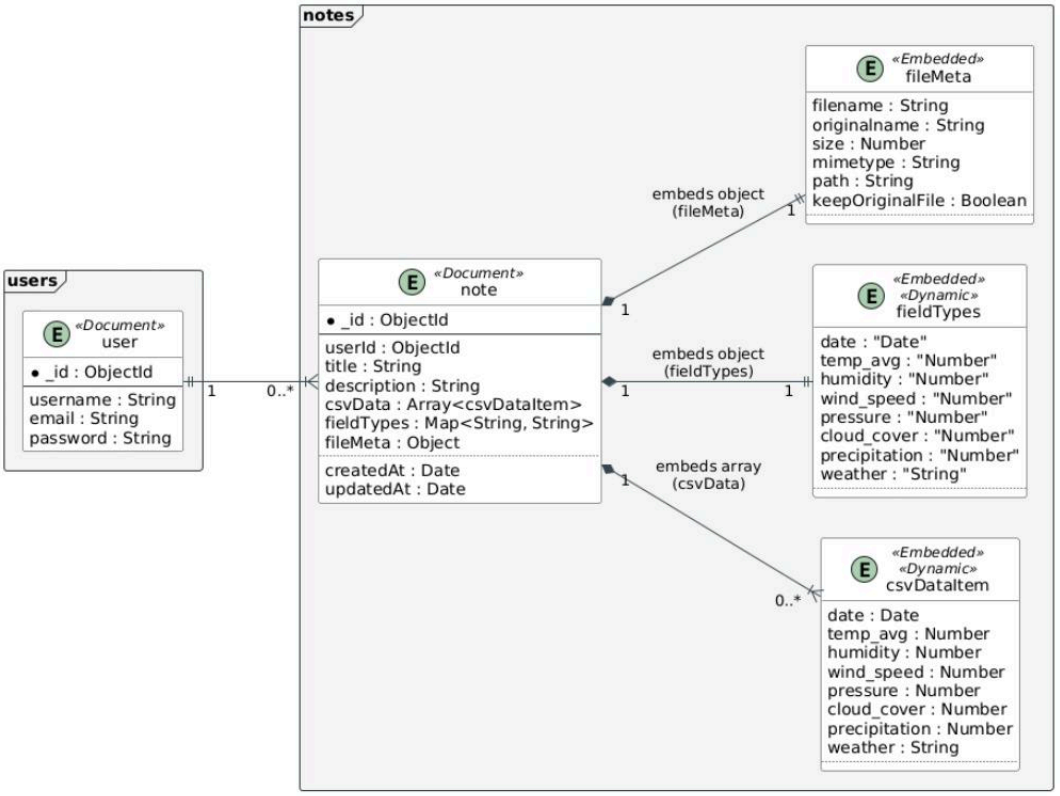
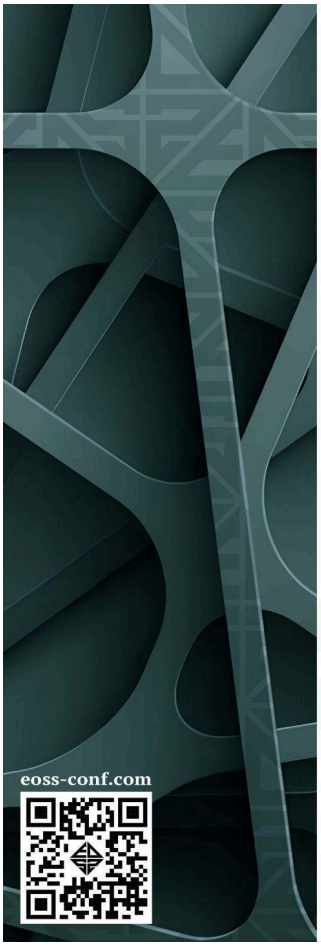


Figure 1. Conceptual UML model of the database

As a result, this work presents an integrated web-based application for weather data analysis and prediction using machine learning methods. The system provides a complete workflow from data import and basic exploratory analysis to predictive modeling. It applies fundamental algorithms such as logistic regression, gradient boosting, and multilayer perceptron with fixed hyperparameters, which simplifies configuration and emphasizes result interpretation. The developed application can be used as an educational and demonstrational tool for learning basic machine learning methods, as well as a practical means for predicting weather data from small localized datasets.

References

1. Machine Learning Methods for Weather Forecasting: A Survey / H. Zhang та ін. Atmosphere. 2025. Т. 16, № 1. С. 82. URL: https://doi.org/10.3390/atmos16010082 (дата звернення: 07.11.2025).
2. Geron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. – 2nd ed. – Sebastopol (CA) : O'Reilly Media, Inc., 2019. – 851 с.
3. Raschka S., Liu Y., Mirjalili V. Machine Learning with PyTorch and Scikit-Learn. – Birmingham : Packt Publishing Ltd., 2022. – 771 с.
4. AMLB: Frameworks. AMLB An AutoML Benchmark. URL: https://openml.github.io/automlbenchmark/frameworks.html (дата звернення: 07.11.2025).
5. What is Exploratory Data Analysis? – GeeksforGeeks. – URL: https://www.geeksforgeeks.org/data-analysis/what-is-exploratory-data-analysis (дата звернення: 07.11.2025).
6. MERN Stack Explained. MongoDB. URL: https://www.mongodb.com/resources/languages/mern-stack (дата звернення: 07.11.2025).
7. Microservice in Python using FastAPI – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/microservice-in-python-using-fastapi/ (дата звернення: 07.11.2025).



EUROPEAN OPEN SCIENCE SPACE

CERTIFICATE
of participation

Anton Posternak

took part in the 2nd International Scientific and Practical Conference

«SCIENTIFIC RESEARCH: EMERGING THEORIES AND PRACTICAL BREAKTHROUGHS»

24 Hours of Participation
(0,8 ECTS credits)

Head of the
organizing committee
Helen Volokitina

EUROPEAN OPEN SCIENCE SPACE 3382508349

EOSS-25/1117-041

November 17-19, 2025, Edinburgh, Scotland

ДОДАТОК Б

Стаття в журналі «Наукові записки Державного університету інформаційно-комунікаційних технологій»

Аналіз та передбачення погодних умов на основі машинного навчання з інтеграцією веб-технологій

УКР.НТУУ“КПІ ім. Ігоря Сікорського” _ІАТЕ_ЦТЕ_ТР-41мп

Аркушів 9

Київ – 2025

УДК 004.8:004.42:004.65

DOI:

Ю.В. Мельник, доктор техн. наук, професор;
Державний університет інформаційно-комунікаційних технологій
С.І. Отрох, доктор техн. наук, професор;
А.В. Постернак;
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

АНАЛІЗ ТА ПЕРЕДБАЧЕННЯ ПОГОДНИХ УМОВ НА ОСНОВІ МАШИННОГО НАВЧАННЯ З ІНТЕГРАЦІЄЮ ВЕБ-ТЕХНОЛОГІЙ

ANALYSIS AND PREDICTION OF WEATHER CONDITIONS BASED ON MACHINE LEARNING WITH INTEGRATION OF WEB TECHNOLOGIES

Ю.В. Мельник, С.І. Отрох, А.В. Постернак. Аналіз та передбачення погодних умов на основі машинного навчання з інтеграцією веб-технологій. Стаття висвітлює актуальність застосування методів машинного навчання для прогнозування погодних умов і аналізує сучасні AutoML-підходи, які спрощують процес вибору моделей і налаштування гіперпараметрів. Запропоновано інтегрований програмний засіб, що забезпечує повний цикл роботи з даними: імпорт, базовий дослідницький аналіз і побудову прогнозів на основі фундаментальних моделей з фіксованими гіперпараметрами. Архітектура системи охоплює клієнтський інтерфейс і сервер прикладної логіки, реалізовані на основі стеку MERN, а також окремий модуль машинного навчання, побудований із використанням бібліотек мови Python. Для передбачення застосовано логістичну регресію, градієнтний бустинг і багатошаровий персептрон. Розроблене рішення може використовуватися як інструмент швидкої перевірки гіпотез, у навчально-демонстраційних цілях та для отримання передбачень допустимої точності на локалізованих наборах погодних даних.

Ключові слова: машинне навчання, AutoML, передбачення погодних умов, веб-застосунок, MERN, Scikit-learn.

Y.V. Melnyk, S.I. Otrokh, A.V. Posternak. Analysis and prediction of weather conditions based on machine learning with integration of web technologies. The article highlights the relevance of applying machine learning methods to weather prediction and analyzes modern AutoML approaches that simplify model selection and hyperparameter tuning. An integrated software solution is proposed, ensuring a complete data workflow: from dataset import and basic exploratory analysis to model construction and the generation of forecasts based on fundamental methods with fixed hyperparameters. The system architecture is composed of a client interface and an application logic server implemented on the MERN stack, as well as a separate machine learning module developed with Python libraries. The developed tool emphasizes transparency and ease of use, allowing non-expert users to perform experiments with minimal technical effort. Logistic regression, gradient boosting, and multilayer perceptron models are used to predict values in localized climate datasets. The developed application can be used as a tool for rapid hypothesis verification, educational and demonstration purposes, and for obtaining forecasts with acceptable accuracy on localized weather datasets. Future development of the system may focus on integrating time-series forecasting models such as ARIMA and SARIMA to account for seasonal and autocorrelated characteristics of climatic processes.

Keywords: machine learning, AutoML, weather prediction, web application, MERN, Scikit-learn.

Постановка проблеми. Сучасні підходи до прогнозування погодних умов дедалі частіше ґрунтуються на методах машинного навчання, які здатні виявляти складні нелінійні залежності у кліматичних даних та використовувати їх для підвищення точності прогнозу. Однак, практичне застосування таких методів залишається прерогативою спеціалістів, які володіють навичками програмування та роботи з відповідними бібліотеками як Scikit-learn, PyTorch чи TensorFlow. Це створює суттєвий бар'єр входу для широкого кола дослідників суміжних галузей, студентів чи фахівців-практиків, наприклад, в агрономії чи локальному управлінні.

Водночас, побудові моделі прогнозу, як правило, передує не менш важливий етап – попередній дослідницький аналіз. Застосування оцінки базових статистичних характеристик, аналізу кореляційних зв'язків та візуалізації часових рядів є важливим кроком для повноцінного розуміння структури даних, що дозволяє своєчасно виявити ключові тенденції, зв'язки між ознаками та потенційні проблеми, а також забезпечує якісну основу для побудови ефективних моделей. І тут постає споріднена проблема: наявні програмні інструменти, що використовуються для аналізу та передбачення, часто характеризуються надмірною складністю конфігурації. Вони вимагають від користувача ґрунтовного розуміння методології, осмисленого вибору методів та налаштування гіперпараметрів, що ускладнює процес швидкої перевірки гіпотез.

Таким чином, постає актуальна задача розробки інтегрованого веб-орієнтованого програмного рішення для аналізу погодних даних і передбачення погодних умов методами машинного навчання. Існує потреба в застосунку, що забезпечує повний цикл роботи: від імпорту користувацьких наборів даних до їх попереднього базового аналізу та створення моделей прогнозування. Ключовий елемент – модуль передбачення на основі фундаментальних моделей із фіксованими гіперпараметрами. Такий підхід дозволяє користувачу здійснювати швидку перевірку гіпотез щодо взаємозв'язку між характеристиками даних та ефективністю моделі у прогнозуванні. Рішення може слугувати як ефективним освітньо-демонстраційним інструментом, так і практичним засобом для отримання передбачень допустимої точності на локалізованих невеликих наборах погодних даних.

Аналіз останніх досліджень і публікацій. Прогнозування погоди є надзвичайно активною сферою досліджень, у якій методи машинного навчання (ML) демонструють виняткову здатність обробляти складні, багатовимірні набори даних і використовувати великі обсяги історичної інформації. У сучасних дослідженнях провідне місце посідають методи глибокого навчання, які завдяки здатності автоматично виявляти складні нелінійні та просторово-часові залежності перевершують традиційні підходи. Репрезентативними є згорткові (CNN) і рекурентні (RNN) нейронні мережі, а також новітні графові мережі (GNN; зокрема GraphCast) та трансформерні моделі (наприклад, Pangu-Weather), що демонструють високу точність у задачах глобального прогнозування [1, 2].

Ключовим викликом для машинного навчання донедавна було не лише підвищення точності прогнозів, а й моделювання невизначеності. Більшість моделей ML залишалися детермінованими, тобто повертали єдиний прогноз без оцінки ймовірного розподілу можливих результатів, поступаючись у цьому аспекті ансамблевим підходам чисельного прогнозування погоди (NWP). Проривом стало впровадження глибоких генеративних методів, зокрема дифузійних моделей. У дослідженні [2] представлено GenCast – імовірнісну дифузійну модель, здатну самостійно генерувати ансамбль прогнозів. За результатами тестування, GenCast перевищила точність та швидкодію провідної глобальної системи ENS у 97,2 % випадків, що свідчить про здатність сучасних ML-підходів ефективно моделювати не лише середні стани, а й невизначеність і ризики екстремальних погодних явищ [1, 2].

Поряд із розробкою складних DL-моделей, для практичних завдань залишається актуальним застосування класичних методів ML. Для цього активно використовуються хмарні середовища, як-от Google Colaboratory та Kaggle Notebooks. Вони надають готове обчислювальне середовище, дозволяючи користувачам завантажувати власні дані (або обирати з репозиторіїв Kaggle) для аналізу та побудови моделей. Проте такий підхід все одно вимагає від користувача навичок програмування, зазвичай мовою Python, та розуміння методології ML для коректного вибору моделей, налаштування гіперпараметрів та інтерпретації результатів.

Для розв'язання даної проблеми можуть застосовуватися фреймворки автоматизованого машинного навчання (AutoML), що забезпечують автоматизацію процесів вибору методів передбачення та оптимізації їх гіперпараметрів. У галузі обробки табличних даних розроблено низку бібліотек, що реалізують різні методологічні підходи [5].

Auto-Sklearn 2.0 спрямований на досягнення високої результативності за жорстких часових обмежень: підхід PoSH Auto-sklearn поєднує портфель заздалегідь відібраних конфігурацій із розподілом обчислювального бюджету за схемою послідовного відсікання (Successive Halving), що дає змогу виділяти більше ресурсів перспективним конвеєрам і швидше досягати якісних рішень [3].

Натомість, AutoGluon-Tabular не зводить процес до класичної задачі поєднаного вибору алгоритмів і гіперпараметрів (CASH), а робить ставку на потужні ансамблі: багатошаровий стекінг з коректним використанням out-of-fold прогнозів і повторюваний k-fold bagging для підвищення стабільності [4].

До інших релевантних інструментів належить FLAML – легка бібліотека, спроектована для мінімізації обчислювальних витрат під час автоматизованого добору алгоритмів і гіперпараметрів та ефективної роботи за обмежених ресурсів. У свою чергу, платформа H2O AutoML виконує випадковий перебір (random grid search) гіперпараметрів для низки базових алгоритмів і наприкінці формує підсумкові стекінгові ансамблі, що визначають підсумковий рейтинг моделей [5].

Таким чином, подібні фреймворки можуть складати технологічну основу для розробки програмних рішень, які абстрагують складність процесів машинного навчання від кінцевого користувача [5].

Метою роботи є розроблення інтегрованого веб-застосунку, що забезпечує автоматизований повний цикл роботи з погодними даними – від завантаження та базового дослідницького аналізу до побудови моделей передбачення. Застосунок має забезпечувати спрощений доступ до методів машинного навчання для користувачів без спеціальної підготовки, використовуючи фундаментальні моделі з фіксованими гіперпараметрами. Такий підхід дозволить використовувати систему як інструмент швидкої перевірки гіпотез, а також в освітньо-демонстраційних цілях.

Виклад основного матеріалу дослідження

Попередній дослідницький аналіз є важливим етапом роботи з даними, що передбачає вивчення основних характеристик набору, виявлення закономірностей і візуалізацію зв'язків. Його головна мета полягає не лише у початковому дослідженні даних, а й у формуванні надійного підґрунтя для побудови ефективних моделей прогнозування [6, 7].

У реалізованому модулі застосовано основні методики попереднього дослідницького аналізу даних. Обчислюються описові статистики, які дають змогу оцінити міри центральної тенденції (середнє значення та медіану) та розсіювання (стандартне відхилення і міжквартильний розмах). Значна різниця між середнім значенням і медіаною свідчить про наявність асиметрії у розподілі даних. Високе стандартне відхилення може вказувати на потребу масштабування ознак (наприклад, стандартизації) перед використанням у моделях, чутливих до масштабу, зокрема у нейронних мережах. Гістограми дають змогу візуально оцінити форму розподілу, що є важливим для перевірки статистичних припущень окремих алгоритмів, наприклад, лінійної регресії [7].

Кореляційна матриця використовується для виявлення мультиколінеарності – сильного лінійного зв'язку між ознаками-предикторами. Висока мультиколінеарність може дестабілізувати коефіцієнти регресійних моделей, знижуючи їхню інтерпретованість, тому даний аналіз є важливим для відбору ознак. Також, у контексті погодних даних аналіз часових рядів за допомогою лінійних графіків дає змогу візуально виокремити основні компоненти – тренд і сезонність, що є корисним для вибору відповідної моделі прогнозування [7].

Алгоритмічні підходи до передбачення погодних даних. Для задачі прогнозування погодних параметрів на основі структурованих табличних даних, оптимальним є застосування підходів машинного навчання з учителем. На відміну від навчання без учителя чи навчання з підкріпленням, даний підхід дозволяє безпосередньо моделювати залежність між вхідними

ознаками та цільовою змінною. Тому, доцільно розглянути для застосування моделі логістичної регресії, градієнтного бустинга та багатошарового персептрону [9-11].

Логістична регресія – це фундаментальний метод керованого машинного навчання, призначений для задач класифікації. На відміну від лінійної регресії, яка прогнозує неперервні значення, логістична регресія моделює ймовірність належності об'єкта до певного класу [9, 10, 12].

Основою методу є обчислення зваженої суми вхідних ознак та коефіцієнта зміщення. Отриманий результат, який називають логітом, пропускається через сигмоїдну функцію. Така S-подібна функція відображає дійсні числа у діапазон від 0 до 1, що інтерпретується як ймовірність належності до позитивного класу. Рішення про класифікацію приймається шляхом порівняння цих ймовірностей з пороговим значенням, наприклад 0.5 [9, 10, 12].

Навчання логістичної регресії передбачає пошук значень параметрів, які мінімізують функцію втрат. У даному випадку застосовують логістичні втрати, відомі як перехресна ентропія. Дана функція штрафує модель за низьку спрогнозовану ймовірність для правильного класу. Важливою перевагою є її опуклість, яка забезпечує збіжність до глобального мінімуму під час оптимізації, зокрема при застосуванні градієнтного спуску [9, 10, 12].

У випадку багатофакторної класифікації логістична регресія узагальнюється шляхом використання підходу один-проти-всіх або через застосування Softmax регресії. Метод є чутливим до масштабу вхідних даних, тому попереднє масштабування ознак, наприклад шляхом стандартизації, є важливим етапом підготовки даних [12].

Градієнтний бустинг є ансамблевим методом контрольованого машинного навчання, що належить до класу алгоритмів бустингу. Концепція методу ґрунтується на послідовному навчанні слабких моделей, переважно дерев рішень, де кожна наступна модель зменшує залишкові помилки попередніх [9, 10, 13].

На відміну від AdaBoost, який коригує ваги неправильно класифікованих об'єктів на кожній ітерації, градієнтний бустинг розглядає задачу як оптимізацію диференційованої функції втрат. Ітеративний процес зводиться до мінімізації цієї функції, наприклад MSE для регресійних задач або логістичних втрат для задач класифікації, використовуючи підхід, аналогічний градієнтному спуску у функціональному просторі [9, 10, 13].

На кожній ітерації формуються псевдо-залишки, що відповідають негативному градієнту функції втрат відносно поточного прогнозу ансамблю. Наступна базова модель навчається на апроксимацію цих псевдо-залишків, а кінцевий прогноз має адитивну структуру: початкове значення (зазвичай константа, наприклад середнє по цільовій змінній) коригується сумою внесків послідовно побудованих базових моделей [9, 10, 13].

Важливою складовою регуляризації виступає параметр швидкості навчання, який масштабує внесок кожної нової моделі в підсумкове передбачення. Менші значення цього параметра дозволяють зменшити ризик перенавчання, проте потребують більшої кількості ітерацій для досягнення порівняної точності [13].

Багатошаровий персептрон є класом штучних нейронних мереж прямого поширення і вважається базовою архітектурою, на якій ґрунтуються численні підходи глибокого навчання. Його структура містить щонайменше три типи шарів: вхідний, один або декілька прихованих та вихідний. У повнозв'язній архітектурі, характерній для багатошарового персептрона, кожен нейрон одного шару з'єднується з кожним нейроном наступного шару [9, 10, 14].

Інформація у такій мережі поширюється лише в одному напрямку – від входу до виходу. Нейрони прихованих шарів обчислюють зважену суму вхідних сигналів, додають параметр зміщення і застосовують нелінійну функцію активації, наприклад ReLU. Саме нелінійність дає змогу моделі відображати складні нелінійні залежності у даних; без неї навіть глибока мережа мала б потужність, еквівалентну звичайній лінійній моделі [9, 10, 14].

Навчання багатошарового персептрона здійснюється за допомогою алгоритму зворотного поширення помилки. Це метод, що базується на градієнтному спуску, який визначає внесок окремих параметрів у загальну помилку та ітеративно коригує їх для мінімізації функції втрат [9, 10, 14].

Багатошаровий персептрон є універсальним інструментом, який застосовується як у задачах регресії, так і у задачах класифікації. Для задач класифікації у вихідному шарі зазвичай

використовується функція Softmax, яка перетворює вихідні значення мережі на розподіл ймовірностей належності до кожної категорії. Моделі даного типу чутливі до масштабу вхідних ознак, тому їх стандартизація є важливою частиною процесу підготовки даних перед тренуванням [9, 10, 14].

У реалізованому програмному рішенні, що базується на Python бібліотеці Scikit-learn, здійснюється автоматичне визначення типу задачі за типом цільової змінної. Категоріальні ознаки незалежно від наявності або відсутності природного порядку кодуються методом one-hot encoding. Цільова змінна у задачах класифікації перетворюється на числові ідентифікатори шляхом застосування label encoding. Числові ознаки проходять стандартизацію, що забезпечує коректність оптимізації моделей логістичної регресії та багатошарового персептрону. Після завершення етапу підготовки сформований набір даних поділяється на навчальну та тестову підвибірки у співвідношенні 80 до 20.

Для оцінювання ефективності моделей застосовуються метрики, узгоджені з типом задачі. У класифікаційних задачах аналізуються чотири базові показники: точність прогнозування (accuracy), що відображає частку правильних передбачень; точність позитивних рішень (precision), що характеризує частку коректно визначених позитивних об’єктів серед усіх виявлених позитивних; повнота (recall, чутливість), що показує, яку частку позитивних об’єктів модель змогла виявити; та F1-міра (F1-score) як збалансований індикатор, що поєднує точність та повноту у вигляді їх гармонійного середнього.

У регресійних задачах застосовують, зокрема, середню квадратичну помилку (MSE), яка відображає середнє квадратичне відхилення прогнозних значень від фактичних, та коефіцієнт детермінації (R^2), що показує, яку частку варіації цільової змінної пояснює модель [15].

Технологічна платформа та модель даних. Розроблена програмна система побудована за принципами багатокомпонентної архітектури, що охоплює три логічні рівні: клієнтський інтерфейс, сервер прикладної логіки та модуль машинного навчання. Такий підхід забезпечує чітке розмежування функціональних обов’язків між компонентами, спрощує супровід, масштабування та подальший розвиток системи.

Клієнтська частина реалізована з використанням бібліотеки React, що ґрунтується на компонентно-орієнтованій архітектурі та технології односторінкових застосунків. Це забезпечує високий рівень повторного використання елементів інтерфейсу, гнучкість у розробленні користувацьких компонентів і швидке оновлення даних без повного перезавантаження сторінки.

Сервер прикладної логіки розроблено на платформі Node.js із використанням фреймворку Express.js. Серверна частина реалізує RESTful API, підтримує асинхронну обробку запитів і механізм автентифікації користувачів на основі JWT-токенів. Доступ до даних здійснюється за допомогою ODM-бібліотеки Mongoose, що забезпечує зручну роботу з об’єктною моделлю документно-орієнтованої бази даних.

Для збереження інформації використано документно-орієнтовану СКБД MongoDB. Гнучка структура документів у JSON-сумісному форматі дає змогу еволюційно змінювати модель даних без складних міграцій, що спрощує подальший розвиток та підвищує адаптивність системи до змін вимог.

Взаємодію між клієнтським інтерфейсом і сервером прикладної логіки побудовано за принципами RESTful API, що забезпечує стандартизований обмін даними та спрощує інтеграцію.

Маршрутизацію клієнтських запитів реалізовано, зокрема, у модулі notesRoutes.js, який визначає основні кінцеві точки (endpoints) для роботи із погодними записами Note: отримання метаданих усіх або окремих погодних записів, імпорт і оновлення CSV-файлів, видалення записів, а також завантаження оригінальних і модифікованих даних.

Окрім базових CRUD-операцій, передбачено швидкий первинний аналіз даних: спеціальні кінцеві точки повертають унікальні рядкові значення та діапазони числових полів для записів, до яких було завантажено CSV-дані.

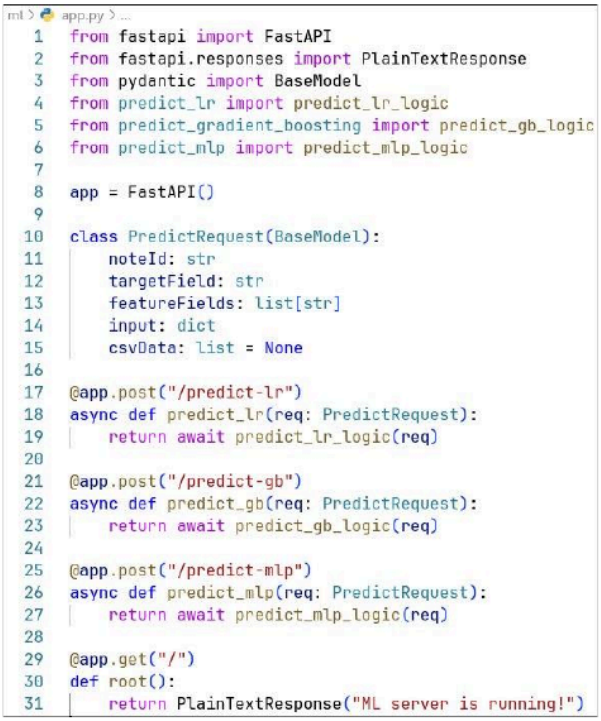
На рисунку 1 наведено фрагмент коду маршрутизатора, який ілюструє відповідність між основними клієнтськими запитами та методами контролера керування записами Note.

```
backend > src > routes > JS notesRoutes.js > ...
import express from "express";
import multer from "multer";
import {
    getAllNotesMeta,
    createNote,
    updateNote,
    deleteNote,
    getNoteById,
    getNoteMetaById,
    downloadCSVOriginal,
    downloadCSVModified,
    getUniqueStringValues,
    getNumericRanges
} from "../controllers/notesController.js";

const router = express.Router();

const storage = multer.memoryStorage();
const upload = multer({ storage: storage });

router.get("/", getAllNotesMeta);
router.get("/:id/meta", getNoteMetaById);
router.get("/:id", getNoteById);
router.get("/:id/download-original", downloadCSVOriginal);
router.get("/:id/download-modified", downloadCSVModified);
router.get("/:id/unique-string-values", getUniqueStringValues);
router.get("/:id/numeric-ranges", getNumericRanges);

router.post("/", upload.single("csvFile"), createNote);
router.put("/:id", upload.single("csvFile"), updateNote);
router.delete("/:id", deleteNote);

export default router;
```

Рис. 1. Фрагмент коду реалізації маршрутизатора notesRoutes.js

Модуль машинного навчання реалізовано як автономний мікросервіс мовою Python на базі FastAPI. Даний компонент виконує ключові етапи опрацювання даних: приймання та попередню обробку вхідної інформації, використання методів машинного навчання з бібліотеки Scikit-learn для одержання передбачень, а також взаємодію з сервером прикладної логіки через стандартні HTTP-запити. Для обчислень використано NumPy, для серіалізації моделей – Joblib. Обрана мікросервісна архітектура сприяє масштабованості, ізоляції компонентів і незалежному розвитку підсистем.

На рисунку 2 наведено структуру Python-сервісу (app.py), який реалізує точки доступу для передбачення погодних даних (зокрема, /predict-lr, /predict-gb, /predict-mlp).



```
ml > app.py > ...
from fastapi import FastAPI
from fastapi.responses import PlainTextResponse
from pydantic import BaseModel
from predict_lr import predict_lr_logic
from predict_gradient_boosting import predict_gb_logic
from predict_mlp import predict_mlp_logic

app = FastAPI()

class PredictRequest(BaseModel):
    noteId: str
    targetField: str
    featureFields: list[str]
    input: dict
    csvData: list = None

@app.post("/predict-lr")
async def predict_lr(req: PredictRequest):
    return await predict_lr_logic(req)

@app.post("/predict-gb")
async def predict_gb(req: PredictRequest):
    return await predict_gb_logic(req)

@app.post("/predict-mlp")
async def predict_mlp(req: PredictRequest):
    return await predict_mlp_logic(req)

@app.get("/")
def root():
    return PlainTextResponse("ML server is running!")
```

Рис. 2. Фрагмент реалізації сервісу машинного навчання на Python із використанням FastAPI

Концептуальна модель даних системи, яка подана на рисунку 3, описує структуру основних інформаційних сутностей – користувача (User) та погодного запису (Note). Реалізація моделі здійснюється засобами документно-орієнтованої бази даних MongoDB, у якій кожна сутність представлена окремим документом.

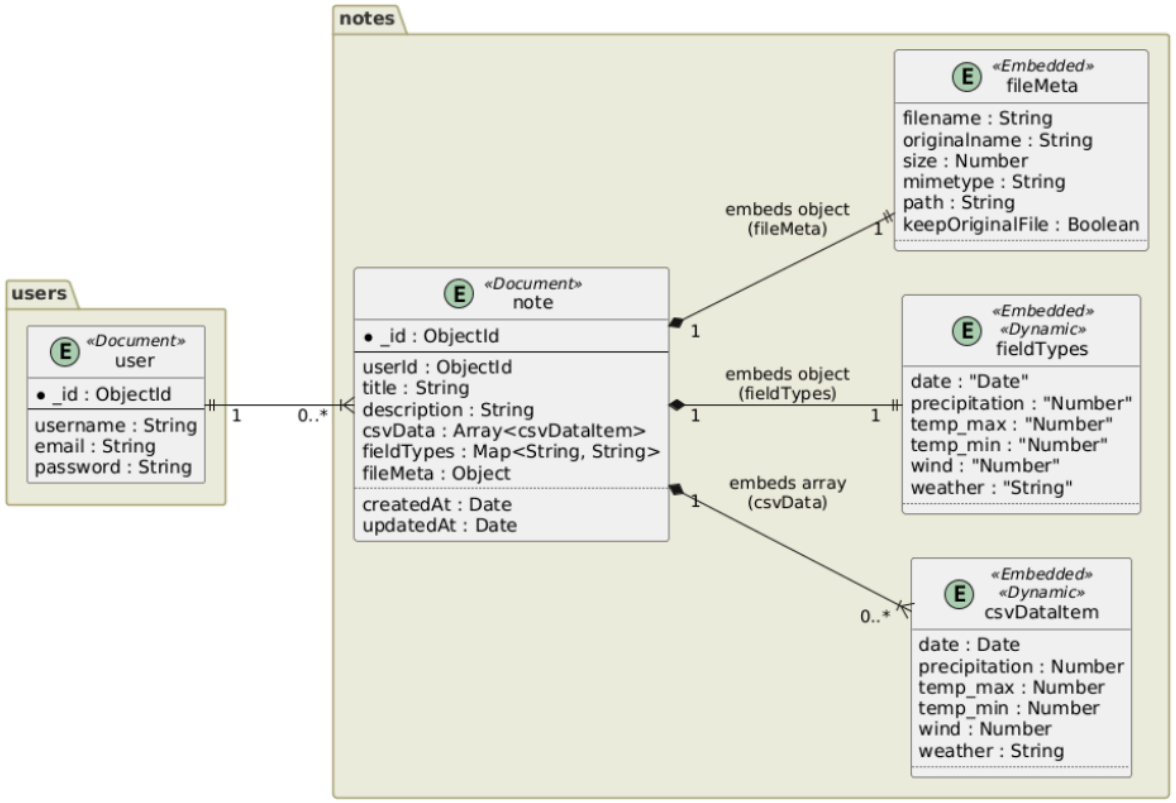


Рис. 3. Концептуальна UML-модель структури бази даних

Документ Note містить посилання на власника, текстові поля заголовка й опису, а також вбудовані структури для збереження табличної інформації, імпортованої користувачем. Поле csvData реалізоване як масив вбудованих об’єктів довільної структури, тоді як fieldTypes є відображенням типів даних для кожної колонки, що формується автоматично під час імпорту.

Підструктура fileMeta є вбудованим об’єктом, який містить метадані файлу – ім’я, шлях, розмір, MIME-тип – і прапорець keepOriginalFile, що визначає політику збереження фізичної копії на диску.

Впроваджена документно-орієнтована модель забезпечує гнучкість, масштабованість і розширюваність структури даних, дозволяючи інтегрувати нові типи інформації без зміни базової схеми – що є досить важливим для систем, орієнтованих на аналіз і візуалізацію користувацьких даних.

Висновки

У статті запропоновано та розроблено інтегроване веб-орієнтоване програмне рішення для аналізу та передбачення погодних умов із використанням методів машинного навчання. Система забезпечує повний цикл роботи з даними: від імпорту невеликих користувацьких наборів і виконання базового дослідницького аналізу до побудови моделей передбачення.

Ключовою особливістю реалізації є застосування фундаментальних методів машинного навчання, зокрема логістичної регресії, градієнтного бустинга та багатошарового персептрону з фіксованими гіперпараметрами. Такий підхід усуває потребу в їх ручному налаштуванні та спрощує процес побудови прогнозів, дозволяючи користувачу зосередитися на виборі цільової змінної, ознак і подальшій інтерпретації результатів.

Розроблений застосунок може використовуватись як освітньо-демонстраційний інструмент для ознайомлення з фундаментальними методами машинного навчання, а також як практичний засіб для отримання передбачень на локальних наборах погодних даних.

Подальший розвиток системи може бути зосереджений на інтеграції моделей прогнозування часових рядів, таких як ARIMA та SARIMA, що дасть змогу враховувати сезонні та автокореляційні властивості кліматичних процесів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Machine Learning Methods for Weather Forecasting: A Survey / H. Zhang та ін. *Atmosphere*. 2025. Т. 16, № 1. С. 82. URL: https://doi.org/10.3390/atmos16010082 (дата звернення: 07.11.2025).
2. Probabilistic weather forecasting with machine learning / I. Price та ін. *Nature*. 2024. URL: https://doi.org/10.1038/s41586-024-08252-9 (дата звернення: 07.11.2025).
3. Feurer M., Eggensperger K., Falkner S., Lindauer M., Hutter F. Auto-sklearn 2.0: Hands-free automl via meta-learning // Journal of Machine Learning Research. – 2022. – Vol. 23, № 261. – С. 1–61.
4. Erickson N., Mueller J., Shirkov A., Zhang H., Larroy P., Li M., Smola A. Autogluon-tabular: Robust and accurate automl for structured data // arXiv preprint arXiv:2003.06505. – 2020. – URL: https://arxiv.org/abs/2003.06505 (дата звернення: 07.11.2025).
5. AMLB: Frameworks. *AMLB An AutoML Benchmark*. URL: https://openml.github.io/automlbenchmark/frameworks.html (дата звернення: 07.11.2025).
6. What is Exploratory Data Analysis? – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/data-analysis/what-is-exploratory-data-analysis (дата звернення: 07.11.2025).
7. Advanced EDA – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/data-science/advanced-eda (дата звернення: 07.11.2025).
8. Geron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. – 2nd ed. – Sebastopol (CA) : O'Reilly Media, Inc., 2019. – 851 с.
9. Raschka S., Liu Y., Mirjalili V. Machine Learning with PyTorch and Scikit-Learn. – Birmingham : Packt Publishing Ltd., 2022. – 771 с.
10. Supervised Machine Learning – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/machine-learning/supervised-machine-learning (дата звернення: 07.11.2025).
11. Logistic Regression in Machine Learning – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/machine-learning/understanding-logistic-regression (дата звернення: 07.11.2025).
12. Gradient Boosting in ML – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/machine-learning/ml-gradient-boosting (дата звернення: 07.11.2025).
13. Multi-layer Perceptron: a Supervised Neural Network Model using Sklearn – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/deep-learning/multi-layer-perceptron-a-supervised-neural-network-model-using-sklearn (дата звернення: 07.11.2025).
14. Gutta S. Machine Learning Metrics in simple terms. *Medium*. URL: https://medium.com/analytics-vidhya/machine-learning-metrics-in-simple-terms-d58a9c85f9f6 (дата звернення: 07.11.2025).

Автори статті

Мельник Юрій – доктор технічних наук, професор, завідувач Кафедри робототехніки та технічних систем, Державний університет інформаційно-комунікаційних технологій, Київ, Україна.

ORCID: 0000-0002-5028-8749

Отрох Сергій – доктор технічних наук, професор, професор Кафедри цифрових технологій в енергетиці, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.
ORCID: 0000-0001-9008-0902
Постернак Антон – студент, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.
ORCID: 0009-0008-5613-9227

Authors of the article

Melnyk Yurii – Doctor of Sciences (technical), Professor, Head of the Department of Robotics and Technical Systems, State University of Information and Communication Technologies, Kyiv, Ukraine.
ORCID: 0000-0002-5028-8749
Otrokh Serhii – Doctor of Sciences (technical), Professor, Professor of the Department of Digital Technologies in Energy, National Technical University of Ukraine “Ihor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine.
ORCID: 0000-0001-9008-0902
Posternak Anton – student, National Technical University of Ukraine “Ihor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine.
ORCID: 0009-0008-5613-9227