УДК 004.8:004.42:004.65 DOI: 10.31673/2786-8362.2025.023547

Мельник Ю.В., д.т.н.; **Отрох С.І.,** д.т.н.;
Беспала О.М., Постернак А.В.

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

Melnyk Yu.V., Otrokh S.I., Bespala O.M., Posternak A.V. 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

Мельник Ю.В., Отрох С.І., Беспала О.М., Постернак А.В. 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

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. 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.

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).

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

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.

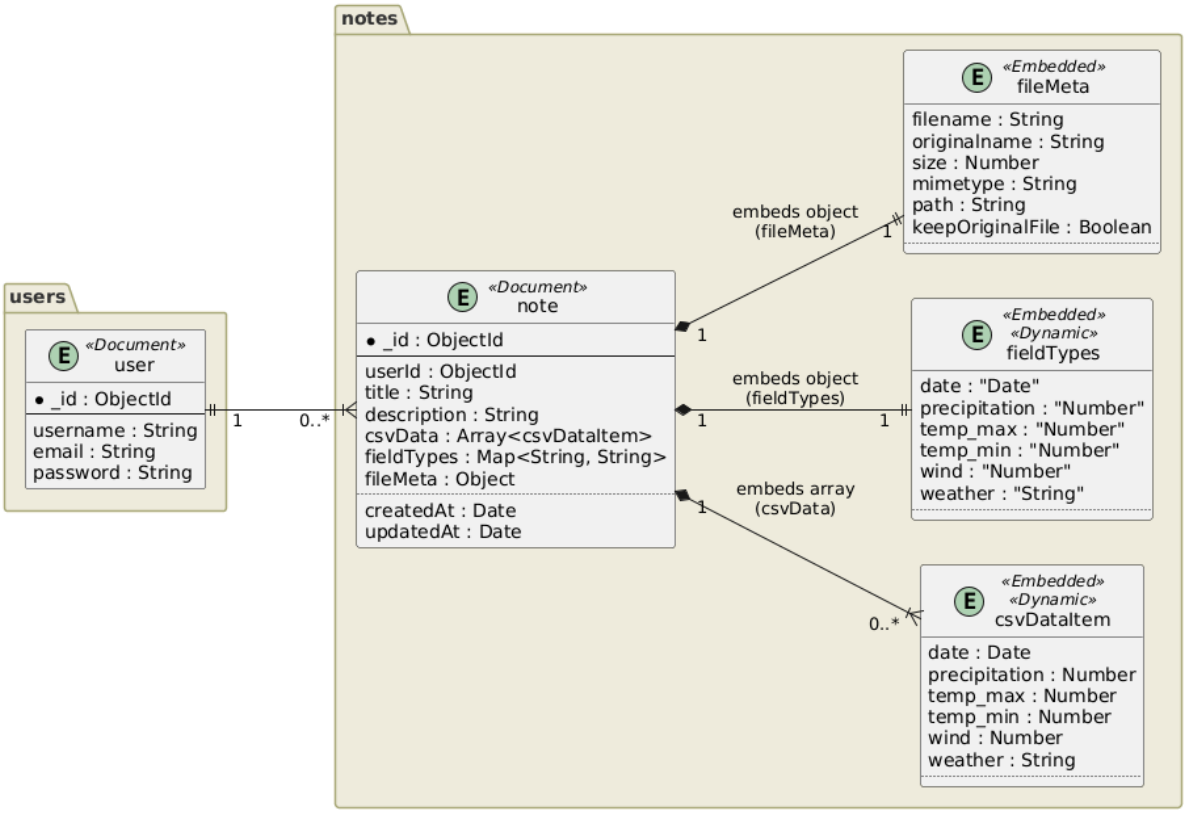

Рис. 3. Концептуальна UML-модель структури бази даних

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.

2. Probabilistic weather forecasting with machine learning / I. Price та ін. *Nature*. 2024. URL: https://doi.org/10.1038/s41586-024-08252-9.

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.

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.

7. Advanced EDA – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/data-science/advanced-eda.

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.

11. Logistic Regression in Machine Learning – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/machine-learning/understanding-logistic-regression.

12. Gradient Boosting in ML – *GeeksforGeeks*. – URL: https://www.geeksforgeeks.org/machine-learning/ml-gradient-boosting.

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.

14. Gutta S. Machine Learning Metrics in simple terms. *Medium*. URL: https://medium.com/analytics-vidhya/machine-learning-metrics-in-simple-terms-d58a9c85f9f6.

Автори статті

Мельник Юрій – доктор технічних наук, професор, Державний університет інформаційно-комунікаційних технологій, Київ, Україна.

ORCID: 0000-0002-5028-8749

Отрох Сергій – доктор технічних наук, професор, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

ORCID: 0000-0001-9008-0902

Беспала Ольга – старший викладач, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна.

ORCID: 0000-0003-3285-2585

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, State University of Information and Communication Technologies, Kyiv, Ukraine.

ORCID: 0000-0002-5028-8749

Otrokh Serhii – Doctor of Sciences (technical), Professor, National Technical University of Ukraine "Ihor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine.

ORCID: 0000-0001-9008-0902

Bespala Olha – Senior Lecturer, National Technical University of Ukraine “Ihor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine.
ORCID: 0000-0003-3285-2585
Posternak Anton – student, National Technical University of Ukraine “Ihor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine.
ORCID: 0009-0008-5613-9227