МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ

В. о. завідувача кафедри інтелектуальних інформаційних систем

_____ Євген СІДЕНКО

«____»_____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

ІНТЕЛЕКТУАЛЬНА СИСТЕМА КЛАСИФІКАЦІЇ ЕКОЛОГІЧНИХ ПОКАЗНИКІВ З ВРАХУВАННЯМ ДИСБАЛАНСУ КЛАСІВ

Спеціальність 122 Комп'ютерні науки

Освітня програма «Інтелектуальні інформаційні системи»

Здобувач _____ Єлизавета КУЛІШОВА

«___»_____ 2025 р.

Керівник д-р. техн. наук, професор _____ Ірина КАЛІНІНА

«___»_____ 2025 р.

Миколаїв – 2025

Чорноморський національний університет імені Петра Могили
(повне найменування закладу вищої освіти)

Факультет	Комп’ютерних наук
Кафедра	Інтелектуальних інформаційних систем
Рівень вищої освіти	Другий (магістерський)
Освітній ступень	Магістр
Спеціальність	122 Комп’ютерні науки
Освітня програма	Інтелектуальні інформаційні системи

ЗАТВЕРДЖУЮ

В. о. завідувача кафедри інтелектуальних інформаційних систем

_____ Євген СІДЕНКО

«____» _____ 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу здобувача

Кулішової Єлизавети Сергіївни
(прізвище, ім’я, по батькові здобувача)

1. Тема кваліфікаційної роботи: «Інтелектуальна система класифікації екологічних показників з врахуванням дисбалансу класів».

Керівник роботи: Калініна Ірина Олександрівна, професор кафедри ІІС, доктор техн. наук, професор.

Затверджена наказом ЧНУ ім. Петра Могили від «7» липня 2025 р. № 184.

2. Строк представлення кваліфікаційної роботи «15» грудня 2025 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні: розробка інтелектуальної системи класифікації екологічних показників якості з урахуванням дисбалансу класів, яка забезпечує підвищення точності розпізнавання рідкісних та небезпечних екологічних станів. Досягнення цього результату передбачає проведення аналізу предметної області, застосування сучасних методів препроцесінгу, балансування даних та алгоритмів машинного навчання, а також

порівняння їхньої ефективності шляхом експериментальних досліджень. Початковими даними є екологічні показники якості, наукові джерела, електронні набори даних, програмне забезпечення RStudio та бібліотеки для аналізу даних і моделювання.

4. Перелік питань, що підлягають розробці: аналіз сучасних підходів до класифікації екологічних показників та дослідити проблему дисбалансу класів; методи побудови інтелектуальних систем та алгоритми машинного навчання, орієнтовані на роботу з даними в яких наявний дисбаланс; попередня обробка екологічних даних: очищення, нормалізація, обробка пропусків, балансування вибірки спеціальними методами; формування навчальної та тестової вибірки для моделювання; реалізація та дослідження ефективності моделей класифікації (методи чутливі до втрат, методи ансамблевого навчання); порівняння моделей за метриками, стійкими до дисбалансу; експериментальна перевірка роботи системи та сформовані висновки.

5. Перелік графічних матеріалів: презентація.

Керівник роботи _____ Ірина КАЛІНІНА
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Здобувач _____ Єлизавета КУЛІШОВА
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата видачі завдання «28» червня 2025 р.

КАЛЕНДАРНИЙ ПЛАН
кваліфікаційної роботи

Тема: Інтелектуальна система класифікації екологічних показників з врахуванням дисбалансу класів

№	Найменування роботи	Початок	Закінчення	Примітки
1	Отримання завдання на виконання КР	27.06.2025	29.06.2025	
2	Аналіз предметної області та постановка задачі	30.06.2025	20.07.2025	
3	Огляд літературних джерел за темою кваліфікаційної роботи, зокрема аналіз публікацій, вивчення проблеми дисбалансу класів та сучасних підходів	21.07.2025	30.07.2025	
4	Дослідження та реалізація методів навчання моделей при дисбалансі	01.09.2025	25.10.2025	
5	Експериментальні дослідження та отримання результатів	26.10.2025	21.11.2025	
6	Перший попередній захист КР на засіданні комісії кафедри	25.11.2025	26.11.2025	
7	Корегування роботи за результатами попереднього захисту	27.11.2025	05.12.2025	
8	Другий попередній захист КР на засіданні комісії кафедри	09.12.2025	09.12.2025	
9	Доробка та остаточне оформлення КР	10.12.2025	12.12.2025	
10	Подання КР, її електронної копії та інших документів (відгуку, рецензії) до захисту	15.12.2025	16.12.2025	

Керівник роботи _____ Ірина КАЛІНІНА
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Здобувач _____ Єлизавета КУЛІШОВА
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата складання календарного плану
«6» липня 2025 р.

АНОТАЦІЯ

до кваліфікаційної роботи
здобувачки групи 601 ЧНУ ім. Петра Могили

Кулішової Єлизавети Сергіївни

на тему: "ІНТЕЛЕКТУАЛЬНА СИСТЕМА КЛАСИФІКАЦІЇ ЕКОЛОГІЧНИХ ПОКАЗНИКІВ З ВРАХУВАННЯМ ДИСБАЛАНСУ КЛАСІВ"

Актуальність даного дослідження полягає у необхідності підвищення точності автоматизованого аналізу екологічних даних та раннього виявлення небезпечних станів довкілля. Сучасні системи моніторингу часто стикаються з проблемою дисбалансу класів, що ускладнює коректне розпізнавання рідкісних, але критично важливих екологічних відхилень. Використання методів машинного навчання у поєднанні з ефективними способами балансування вибірки дає можливість підвищити якість класифікації та забезпечити більш надійну підтримку прийняття рішень у сфері екологічної безпеки.

Об'єктом дослідження є процес класифікації екологічних показників якості.

Предметом дослідження є методи машинного навчання, орієнтовані на класифікацію екологічних даних із урахуванням дисбалансу класів.

Метою дослідження є розробка інтелектуальної системи класифікації якості екологічних показників на основі їх параметрів з урахуванням дисбалансу класів для підвищення точності розпізнавання небезпечних станів.

У результаті виконання роботи було застосовано методи статистичного аналізу, алгоритми препроцесінгу даних, способи балансування вибірки, алгоритми машинного навчання та методи оцінювання моделей, стійкі до дисбалансу. Проведено порівняльний аналіз моделей та визначено оптимальні підходи до класифікації екологічних показників у задачах із високою нерівномірністю розподілу класів. Практичну реалізацію системи виконано у середовищі RStudio.

Дана робота складається з чотирьох розділів. Кожен розділ відповідно присвячений: аналізу предметної області та проблеми дисбалансу класів; методам препроцесінгу, очищення та балансування екологічних даних; дослідженню алгоритмів машинного навчання, чутливих до дисбалансу, та методів оцінювання моделей; практичній реалізації інтелектуальної системи, проведенню експериментів та аналізу отриманих результатів. Загальний обсяг роботи – 125 сторінки. Кваліфікаційна робота містить 1 додаток, 65 рисунків, 9 таблиць і 45 джерел посилання.

Ключові слова: екологічні дані, класифікація, машинне навчання, дисбаланс класів, препроцесінг, балансування вибірки, RStudio, інтелектуальна система.

ABSTRACT

to the qualification work by the student of the group 601 of Petro Mohyla Black Sea National University

Kulishova Yelyzaveta

"INTELLIGENT SYSTEM FOR CLASSIFICATION OF ENVIRONMENTAL INDICATORS WITH CONSIDERATION OF CLASS IMBALANCE"

The relevance of this research lies in the need to improve the accuracy of automated analysis of environmental data and the early detection of hazardous environmental conditions. Modern monitoring systems often face the problem of class imbalance, which complicates the correct recognition of rare but critically important environmental anomalies. The use of machine learning methods in combination with effective sampling balancing techniques makes it possible to increase classification quality and ensure more reliable decision-making support in the field of environmental safety.

The object of the study is the process of classifying environmental quality indicators.

The subject of the study is machine learning methods aimed at classifying environmental data with consideration of class imbalance.

The purpose of the study is to develop an intelligent system for classifying environmental quality indicators based on their parameters, taking into account class imbalance, in order to improve the accuracy of detecting hazardous conditions.

In the course of the work, methods of statistical analysis, data preprocessing algorithms, sampling balancing techniques, machine learning algorithms, and model evaluation methods resistant to class imbalance were applied. A comparative analysis of models was conducted, and optimal approaches for classifying environmental indicators in tasks with a highly imbalanced class distribution were identified. The practical implementation of the system was carried out in the RStudio environment.

This work consists of four chapters. Each chapter is devoted respectively to: the analysis of the subject area and the problem of class imbalance; methods of preprocessing, cleaning, normalization, and balancing environmental data; the study of machine learning

algorithms sensitive to class imbalance and methods for evaluating model performance; the practical implementation of the intelligent system, experiments, and analysis of the obtained results. The total volume of the work is 125 pages. The qualification thesis includes 1 appendix, 65 figures, 9 tables, and 45 references.

Key words: environmental data, classification, machine learning, class imbalance, preprocessing, sampling balancing, RStudio, intelligent system.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ІА – інтелектуальний алгоритм

ІС – інтелектуальна система

КР – кваліфікаційна робота

МН – машинне навчання

ADASYN – Adaptive Synthetic Sampling Approach for Imbalanced Learning (Адаптивний синтетичний підхід до вибірки для незбалансованого навчання)

AUC – Area Under Curve (Площа під кривою)

FN – False Negative (Хибно негативний результат)

FP – False Positive (Хибно позитивний результат)

KNN – Nearest Neighbors (Метод k-найближчих сусідів)

MCC – Matthews Correlation Coefficient (Коефіцієнт кореляції Меттьюса)

PCA – Principal Component Analysis (Метод головних компонент)

RBF – Radial Basis Function (Радіальна базисна функція)

ROC – Receiver Operating Characteristic (Робоча характеристика приймача)

ROSE – Random Over-Sampling Examples (Генерація нових прикладів з урахуванням розподілу класів)

RUS – Random UnderSampling (Випадкове зменшення вибірки)

SMOTE – Synthetic Minority Over-sampling Technique (Техніка синтетичного генерування меншості)

SVM – Support Vector Machine (Метод опорних векторів)

TN – True Negative (Істинно негативний результат)

TP – True Positive (Істинно позитивний результат)

XGBoost – Extreme Gradient Boosting (Екстремальний градієнтний бустинг)

ВСТУП

В умовах глобальної урбанізації, розвитку промисловості та інтенсивного використання природних ресурсів питання збереження екологічної безпеки набуває особливої актуальності. Стан навколишнього середовища безпосередньо впливає на якість життя населення, здоров’я людей і стабільність природних екосистем. Одним із найважливіших аспектів екологічного моніторингу є контроль за якістю води, адже саме водні ресурси забезпечують функціонування більшості сфер людської діяльності. Забруднення водних об’єктів токсичними речовинами, важкими металами чи мікроорганізмами призводить до деградації природних систем, втрати біорізноманіття та загрози для здоров’я людей.

Сучасний рівень розвитку інформаційних технологій дозволяє застосовувати інтелектуальні методи для аналізу великих обсягів екологічних даних. Такі підходи забезпечують автоматизацію процесів моніторингу, аналізу тенденцій і прогнозування екологічних ризиків. Проте, при роботі з реальними екологічними наборами даних виникає низка проблем, серед яких найважливішою є дисбаланс класів. У більшості випадків дані про забруднення чи аномальні ситуації представлені значно меншою кількістю спостережень, ніж нормальні умови. Це призводить до того, що алгоритми машинного навчання (МН) орієнтуються переважно на більшість і недостатньо ефективно розпізнають критичні випадки, які мають найбільшу екологічну цінність.

У зв’язку з цим особливої ваги набуває розробка інтелектуальних систем (ІС), здатних враховувати дисбаланс класів під час класифікації екологічних показників. Такі системи повинні забезпечувати точне розпізнавання рідкісних, але небезпечних екологічних станів, що є ключовим для своєчасного реагування та запобігання екологічним катастрофам.

Актуальність теми полягає в необхідності створення ефективних інструментів автоматизованого аналізу екологічних показників на основі їх хімічних та фізичних показників із використанням сучасних методів МН. Розвиток систем екологічного моніторингу вимагає впровадження інтелектуальних підходів,

які не лише підвищують точність класифікації, але й враховують особливості даних, притаманні природним процесам, зокрема їх нерівномірність, варіативність та неповноту.

Робота спирається на загальні підходи до аналізу екологічних даних та є логічним продовженням сучасних досліджень, присвячених розробці інтелектуальних систем екологічного призначення. При цьому вона не є продовженням інших робіт автора, але використовує напрацювання у галузі МН, балансування вибірок та методів екологічного моніторингу.

Метою кваліфікаційної роботи (КР) є розробка ІС класифікації екологічних показників на основі їх екологічних параметрів з урахуванням дисбалансу класів для підвищення точності прийняття рішень щодо її придатності до споживання.

Для досягнення мети поставлено такі **основні завдання**, як:

– провести аналіз предметної області;

– вивчити сучасні методи МН, які застосовуються для класифікації екологічних даних;

– розглянути існуючі методи обробки дисбалансних даних та оцінити їх ефективність у задачах класифікації;

– розробити алгоритм ІС, що враховує дисбаланс класів при навчанні моделей;

– реалізувати систему класифікації мовою програмування R у середовищі Rstudio;

– провести експериментальне дослідження та оцінку якості роботи системи з використанням збалансованих і незбалансованих наборів даних;

– порівняти ефективність різних моделей класифікації за метриками, стійкими до дисбалансу класів.

Об’єктом дослідження є процес класифікації екологічних показників якості.

Предметом дослідження є методи МН для класифікації екологічних даних з урахуванням дисбалансу класів у вхідних наборах.

Методи дослідження включають аналіз наукових джерел, статистичний аналіз, методи попередньої обробки даних (препроцесінгу), алгоритми МН, методи балансування даних, а також методи оцінювання ефективності моделей. Практична реалізація здійснюється у середовищі RStudio з використанням спеціальних бібліотек.

Наукова новизна роботи полягає у поєднанні методів балансування даних та інтелектуальних алгоритмів (ІА) класифікації для підвищення точності розпізнавання екологічних станів. У межах дослідження передбачається розроблення підходу, який дозволить мінімізувати вплив домінуючих класів, забезпечуючи справедливе навчання моделей навіть у разі значного перекосу у вибірці.

Практичне значення результатів полягає у можливості застосування розробленої системи в установах, що займаються екологічним моніторингом та управлінням якістю водних ресурсів. Система може бути використана для автоматизованого контролю стану, виявлення небезпечних тенденцій і прогнозування ризиків забруднення. Крім того, результати дослідження можуть бути адаптовані для інших типів екологічних даних – якості повітря, ґрунтів або атмосферних показників.

Розроблена система дозволить суттєво зменшити час аналізу великих обсягів даних, підвищити точність класифікації, забезпечити раннє виявлення відхилень і сприяти ухваленню ефективних управлінських рішень у сфері екологічної безпеки.

Теоретичне значення полягає в узагальненні та систематизації підходів до класифікації екологічних показників із використанням методів МН та балансування даних. Отримані результати можуть бути використані як основа для подальших наукових досліджень у галузі екологічної інформатики та інтелектуального аналізу даних.

Для досягнення поставленої мети у роботі використано комплексний підхід, що включає:

- аналітичний етап – вивчення предметної області та наукових джерел;

- експериментальний етап – створення та навчання моделей класифікації;
- оцінювальний етап – перевірку ефективності системи на реальних або змодельованих даних.

Структура КР відповідає поставленій меті та логіці дослідження. Вона складається зі вступу, чотирьох розділів, висновків, списку використаних джерел та додатків.

У **першому розділі** розглянуто теоретичні основи інтелектуальної класифікації екологічних даних, поняття екологічних показників, принципи побудови ІС і проблему дисбалансу класів у задачах класифікації.

Другий розділ присвячено етапам підготовки даних до моделювання: очищенню, нормалізації, заповненню пропусків, а також застосуванню методів балансування вибірки. Особливу увагу приділено методам врахування дисбалансу.

У **третьому розділі** описано процес навчання моделей МН, проведено експериментальні дослідження, порівняно різні алгоритми та оцінено їхню ефективність на дисбалансних наборах даних.

Четвертий розділ передбачає розробку практичного застосування системи в середовищі RStudio, реалізацію інтерфейсу для завантаження екологічних даних, візуалізацію результатів класифікації та аналіз отриманих метрик.

КР спрямована на створення ІС, здатної підвищити ефективність аналізу екологічних даних за рахунок врахування дисбалансу класів. Вона поєднує теоретичні основи МН, сучасні підходи до обробки даних і практичну реалізацію в середовищі R. Отримані результати сприятимуть розвитку екологічного моніторингу, забезпеченню сталого управління водними ресурсами та підвищенню рівня екологічної безпеки.

1 ТЕОРЕТИЧНІ ОСНОВИ ІНТЕЛЕКТУАЛЬНОЇ КЛАСИФІКАЦІЇ ЕКОЛОГІЧНИХ ДАНИХ

1.1 Аналіз предметної області

Екологічні показники є основними характеристиками, що відображають стан природного середовища та рівень його забруднення. Вони дозволяють кількісно оцінювати вплив антропогенних і природних факторів на довкілля, визначати тенденції його змін і розробляти заходи для збереження екологічної рівноваги. До основних груп екологічних показників належать показники якості повітря, води, ґрунту, рівня шуму, радіаційного фону та обсягів викидів у навколишнє середовище.

Якість повітря характеризується концентраціями шкідливих речовин, таких як діоксид азоту (NO_2), діоксид сірки (SO_2), чадний газ (CO), тверді частинки (PM2.5, PM10) та озон (O_3) [1]. Ці речовини впливають на здоров’я людини, кліматичні процеси та стан екосистем. Показники якості води включають значення кислотності (pH), хімічного споживання кисню (ХСК), вміст нітратів, фосфатів, важких металів і мікроорганізмів. Забруднення водних ресурсів безпосередньо впливає на стан біосфери та якість питної води.

Ґрунтові показники відображають концентрацію поживних елементів, рівень токсичних речовин, залишки пестицидів і радіонуклідів. Зміна цих параметрів свідчить про деградацію земель, що є серйозною екологічною проблемою. Крім того, важливими є *показники кліматичного стану*, такі як температура, вологість, атмосферний тиск, швидкість вітру, кількість опадів. Ці фактори впливають на процеси поширення забруднювачів у навколишньому середовищі.

Моніторинг екологічних показників є необхідним для забезпечення сталого розвитку та охорони природи. Система моніторингу здійснює регулярний збір, зберігання, аналіз та інтерпретацію екологічних даних. Основна мета такого моніторингу – виявлення негативних тенденцій у стані довкілля та запобігання екологічним катастрофам. Зібрані показники використовуються для прийняття

управлінських рішень у сфері екологічної безпеки, планування територій і контролю діяльності промислових підприємств.

Проте обсяг екологічних даних постійно зростає, і ручна їх обробка стає практично неможливою. Дані надходять з численних сенсорів, супутників, лабораторій і спостережних станцій. У зв'язку з цим виникає необхідність *автоматизації класифікації екологічних показників*. Автоматизовані системи здатні швидко аналізувати великі масиви даних, виявляти приховані закономірності, визначати рівні забруднення та прогнозувати подальші зміни.

Класифікація екологічних показників передбачає розподіл даних за певними категоріями, наприклад за рівнем екологічної небезпеки або типом забруднення. Це дозволяє підвищити ефективність прийняття рішень у сфері екологічного управління. Автоматизовані методи забезпечують точність, стабільність та об'єктивність оцінок, усуваючи суб'єктивний фактор, притаманний ручним методам аналізу.

Важливу роль у сучасному екологічному аналізі відіграють *інтелектуальні системи*, які використовують алгоритми МН для автоматичного розпізнавання, класифікації та прогнозування екологічних процесів. Такі системи не лише аналізують поточні показники, а й здатні виявляти аномалії, визначати джерела забруднення та передбачати можливі наслідки.

Однією з ключових причин автоматизації є *підвищення швидкості реагування на зміни в екологічних даних*. Наприклад, автоматична класифікація може виявити перевищення гранично допустимих концентрацій у реальному часі та попередити відповідні служби. Це дає змогу оперативно вживати заходів для запобігання небезпечним екологічним ситуаціям.

У світі існує низка систем екологічного моніторингу, що активно використовуються для збору й аналізу даних. Європейська система *EIONET (European Environment Information and Observation Network)* [2] координує обмін інформацією між країнами ЄС і забезпечує аналітичну підтримку екологічних

політик. Американська система *AirNow* [3] надає відкриті дані про якість повітря в режимі реального часу для громадян та дослідників.

Серед глобальних систем варто відзначити *Copernicus Atmosphere Monitoring Service (CAMS)* [4], яка використовує супутникові технології для моніторингу стану атмосфери. В Україні функціонує національна система *EcoCity* [5], яка збирає та обробляє дані з автоматизованих станцій спостереження за якістю повітря. Ці системи забезпечують не лише контроль, а й підтримку державних рішень у сфері екологічної політики.

У сучасних умовах дедалі більше систем переходять до використання *інтелектуальних алгоритмів* для автоматичної обробки та класифікації екологічних показників. Методи МН дозволяють підвищити точність визначення класів забруднення та оптимізувати аналіз даних із різних джерел. Наприклад, штучні нейронні мережі застосовуються для аналізу часових рядів показників забруднення, а алгоритми кластеризації – для групування схожих регіонів за рівнем екологічного ризику.

Важливою особливістю екологічних даних є їхня *висока варіативність і нерівномірність розподілу*, що ускладнює процес класифікації. У багатьох випадках дані містять пропуски або спостереження, які належать до рідкісних класів, наприклад до випадків надзвичайного забруднення. Тому створення систем, які враховують *дисбаланс класів*, є актуальним завданням для підвищення точності класифікації.

Загалом екологічні показники є основою для створення аналітичних і прогнозних моделей, що підтримують прийняття рішень у сфері охорони навколишнього середовища. Їх автоматизований аналіз дозволяє швидко реагувати на зміни, прогнозувати можливі наслідки забруднення та підвищувати ефективність екологічного менеджменту.

Таким чином, предметна область дослідження охоплює широкий спектр завдань, пов’язаних із збором, обробкою та класифікацією екологічних показників. Розроблення *ІС класифікації екологічних показників з урахуванням дисбалансу*

класів є важливим науковим і практичним завданням, спрямованим на підвищення точності оцінки стану довкілля.

У межах цієї роботи планується створити модель, здатну ефективно аналізувати екологічні дані, враховувати їх нерівномірність та адаптуватися до різних типів показників. Результати такої системи можуть бути використані науковими центрами та екологічними організаціями для прийняття рішень, спрямованих на поліпшення стану навколишнього середовища та забезпечення екологічної безпеки.

1.2 Інтелектуальні системи класифікації: сутність і принципи побудови

Інтелектуальні системи є одним із ключових напрямів сучасних інформаційних технологій, що дозволяють автоматизувати процеси аналізу, прогнозування та прийняття рішень. Вони базуються на використанні штучного інтелекту, машинного навчання, обробки даних і статистичного аналізу. Основна мета таких систем – імітувати людське мислення під час вирішення складних задач, де кількість інформації надто велика для традиційних підходів.

ІС може сприймати, аналізувати та інтерпретувати дані, робити висновки, навчатися на основі попереднього досвіду й адаптуватися до нових умов. У контексті екологічних досліджень такі системи допомагають аналізувати величезні обсяги екологічних показників і автоматично визначати закономірності у стані довкілля. Вони здатні класифікувати показники за рівнем забруднення, прогнозувати екологічні ризики або виявляти аномалії у даних.

Поняття класифікатора тісно пов’язане з ІС. Класифікатор – це алгоритм або модель, яка визначає належність об’єкта до певного класу на основі його ознак. Наприклад, у системі екологічного моніторингу класифікатор може розподіляти проби води на категорії «чиста», «помірно забруднена» та «забруднена». Ефективність класифікатора залежить від якості навчальних даних, обраних ознак та обробки дисбалансу класів.

Основою побудови класифікаційних систем є МН. Це галузь штучного інтелекту, яка дозволяє комп'ютерам навчатися з даних без явного програмування правил. Алгоритми МН аналізують вхідні дані, виявляють закономірності та формують модель, яка здатна робити передбачення для нових прикладів. Існують три основні типи МН: з учителем (supervised learning), без учителя (unsupervised learning), напівкероване навчання (semi-supervised learning) та з підкріпленням (reinforcement learning) [6].

На рис. 1.1 зображено типи машинного навчання.

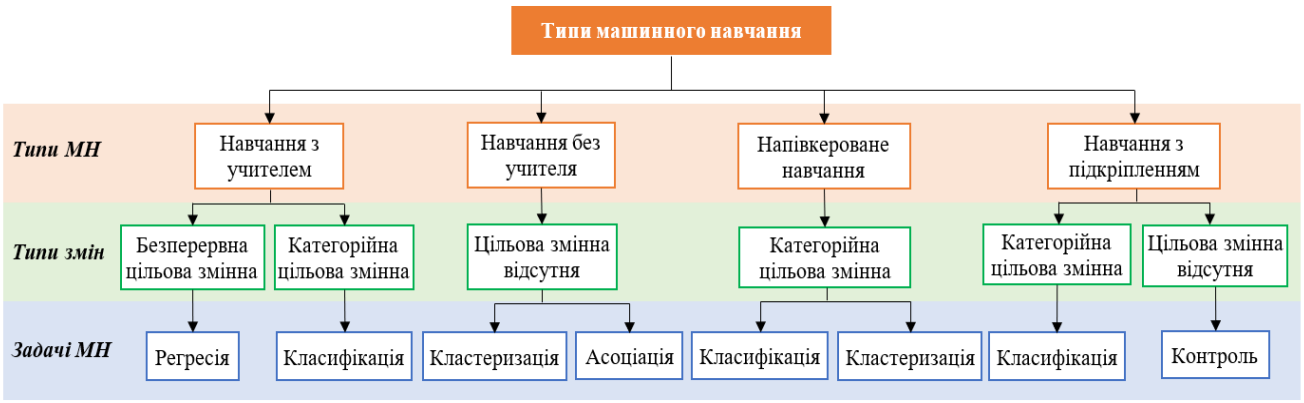

Рисунок 1.1 – Схема типів машинного навчання

Для задачі класифікації використовується навчання з учителем, коли система має набір прикладів із відомими класами. На основі цих даних алгоритм вчиться розпізнавати закономірності, що дозволяють передбачити клас для нових об'єктів. Наприклад, у задачі визначення якості води система навчається на даних про хімічні показники, після чого може прогнозувати клас чистоти для нових проб.

Побудова ІС класифікації складається з кількох основних етапів, таких як:

– збір даних, під час якого формується навчальний набір із різних джерел: сенсорів, лабораторних досліджень, відкритих екологічних баз даних чи супутникових спостережень;

– попередня обробка даних, що включає очищення від шумів, нормалізацію, обробку пропусків і перетворення ознак у придатну для аналізу форму;

– розділення даних на навчальну та тестову вибірки, що дозволяє перевіряти узагальнювальну здатність моделі;

– навчання моделі, коли класифікатор оптимізує свої параметри, щоб мінімізувати похибку класифікації;

– тестування та оцінка результатів, під час якого перевіряється точність, повнота, F1-міра або ROC-AUC.

Важливим етапом побудови ІС є врахування якості даних. Екологічні набори часто містять пропуски, шум або нерівномірний розподіл класів. Тому система має включати етапи балансування даних, щоб уникнути переваги більш поширеного класу.

Крім того, ІС класифікації можуть бути адаптивними – тобто здатними оновлювати свої знання після появи нових даних. Це особливо актуально для екологічних систем, де параметри довкілля змінюються з часом під впливом клімату або діяльності людини.

Використання таких систем дозволяє значно підвищити ефективність екологічного моніторингу. Вони забезпечують швидкий аналіз даних у реальному часі, зменшують ризик людських помилок та дозволяють прогнозувати негативні екологічні тенденції.

Важливою характеристикою ІС є інтерпретованість результатів. Для прийняття управлінських рішень у сфері екології недостатньо просто знати прогноз, потрібно також розуміти причини такого результату – які параметри найбільше впливають на якість повітря, води чи ґрунту.

У межах КР «Інтелектуальна система класифікації екологічних показників з урахуванням дисбалансу класів» передбачається створення системи, яка поєднує сучасні методи машинного навчання та обробки дисбалансних даних. Така система

дозволить точніше класифікувати екологічні показники, підвищити достовірність результатів і забезпечити автоматизований аналіз стану довкілля.

ІС класифікації є фундаментом для побудови ефективних інструментів екологічного моніторингу. Вони забезпечують глибоке розуміння даних, адаптацію до нових умов і підтримку прийняття рішень у сфері сталого розвитку.

1.3 Проблема дисбалансу класів у задачах класифікації екологічних показників

Однією з основних проблем у задачах класифікації є дисбаланс класів, який проявляється тоді, коли розподіл прикладів між класами є значно нерівномірним. У контексті екологічних даних мажоритарний клас (majority class) зазвичай представлений нормальними станами довкілля, такими як чиста вода, допустимий рівень забруднення повітря або відсутність лісових пожеж. Алгоритми МН легко навчаються на таких даних, оскільки їх кількість значно більша. Міноритарний клас (minority class), навпаки, включає рідкісні події, що відображають критичні або аномальні стани довкілля, наприклад забруднену воду, перевищення допустимих норм викидів або поява лісових пожеж. Ці дані є найбільш цінними з точки зору прийняття рішень, оскільки саме вони сигналізують про потенційні загрози.

Мажоритарний та міноритарний класи було схематично показано як елементи імбалансу класів на рис. 1.2.

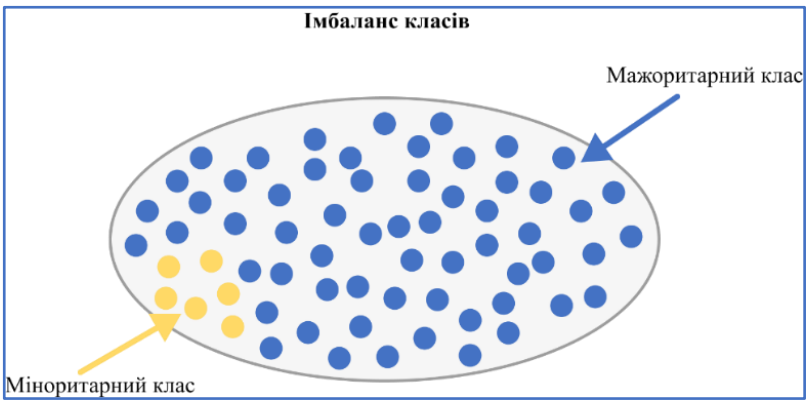

Рисунок 1.2 – Елементи імбалансу класів

Імбаланс класів виникає, коли кількість прикладів одного класу суттєво переважає інший. У реальних екологічних наборах даних спостерігається саме такий перекіс, оскільки аномальні події трапляються значно рідше за нормальні.

Типи дисбалансу класів можна умовно поділити на три категорії: легкий, середній та сильний (див. рис. 1.3-1.4).

У випадку легкого дисбалансу можна застосовувати стандартні методи класифікації без значних змін.

При середньому та сильному дисбалансі рекомендується використовувати спеціалізовані методи балансування, такі як undersampling (зменшення кількості прикладів більшого класу), oversampling (збільшення кількості прикладів меншого класу), SMOTE (Synthetic Minority Oversampling Technique, генерація синтетичних прикладів меншості), та ROSE (Random OverSampling Examples, генерація нових прикладів з урахуванням розподілу класів) [7].

Рисунок 1.3 – Типи дисбалансу класів

Таблиця 1.1 – Типи дисбалансу класів

Ступінь дисбалансу	**Співвідношення класів (Majority:Minority)**	**Відсоток меншості (%)**	**Приклад**
Легкий	2:1 – 5:1	20% – 40%	1000 «чисте» vs 250 «забруднене»
Середній	10:1 – 50:1	2% – 10%	1000 «чисте» vs 50 «забруднене»
Сильний	>100:1	<1%	10000 «чисте» vs 50 «забруднене»

За складністю дисбаланс буває: збалансований, міжкласовий, внутрішньокласовий, змішаний.

Збалансований класовий розподіл – це ситуація, коли кожен клас у наборі даних представлений приблизно однаковою кількістю прикладів. У такому випадку навчальні алгоритми отримують рівномірну інформацію про всі класи, що дозволяє моделі навчатися без упередження до будь-якої з категорій.

Збалансований класовий розподіл відображено на рис. 1.4.

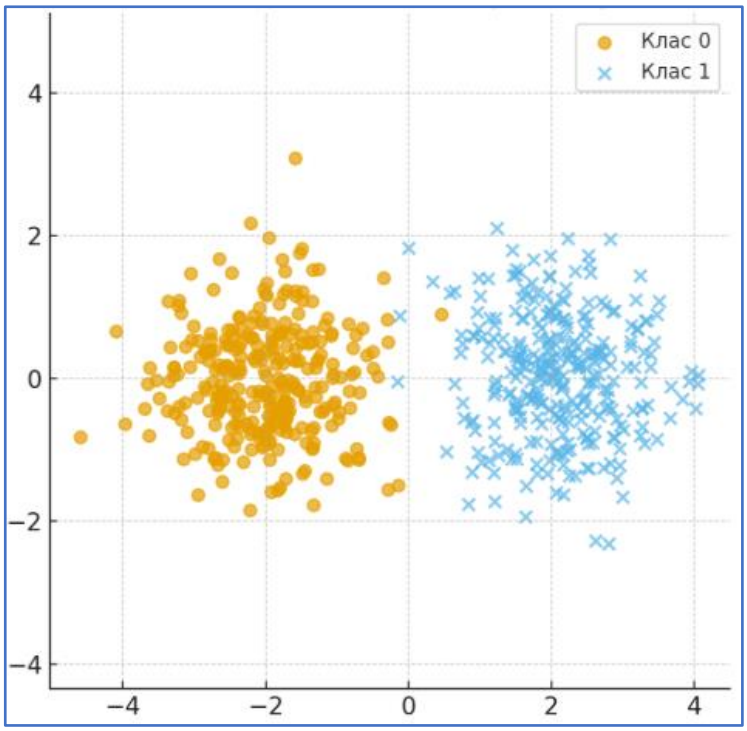

Рисунок 1.4 – Збалансований класовий розподіл

Дисбаланс класів може проявлятися як міжкласовий та внутрішньокласовий. Міжкласовий дисбаланс означає, що класи мають різну кількість прикладів. Проблема міжкласового дисбалансу полягає у тому, що базові метрики оцінки моделі вводять в оману: модель, яка завжди передбачає більший клас, демонструє високу точність, проте рідкісні приклади меншості ігноруються.

Міжкласовий дисбаланс відображено на рис. 1.5.

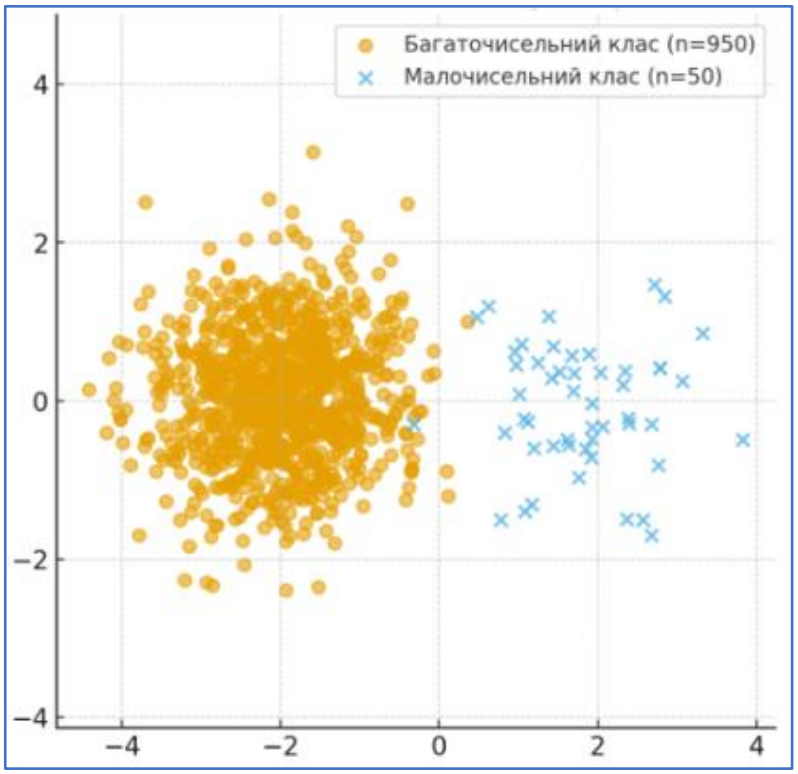

Рисунок 1.5 – Міжкласовий дисбаланс

Внутрішньокласовий дисбаланс виникає, коли в одному класі присутні підгрупи з різним рівнем представленості. Наприклад, клас позитивних прикладів може містити одну велику підгрупу та одну або кілька рідкісних підгруп з окремими характеристиками.

Внутрішньокласовий дисбаланс відображено на рис. 1.6.

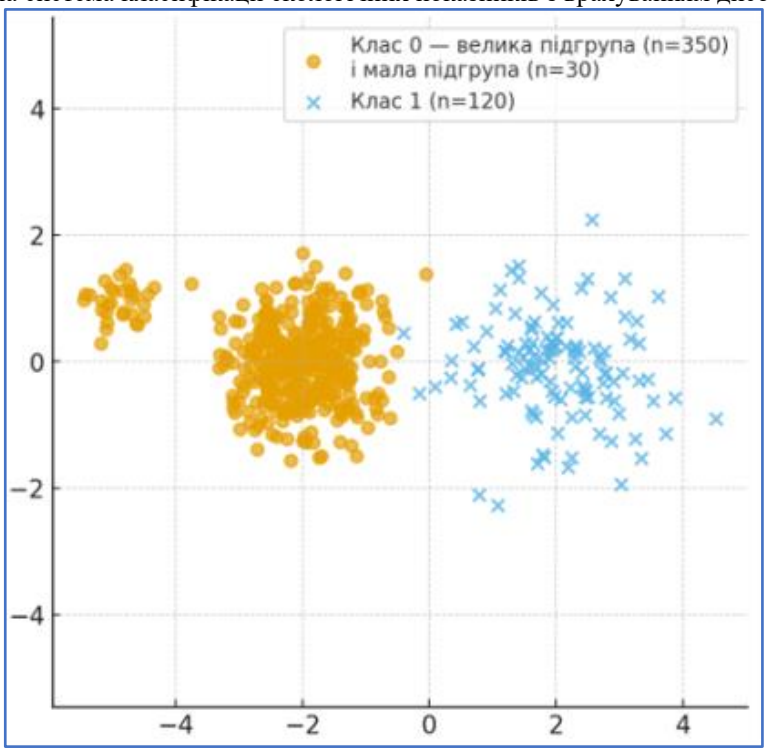

Рисунок 1.6 – Внутрішньокласовий дисбаланс

У такому випадку модель, тренуючись на великій підгрупі, не навчається розпізнавати рідкісні підгрупи, які фактично стають «невидимими». Застосування стандартних методів oversampling, таких як SMOTE, може створювати синтетичні приклади між різними підгрупами, що призводить до змивання структури даних або пропуску рідкісних підгруп.

Складним випадком є поєднання дисбалансу та перекриття класів, коли класи не тільки нерівні, але й сильно «змішані». Навіть при вирівнюванні кількості прикладів за допомогою oversampling не гарантується покращення роздільної здатності моделі: синтетичні точки часто потрапляють у зони перекриття, що підвищує рівень шуму та збільшує кількість помилкових спрацьовувань (False Positives) і пропусків (False Negatives), особливо для міноритарного класу.

Змішаний дисбаланс відображено на рис. 1.7.

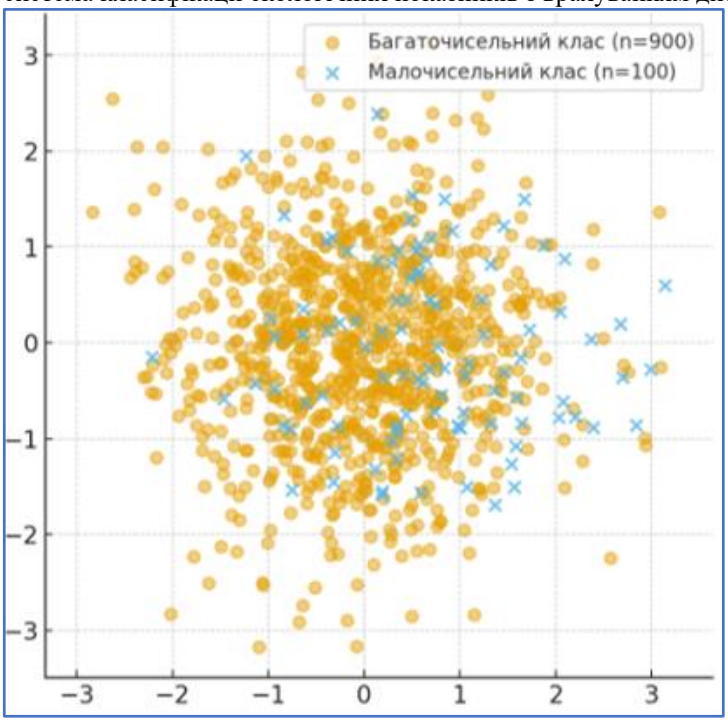

Рисунок 1.7 – Змішаний дисбаланс

Групи методів врахування дисбалансу класів – це класифікація підходів, які використовуються для роботи з даними, де класи представлені нерівномірно, щоб покращити ефективність моделей МН (див. рис. 1.8) [8]. Це різні способи вирішення проблеми, коли рідкісні (міноритарні) класи можуть бути ігноровані моделлю через їхню малу кількість у наборі даних.

Рисунок 1.8 – Групи методів врахування дисбалансу класів

Для боротьби з дисбалансом класів на етапі препроцесінгу даних можна застосовувати методи повторної вибірки. До oversampling підходів належать Random Oversampling (просте дублювання прикладів), SMOTE, Borderline-SMOTE, ADASYN та ROSE. Undersampling включає Random Undersampling, Tomek Links, Edited Nearest Neighbors (ENN) та NearMiss. Існують також гібридні методи, що поєднують обидва підходи, наприклад SMOTE + Tomek Links або SMOTE + ENN.

Методи навчання, чутливого до втрат, реалізуються безпосередньо на етапі навчання моделі та враховують дисбаланс класів у функції втрат. До них належать встановлення ваг класів у Logistic Regression (логістична регресія), SVM (Support Vector Machine, метод опорних векторів) або Random Forest (випадковий ліс), використання зваженої крос-ентропії у нейронних мережах, застосування cost matrix (матриці затрат) та зміщення порогу класифікації. Однак ці підходи поки не використовуються на етапі препроцесінгу.

Варто зазначити, що звичні метрики оцінки класифікації, такі як Accuracy (точність), у разі дисбалансу класів не відображають реальної якості моделі. Для оцінювання застосовуються метрики, стійкі до дисбалансу: F1-score (гармонійне середнє точності та повноти), Balanced Accuracy (збалансована точність), AUC-ROC (Area Under the Receiver Operating Characteristic curve, площа під кривою операційної характеристики приймача), AUC-PR та MCC (Matthews Correlation Coefficient, коефіцієнт кореляції Меттьюса). Використання стратифікованої крос-валідації матриці неточностей дозволяє детальніше аналізувати результати.

В сучасних дослідженнях для задач з дисбалансом класів успішно застосовуються ансамблеві методи, які демонструють кращу ефективність порівняно з окремими сильними алгоритмами, такими як SVM, KNN (K-Nearest Neighbors, k-найближчих сусідів) або нейронні мережі. Серед таких методів – BalancedBagging, RUSBoost, SMOTEBoost, EasyEnsemble, BalanceCascade. Вони дозволяють врахувати нерівність класів під час навчання і підвищують якість класифікації рідкісних, але важливих прикладів.

Додатково можуть використовуватися методи генерації синтетичних даних за допомогою GANs (Generative Adversarial Networks, генеративно-змагальні нейронні мережі) або VAEs (Variational Autoencoders, варіаційні автоенкодери), перенесення знань з іншої задачі (transfer learning) для перенесення знань з іншої задачі, а також аугментація даних (data augmentation) для тексту, зображень чи аудіо – наприклад, повороти, віддзеркалення, додавання шуму або синонімічна заміна. Залучення зовнішніх джерел даних для менш представленого класу також підвищує якість навчання.

Проблема дисбалансу класів є важливою для інтелектуальної системи класифікації екологічних показників, оскільки забезпечує коректне розпізнавання рідкісних, але критично важливих станів довкілля. Методи балансування на етапі препроцесінгу дозволяють ефективно підготувати дані для навчання моделей та підвищити їхню практичну цінність для прийняття рішень у сфері екологічного моніторингу.

1.4 Аналіз сучасних підходів і програмних рішень

Сучасні дослідження в галузі класифікації екологічних показників із врахуванням дисбалансу класів охоплюють широкий спектр методів – від класичних технік балансування вибірки до інтегрованих гібридних моделей МН. Проведений аналіз наукових публікацій дозволяє визначити основні тенденції та обмеження сучасних підходів, а також обґрунтувати вибір методів для побудови ІС у даній КР.

Однією з базових праць у цій сфері є дослідження H. Chawla та співавт., у якому було запропоновано метод SMOTE [9]. Автори показали, що синтетичне створення нових зразків міноритарного класу за допомогою інтерполяції між існуючими прикладами дозволяє підвищити точність класифікації та зменшити переобучення, властиве простому дублюванню даних. Метод показав покращення метрик ROC у порівнянні з базовими підходами. Однак було виявлено, що SMOTE може створювати нереалістичні приклади у зонах перетину класів, що призводить

до появи шуму. Незважаючи на це, підхід став фундаментальним у подальших дослідженнях балансування даних у різних галузях, зокрема в екологічному моніторингу. У межах екологічних задач SMOTE забезпечує базу для ефективного вирівнювання дисбалансу між класами показників забруднення.

Подальший розвиток цієї ідеї представлено у роботі G. Douzas та F. Bacao, де запропоновано Geometric SMOTE (G-SMOTE) – узагальнення класичного SMOTE 10]. У цьому підході використовується геометричний механізм створення синтетичних зразків, що враховує локальну структуру простору ознак. Автори довели ефективність методу на задачах класифікації покриття земної поверхні, де традиційний SMOTE не забезпечував належної точності. Перевагою є зменшення кількості артефактів і більш реалістичне відтворення розподілу даних. Недоліком є підвищена складність налаштувань і залежність від метрики відстані. Такий підхід є перспективним для використання у класифікації просторових екологічних показників, зокрема у задачах моніторингу земель і водних ресурсів.

У систематичному огляді G. He та E. Garcia «Learning from Imbalanced Data» детально проаналізовано існуючі методи навчання на дисбалансних наборах [11]. Автори класифікують методи за трьома рівнями – рівнем даних, рівнем алгоритму та рівнем оцінювання результатів. У роботі наголошується на важливості застосування адекватних метрик, таких як F1-score, AUC та Precision-Recall, які точніше відображають якість класифікації у випадках з дисбалансом класів. Дослідники також вказують на доцільність комбінування методів балансування з алгоритмічними підходами, наприклад, cost-sensitive навчанням. Зроблені висновки залишаються актуальними для екологічних систем, де рідкісні події, як-от аварійні викиди чи перевищення норм, мають ключове значення. Цей огляд формує методологічну основу для розробки стратегії боротьби з дисбалансом у даній роботі.

Робота D. Roberts та ін. «Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure» підкреслює важливість правильної організації крос-валідації для екологічних даних [12]. Автори довели, що випадкове

розбиття вибірки часто призводить до завищення результатів, якщо дані мають просторову або часову кореляцію. Вони запропонували методики просторово-часової валідації, які дозволяють уникнути витоку інформації між тренувальними та тестовими підвибірками. Такі стратегії забезпечують більш реалістичну оцінку продуктивності моделей. Робота є важливою для побудови систем класифікації екологічних показників, адже екологічні дані зазвичай мають географічну залежність.

Схожі висновки представлено у дослідженні P. Ploton та співавт. «Spatial validation reveals poor predictive performance of large-scale mapping models» [13]. Автори довели, що більшість екологічних моделей демонструють завищену продуктивність через ігнорування просторової незалежності даних у процесі валідації. Їх результати свідчать, що при врахуванні просторової структури точність моделей істотно зменшується, що підкреслює важливість правильної оцінки результатів. Це актуально для задач картографування забруднення чи якості середовища. Дослідники закликають до використання відкритих кодів і відтворюваних методологій. Для створення достовірної системи класифікації екологічних показників у даній роботі враховуються висновки Ploton та колег.

H. Ke та ін. у праці «A hybrid XGBoost–SMOTE model for optimization of forecasted PM2.5 and O3 concentrations» [14] розробили гібридну модель, що поєднує градієнтний бустинг (XGBoost) та метод SMOTE. Запропонований підхід дозволив підвищити точність прогнозування концентрацій PM2.5 та O3, особливо в умовах дефіциту даних про екстремальні події. Автори використали метод пояснюваності SHAP для аналізу впливу ознак. Вони дійшли висновку, що балансування вибірки покращує здатність моделі розпізнавати аномальні ситуації забруднення. Недоліком є потенційне перенавчання на синтетичних даних, тому автори рекомендують ретельну перевірку на просторових вибірках. Ця робота демонструє ефективність поєднання методів балансування та сучасних алгоритмів МН у сфері екологічного прогнозування.

Схожі підходи застосовано у роботі W. Wong та співавт. «A Stacked Ensemble Deep Learning Approach for Imbalanced Multi-class Water Quality Index Prediction» [15]. Автори використали ансамблеву архітектуру на основі глибоких нейронних мереж для класифікації індексу якості води. Було реалізовано стекінг моделей та техніки балансування, що підвищили точність для рідкісних класів низької якості. Дослідники також застосували методи пояснюваності для підвищення інтерпретованості моделі. Отримані результати засвідчили перевагу ансамблевих моделей над окремими підходами. Ця праця підкреслює важливість балансування даних у багатокласових задачах екологічного моніторингу.

L. Liu та колеги у дослідженні «Solving the class imbalance problem using ensemble algorithm» [16] представили комбінований підхід, який поєднує undersampling, oversampling, cost-sensitive навчання та ансамблеві методи. Автори показали, що інтеграція різних технік дозволяє підвищити стабільність та узагальнювальну здатність моделей. Їхній метод продемонстрував покращення метрик recall і F1-score на медичних наборах даних із високим дисбалансом. Цей підхід має практичну цінність для екологічних задач, де спостерігаються схожі проблеми рідкісних класів. Дослідження також наголошує на важливості ретельної оцінки моделей за допомогою збалансованих метрик. Отже, комбіновані стратегії є ефективними при побудові стійких систем класифікації.

У сучасному огляді W. Chen та ін. «A survey on imbalanced learning: latest research» [17] узагальнено новітні методи боротьби з дисбалансом класів. Автори приділили увагу глибоким генеративним моделям, модифікованим функціям втрат та мета-навчанню. Особливу увагу приділено ризикам створення синтетичних прикладів, що спотворюють реальний розподіл даних. Дослідження рекомендує поєднання алгоритмічних і попередньо обробних підходів.

A. Masood та співавт. у роботі «Improving PM2.5 prediction using a hybrid model» [18] розробили гібридну архітектуру, яка поєднує нейронні мережі з алгоритмами оптимізації для покращення прогнозу концентрацій забруднюючих речовин. Автори довели, що попередня обробка, вибір релевантних ознак і

балансування даних мають суттєвий вплив на кінцеву точність. Модель виявила підвищену чутливість до рідкісних, але критично важливих подій. Дослідники також підкреслюють значення просторово-часової валідації для коректної оцінки моделі. Цей підхід підтверджує доцільність використання інтегрованих рішень у задачах прогнозування екологічних показників.

Сучасні дослідження свідчать про ефективність комбінованих методів, що поєднують балансування вибірки, адаптацію функцій втрат і ансамблеві підходи. Найбільш перспективними для екологічних даних є гібридні системи, що враховують просторово-часові залежності та особливості дисбалансу класів.

1.5 Структура інтелектуальної системи

Розроблена структура ІС спрямована на аналіз даних якості води, попередню обробку, балансування вибірки та побудову моделей МН для прогнозування якості екологічних показників. Основною метою є підвищення точності класифікації шляхом усунення дисбалансу класів та використання ансамблевих методів навчання, здатних ефективно працювати з нерівномірно розподіленими даними.

Структура інтелектуальної системи складається з кількох взаємопов’язаних підсистем і модулів, які забезпечують повний цикл обробки даних, моделювання, оцінювання та подання результатів користувачу. Система побудована модульно, що підвищує гнучкість, масштабованість та можливість подальшого удосконалення її компонентів. Усі підсистеми працюють послідовно, забезпечуючи безперервний процес аналізу екологічних показників й побудови моделей прогнозування. (див. рис. 1.9).

Інтелектуальна система для аналізу екологічних даних складається з декількох підсистем, які забезпечують повний цикл роботи з інформацією – від її завантаження до формування прогнозів та візуалізації результатів. Основою системи є підсистема завантаження і зберігання даних, яка включає модуль імпорту даних, базу даних та інтерфейс користувача. Модуль імпорту дозволяє завантажувати початкові набори даних із відкритих джерел, зокрема з ресурсу

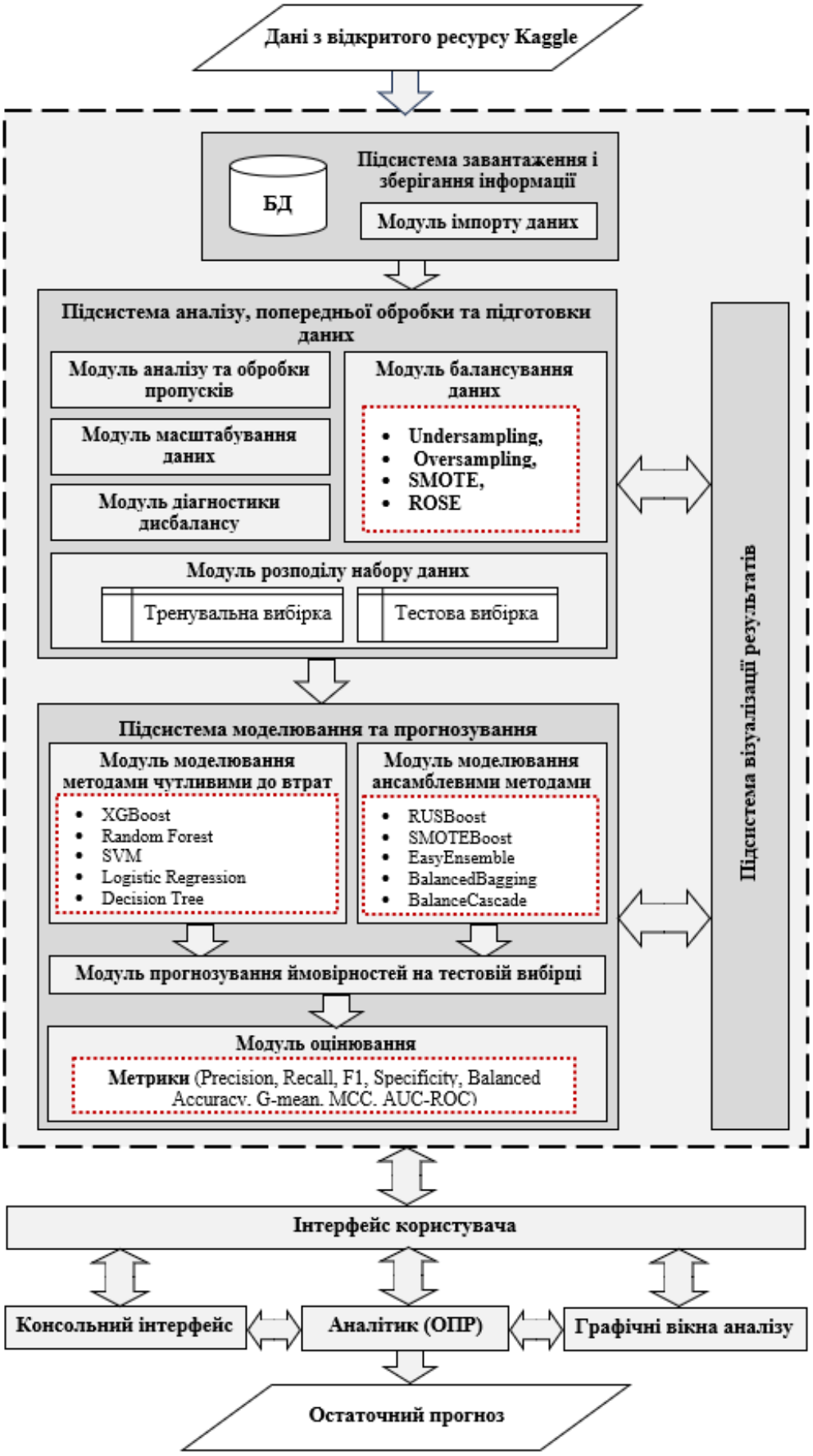

Рисунок 1.9 – Структура інтелектуальної системи

Kaggle, а база даних забезпечує надійне зберігання як початкових, так і оброблених даних. Інтерфейс користувача надає можливість взаємодії з даними через консольний режим та графічні вікна аналізу.

Наступною важливою частиною системи є підсистема аналізу, попередньої обробки та підготовки даних. Вона включає модуль аналізу та обробки пропусків, який здійснює діагностику відсутніх значень та їх заповнення, а також модуль масштабування даних, що проводить нормалізацію числових ознак для підвищення стабільності моделей. Підсистема також містить модулі діагностики дисбалансу та балансування вибірки, які визначають співвідношення класів у цільовій змінній і застосовують різні методи для усунення дисбалансу, такі як Undersampling, Oversampling, SMOTE та ROSE. Важливим етапом є розподіл набору даних на тренувальну та тестову вибірки, що дозволяє підготувати якісний і збалансований датасет для моделювання.

Підсистема моделювання та прогнозування реалізує створення та оцінювання моделей. Вона включає модуль моделювання методами, чутливими до втрат, який охоплює класичні алгоритми, адаптовані до роботи з нерівномірними даними, зокрема XGBoost, Random Forest, SVM, Logistic Regression та Decision Tree. Також система має модуль моделювання ансамблевими методами, що підвищує ефективність прогнозів за рахунок поєднання різних ансамблевих технік, таких як RUSBoost, SMOTEBoost, EasyEnsemble, Balanced Bagging та Balance Cascade. Модуль прогнозування ймовірностей формує прогнозні значення для тестової вибірки на основі побудованих моделей, що дозволяє аналітикам оцінювати ймовірність належності об’єктів до певного класу.

Оцінка результатів моделювання здійснюється через спеціальний модуль, який забезпечує формальний аналіз ефективності моделей та їх порівняння. Використовуються різні метрики, такі як Precision, Recall, F1-score, Specificity, Balanced Accuracy, G-mean, MCC та AUC-ROC. Аналіз проводиться для всіх моделей і ансамблевих методів, що дозволяє визначити найбільш ефективні підходи для прогнозування якості води.

Підсистема візуалізації результатів забезпечує наочне представлення даних та результатів моделювання. Вона включає побудову ROC-кривих, формування матриць плутанини, відображення важливості ознак, графічний аналіз балансування вибірки та виведення остаточних прогнозів. Панель візуалізації доступна через графічні інтерфейси та використовується аналітиками для прийняття рішень на основі результатів моделювання.

Інтерфейс користувача системи забезпечує взаємодію аналітика з усіма підсистемами. Він включає консольний режим, графічні вікна для аналізу та відображення результатів моделювання й прогнозування. Така структура дозволяє ефективно працювати з великими обсягами даних, гнучко додавати нові методи обробки та моделювання, а також швидко оцінювати ефективність моделей.

Отже, представлена система охоплює всі етапи роботи з екологічними даними – від їх завантаження та підготовки до моделювання, оцінювання та візуалізації результатів. Вона забезпечує точне і надійне прогнозування придатності води до споживання, дозволяє інтегрувати нові алгоритми та методи балансування, а також надає зручний інтерфейс для аналітиків, що робить систему ефективним інструментом для прийняття рішень.

Висновки до розділу 1

Перший розділ присвячений теоретичним основам інтелектуальної класифікації екологічних даних і містить комплексний аналіз предметної області. У процесі дослідження було встановлено, що екологічні показники якості води є багатовимірними, нерівномірними та схильними до варіацій, що ускладнює процес їх формальної обробки традиційними статистичними підходами. Показано, що сучасний рівень інформаційних технологій створює умови для впровадження інтелектуальних систем, здатних автоматизувати аналіз великих обсягів даних та оперативно виявляти екологічні ризики. На основі огляду літератури узагальнено визначення та ключові принципи побудови інтелектуальних систем класифікації,

серед яких – модульність, адаптивність, здатність до самонавчання та інтеграція методів машинного навчання.

Одним із центральних аспектів розділу є дослідження проблеми дисбалансу класів, характерної для екологічних даних. Підкреслено, що у сфері моніторингу водних ресурсів рідкісні випадки забруднення або аномальні стани трапляються значно рідше, ніж нормальні, що призводить до викривлення навчального процесу моделей. З огляду на це виявлено, що класичні алгоритми класифікації, орієнтовані на мінімізацію загальної похибки, демонструють недостатню здатність до розпізнавання критичних екологічних подій. Важливим результатом аналізу є визначення необхідності використання спеціалізованих методів балансування та модифікованих моделей навчання.

Також у розділі розглянуто сучасні програмні засоби та алгоритмічні підходи, що застосовуються для аналізу екологічних даних. Узагальнено їхні переваги, недоліки та сфери застосування. Окреслено архітектуру типової інтелектуальної системи, яка включає модулі збору даних, препроцесінгу, моделювання та інтерпретації результатів. Таким чином, розділ формує теоретичне підґрунтя дослідження та підтверджує необхідність створення інтелектуальної системи класифікації з урахуванням дисбалансу класів для підвищення точності екологічного моніторингу.

2 ПРЕПРОЦЕСІНГ ЕКОЛОГІЧНИХ ДАНИХ

2.1 Теоретичні відомості про препроцесінг даних

Препроцесінг даних є одним із ключових етапів у процесі побудови ІС аналізу, оскільки якість і достовірність результатів моделювання безпосередньо залежать від якості вхідних даних. У реальних умовах екологічні дані, як правило, мають складну структуру, можуть містити пропуски, викиди, шум, різні одиниці вимірювання або бути зібраними з різних джерел, що ускладнює їх подальше використання у МН. Тому препроцесінг виступає проміжним, але надзвичайно важливим етапом між збором даних та їх моделюванням.

Основна мета препроцесінгу полягає у *підвищенні якості даних*, забезпеченні їх *узгодженості, повноти* та *придатності для аналізу*. Цей процес включає виявлення та усунення похибок, приведення показників до єдиної шкали, обробку пропусків і аномалій, а також балансування класів у випадку дисбалансних наборів.

Процес препроцесінгу складається з низки взаємопов’язаних етапів (див. рис. 2.1), що реалізуються у вигляді послідовного алгоритму. Алгоритм препроцесінгу даних – це послідовність дій, що охоплює повний цикл підготовки від первинного збору до формування структурованого, якісного й аналітично придатного набору даних. Такий підхід гарантує точність, стабільність і достовірність результатів, а також підвищує ефективність побудованих моделей у завданнях екологічного моніторингу та прогнозування стану довкілля.

Першим етапом є початок процесу препроцесінгу, який визначає **старт** підготовки даних. Після цього формується **набір вхідних даних** – здійснюється збір усіх доступних екологічних показників, таких як температура, рівень вологості, вміст шкідливих речовин, концентрація кисню тощо. Ці дані можуть бути отримані з різних сенсорних систем, лабораторних спостережень або відкритих екологічних баз.

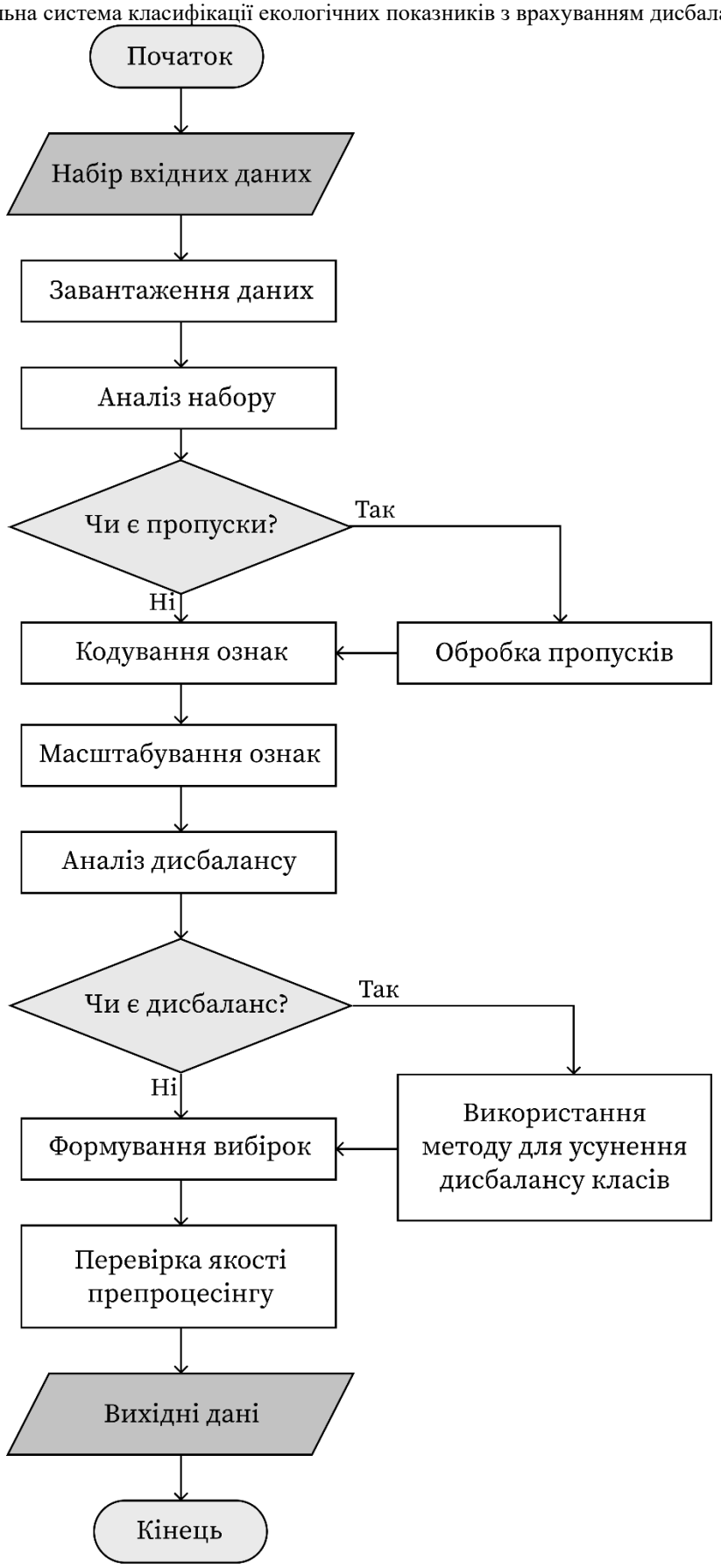

Рисунок 2.1 – Алгоритм попередньої обробки даних

Далі виконується **завантаження даних**, тобто імпорт отриманого набору до програмного середовища (наприклад, Python, R або MATLAB) для подальшої обробки. На цьому етапі важливо забезпечити правильне зчитування файлів, форматів і типів змінних.

Наступним кроком є **аналіз набору даних**, що передбачає початкове дослідження структури, статистичних характеристик та взаємозв'язків між змінними. Проводиться оцінка кількості спостережень, типів ознак (числових, категоріальних), діапазонів значень, а також виявлення потенційних проблем – пропусків, аномалій або дубльованих записів.

Після цього здійснюється **перевірка на наявність пропусків**. Якщо пропущені значення виявлено, виконується обробка пропусків, яка може включати заповнення середнім, медіаною, модою, інтерполяцією або застосуванням методів імпутації, таких як KNN або регресійні підходи. У разі, якщо пропусків немає, алгоритм переходить безпосередньо до наступного етапу.

Подальшим кроком є **кодування ознак** – процес перетворення категоріальних даних у числові форми, які можна використовувати в алгоритмах МН.

Після кодування відбувається **масштабування ознак**, яке забезпечує порівнянність усіх показників. Оскільки екологічні дані можуть мати різні одиниці вимірювання (наприклад, °C, мг/л, %), необхідно застосовувати нормалізацію або стандартизацію, щоб уникнути домінування змінних із великими числовими діапазонами. Це покращує стабільність і точність роботи моделей.

Далі здійснюється **аналіз дисбалансу класів** – оцінка рівномірності розподілу цільових категорій. У екологічних задачах класи можуть бути представлені нерівномірно: наприклад, кількість спостережень нормального стану навколишнього середовища може значно перевищувати кількість записів про забруднення. Якщо виявлено дисбаланс, застосовуються методи балансування класів, такі як oversampling (збільшення меншості), undersampling (зменшення більшості) або алгоритми SMOTE, ADASYN, ROSE. Ці методи допомагають

уникнути зміщення моделі в бік більшості та підвищити точність розпізнавання рідкісних екологічних явищ.

Якщо ж дані збалансовані, процес переходить безпосередньо до етапу **формування навчальної та тестової вибірок**. Зазвичай 70–80% даних виділяють для навчання моделі, а решту – для тестування, щоб оцінити її узагальнювальну здатність і запобігти перенавчанню. У деяких випадках також формується валідаційна вибірка для тонкого налаштування параметрів моделі.

Після формування вибірок виконується перевірка якості препроцесінгу, яка включає оцінку правильності заповнення пропусків, відсутність дублювання, коректність масштабування та кодування ознак. Це дозволяє переконатися, що дані повністю підготовлені до подальшого моделювання.

На виході отримуються готові, очищені та збалансовані дані, які можна безпосередньо використовувати для аналізу, побудови прогнозних моделей або виявлення закономірностей у поведінці екосистем.

Завершальним етапом є кінець процесу препроцесінгу, який означає готовність підготовленого набору до етапу моделювання або МН.

Таким чином, препроцесінг екологічних даних є необхідним етапом, який гарантує коректність подальшого моделювання. Він створює основу для формування якісної аналітичної системи, здатної ефективно виявляти закономірності в природних процесах, прогнозувати зміни стану навколишнього середовища та сприяти прийняттю обґрунтованих управлінських рішень у сфері екологічної безпеки.

2.2 Характеристика вхідних екологічних показників

У рамках дослідження для моделювання та аналізу було обрано два набори даних, які відображають різні аспекти екологічного моніторингу: якість води та виявлення розливів нафти. Обидва набори використовуються в наукових і навчальних цілях для тестування алгоритмів МН, що вирішують задачі екологічної класифікації.

Перший набір даних – «Water Potability» (Придатність води до пиття) – містить 3276 спостережень, які описують фізико-хімічні властивості зразків води [19]. Кожен запис характеризується набором з 9 кількісних показників, серед яких:

– pH – рівень кислотності або лужності води;

– Hardness – жорсткість, що визначається концентрацією кальцію та магнію;

– Solids – кількість розчинених твердих речовин у мг/л;

– Chloramines – концентрація хлораміну, що впливає на бактеріологічну безпеку;

– Sulfate – вміст сульфатів, важливий показник якості води;

– Conductivity – електропровідність, яка характеризує загальну мінералізацію;

– Organic Carbon – рівень органічного вуглецю, що впливає на органолептичні властивості води;

– Trihalomethanes – концентрація побічних продуктів хлорування;

– Turbidity – каламутність, яка вказує на наявність завислих частинок;

– цільова змінна Potability має бінарний характер (1 – вода придатна до пиття, 0 – непридатна).

У наборі присутні пропуски даних – переважно у змінних Sulfate, pH і Trihalomethanes, що вимагає застосування методів імпутації або очищення перед аналізом. Окрім того, частина ознак має різні одиниці вимірювання, тому нормалізація та стандартизація є обов’язковими кроками препроцесінгу.

Цей набір даних цінний тим, що поєднує простоту структури з практичною значущістю: він дозволяє створювати моделі для прогнозування придатності води на основі хімічних показників, що є важливим завданням у сфері охорони здоров’я та управління водними ресурсами.

На рис. 2.2 зображено за допомогою гістограми дисбаланс класів у першому наборі даних.

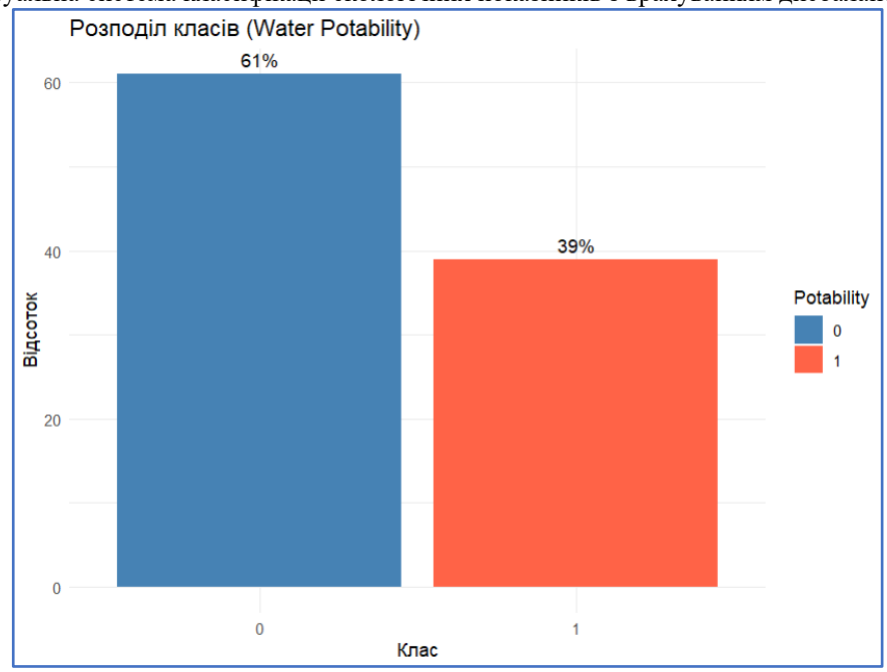

Рисунок 2.2 – Дисбаланс класів у першому наборі даних

Другий набір даних – «Oil Spill Detection» (Виявлення розливів нафти на супутникових знімках) – представлений 937 спостереженнями, кожне з яких відповідає окремому фрагменту супутникового зображення поверхні океану [20]. Кожен запис описується 48 числовими ознаками, отриманими шляхом комп'ютерного аналізу супутникових даних (зокрема радарних зображень). Ці ознаки відображають текстурні, спектральні та статистичні характеристики ділянок води, що дозволяють моделі розрізняти ділянки з нафтопродуктами від чистої води.

Цільова змінна є бінарною – «1» означає наявність нафтової плями, «0» – відсутність. Головною особливістю набору є сильний дисбаланс класів: лише 41 спостереження (≈4%) належить до класу «розлив нафти», тоді як 896 прикладів (≈96%) – до класу «без розливу». Така нерівномірність розподілу даних значно ускладнює навчання моделей, оскільки більшість алгоритмів схильні «ігнорувати» рідкісний клас, що призводить до високої кількості хибнонегативних результатів.

Окрім дисбалансу, у наборі спостерігається велика різниця масштабів між ознаками: деякі мають дуже малі значення, інші – великі порядки величин. Це

зумовлює необхідність масштабування або стандартизації для забезпечення стабільності навчання моделей.

Набір Oil Spill Detection є типовим прикладом складного екологічного завдання, де потрібно виявляти рідкісні події (аналогічно задачам медичної діагностики чи розпізнавання дефектів).

На рис. 2.3 зображено за допомогою гістограми дисбаланс класів у другому наборі даних.

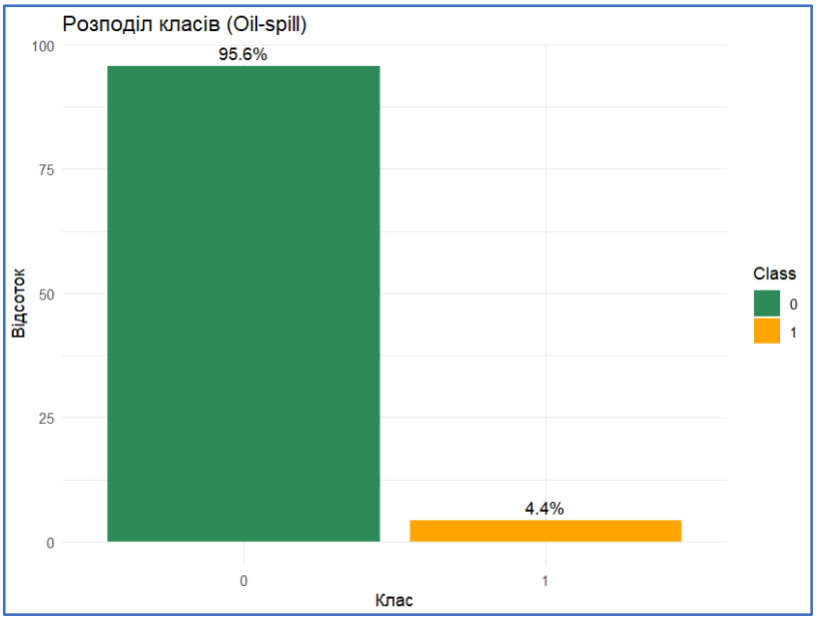

Рисунок 2.3 – Дисбаланс класів у другому наборі даних

Обидва набори – «Water Potability» і «Oil Spill Detection» – відображають два різних, але взаємопов’язаних напрями екологічного аналізу: контроль якості природних ресурсів (води) та виявлення антропогенних забруднень (нафти). Вони доповнюють одне одного, дозволяючи дослідити, як різні екологічні набори даних проходять етапи препроцесінгу, очищення, балансування та підготовки для побудови ефективних моделей класифікації.

Для роботи з наборами даних треба завантажити необхідні бібліотеки та самі набори даних у середовище RStudio.

На етапі завантаження бібліотек для роботи виконується підготовка робочого середовища: встановлюються та завантажуються всі необхідні пакети для подальшого аналізу даних і роботи з дисбалансом класів (див. рис. 2.4), зокрема:

– dplyr, ggplot2, gridExtra – для обробки та візуалізації даних;

– caret – для побудови та оцінювання моделей;

– ROSE і smotefamily – для генерації синтетичних прикладів та балансування вибірки;

– Amelia – для візуалізації та обробки пропущених значень;

– pROC – для роботи з ROC-кривими.

```
# Встановлення та завантаження потрібних пакетів --------------------------
packages <- c("dplyr", "ggplot2", "caret", "ROSE", "Amelia", "smotefamily", "gr

installed <- packages %in% rownames(installed.packages())
if (any(!installed)) {
  install.packages(packages[!installed])
}

library(pROC)
library(ggplot2)
library(gridExtra)
library(dplyr)
library(caret)
library(ROSE)
library(Amelia)
library(smotefamily)
```

Рисунок 2.4 – Завантаження необхідних для роботи бібліотек

Завантаження даних та аналіз структури передбачає завантаження наборів у робоче середовище RStudio та первинне ознайомлення зі структурою вибірки за допомогою функцій. Для наборів *water_potability* та *oil-spill* здійснюється імпорт даних у середовище R, перевірка кількості спостережень і змінних, а також визначення їх типів (числові, цілі, факторні) (див. рис. 2.5, 2.7). Додатково формується описова статистика (мінімальні, максимальні значення, медіани, середні, кількість пропусків) (див. рис. 2.6, 2.8), що дозволяє виявити можливі проблеми – наявність пропусків, аномалій чи дисбалансу класів у цільовій змінній.

Більшість змінних першого набору даних мають тип numeric, а Potability – integer (див. рис. 2.5). У наборі є пропущені значення: ph, Sulfate, Trihalomethanes.

Значення ознак мають широкий діапазон, наприклад, ph варіює від 0 до 14, Hardness – від 47 до 323, Solids – до 61 227.2 (див. рис. 2.6). Середні значення показують, що вода в середньому має нейтральний pH ≈ 7.08, помірну жорсткість (≈ 196), а Potability має середнє 0.39, тобто лише близько 39% зразків є питними. Дані потребують попередньої обробки (усунення пропусків та масштабування).

```
--- Структура даних ---
> str(water_data)
'data.frame':   3276 obs. of  10 variables:
 $ ph              : num  NA 3.72 8.1 8.32 9.09 ...
 $ Hardness        : num  205 129 224 214 181 ...
 $ Solids          : num  20791 18630 19910 22018 17979 ...
 $ Chloramines     : num  7.3 6.64 9.28 8.06 6.55 ...
 $ Sulfate         : num  369 NA NA 357 310 ...
 $ Conductivity    : num  564 593 419 363 398 ...
 $ Organic_carbon  : num  10.4 15.2 16.9 18.4 11.6 ...
 $ Trihalomethanes: num  87 56.3 66.4 100.3 32 ...
 $ Turbidity       : num  2.96 4.5 3.06 4.63 4.08 ...
 $ Potability      : int  0 0 0 0 0 0 0 0 0 0 ...
```

Рисунок 2.5 – Структура першого набору даних

```
--- Описова статистика ---
> summary(water_data)
       ph            Hardness          Solids         Chloramines
 Min.   : 0.000   Min.   : 47.43   Min.   :  320.9   Min.   : 0.352
 1st Qu.: 6.093   1st Qu.:176.85   1st Qu.:15666.7   1st Qu.: 6.127
 Median : 7.037   Median :196.97   Median :20927.8   Median : 7.130
 Mean   : 7.081   Mean   :196.37   Mean   :22014.1   Mean   : 7.122
 3rd Qu.: 8.062   3rd Qu.:216.67   3rd Qu.:27332.8   3rd Qu.: 8.115
 Max.   :14.000   Max.   :323.12   Max.   :61227.2   Max.   :13.127
 NA's   :491
    Sulfate       Conductivity   Organic_carbon  Trihalomethanes     Turbidity
 Min.   :129.0   Min.   :181.5   Min.   : 2.20   Min.   :  0.738   Min.   :1.450
 1st Qu.:307.7   1st Qu.:365.7   1st Qu.:12.07   1st Qu.: 55.845   1st Qu.:3.440
 Median :333.1   Median :421.9   Median :14.22   Median : 66.622   Median :3.955
 Mean   :333.8   Mean   :426.2   Mean   :14.28   Mean   : 66.396   Mean   :3.967
 3rd Qu.:360.0   3rd Qu.:481.8   3rd Qu.:16.56   3rd Qu.: 77.337   3rd Qu.:4.500
 Max.   :481.0   Max.   :753.3   Max.   :28.30   Max.   :124.000   Max.   :6.739
 NA's   :781                                     NA's   :162
   Potability
 Min.   :0.0000
 1st Qu.:0.0000
 Median :0.0000
 Mean   :0.3901
 3rd Qu.:1.0000
 Max.   :1.0000
```

Рисунок 2.6 – Описова статистика першого набору даних

Більшість змінних мають тип numeric, частина – integer (див. рис. 2.7). Пропущені значення відсутні. Діапазони значень дуже різноманітні – від невеликих дробових чисел до десятків мільйонів (див. рис. 2.8). Це вказує на наявність як нормованих, так і вимірювальних ознак різних масштабів. Остання змінна V50 ймовірно є бінарною цільовою (0 або 1). Деякі колонки мають сталі або

повторювані значення (V22, V25, V48), що може свідчити про технічні параметри або ознаки, які не несуть значущої варіації. Загалом набір даних характеризується високою розмірністю, значними коливаннями масштабів змінних і потребує нормалізації перед аналізом

```
--- Структура даних ---
> str(data_oil)
'data.frame':   937 obs. of  50 variables:
 $ V1   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ V2   : int  2558 22325 115 1201 312 54 116 57 188 64 ...
 $ V3   : num  1506.1 79.1 1449.8 1562.5 950.3 ...
 $ V4   : num  457 841 608 296 441 ...
 $ V5   : int  90 180 88 66 37 82 97 141 22 33 ...
 $ V6   : num  6395000 55812500 287500 3002500 780000 ...
 $ V7   : num  40.9 51.1 40.4 42.4 41.4 ...
 $ V8   : num  7.89 1.21 7.34 7.97 7.03 6.92 6.24 0.83 5.89 7.9 ...
 $ V9   : num  29780 61900 3340 18030 3350 ...
 $ V10  : num  0.19 0.02 0.18 0.19 0.17 0.15 0.15 0.02 0.15 0.19 ...
 $ V11  : num  214.7 901.7 86.1 166.5 232.8 ...
 $ V12  : num  0.21 0.02 0.21 0.21 0.15 0.18 0.21 0.02 0.19 0.19 ...
 $ V13  : num  0.26 0.03 0.32 0.26 0.19 0.25 0.3 0.03 0.24 0.25 ...
 $ V14  : num  0.49 0.11 0.5 0.48 0.35 0.53 0.49 0.07 0.38 0.36 ...
 $ V15  : num  0.1 0.01 0.17 0.1 0.09 0.15 0.14 0.02 0.09 0.12 ...
 $ V16  : num  0.4 0.11 0.34 0.38 0.26 0.38 0.35 0.05 0.29 0.24 ...
 $ V17  : num  99.6 6058.2 71.2 120.2 289.2
```

Рисунок 2.7 – Структура другого набору даних

```
--- Описова статистика ---
> summary(data_oil)
       V1               V2               V3               V4
 Min.   :  1.00   Min.   :   10.0   Min.   :   1.92   Min.   :   1.0
 1st Qu.: 31.00   1st Qu.:   20.0   1st Qu.:  85.27   1st Qu.: 444.2
 Median : 64.00   Median :   65.0   Median : 704.37   Median : 761.3
 Mean   : 81.59   Mean   :  332.8   Mean   : 698.71   Mean   : 871.0
 3rd Qu.:124.00   3rd Qu.:  132.0   3rd Qu.:1223.48   3rd Qu.:1260.4
 Max.   :352.00   Max.   :32389.0   Max.   :1893.08   Max.   :2724.6
       V5               V6               V7               V8
 Min.   :  0.00   Min.   :   70312   Min.   :21.24   Min.   : 0.830
 1st Qu.: 54.00   1st Qu.:  125000   1st Qu.:33.65   1st Qu.: 6.750
 Median : 73.00   Median :  186300   Median :39.97   Median : 8.200
 Mean   : 84.12   Mean   :  769696   Mean   :43.24   Mean   : 9.128
 3rd Qu.:117.00   3rd Qu.:  330468   3rd Qu.:52.42   3rd Qu.:10.760
 Max.   :180.00   Max.   :71315000   Max.   :82.64   Max.   :24.690
       V9              V10              V11              V12
 Min.   :   667   Min.   :0.020   Min.   : 41.0   Min.   :0.0200
 1st Qu.:  1371   1st Qu.:0.160   1st Qu.: 83.5   1st Qu.:0.2000
 Median :  2090   Median :0.200   Median : 99.8   Median :0.2400
 Mean   :  3941   Mean   :0.221   Mean   :109.9   Mean   :0.2514
```

Рисунок 2.8 – Описова статистика другого набору даних

2.3 Очищення даних: обробка пропусків

Очищення даних є одним із найважливіших етапів препроцесінгу, оскільки якість вхідної інформації безпосередньо визначає точність, узагальнювальну здатність та надійність побудованих моделей. У реальних екологічних даних часто

спостерігаються пропущені або некоректні значення, спричинені як технічними, так і природними факторами. Наприклад, збої у роботі сенсорів, помилки під час ручного введення або переривання зв’язку при зчитуванні даних можуть призвести до відсутності вимірювань окремих показників. Крім того, частина екологічних спостережень може бути пропущена через вплив погодних умов, аварійне вимкнення обладнання чи обмеження у часі та місці збору даних.

Процес очищення даних передбачає послідовне виконання кількох завдань: виявлення, аналізу, оцінки впливу та заповнення пропусків.

На першому етапі здійснюється аналіз наявності пропусків у кожній змінній. Для цього обчислюється кількість та частка відсутніх значень у кожному стовпці. У більшості випадків цей аналіз супроводжується візуалізацією пропусків за допомогою графічних інструментів, зокрема функції *missmap()* (див. рис. 2.9-2.10). Такі візуалізації дозволяють легко ідентифікувати змінні з найбільшою кількістю пропусків і виявити можливі закономірності їх появи. Якщо пропуски зосереджені в певних ділянках даних, це може свідчити про систематичну помилку під час збору або зчитування показників.

```
# 2. Аналіз пропусків ------------------------------------------------------
cat("\n--- Кількість пропусків ---\n")
print(colSums(is.na(water_data)))
missmap(water_data, main = "Пропущені значення")
```

Рисунок 2.9 – Аналіз пропусків з функцією missmap() для першого набору даних

На наступному етапі здійснюється оцінка масштабів пропусків. Якщо кількість відсутніх даних незначна (до 5–10% від загального обсягу), їх можна заповнити методами статистичної імпутації. У випадках, коли частка пропусків перевищує 30–40%, необхідно розглянути доцільність видалення таких ознак або застосування складніших моделей прогнозування пропущених значень.

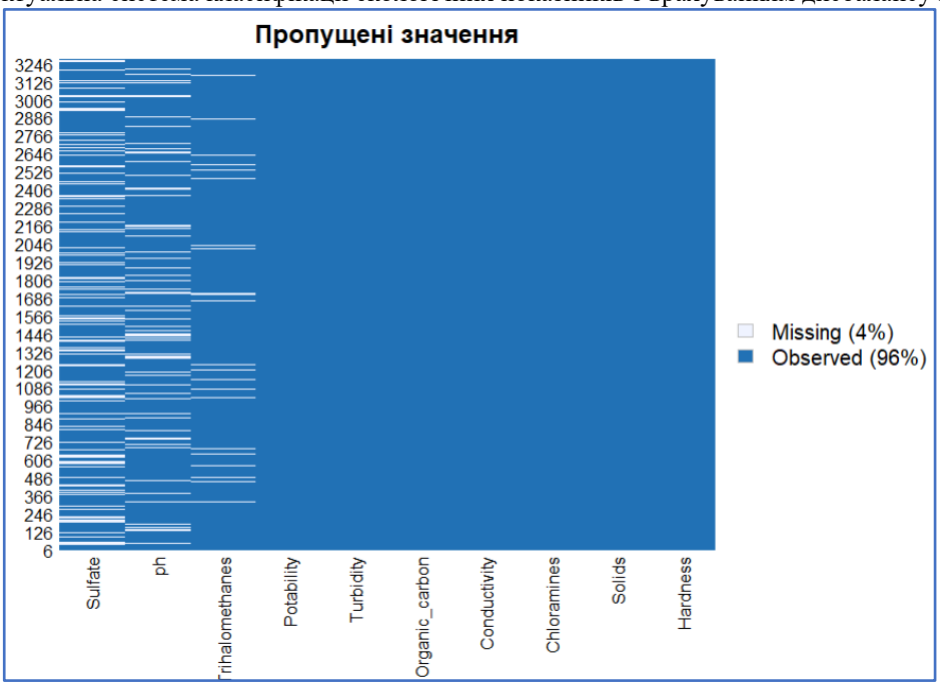

Рисунок 2.10 – Візуалізація пропусків функцією missmap() (перший набір даних)

Для першого набору даних найбільшу кількість пропусків виявлено у змінних Sulfate, pH та Trihalomethanes, що є типовим для екологічних вимірювань, оскільки ці показники часто залежать від умов відбору проб і калібрування приладів.

Після виявлення пропусків обирається метод їх обробки. Існують три основні підходи, такі як:

– видалення спостережень із пропусками. Цей метод застосовується лише тоді, коли відсутня незначна кількість записів, і їх видалення не впливає на репрезентативність вибірки;

– заповнення сталими або статистичними характеристиками (імпутація). Найчастіше використовуються середнє, медіана або мода залежно від типу змінної;

– імпутація на основі моделей. У більш складних випадках пропуски заповнюються з використанням методів МН (лінійна регресія, KNN-імпутація, дерева рішень, моделі ARIMA для часових рядів тощо).

У даній роботі для забезпечення узгодженості та збереження структури розподілу обрано заміщення пропусків медіанними значеннями. Цей метод має низку переваг, такі як:

– стійкість до викидів – на відміну від середнього, медіана не спотворюється під впливом аномально великих або малих значень;

– збереження розподілу даних – заповнення медіаною дозволяє підтримувати форму емпіричного розподілу без суттєвого зміщення;

– простота та ефективність – метод не потребує складних розрахунків і підходить для швидкої підготовки великих наборів даних;

– універсальність – може бути застосований як для числових, так і для нормалізованих ознак.

Для описаного процесу обробки пропусків є формула, яка використовувалась при заміщенні відсутніх значень медіаною (див. рис. 2.11-2.12).

$$X_i' = \begin{cases} X_i, & \text{якщо } X_i \text{ не є пропущеним значенням} \\ median(X), & \text{якщо } X_i \text{ є пропущеним значенням} \end{cases} \quad (2.1)$$

де X_i – вихідне значення змінної в і-му рядку;

X_i' – значення змінної після заміщення пропусків;

$median(X)$ – медіана вибірки X для відповідного атрибуту.

```
> # 3. Обробка пропусків (медіана) ----------------------------------------
> water_data <- water_data %>%
+   mutate(across(everything(), ~ ifelse(is.na(.), median(., na.rm = TRUE), .)))
```

Рисунок 2.11 – Обробка пропусків за допомогою медіани

```
--- Кількість пропусків ---
> print(colSums(is.na(water_data)))
             ph        Hardness          Solids     Chloramines         Sulfate    Conductivity
            491               0               0               0             781               0
 Organic_carbon Trihalomethanes       Turbidity      Potability
              0             162               0               0
```

Рисунок 2.12 – Вивід кількості пропусків

Після заповнення пропусків виконується перевірка результату очищення – повторна візуалізація матриці пропусків, щоб переконатися, що відсутні значення успішно замінено. Крім того, проводиться контроль статистичних характеристик (середнього, медіани, стандартного відхилення) до та після імпутації, щоб упевнитися у відсутності спотворень у структурі даних (див. рис. 2.13).

```
--- Перевірка після заповнення ---
> print(colSums(is.na(water_data)))
             ph        Hardness          Solids     Chloramines         Sulfate
              0               0               0               0               0
   Conductivity  Organic_carbon Trihalomethanes       Turbidity      Potability
              0               0               0               0               0
```

Рисунок 2.13 – Вивід результату після обробки пропусків

У другому наборі даних не було знайдено пропущені дані (див. рис. 2.14).

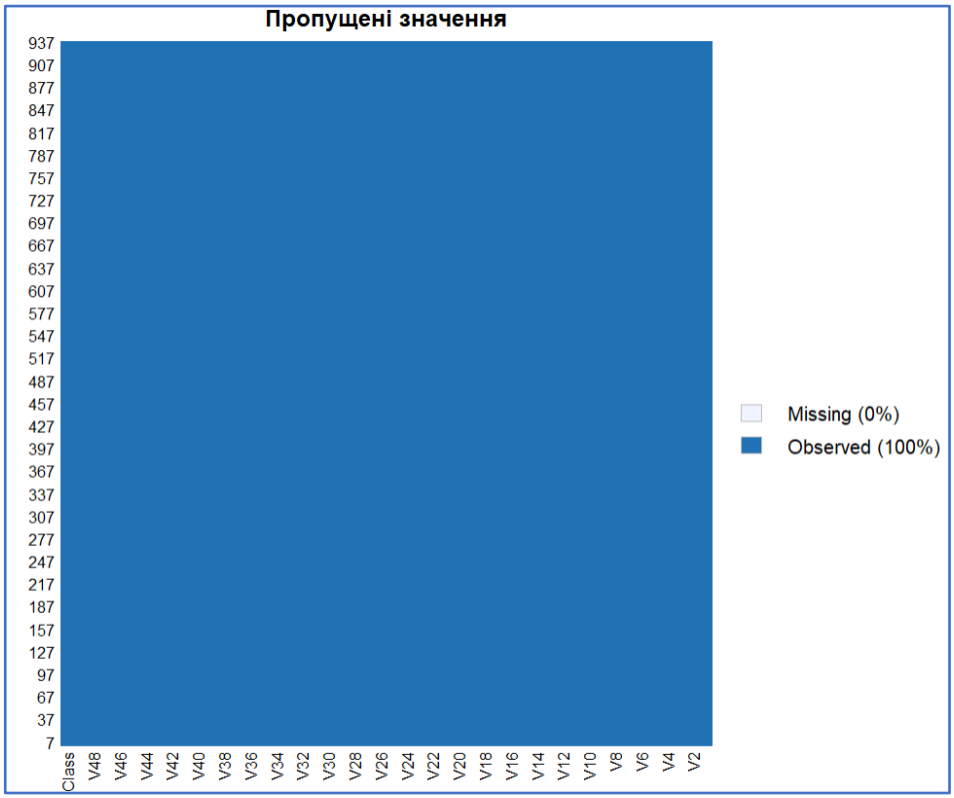

Рисунок 2.14 – Візуалізація пропусків за функцією missmap() для другого набору даних

Процес очищення екологічних даних від пропусків дозволяє підвищити якість та надійність інформаційної бази, мінімізувати ризики появи систематичних похибок і забезпечити стабільність результатів під час побудови моделей МН.

2.4 Масштабування та нормалізація числових ознак

Масштабування ознак є одним із ключових етапів підготовки даних перед застосуванням алгоритмів МН. Його мета – привести всі числові змінні до єдиного масштабу, що дозволяє уникнути переважання однієї ознаки над іншою через різницю в діапазонах значень. Це особливо важливо для алгоритмів, які залежать від відстаней між точками, таких як KNN, SVM, а також для моделей, що використовують градієнтний спуск, наприклад, нейронних мереж або логістичної регресії.

У роботі було виконано масштабування числових ознак для двох наборів даних. Спочатку для кожного набору даних категоріальні змінні були перетворені на фактори (Potability для води та Class для нафти), що дозволило виключити їх із процесу масштабування числових ознак.

Масштабування виконувалося за формулою стандартного відхилення (Z-score normalization):

$$X' = \frac{X-\mu}{\sigma} \tag{2.2}$$

де X'– нормалізоване значення ознаки;

X – оригінальне значення ознаки;

μ – середнє значення ознаки;

σ – стандартне відхилення.

Після цього середнє значення кожної ознаки стає рівним 0, а стандартне відхилення – 1. У результаті всі ознаки знаходяться в порівнянному масштабі, що запобігає спотворенню ваги ознак у моделі.

Для набору water_data було створено новий датафрейм *data_scaled_water*, у якому числові змінні були масштабовані за формулою Z-score.

```
# 4. Масштабування ознак ---------------------------
data_scaled_water <- water_data %>%
  mutate(Potability = as.factor(Potability)) %>%
  mutate(across(-Potability, scale))
```

Рисунок 2.15 – Масштабування першого набору даних

Візуалізація розподілу ознак першого набору даних до та після масштабування показала, що форма розподілу залишається незмінною, проте всі значення приведені до однакового масштабу (див. рис. 2.16). Це дозволяє легше порівнювати різні параметри води, такі як pH, твердості, вміст солей та інші.

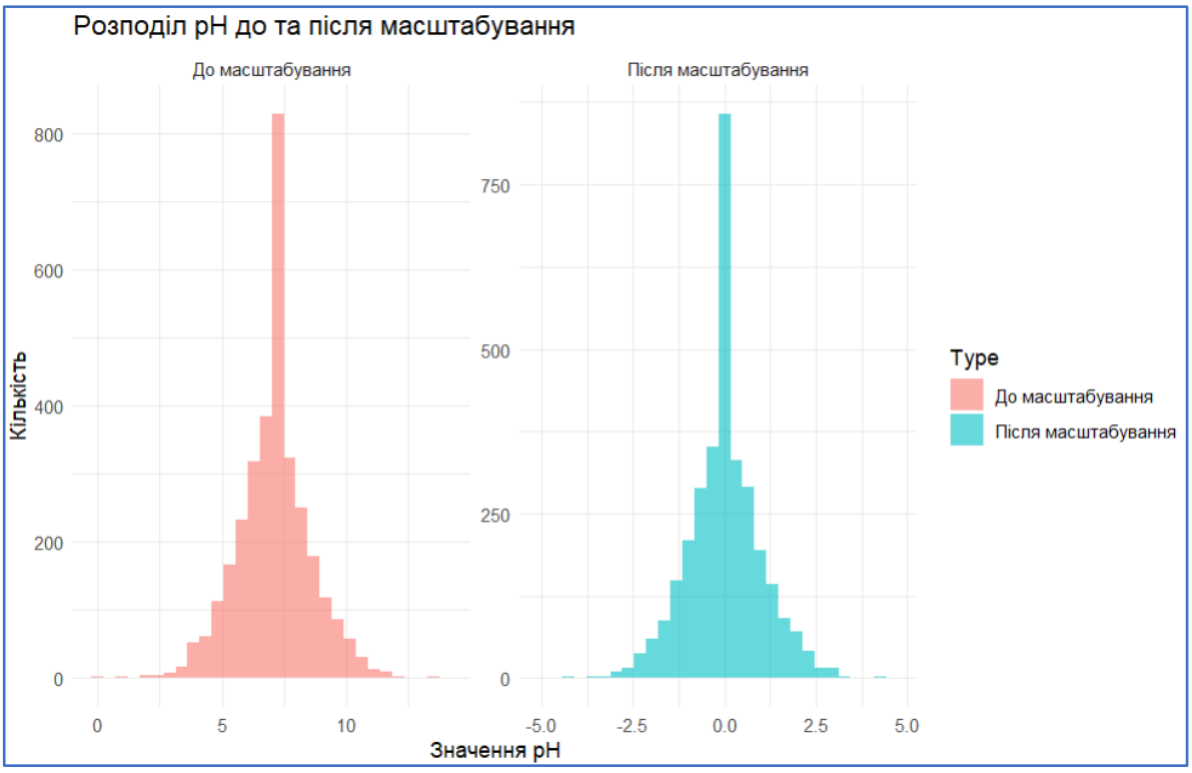

Рисунок 2.16 – Розподіл pH до та після масштабування

Аналогічно, для набору data_oil було створено *data_scaled_oil*. Окрім масштабування, були видалені ознаки з постійними значеннями, оскільки вони не несуть інформації для моделей МН. Це було виконано за допомогою перевірки

кількості унікальних значень для кожної змінної. Всі числові ознаки були перетворені у формат numeric після масштабування.

Для візуальної перевірки результатів масштабування у наборі нафти було порівняно розподіл однієї ознаки (наприклад, V10) до та після масштабування (див. рис. 2.17). Гістограми показали, що після масштабування значення розподілені навколо нуля, але загальна форма розподілу залишилася такою ж.

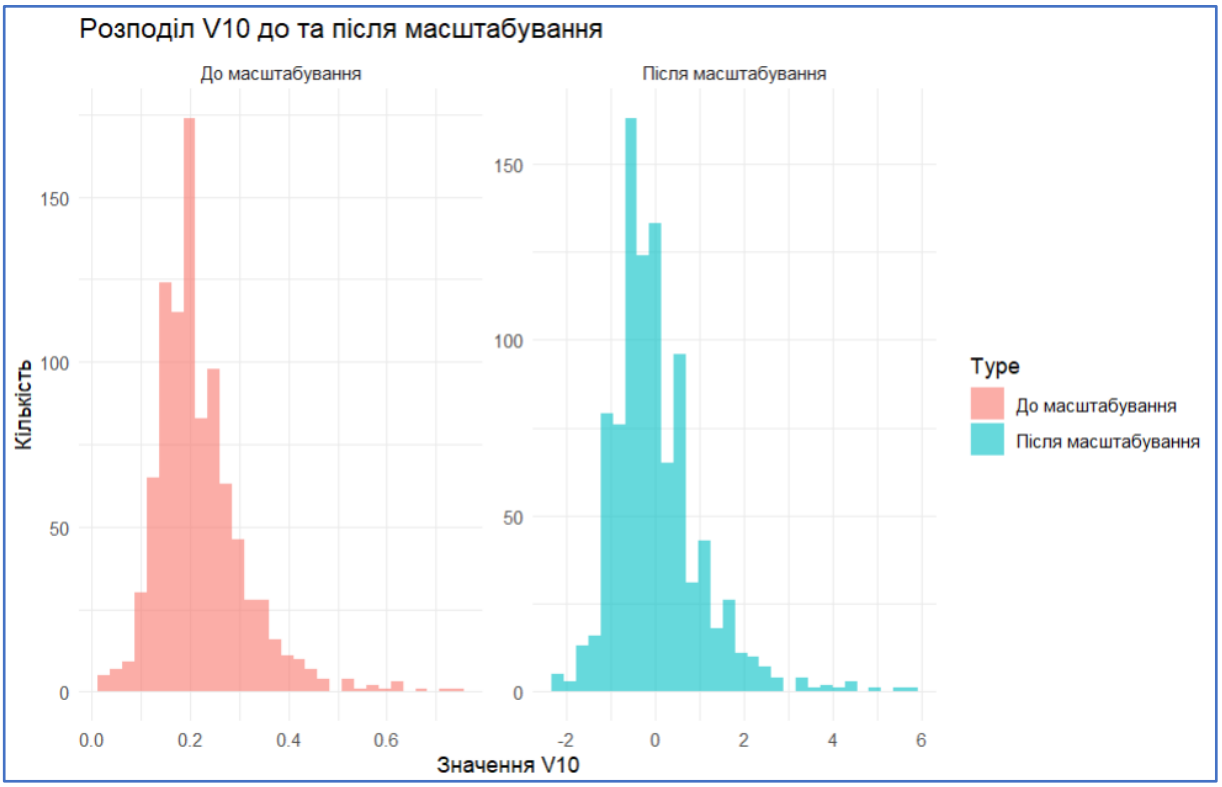

Рисунок 2.17 – Розподіл V10 до та після масштабування

Масштабування числових ознак виконує такі процеси, як:

– прискорити процес навчання моделей МН;

– запобігти переважанню ознак з великими діапазонами;

– забезпечити коректну роботу метрик відстані;

– покращити стабільність алгоритмів, що використовують градієнтний спуск.

У даному дослідженні масштабування було застосовано до всіх числових змінних, крім категоріальних, для обох наборів даних. Кожна ознака була стандартизована окремо, що дозволяє зберегти її відносні відмінності всередині набору. Масштабування не змінює інформацію, яка міститься у даних, а лише приводить її до зручного для моделей вигляду.

2.5 Методи балансування класів

На етапі препроцесінгу даних ефективним інструментом для боротьби з дисбалансом є методи повторної вибірки. Вони дозволяють регулювати кількість прикладів у різних класах, що робить навчальну вибірку більш збалансованою і сприяє підвищенню якості моделей. Завдяки своїй простоті та гнучкості методи повторної вибірки широко використовуються на практиці як на етапі підготовки даних, так і у складі комплексних підходів до вирішення задач класифікації.

Undersampling – це метод балансування класів, який полягає у зменшенні кількості прикладів більшості класу для досягнення більш рівномірного розподілу між класами. Суть методу полягає у виборі підмножини даних більшості класу таким чином, щоб їхня кількість стала порівнянною з кількістю прикладів меншості (див. рис. 2.18). Відбір може здійснюватися випадковим чином, що є найпростішим та найшвидшим підходом, або за певними стратегічними критеріями, наприклад, шляхом видалення прикладів, які є надмірно схожими на сусідні або знаходяться далеко від межі розділення класів.

Основними перевагами undersampling є простота реалізації та зменшення розміру навчальної вибірки, що дозволяє скоротити час навчання моделей.

Водночас метод має суттєвий недолік – при зменшенні числа прикладів більшості класу втрачається частина інформації, що може погіршити здатність моделі адекватно відображати різноманітність даних більшості класу. Попри це, undersampling залишається зручним і ефективним інструментом на етапі препроцесінгу даних, особливо коли потрібно швидко збалансувати вибірку за класами (див. рис. 2.19). Цей метод обчислено математично.

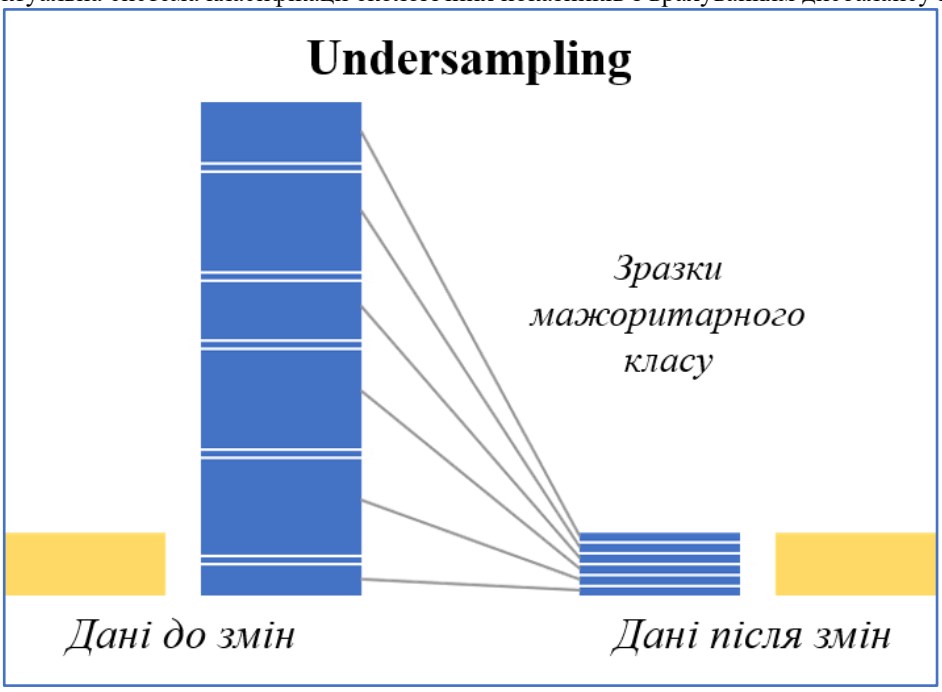

Рисунок 2.18 – Схематичне відображення методу Undersampling

$$S' = S'_{maj} \cup S_{min} \tag{2.3}$$

де S' – нова збалансована вибірка;

n_{maj} – кількість об’єктів у більшості;

n_{min} – кількість об’єктів у меншості;

S_{maj} – множина об’єктів більшості;

S_{min} – множина об’єктів меншості;

$S'_{maj} \subset S_{maj}$ – підмножина об’єктів більшості після undersampling;

$|S'_{maj}| = n_{min}$ – чисельність підмножини більшості після undersampling.

```
## 6.1. Undersampling
undersample_data_water <- downSample(
  x = data_scaled_water %>% select(-Potability),
  y = data_scaled_water$Potability
)
cat("\n--- Розподіл класів після undersampling ---\n")
print(table(undersample_data_water$Class))
```

```
## 6.1. Undersampling
undersample_data_oil <- downSample(
  x = data_scaled_oil %>% select(-Class),
  y = data_scaled_oil$Class
)
cat("\n--- Розподіл класів після undersampling ---\n")
print(table(undersample_data_oil$Class))
```

Рисунок 2.19 – Виконання Undersampling для двох наборів даних

Oversampling – це метод балансування класів, який полягає у збільшенні кількості прикладів меншості для досягнення більш рівномірного розподілу між класами (див. рис. 2.20). Найпростіший підхід передбачає дублювання існуючих прикладів меншого класу, що дозволяє врівноважити кількість зразків без втрати даних більшості класу. Завдяки цьому модель отримує більше інформації про менш представлений клас, що сприяє підвищенню точності її прогнозів на цих прикладах.

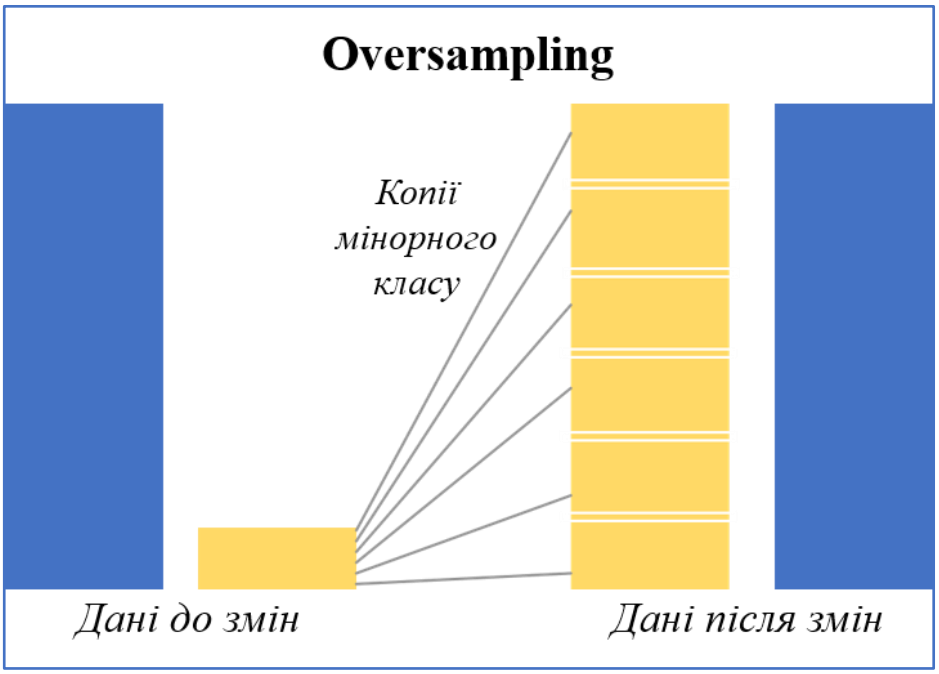

Рисунок 2.20 – Схематичне відображення методу Oversampling

Однак такий метод має певні обмеження, оскільки повторювані приклади можуть призводити до перенавчання, коли модель запам’ятовує конкретні зразки замість того, щоб навчитися загальним закономірностям. Oversampling є зручним і широко використовуваним інструментом на етапі препроцесінгу даних, особливо коли важливо зберегти всю інформацію більшості класу та підсилити представлення меншості (див. рис. 2.21). Цей метод обчислено математично.

$$|S'_{min}| = n_{maj} \tag{2.4}$$

де $S'_{min} = S_{min} \cup R$;

R – множина випадково вибраних (з повторенням) елементів з S_{min}.

Після цього формується нова вибірка.

$$S' = S_{maj} \cup S'_{min} \tag{2.5}$$

де S' – нова збалансована вибірка;

S_{maj} – множина об’єктів більшості;

S_{min} – множина об’єктів меншості;

n_{maj} – кількість об’єктів у більшості;

S'_{min} – підмножина меншості після oversampling;

R – множина випадково вибраних (з повторенням) елементів з S_{min}.

```r
## 6.2. Oversampling
oversample_data_water <- upSample(
  x = data_scaled_water %>% select(-Potability),
  y = data_scaled_water$Potability
)
cat("\n--- Розподіл класів після oversampling ---\n")
print(table(oversample_data_water$Class))
```

```r
## 6.2. Oversampling
oversample_data_oil <- upSample(
  x = data_scaled_oil %>% select(-Class),
  y = data_scaled_oil$Class
)
cat("\n--- Розподіл класів після oversampling ---\n")
print(table(oversample_data_oil$Class))
```

Рисунок 2.21 – Виконання Oversampling для двох наборів даних

SMOTE – це метод oversampling, який створює нові синтетичні приклади для менш представленого класу шляхом інтерполяції між існуючими точками та їхніми найближчими сусідами. На відміну від простого дублювання прикладів, SMOTE «заповнює прогалини» в просторі класу меншості, роблячи його більш щільним і дозволяючи моделі краще вивчати межі між класами. Такий підхід сприяє підвищенню здатності алгоритму класифікації розпізнавати менш представлені приклади та зменшує ризик перенавчання, характерний для простого дублювання. Водночас метод має певні обмеження: синтетичні точки можуть створюватися у зонах, де класи перетинаються, що іноді погіршує точність класифікації (див. рис. 2.22).

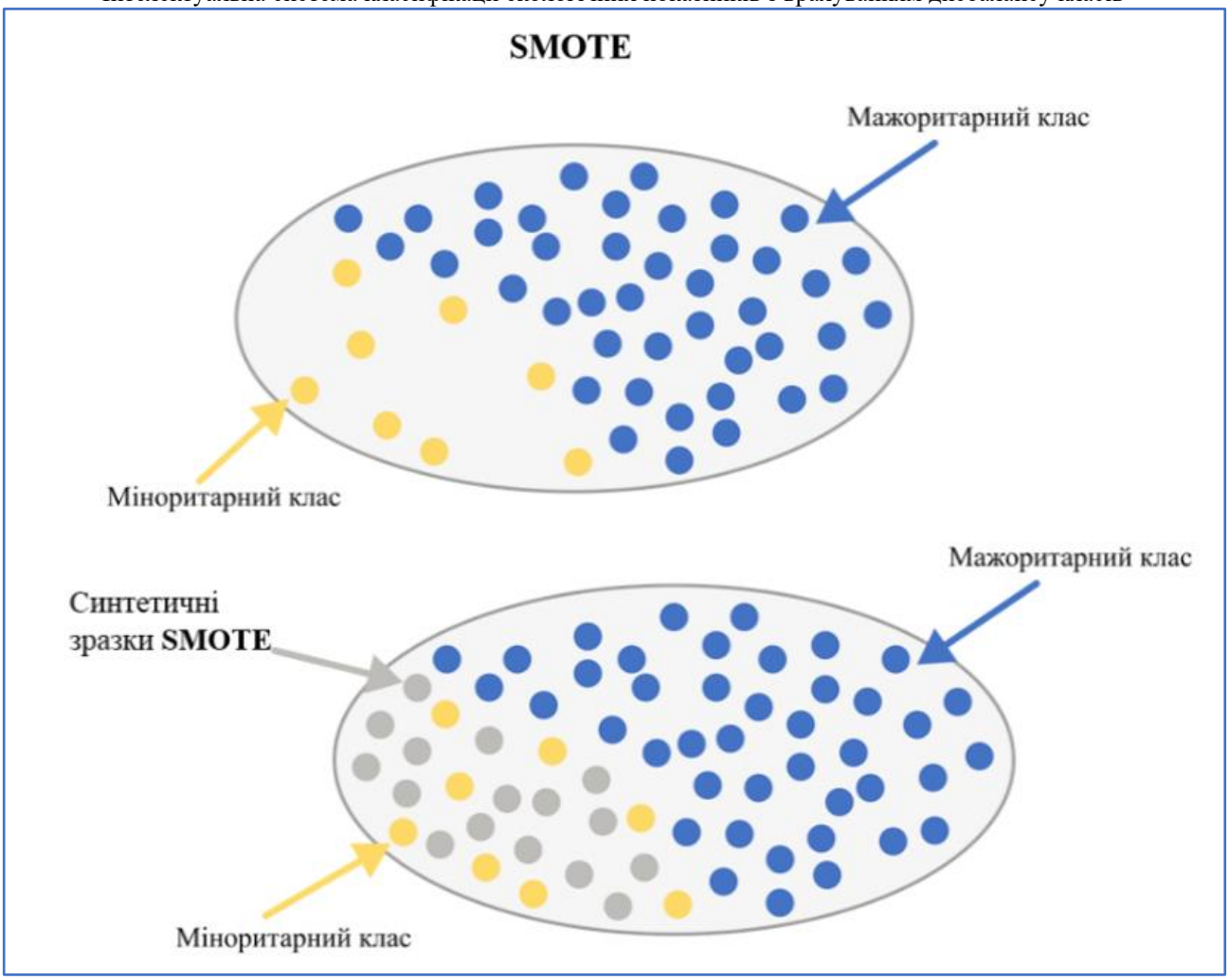

Рисунок 2.22 – Схематичне відображення методу SMOTE

Математично метод SMOTE формулюється так: для кожного зразка меншості x_i, випадково обирається один із його k-найближчих сусідів x_{zi} (див. рис. 2.23). Було створено новий синтетичний приклад.

$$x_{new} = x_i + \delta \times (x_{zi} - x_i) \tag{2.6}$$

де x_{new} – новий синтетичний приклад;

x_i – зразок класу меншості;

x_{zi} – випадково вибраний один із k-найближчих сусідів зразка x_i;

$\delta \sim U(0,1)$ – випадкове число з інтервалу [0, 1], рівномірно розподілене.

```r
## 6.3. SMOTE
target_num_water <- as.numeric(as.character(data_scaled_water$Potability))

set.seed(123)
smote_out_water <- SMOTE(
  X = data_scaled_water %>% select(-Potability),
  target = target_num_water,
  K = 5,
  dup_size = 1
)

smote_data_water <- smote_out_water$data %>%
  rename(Potability = class) %>%
  mutate(Potability = as.factor(Potability))
```

```r
## 6.3. SMOTE
smote_out_oil <- SMOTE(
  X = data_scaled_oil %>% select(-Class),
  target = data_scaled_oil$Class,
  K = 5
)
smote_data_oil <- smote_out_oil$data %>%
  rename(Class = class) %>%
  mutate(Class = as.factor(Class))

cat("\n--- Розподіл класів після SMOTE ---\n")
print(table(smote_data_oil$Class))
```

Рисунок 2.23 – Виконання SMOTE для двох наборів даних

ROSE – це метод oversampling, який полягає у збільшенні кількості прикладів меншості шляхом простого дублювання наявних зразків (див. рис. 2.24). На відміну від синтетичних підходів, нові точки не змінюють геометрію чи конфігурацію простору класу, оскільки вони розташовуються точно в тих же позиціях, що й оригінальні приклади меншості. Завдяки цьому підхід дозволяє зберегти існуючу структуру даних більшості класу та підсилити представлення меншості без ускладнення простору ознак. Недоліком методу є те, що повторювані приклади можуть підвищувати ризик перенавчання моделі, адже алгоритм бачить однакові зразки кілька разів.

Було описано математично метод ROSE.

$$x_{new} = x_i + \varepsilon \tag{2.7}$$

де x_{new} – новий синтетичний приклад;

x_i – випадково вибраний зразок із класу y_i;

$\varepsilon \sim N(0, \Sigma_y)$ – випадковий шум з нормального розподілу з матрицею коваріації Σ_y, оціненою для класу y (див. рис. 2.25).

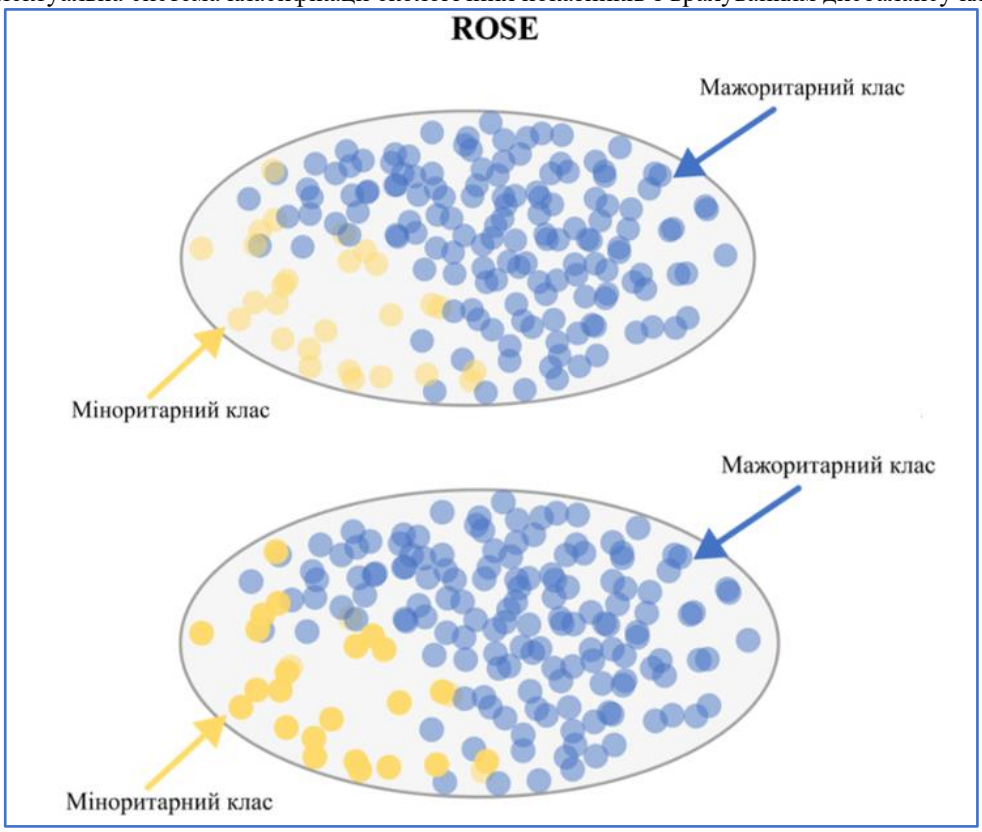

Рисунок 2.24 – Схематичне відображення методу ROSE

```r
## 6.4. ROSE
data_scaled_water <- water_data %>%
  mutate(Potability = as.factor(Potability)) %>%
  mutate(across(-Potability, ~ (scale(.) %>% as.numeric())))

set.seed(123)
rose_data_water <- ROSE(Potability ~ ., data = data_scaled_water, seed = 1)
cat("\n--- Розподіл класів після ROSE ---\n")
print(table(rose_data_water$Potability))
```

```r
## 6.4. ROSE
set.seed(123)

rose_data_oil <- ROSE(
  Class ~ .,
  data = data_scaled_oil,
  N = nrow(data_scaled_oil),
  seed = 123
)$data

cat("\n--- Розподіл класів після ROSE ---\n")
print(table(rose_data_oil$Class))
```

Рисунок 2.25 – Виконання ROSE для двох наборів даних

Результати *першого набору даних*.

До балансування: незначний дисбаланс (Клас0 = 1998, Клас1 = 1278).

Undersampling: більший клас скоротили до 1278, тепер обидва класи рівні – баланс досягнуто, але втрачено 720 прикладів класу 0.

Oversampling: менший клас збільшили до 1998 – баланс і збереження всіх даних більшого класу, але додані копії прикладів класу 1.

SMOTE: синтетично створено нові приклади класу 1, щоб зрівняти його з

класом 0 (1998 проти 1998). Це більш природне заповнення простору, ніж просте дублювання.

ROSE: обидва класи наближено зрівняні (1648 і 1628) за рахунок генерації нових точок із шумом. Результат – майже баланс, але з меншим відхиленням.

Результати *другого набору даних*.

До балансування: дуже сильний дисбаланс (Клас0 = 896, Клас1 = 41).

Undersampling: обидва класи зменшені до 41 – баланс є, але втрачено величезну кількість даних класу 0.

Oversampling: клас 1 збільшили до 896 – повний баланс, але створено велику кількість копій, що може призвести до перенавчання.

SMOTE: клас 1 збільшено до 861 (майже рівно з 896) за рахунок синтетичних точок – баланс досягнуто без простого дублювання.

ROSE: обидва класи майже зрівняні (471 і 466) – баланс, але обидві кількості менші за початкові. Метод створив «розмитий» баланс з урахуванням розподілу.

Результати подані таблично та графічно на рис. 2.26 та на рис. 2.27.

```
> print(dist_water)
             Метод Клас0 Клас1
1 До балансування  1998  1278
2   Undersampling  1278  1278
3    Oversampling  1998  1998
4           SMOTE  1998  1998
5           ROSE  1648  1628
```

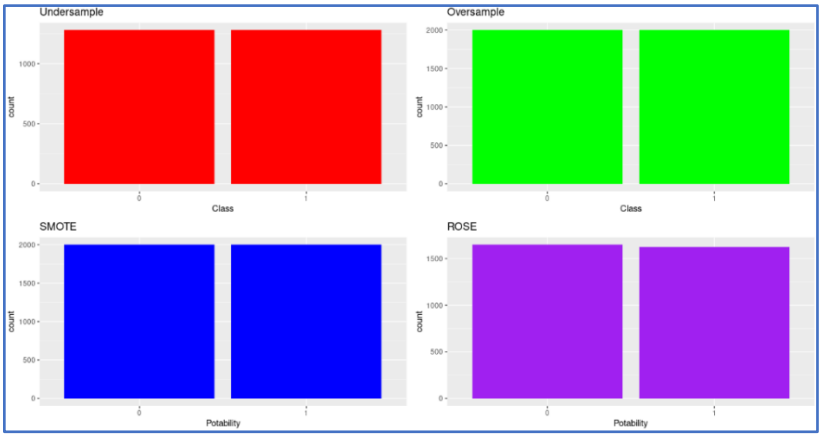

Рисунок 2.26 – Табличні та графічні результати виконаного балансування для першого набору даних

```
> print(dist_oil)
             Метод Клас0 Клас1
1 До балансування   896    41
2   Undersampling    41    41
3    Oversampling   896   896
4           SMOTE   896   861
5           ROSE    471   466
```

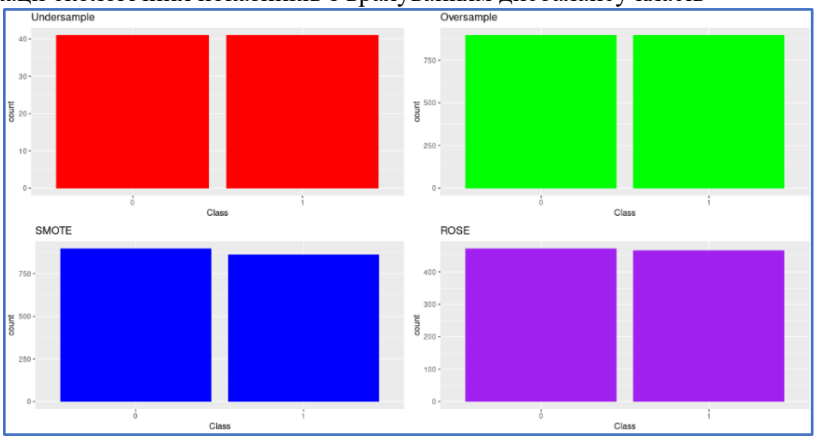

Рисунок 2.27 – Табличні та графічні результати виконаного балансування для другого набору даних

Для оцінки якості різних методів балансування було використано *ROC-кривy*, що демонструє співвідношення чутливості (Sensitivity, True Positive Rate) та 1-специфічності (False Positive Rate) для різних порогів класифікації. Чим ближче крива до верхнього лівого кута, тим краща модель.

AUC – площа під ROC-кривою, числовий показник якості класифікації: 1.0 – ідеальна модель; 0.5 – випадкове вгадування.

На рис. 2.28 видно, що AUC ≈ 0.5 означає, що модель класифікує майже на рівні випадкового вгадування. Такий результат є через те, що у першому наборі даних був незначний дисбаланс. А для даних з незначним дисбалансом використання методів для усунення дисбалансу є малоефективним.

Найкращий результат дав ROSE (AUC = 0.55), але це лише трохи краще за випадковість. Усі інші методи балансування (undersampling, oversampling, SMOTE) суттєво не покращили якість (AUC тримається на рівні 0.52–0.53).

У другому наборі даних усі методи, крім ROSE, показують майже ідеальну роботу моделі (AUC ≈ 1) (див. рис. 2.29). Undersampling дав AUC = 1, але треба перевірити, чи не через втрату даних модель стала занадто простою. Oversampling і SMOTE – оптимальні методи, бо зберігають усю інформацію і дають високу якість. ROSE у вашому випадку працює гірше (AUC < 0.9), тому його краще уникати.

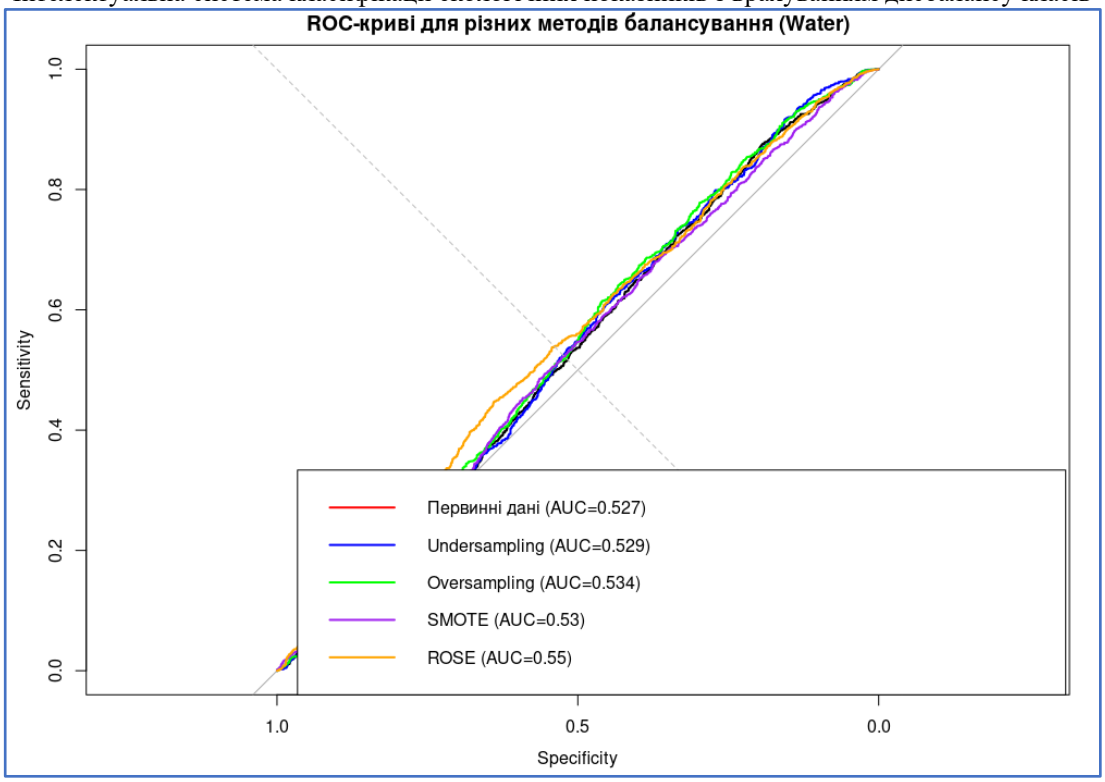

Рисунок 2.28 – ROC-криві для першого набору даних

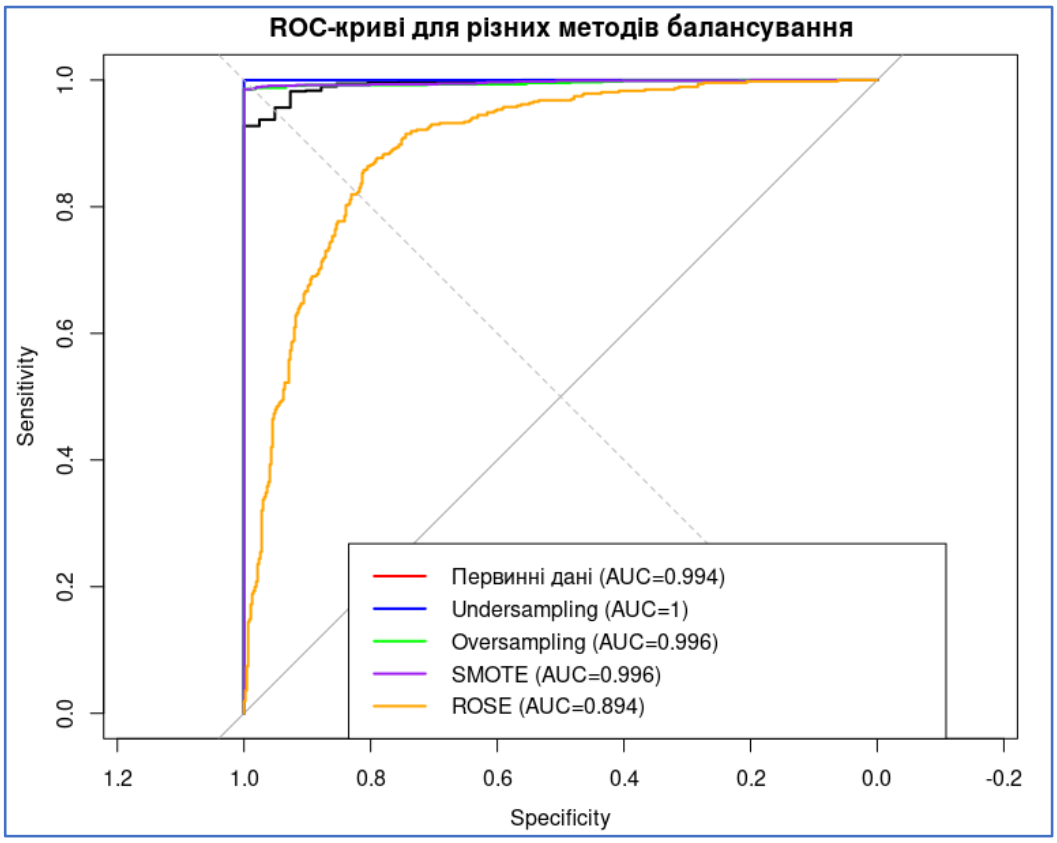

Рисунок 2.29 – ROC-криві для другого набору даних

2.6 Формування навчальної та тестової вибірки для подальшого моделювання

Для оцінки здатності моделі МН узагальнювати знання на нових даних використовують поділ початкового набору даних на тренувальну та тестову вибірки. Тренувальна вибірка призначена для побудови моделі та навчання алгоритму, тоді як тестова вибірка застосовується лише для оцінки якості моделі після навчання (див. рис. 2.30). Такий підхід дозволяє уникнути перенавчання, оскільки модель не бачить тестові дані під час процесу навчання і, відповідно, не підлаштовується під них.

```
--- Розподіл у train ---
> print(table(train_data_water$Potability))

   0    1
1399 1399
> cat("\n--- Розподіл у test ---\n")

--- Розподіл у test ---
> print(table(test_data_water$Potability))

  0   1
599 599
```

```
> cat("\n--- Розміри train/test ---\n")

--- Розміри train/test ---
> print(dim(train_data_oil))
[1] 657  49
> print(dim(test_data_oil))
[1] 280  49
```

Рисунок 2.30 – Розподіл двох наборів даних на навчальну та тестову вибірки

У випадках, коли дані мають дисбаланс класів, балансування необхідно виконувати лише на тренувальній вибірці. Це запобігає витоку інформації з тестової частини і забезпечує коректність оцінювання результатів. Одним із поширених методів балансування є SMOTE, який створює синтетичні приклади менш представленого класу. Завдяки цьому мінорний клас отримує додаткові приклади, що робить розподіл класів більш рівномірним і дозволяє моделі ефективніше навчатися на тренувальних даних.

Для візуального оцінювання впливу балансування на структуру даних застосовується метод PCA-візуалізації (Principal Component Analysis), який дозволяє проєктувати багатовимірні дані у дво- або тривимірний простір, зберігаючи при цьому максимально можливу інформацію про структуру та варіації

даних. На графіках PCA можна побачити, як балансування за допомогою SMOTE впливає на розподіл прикладів: до застосування методу набори даних демонструють помітний дисбаланс, а після створення синтетичних прикладів мінорного класу розподіл стає більш збалансованим (див. рис. 2.31). Це покращує представлення меншості у просторі ознак і сприяє більш стабільному та точному навчанню моделей на обох наборах даних.

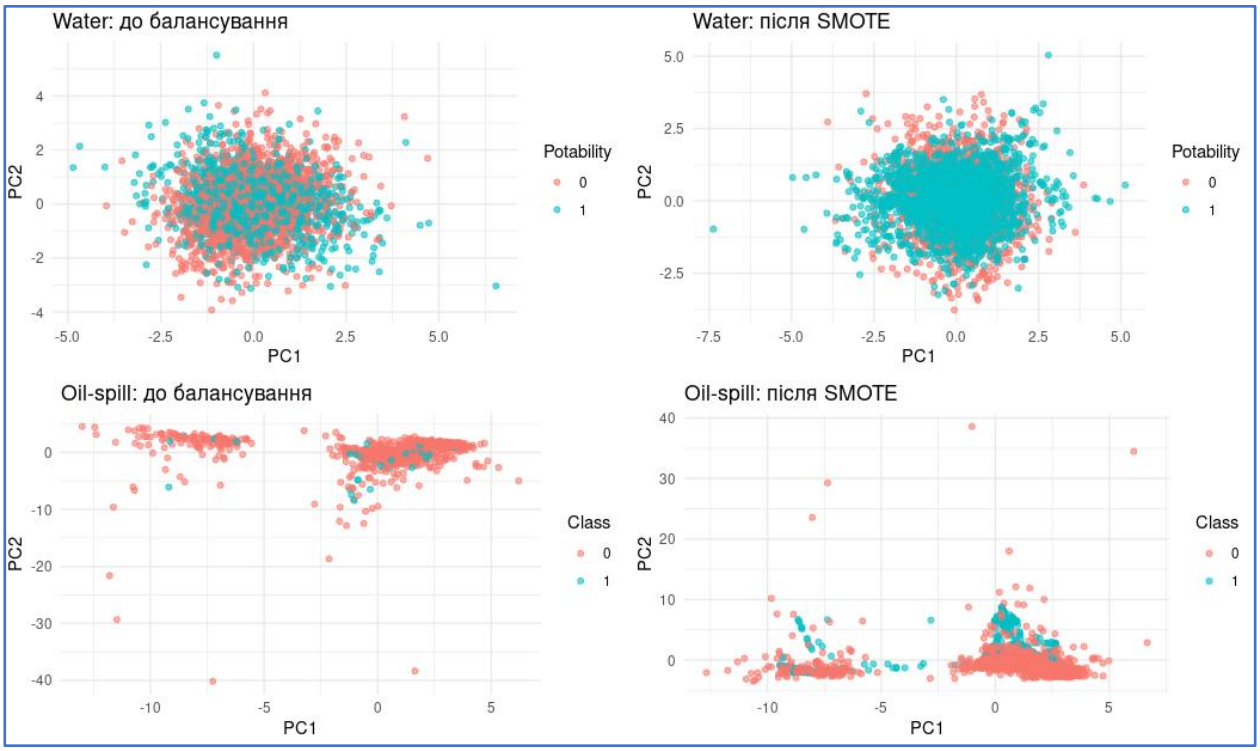

Рисунок 2.31 – Набори даних до та після балансування

Комбінація правильного поділу на тренувальну і тестову вибірки та застосування балансування мінорного класу дозволяє отримати надійніші та узагальнювальні моделі МН, що адекватно працюють з новими, раніше невідомими даними.

Висновки до розділу 2

Другий розділ зосереджений на етапах попередньої обробки екологічних даних, що є обов’язковою умовою для побудови коректних і стійких моделей

класифікації. На основі проведеного аналізу встановлено, що екологічні показники характеризуються наявністю пропусків, шумів, аномальних значень та різними масштабами величин, що потребує комплексного підходу до препроцесінгу. Детально розглянуті методи очищення даних дозволяють зменшити вплив некоректних вимірювань і сформувати якісну навчальну вибірку.

У розділі обґрунтовано вибір методів заповнення пропусків, серед яких зокрема статистичні підходи та алгоритмічні методи. Показано, що правильне заповнення пропусків суттєво впливає на підсумкову якість моделі, особливо коли мова йде про рідкісні класи. Масштабування й нормалізація числових ознак визначені як ключові кроки для коректної роботи алгоритмів, чутливих до діапазону значень.

Особливу увагу в розділі приділено методам балансування класів. Систематизовано їх переваги та недоліки, визначено умови, за яких кожен метод є найбільш ефективним. Показано, що використання балансування перед моделюванням суттєво підвищує ефективність алгоритмів у задачах з нерівномірною структурою даних.

Заключна частина розділу присвячена формуванню навчальної та тестової вибірок, де наголошено на важливості коректного поділу даних для уникнення переобучення. У результаті виконаних досліджень обґрунтовано, що якісний препроцесінг є критичним етапом, який визначає ефективність подальших експериментів. Таким чином, розділ підтвердив, що належна підготовка даних є необхідною умовою для створення точної та надійної інтелектуальної системи класифікації.

3 НАВЧАННЯ МОДЕЛІ КЛАСИФІКАЦІЇ

3.1 Загальна характеристика підходів до навчання моделей при дисбалансі класів

На етапі препроцесінгу даних можуть застосовуватися різноманітні техніки вирівнювання вибірки. Проте навіть після балансування даних залишковий вплив дисбалансу може проявлятися під час навчання моделі. Тому важливо враховувати цю проблему і на етапі моделювання. Саме на цьому етапі застосовуються підходи, які безпосередньо змінюють принципи навчання або оцінювання втрат, дозволяючи алгоритму надавати більшу вагу помилкам при класифікації рідкісного класу.

Із сучасних методів для навчання при дисбалансі класів можна виокремити на дві основні групи: методи навчання, чутливі до втрат (cost-sensitive learning), та ансамблеві методи. Перша група спрямована на модифікацію функції втрат або параметрів навчання з урахуванням ваг класів. Друга група використовує комбінацію кількох моделей для підвищення стійкості та точності прогнозування, часто поєднуючи бустинг, беггінг або ресемплінг усередині ансамблю.

Методи навчання, чутливі до втрат, передбачають, що кожен клас у задачі класифікації має власну вагу, яка відображає його значущість або рідкість. Наприклад, помилка при класифікації об’єкта з менш представленого класу може бути «дорожчою», ніж помилка при класифікації об’єкта з домінуючого класу. Такий підхід дозволяє моделі компенсувати дисбаланс, фокусуючись на важливіших, але менш численних прикладах. У практичних реалізаціях це досягається через параметри class_weight або scale_pos_weight у таких алгоритмах, як Logistic Regression, SVM, Random Forest чи XGBoost. Крім того, для дерев рішень може використовуватися матриця витрат, де задається різна вартість помилок різних типів.

Інша група – ансамблеві методи – базується на об’єднанні кількох моделей, кожна з яких може працювати із різними підвибірками даних або мати власні ваги для прикладів. Такий підхід дозволяє зменшити вплив випадкових помилок окремих моделей і забезпечує більш збалансовані результати. У випадку задач із дисбалансом класів ансамблеві методи часто поєднують техніки ресемплінгу із бустингом або беггінгом. Наприклад, методи RUSBoost і SMOTEBoost комбінують undersampling або oversampling із алгоритмом AdaBoost, тоді як Balanced Bagging і EasyEnsemble формують кілька збалансованих підвибірок даних для тренування незалежних моделей.

Завдяки своїй гнучкості ансамблеві підходи часто перевершують окремі базові алгоритми, зокрема SVM, KNN чи нейронні мережі, у задачах, де дисбаланс класів є суттєвим. Вони підвищують стабільність прогнозів і дозволяють покращити показники F1-score, AUC-ROC та Balanced Accuracy, зберігаючи при цьому високу узагальнювальну здатність моделі.

Обидві групи методів – і чутливі до втрат, і ансамблеві – мають спільну мету: покращити якість класифікації рідкісного класу без додаткової зміни вибірки. Якщо на етапі препроцесінгу основна увага зосереджується на зміні структури даних (додавання або видалення прикладів), то на етапі навчання фокус переноситься на зміну поведінки моделі. Це дозволяє зберігати цілісність вихідних даних і водночас підвищувати чутливість моделі до менш представлених спостережень.

Етап навчання моделі класифікації у задачах із дисбалансом класів є критично важливим і повинен враховувати не лише структуру даних, а й особливості функції втрат та поведінки алгоритмів. Застосування методів, чутливих до втрат, та ансамблевих підходів забезпечує підвищення точності і стійкості моделей, дозволяючи більш ефективно виявляти рідкісні, але значущі випадки. У подальших підрозділах розглянуто особливості реалізації цих методів у межах розробленої системи та їх порівняльну ефективність.

3.2 Методи навчання, чутливі до втрат

Методи навчання, чутливі до втрат, є ефективним підходом для вирішення проблеми дисбалансу класів без зміни структури вибірки. Їхня основна ідея полягає у введенні різних ваг для кожного класу у функцію втрат, щоб надати більшої значущості помилкам при класифікації рідкісних випадків. Тобто система «карає» модель сильніше, якщо вона неправильно класифікує приклад із менш представленого класу. Завдяки цьому модель навчається приділяти більше уваги прикладам меншості, що підвищує показники чутливості та F1-міри для цього класу. Обчислено загальний вигляд функції втрат із вагами.

$$L = -\sum_{i=1}^{N} w_{y_i}[y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)] \quad (3.1)$$

де L – значення функції втрат;

N – загальна кількість зразків;

i – індекс поточного зразка;

w_{y_i} – вага класу, до якого належить зразок y_i;

y_i – справжнє значення класу;

$\hat{y}_i$ – прогнозована ймовірність приналежності зразка до позитивного класу.

Логістична регресія

Логістична регресія є однією з базових моделей класифікації, яка передбачає ймовірність належності об’єкта до певного класу. Для задач із дисбалансом класів у функцію втрат логістичної регресії вводяться ваги, що компенсують нерівність між кількістю прикладів кожного класу. У пакеті R це реалізується через аргумент *weights* у функції *glm*() [21]. Ваги для менш представленого класу розраховуються пропорційно до відношення кількості негативних і позитивних прикладів.

$$w_1 = \frac{N_{neg}}{N_{pos}}, w_0 = 1 \quad (3.2)$$

де w_1 – вага позитивного класу;

w_0 – вага негативного класу;

N_{neg} – кількість зразків негативного класу;

N_{pos} – кількість зразків позитивного класу.

Таким чином, якщо клас «1» зустрічається рідше, його приклади отримують більшу вагу при навчанні, що зменшує кількість хибно негативних прогнозів. Обчислено функцію втрат для логістичної регресії з вагами (див. формулу 3.1).

За результатами тестування для першого набору даних: AUC (Logistic Regression)=0,508.

Результат свідчить про слабку здатність моделі до класифікації в умовах значного дисбалансу. Логістична регресія залишається базовим методом, однак для таких задач потребує додаткової обробки або ресемплінгу.

Для візуалізації результату було побудовано щільність прогнозованих ймовірностей. Графік показує, що прогнозовані ймовірності мають сильне перекриття між класами, що пояснює низьке значення AUC (див. рис. 3.1).

За результатами тестування для другого набору даних: AUC (Logistic Regression) = 0,78.

На рис. 3.2 показано щільність прогнозованих ймовірностей для моделі логістичної регресії. Видно, що більшість спостережень класу 0 мають низькі ймовірності (<0.25), тоді як приклади класу 1 розподілені ближче до області високих значень (0.75–1.0). Отримане значення AUC підтверджує помірну здатність моделі розрізняти класи, покращену завдяки використанню ваг, пропорційних дисбалансу вибірки.

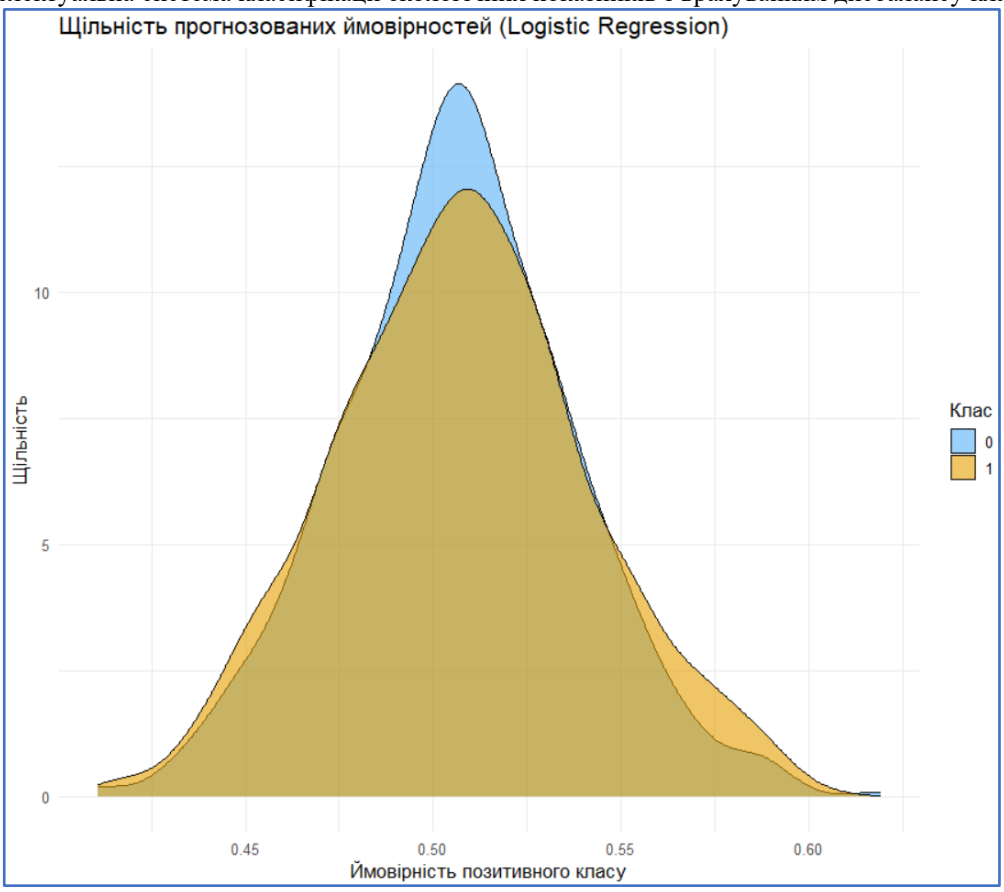

Рисунок 3.1 – Щільність прогнозованих ймовірностей (перший набір даних)

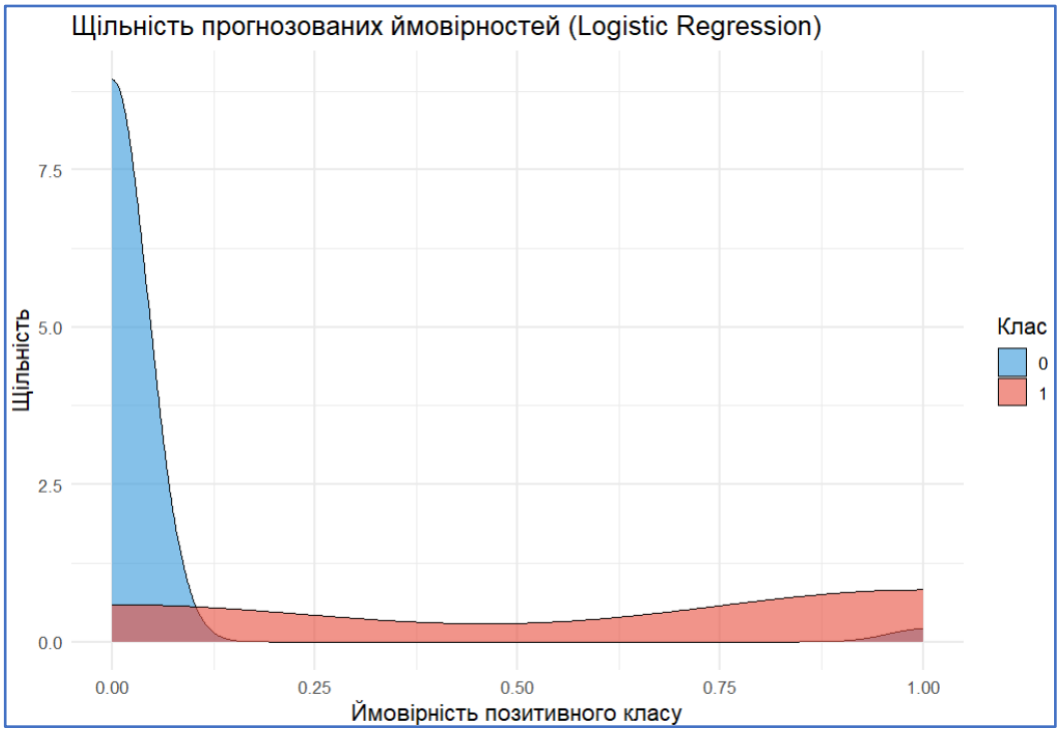

Рисунок 3.2 – Щільність прогнозованих ймовірностей (другий набір даних)

Метод опорних векторів (SVM)

Метод опорних векторів також підтримує навчання, чутливе до втрат, через введення ваг для кожного класу. У SVM ваги враховуються у функції оптимізації.

$$min\frac{1}{2}||w||^2 + C\sum_{i=1}^{N} w_{y_i}\xi_i \quad (3.3)$$

де w – вектор вагових коефіцієнтів моделі;

$||w||^2$ – квадрат евклідової норми вектора w;

C – параметр регуляризації , що контролює баланс між мінімізацією похибок і нормою ваг;

ξ_i – змінна похибки для зразка *i*;

w_{y_i} – ваговий коефіцієнт для класу;

N – загальна кількість зразків.

У коді це реалізовано за допомогою параметра class.weights у функції svm() з бібліотеки **e1071** [22]. Модель використовувала RBF-ядерну функцію для розділення класів у нелінійному просторі.

Ваги класів розраховувалися аналогічно: клас із меншою кількістю прикладів отримував вагу, пропорційну співвідношенню neg/pos. Це дозволило алгоритму приділяти більше уваги менш представленому класу при побудові межі розділення.

За результатами тестування для першого набору даних: AUC (SVM)=0,733.

Результат є дещо нижчим за інші методи, але демонструє стабільність навіть при нелінійних залежностях.

У якості візуалізації було побудовано розподіл класів у просторі головних компонент – двовимірне зображення після зведення ознак за допомогою PCA (див. рис. 3.3). Воно показує, що класи частково перекриваються, що ускладнює точне розділення навіть для нелінійних моделей.

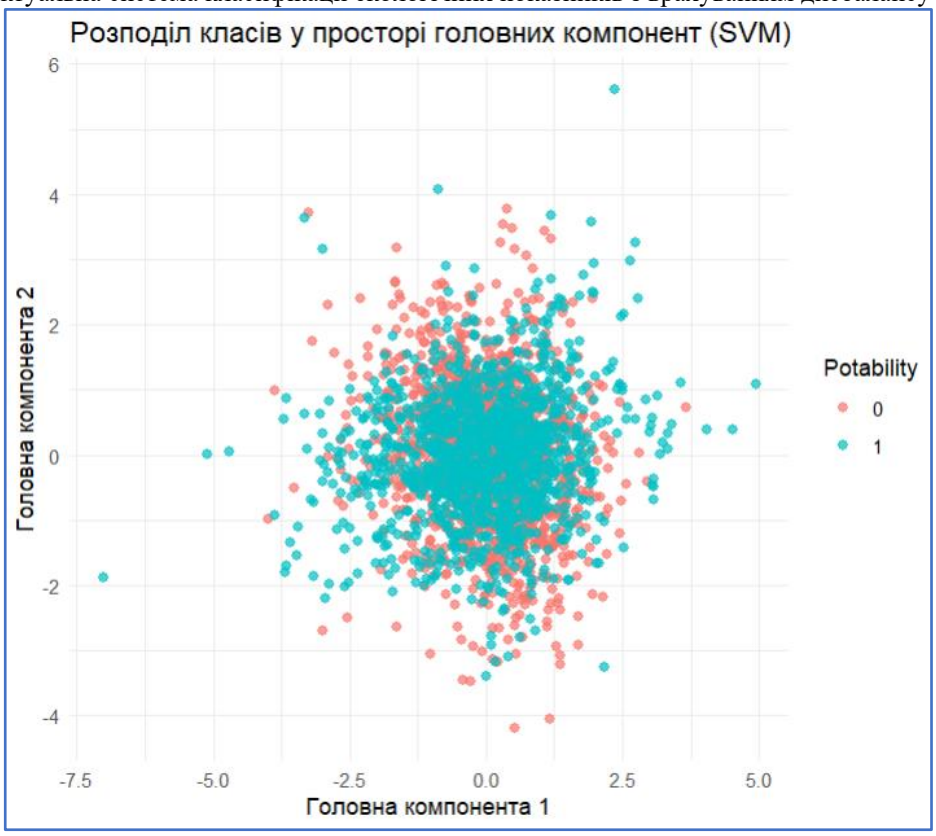

Рисунок 3.3 – Розподіл класів у просторі головних компонент (перший набір даних)

SVM-модель другого набору навчалася аналогічно першому набору – з радіальним базисним ядром (RBF) та вагами класів, заданими через параметр class.weights.

За результатами тестування для другого набору даних: AUC (SVM) = 0,854.

Отримане значення демонструє кращу здатність SVM моделювати нелінійні межі між класами, зберігаючи стабільність результатів навіть у випадку нерівномірного розподілу даних.

На рис. 3.4. видно, що класи частково перекриваються, однак модель здатна розмежовувати їх завдяки нелінійній границі, що узгоджується з характеристиками RBF-ядра. Скупчення точок поблизу нульових значень головних компонент свідчить про наявність основної маси даних у спільній області, де класи мають складну структуру розділення.

Рисунок 3.4 – Розподіл класів у просторі головних компонент (другий набір даних)

Random Forest

Random Forest – метод випадкових дерев рішень, який також може враховувати ваги класів. Принцип його роботи базується на навчанні багатьох дерев на випадкових підмножинах даних і ознак, після чого результати голосування агрегуються. Для корекції дисбалансу у функції навчання кожного дерева використовується коефіцієнт ваги класів.

$$L = \sum_{i=1}^{N} w_{y_i} * I(\hat{y}_i \neq y_i) \quad (3.4)$$

де L – значення функції втрат;

N – загальна кількість зразків;

i – індекс поточного зразка;

w_{y_i} – ваговий коефіцієнт для класу, до якого належить зразок y_i;

y_i – справжнє значення класу;

$\hat{y}_i$ – прогнозоване значення класу;

$I(\hat{y}_i \neq y_i)$ – індикатор помилки , який дорівнює 1, якщо прогноз не збігається з істинним класом, і 0 інакше.

У реалізованій системі це здійснено через параметр *classwt* у функції *randomForest*() з бібліотеки *randomForest* [23]. Для менш представленого класу встановлювалася вага *neg/pos*, що збільшувала його вплив у процесі побудови дерев. Такий підхід дозволяє алгоритму краще враховувати рідкісні випадки при виборі гілок і підвищує стабільність результатів при великих вибірках.

За результатами тестування для першого набору даних: AUC (Random Forest)=0,81.

Результат виявився найкращим серед протестованих алгоритмів. Це свідчить, що цей метод з урахуванням ваг класів добре справляється з дисбалансом.

Важливість ознак – діаграма демонструє відносний внесок кожної змінної у прогноз (див. рис. 3.5). Найбільш інформативні показники pH, розчинений кисень, твердість води. Вони збігаються з результатами XGBoost.

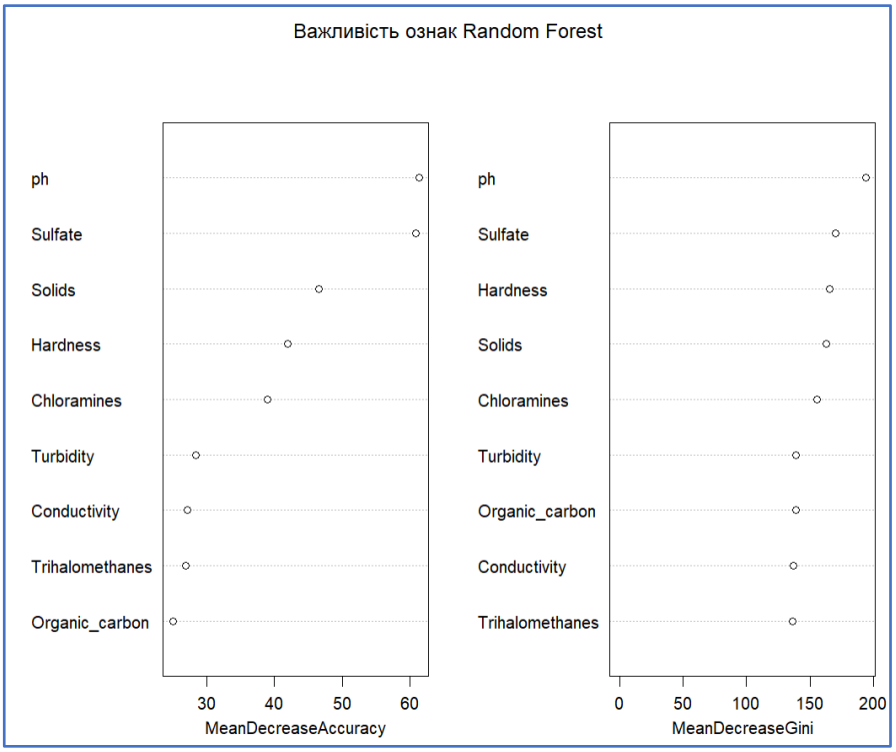

Рисунок 3.5 – Важливість ознак (перший набір даних)

Модель другого набору даних навчалася на 500 деревах, із добором кількості ознак *mtry = sqrt(p)*. Ваги меншості сприяли підвищенню уваги до рідкісних спостережень.

За результатами тестування для другого набору даних: AUC (Random Forest) = 0,906.

Найбільший внесок у точність класифікації мають ознаки V47, V48, V3, V25 та V49, які суттєво впливають на результати моделі (див. рис. 3.6). Решта ознак демонструють нижчу, але помітну інформативність. Такий розподіл свідчить, що кілька ключових змінних домінують у формуванні рішень дерев, тоді як більшість інших відіграють допоміжну роль

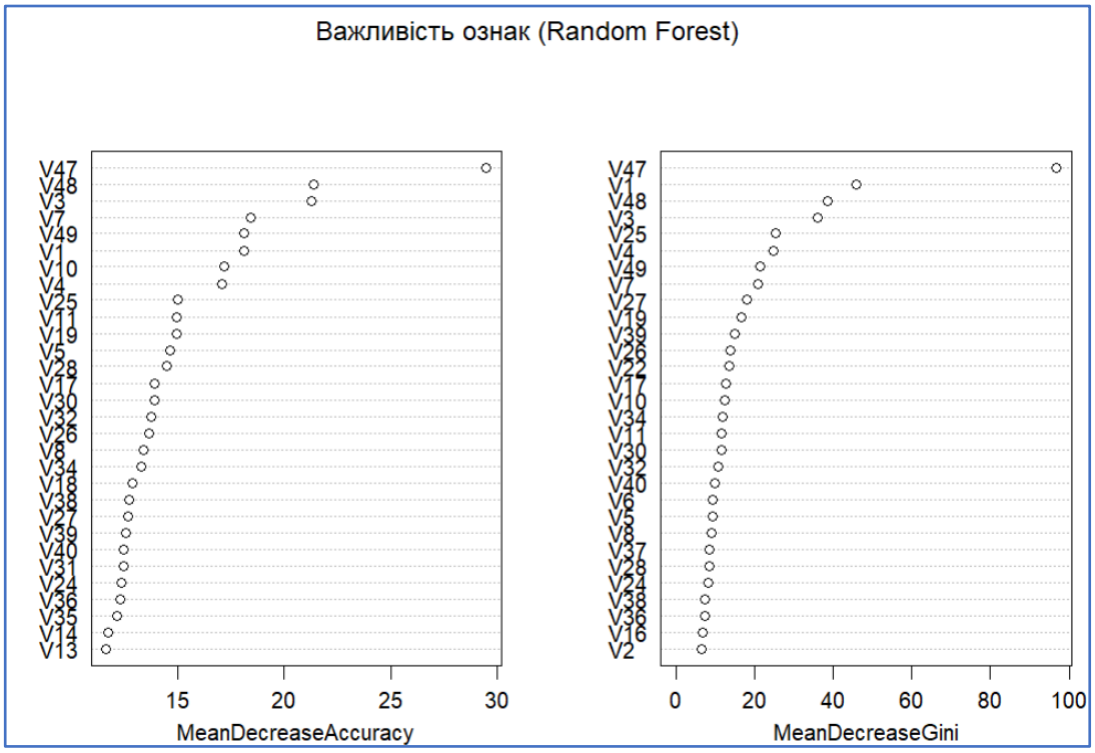

Рисунок 3.6 – Важливість ознак (другий набір даних)

XGBoost

XGBoost – один із найпотужніших алгоритмів бустингу, що має вбудовану підтримку навчання, чутливого до втрат. Для цього використовується параметр

scale_pos_weight, який встановлює вагу для позитивного класу в залежності від дисбалансу.

$$scale\_pos\_weight = \frac{N_{neg}}{N_{pos}} \tag{3.5}$$

де $scale\_pos\_weight$ – коефіцієнт, що масштабує вагу позитивного класу;

N_{neg} – кількість зразків негативного класу;

N_{pos} – кількість зразків позитивного класу.

Це означає, що під час обчислення градієнтів для оновлення дерев втрати від меншого класу зважуються сильніше.

У коді XGBoost реалізовано за допомогою бібліотеки *xgboost* з параметрами *objective = "binary:logistic"* і *eval_metric = "auc"* [24]. Значення scale_pos_weight обчислювалося автоматично на основі співвідношення кількості класів у тренувальній вибірці. Обчислено функцію втрат у XGBoost для задачі двокласової класифікації.

$$L = \sum_{i=1}^{N} w_{y_i} * l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{3.6}$$

де L – значення функції втрат для задачі двокласової класифікації;

N – загальна кількість зразків;

i – індекс поточного зразка;

w_{y_i} – ваговий коефіцієнт для класу, до якого належить зразок y_i;

y_i – справжнє значення класу;

$\hat{y}_i$ – прогнозоване значення класу;

K – кількість дерев у моделі;

f_k – k-те дерево моделі;

$\Omega(f_k)$ – регуляризаційний член для k-го дерева, який запобігає переобученню.

Для моделі XGBoost параметр scale_pos_weight було обчислено автоматично як відношення кількості негативних і позитивних прикладів. Це дало змогу компенсувати дисбаланс між класами. Модель навчалася із параметрами, такими як:

– objective = "binary:logistic";
– eval_metric = "auc";
– eta = 0.1, max_depth = 6, subsample = 0.8.

За результатами тестування для першого набору даних: AUC (XGBoost)=0,76.

Це свідчить про досить високу якість класифікації, хоча модель все ще допускає певну кількість хибних позитивних прогнозів.

Важливість ознак – діаграма показує, які ознаки найбільше впливають на рішення моделі. Серед них найвагомішими виявилися такі характеристики, як pH, розчинений кисень, твердість води тощо (див. рис. 3.7).

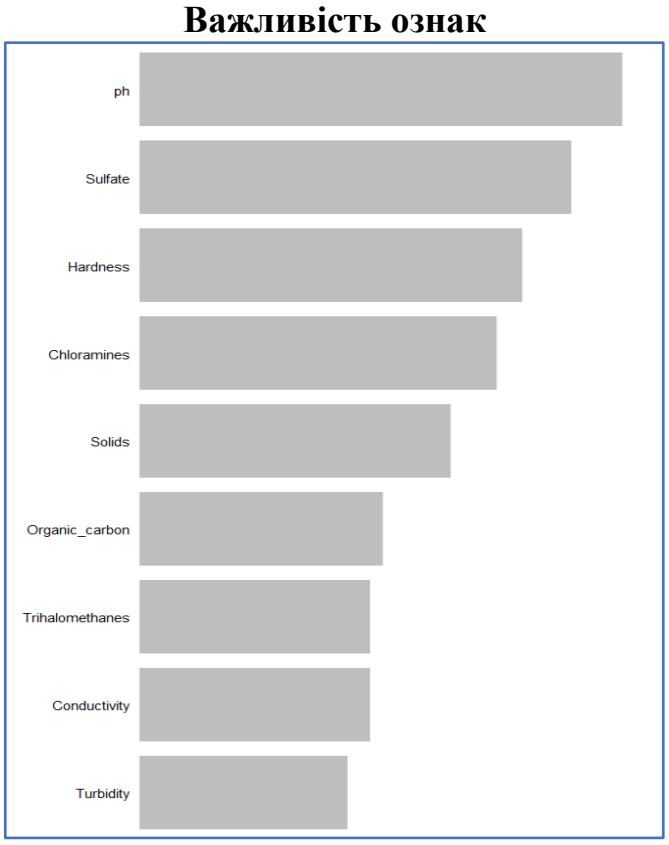

Рисунок 3.7 – Важливість ознак (перший набір даних)

За результатами тестування для другого набору даних: AUC (XGBoost)=0,919.

Отримане значення AUC свідчить про високу точність класифікації, що є найкращим результатом серед протестованих методів. Алгоритм XGBoost найкраще адаптується до складних закономірностей у даних і демонструє найвищу чутливість до позитивного класу.

Рис. 3.8 важливості ознак підтверджує домінування тих самих ключових факторів, що і в моделі Random Forest.

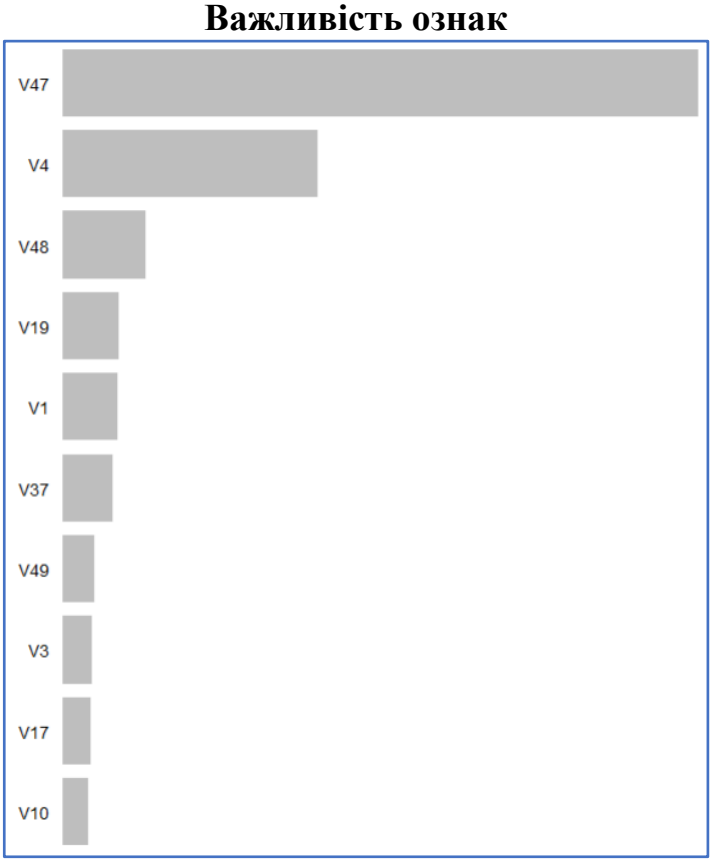

Рисунок 3.8 – Важливість ознак (другий набір даних)

Decision Tree з матрицею витрат

Окремим підходом до навчання, чутливого до втрат, є використання матриці витрат (cost matrix) у дереві рішень. У цьому випадку кожен тип помилки має

власну «вартість». У системі це реалізовано через параметр *parms = list(loss = cost_matrix)* у функції *rpart*() [25]. Було задано матрицю витрат.

$$Cost = \begin{bmatrix} o & C_{01} \\ C_{10} & o \end{bmatrix} \tag{3.7}$$

де $Cost$ – матриця витрат для дерева рішень;

C_{01} – вартість помилки при класифікації «0» як «1»;

C_{10} – вартість помилки при класифікації з класу «1» як «0»;

0 – вартість правильного класифікування відповідного класу.

У розробленій моделі значення $C_{10} = 8$ означає, що помилка пропуску позитивного класу є у вісім разів серйознішою. Це стимулює дерево обережніше приймати рішення щодо рідкісних випадків.

Після навчання модель будувала зрозумілу структуру рішень, яку можна візуалізувати за допомогою *rpart.plot*(), а ROC-крива показала суттєве підвищення чутливості моделі. Таким чином, метод дерев із матрицею витрат дозволяє інтерпретовано врахувати різну ціну помилок у задачах класифікації.

За результатами тестування для першого набору даних: AUC (Decision Tree): 0,61.

На рис. 3.9 представлено структуру дерева рішень, побудованого для класифікації з урахуванням матриці витрат. Кожен вузол містить умову розгалуження та колір, що відображає переважаючий клас і рівень чистоти вузла (зелені – позитивний клас, червоні – негативний, жовті – змішані). Модель поступово ділить простір ознак на підмножини, формуючи зрозумілу ієрархію правил для прийняття рішень. Завдяки встановленню ваги $C_{10} = 8$, дерево приділяє більше уваги правильному виявленню позитивних спостережень, навіть за рахунок зростання кількості помилкових тривог.

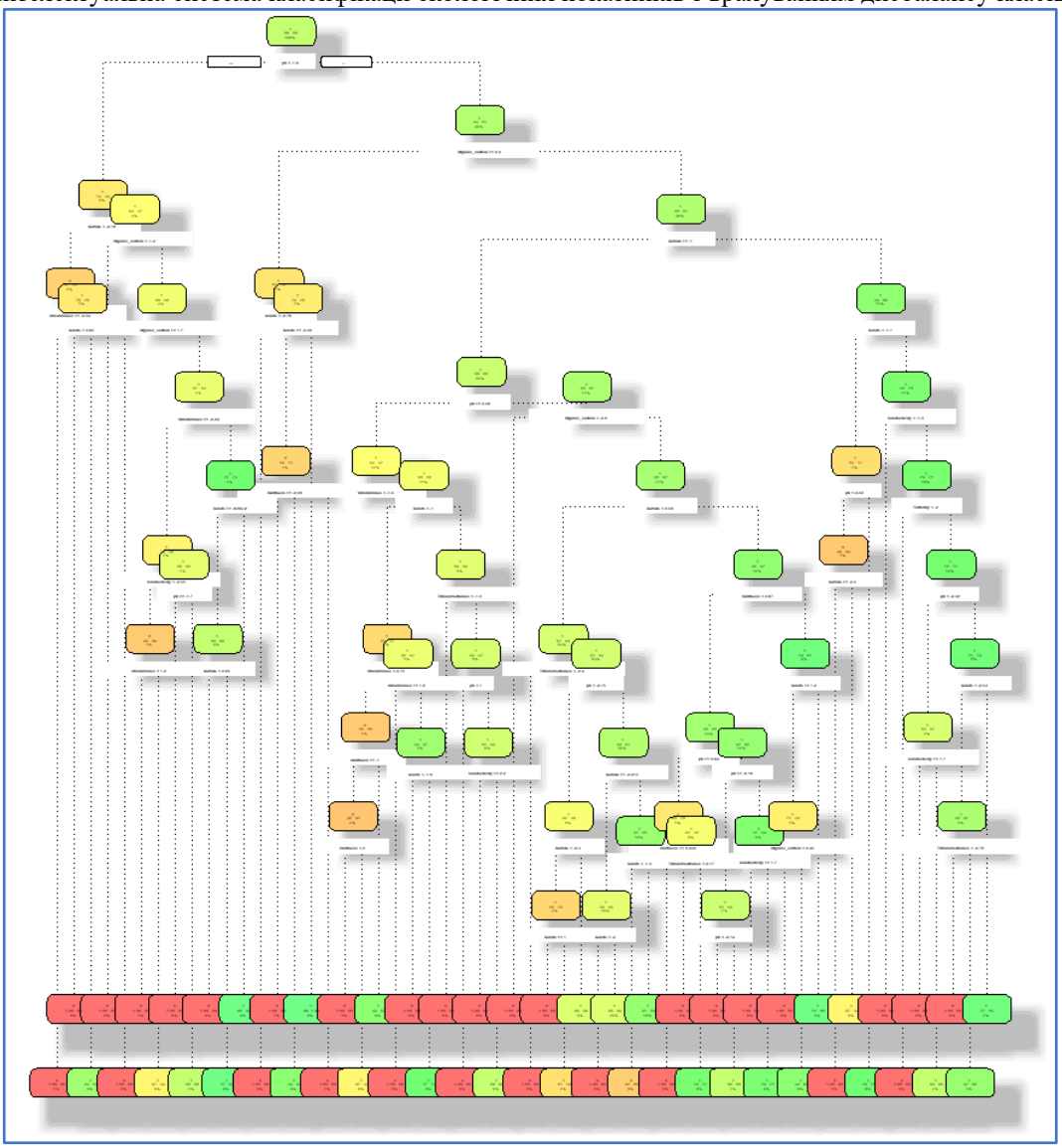

Рисунок 3.9 – Структура дерева рішень (перший набір даних)

За результатами тестування для другого набору даних: AUC (Decision Tree): 0,721.

Попри найнижчий показник серед протестованих моделей другого набору даних, цей підхід залишається цінним через інтерпретованість і можливість наочно оцінити логіку прийняття рішень.

На рис. 3.10 представлено структуру дерева рішень, побудованого для класифікації з урахуванням матриці витрат.

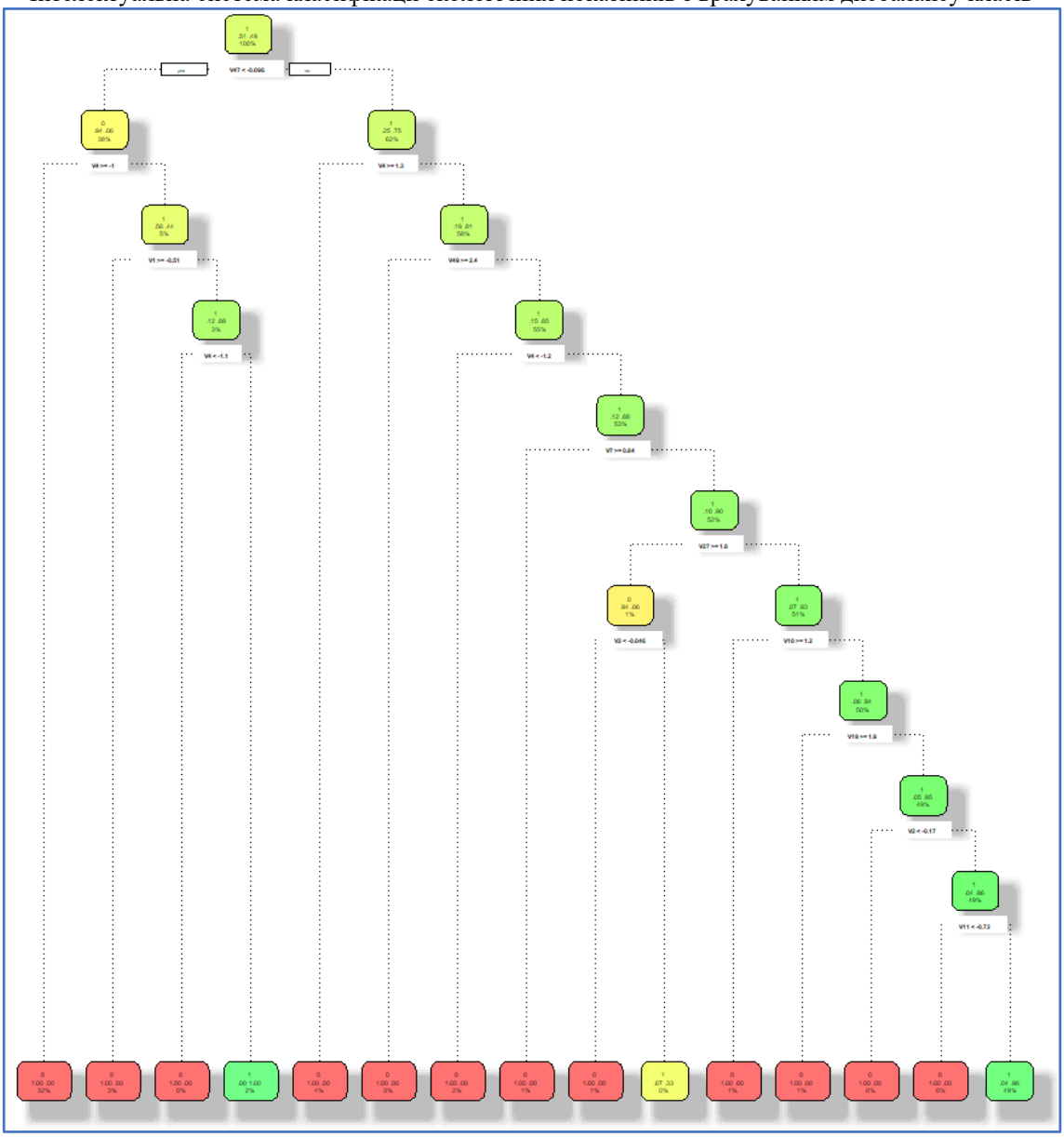

Рисунок 3.10 – Структура дерева рішень (другий набір даних)

Порівняння ROC-кривих моделей

Для порівняння двох наборів даних було побудовано ROC-криві навчених моделей [26].

На першому графіку (див. рис. 3.11) показано порівняння ROC-кривих для кількох моделей МН на першому наборі даних. Найкращі результати демонструють *Random Forest (AUC = 0.81)* і *XGBoost (AUC = 0.76)*, тоді як *Logistic Regression* має найгірший показник. Це свідчить, що на першому наборі даних нелінійні моделі краще відокремлюють класи, ніж лінійна регресія.

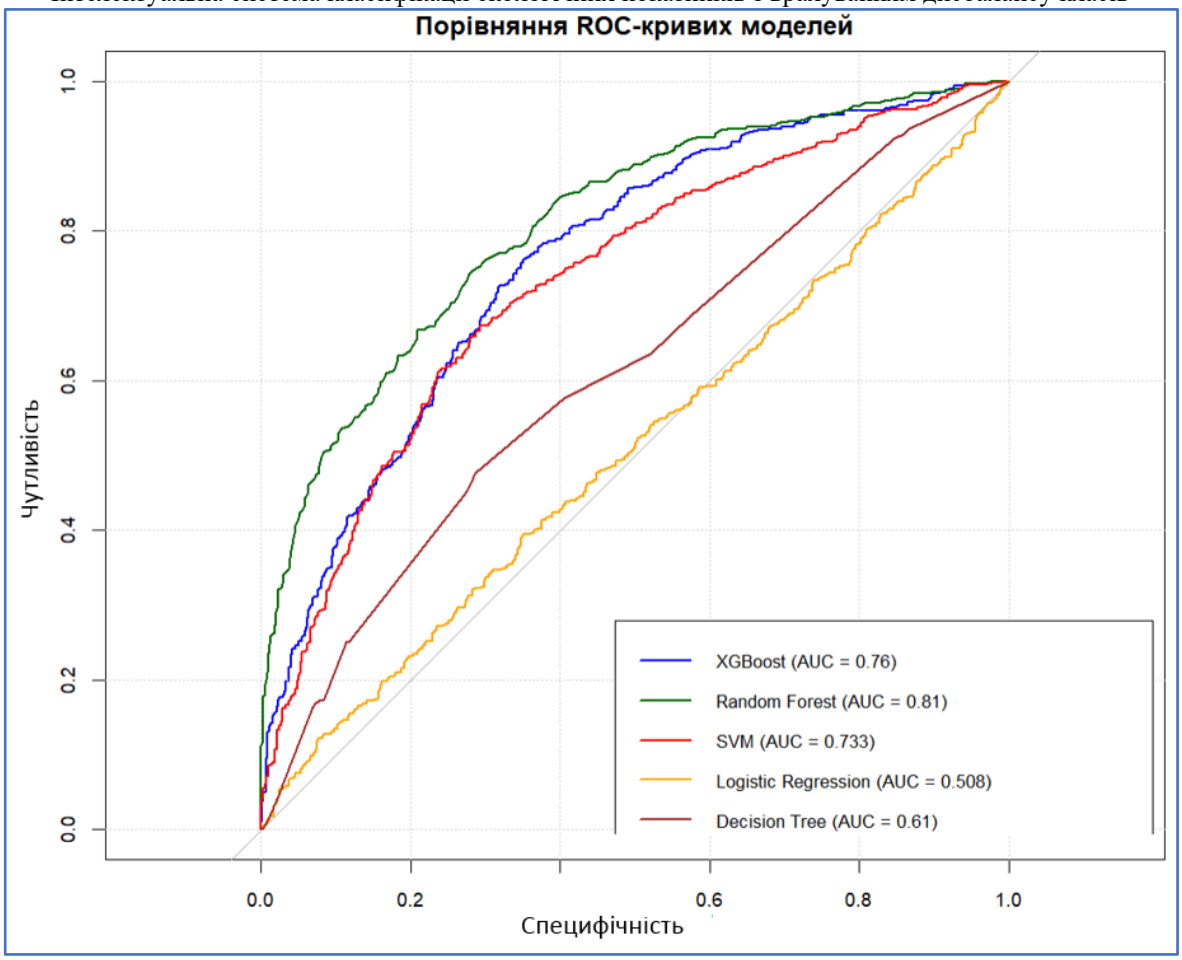

Рисунок 3.11 – Порівняння ROC-кривих моделей (перший набір даних)

На рис. 3.12, який показує результати для другого набору даних, усі моделі значно покращили свої результати (рис. 3.12). Зокрема, *XGBoost (AUC = 0,919)* і *Random Forest (AUC = 0,906)* знову мають найвищі показники, тоді як навіть *Logistic Regression (AUC = 0,777)* показала значне покращення. Це може свідчити про те, що другий набір даних є більш чистим, збалансованим або містить більш інформативні ознаки.

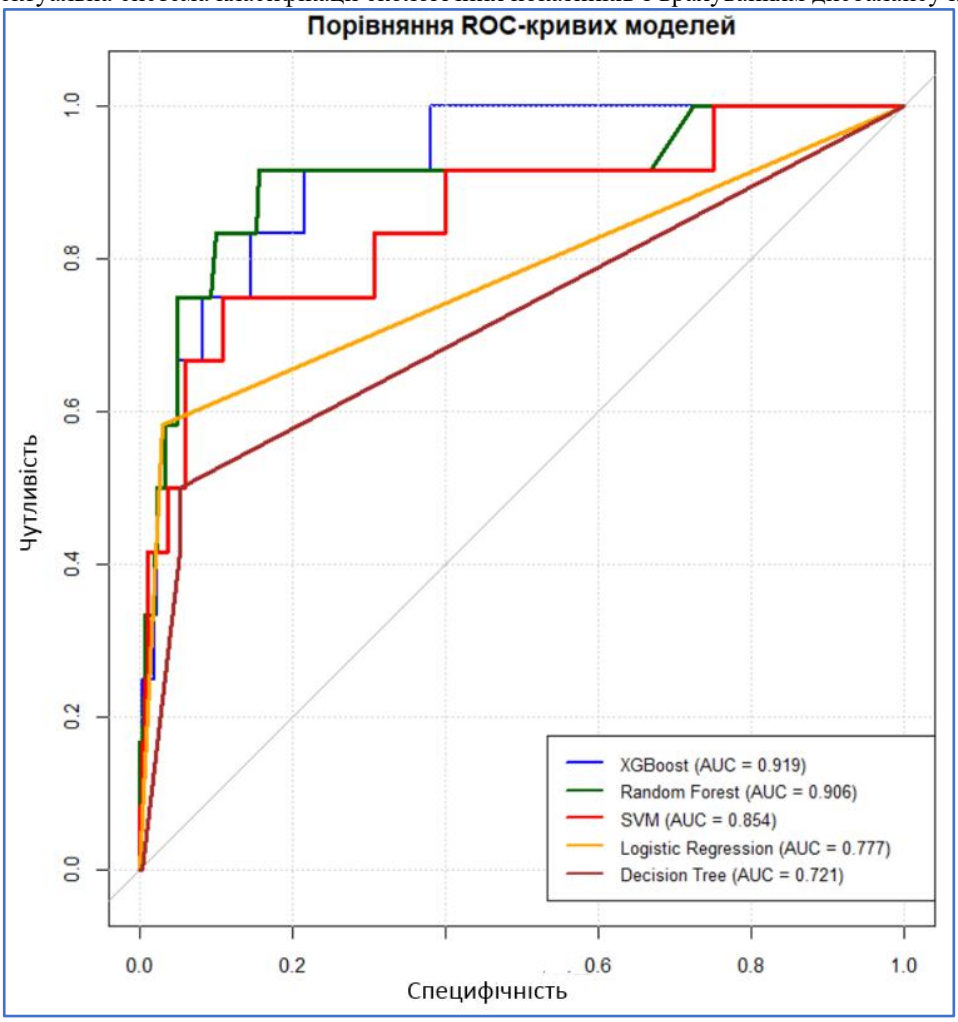

Рисунок 3.12 – Порівняння ROC-кривих моделей (другий набір даних)

Обидва графіки підтверджують стабільну перевагу методів XGBoost, Random Forest над іншими моделями.

3.3 Ансамблеві методи навчання

Ансамблеві методи навчання є одними з найефективніших підходів у задачах класифікації, зокрема у випадках суттєвого дисбалансу класів. Основна ідея ансамблевого навчання полягає в об’єднанні декількох базових моделей (класифікаторів) з метою підвищення загальної продуктивності, стійкості та узагальнювальної здатності моделі [27]. На відміну від окремих класифікаторів, ансамблі здатні зменшувати варіативність, упередженість або дисбаланс даних через комбінування різних підвибірок, ваг, моделей або дерев рішень.

У контексті задачі прогнозування Potability та Class для двох досліджуваних наборів даних найбільш доцільними є ансамблеві методи, що поєднують бустингові та бэггінгові підходи із стратегіями балансування класів, а саме: RUSBoost, SMOTEBoost, EasyEnsemble, Balanced Bagging та BalanceCascade. Ці алгоритми спеціально орієнтовані на роботу з дисбалансом, що робить їх придатними для реальних задач, де менший клас має критично важливе значення (наприклад, дефектні зразки, небезпечні показники, рідкісні події).

RUSBoost

Метод RUSBoost є поєднанням техніки бустингу зі стратегією випадкового зменшення вибірки більшого класу (Random UnderSampling, RUS), що дозволяє ефективно боротися з дисбалансом у даних [28]. Стандартний бустинг має схильність концентруватися на складних прикладах, проте при сильному дисбалансі модель переважно тренується на прикладах більшості, ігноруючи менш представлений клас.

Алгоритм RUSBoost вирішує цю проблему шляхом регулярного видалення випадкової частини даних більшого класу перед кожною ітерацією навчання слабкого класифікатора. Це гарантує рівні пропорції класів у кожному навчальному наборі та знижує систематичну упередженість моделі. Було обчислено ансамблевий класифікатор.

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \quad (3.8)$$

де $H(x)$ – прогноз ансамблевого класифікатора;

x – вхідний об’єкт;

T – кількість базових класифікаторів у ансамблі;

$h_t(x)$ – t-й базовий класифікатор;

α_t – вага t-го класифікатора у ансамблі;

$sign(\cdot)$ – функція знака, яка повертає +1 або −1 залежно від сумарного результату.

У дослідженні метод RUSBoost було реалізовано за допомогою побудови ансамблю з кількох моделей, кожна з яких тренувалась на новому undersampled наборі даних. На кожній ітерації виконувалося випадкове зменшення більшого класу до обсягу меншого, що дозволяло збалансувати початковий дисбаланс. Далі навчався слабкий класифікатор, який у цьому дослідженні представляв собою модифікований варіант бустингової моделі.

Для першого набору даних було використано 200 ітерацій, що забезпечило достатню глибину бустингу та можливість краще адаптуватися до складних прикладів.

Для стабілізації результатів моделювання та зменшення випадкових коливань було побудовано ансамбль із 10 незалежних моделей, прогнози яких усереднювалися. Це дозволило суттєво знизити дисперсію отриманих оцінок і підвищити узагальнювальну здатність моделі. Кожна модель працювала зі своїм варіантом undersampled підвибірки, що розширювало охоплення прикладів більшості класу.

За результатами тестування для першого набору даних: AUC (RUSBoost) =0,747.

Для візуалізації результату було побудовано розподіл прогнозованих ймовірностей. Графік показує, що прогнозовані ймовірності мають середнє перекриття між класами (див. рис. 3.13).

Для другого набору кількість ітерацій було зменшено до 100, що було достатнім через іншу структуру даних та меншу необхідність у глибокому бустуванні.

За результатами тестування для другого набору даних: AUC (RUSBoost) =0,887.

На рис. 3.14 показано розподіл прогнозованих ймовірностей для ансамблевої моделі RUSBoost. Видно, що більшість спостережень класу 0 мають низькі ймовірності (<0.25), а приклади класу 1 розподілені ближче до області високих значень (0.75–1.0).

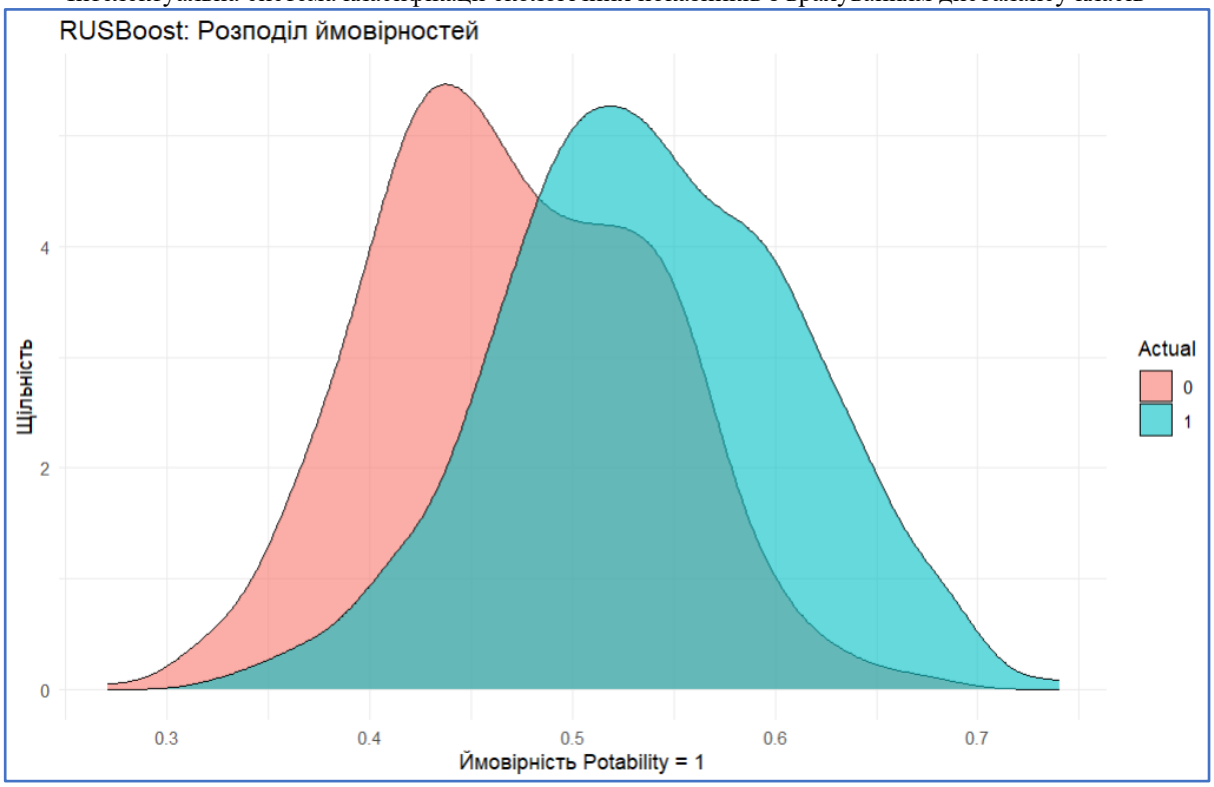

Рисунок 3.13 – Розподіл ймовірностей (перший набір даних)

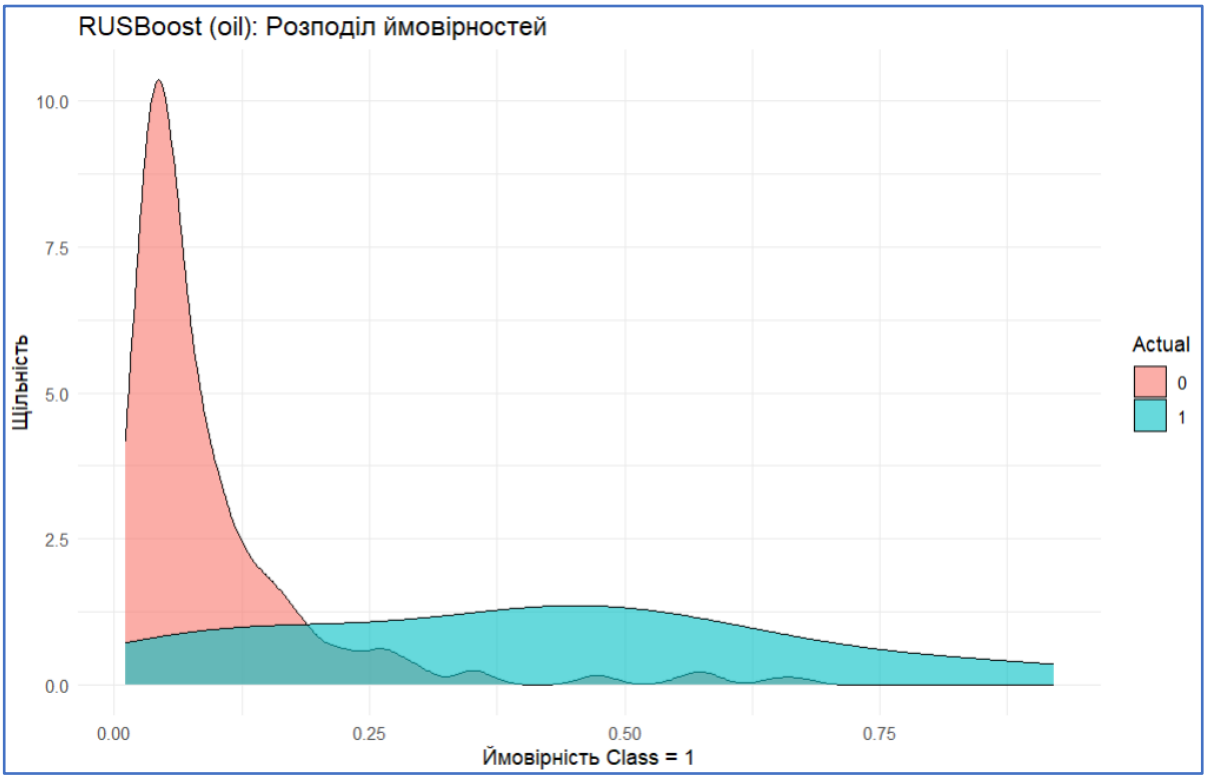

Рисунок 3.14 – Розподіл ймовірностей (другий набір даних)

SMOTEBoost

SMOTEBoost є модифікацією класичного AdaBoost, яка включає синтетичне збільшення меншого класу за допомогою алгоритму SMOTE [29, 32]. Традиційні методи oversampling часто дублюють існуючі приклади, що не додає нової інформації та може спричиняти перенавчання. SMOTE генерує нові об'єкти меншого класу, інтерполюючи їх між наявними точками, що дозволяє створювати більш «плавне» та природне розширення вибірки. У SMOTEBoost цей процес виконується на кожній ітерації бустингу, тому модель поступово покращує представлення меншого класу.

$$x_{new} = x_i + \lambda(x_{nn} - x_i), \lambda \in [0,1] \quad (3.9)$$

де x_{new} − новий синтетичний приклад;

x_i − точка меншого класу;

x_{nn} − один із її k-ближчих сусідів;

λ – випадковий коефіцієнт з інтервалу [0,1].

Було використано SMOTE із параметром кількості сусідів K = 5, що є типовим значенням для генерації якісних синтетичних точок у багатовимірному просторі. Додатковий параметр dup_size = 2 визначав рівень розширення меншого класу на кожній ітерації бустингу. Після застосування SMOTE отримана збалансована вибірка використовувалася як навчальні дані для побудови слабкого класифікатора у структурі AdaBoost.

Було проведено 100 бустингових ітерацій, що дозволило моделі послідовно зменшувати похибки класифікації та адаптувати свою увагу до складних прикладів. На кожному кроці створювалися нові синтетичні точки, завдяки чому структура меншого класу розширювалася плавно та різноманітно.

За результатами тестування для другого набору даних: AUC (SMOTEBoost)=0,716.

Після застосування методу кількість прикладів класу 1 зрівнялася з класом 0, що зробило навчальну вибірку збалансованою (див. рис. 3.15). Це дозволило моделі краще розпізнавати рідкісний клас і зменшити упередженість у бік більш поширеного класу. Значення AUC свідчить, що SMOTEBoost підвищує якість розпізнавання позитивних випадків порівняно з оригінальними даними.

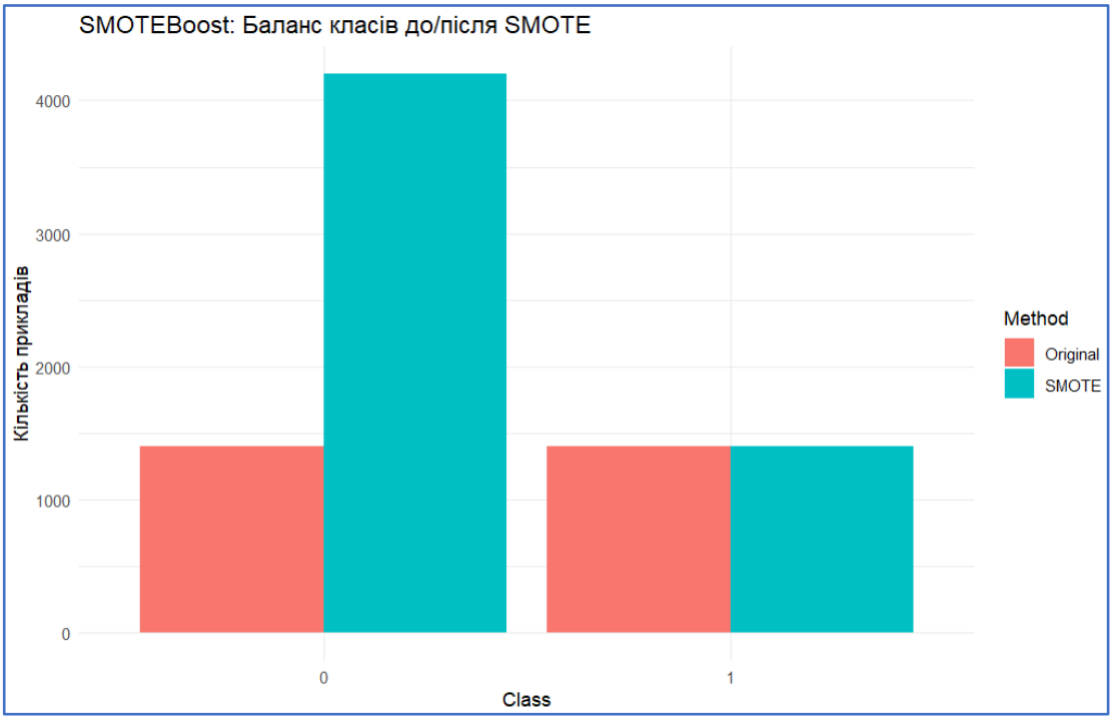

Рисунок 3.15 – Порівняння балансу класів (перший набір даних)

За результатами тестування для другого набору даних: AUC (SMOTEBoost)=0,899.

Застосування SMOTEBoost суттєво збільшило кількість прикладів позитивного класу, що допомогло моделі краще навчитися виявляти рідкісні випадки (див. рис. 3.16).

Результати підтвердили переваги SMOTEBoost для складніших задач класифікації. Отримані результати демонструють, що алгоритм краще адаптується до наборів даних із більш розвиненою структурою ознак.

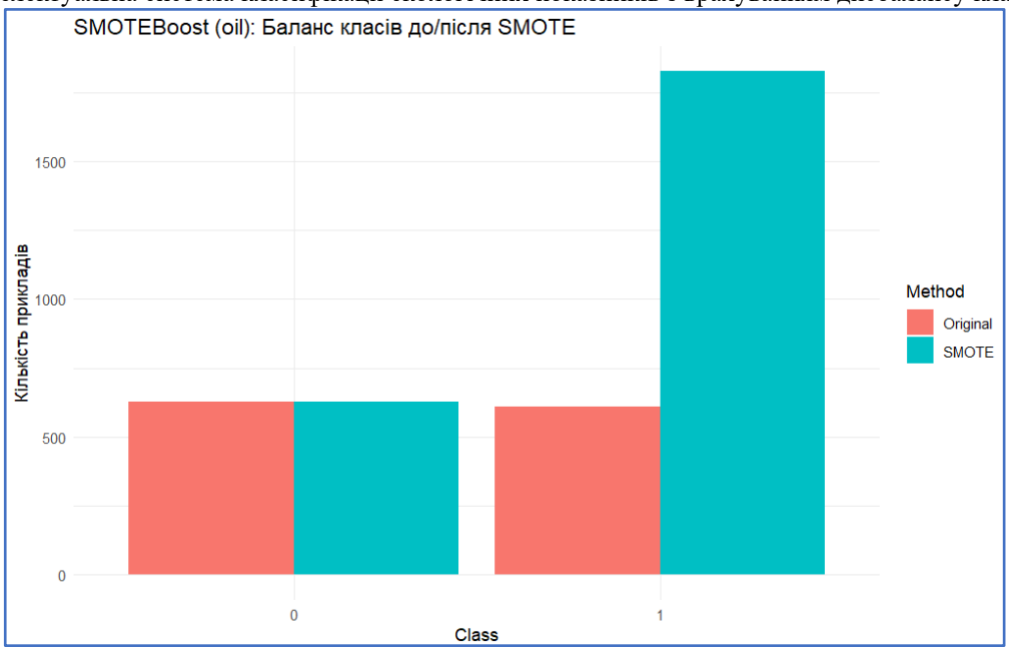

Рисунок 3.16 – Порівняння балансу класів (другий набір даних)

EasyEnsemble

EasyEnsemble є ансамблевим методом, що використовує кілька незалежних випадкових undersampling-підвибірок більшого класу для побудови окремих моделей бустингу. На відміну від простого undersampling, який викидає частину даних назавжди, EasyEnsemble формує багато підвибірок, які разом охоплюють більшу частину інформації. Це дозволяє зберегти корисні приклади більшого класу без втрати інформації. Кожна з підвибірок має збалансовану структуру, що забезпечує справедливе навчання моделі на кожному етапі [30].

На кожній з них навчається окремий boosting-класифікатор, який зосереджується на різних частинах простору даних. Об’єднання прогнозів від кількох моделей дозволяє зменшити дисперсію та підвищити узагальнювальну здатність ансамблю. Обчислено підсумковий прогноз.

$$P(x) = \frac{1}{M}\sum_{i=1}^{M} P_i(x) \tag{3.10}$$

де $P(x)$ – прогноз ймовірності для об’єкта x від ансамблю;

M – кількість підмоделей;

$P_i(x)$ – прогноз ймовірності об’єкта x від i-ої підмоделі.

EasyEnsemble добре працює на великих наборах даних, де повний балансування є ресурсомістким. Метод також є ефективним при високому дисбалансі, оскільки забезпечує багаторазове представлення меншого класу.

У роботі було створено 10 незалежних підвибірок, кожна з яких містила однакову кількість прикладів обох класів. Для кожної підвибірки навчався boosting-класифікатор зі 50 ітерацій, що забезпечувало достатню глибину навчання без надмірного збільшення часу моделювання. Усі моделі тренувалися окремо, що дозволяло кожній підмоделі вловити унікальні закономірності у даних. Після завершення навчання кожний класифікатор генерував ймовірнісний прогноз. Потім усі прогнози усереднювалися, щоб отримати фінальну оцінку для кожного об’єкта.

За результатами тестування для першого набору даних: AUC (EasyEnsemble)=0,743.

На рис. 3.17 видно, що всі десять моделей EasyEnsemble формують подібні розподіли прогнозованих ймовірностей, зосереджені переважно в діапазоні 0.35–0.65, що свідчить про відсутність чітко вираженого розділення між класами. Чорна лінія (усереднений прогноз) вирівнює незначні коливання між моделями, але повторює їхній дзвоноподібний профіль.

Модель має середню якість класифікації та помірну здатність відрізняти позитивний клас від негативного. Незважаючи на стабільність між моделями, ансамбль не отримує сильної користі від undersampling, що може бути наслідком втрати корисної інформації при зменшенні вибірки. Результат демонструє невисокий рівень продуктивності для цього набору даних.

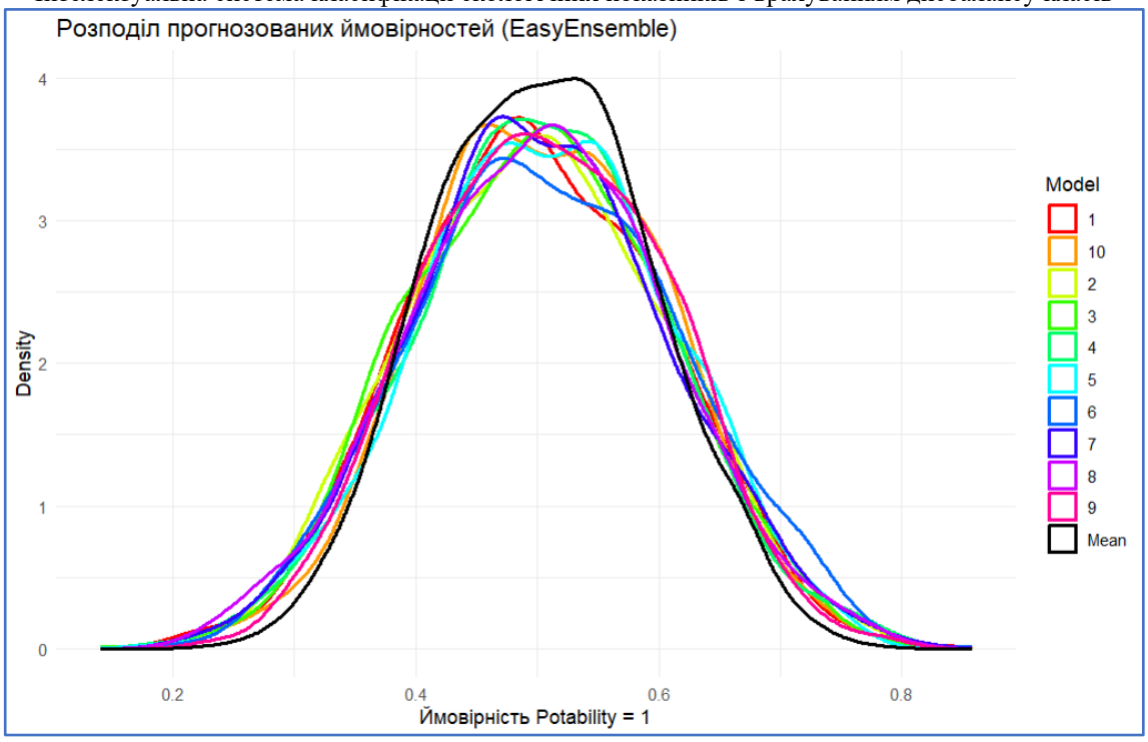

Рисунок 3.17 – Розподіл прогнозованих ймовірностей (перший набір даних)

За результатами тестування для першого набору даних: AUC (EasyEnsemble)=0,883.

Метод EasyEnsemble показав високу стабільність прогнозів. На рис. 3.18 видно, що всі 10 підмоделей формують дуже схожі розподіли ймовірностей, зі зміщенням у бік малих значень, що є типовим для задачі з переважанням негативного класу. Чорна лінія, яка позначає усереднений прогноз, згладжує коливання між моделями і добре відображає спільну тенденцію ансамблю. Метод демонструє помірну варіативність між окремими моделями, але не критичну – їхні криві розташовані близько одна до одної, що свідчить про надійність.

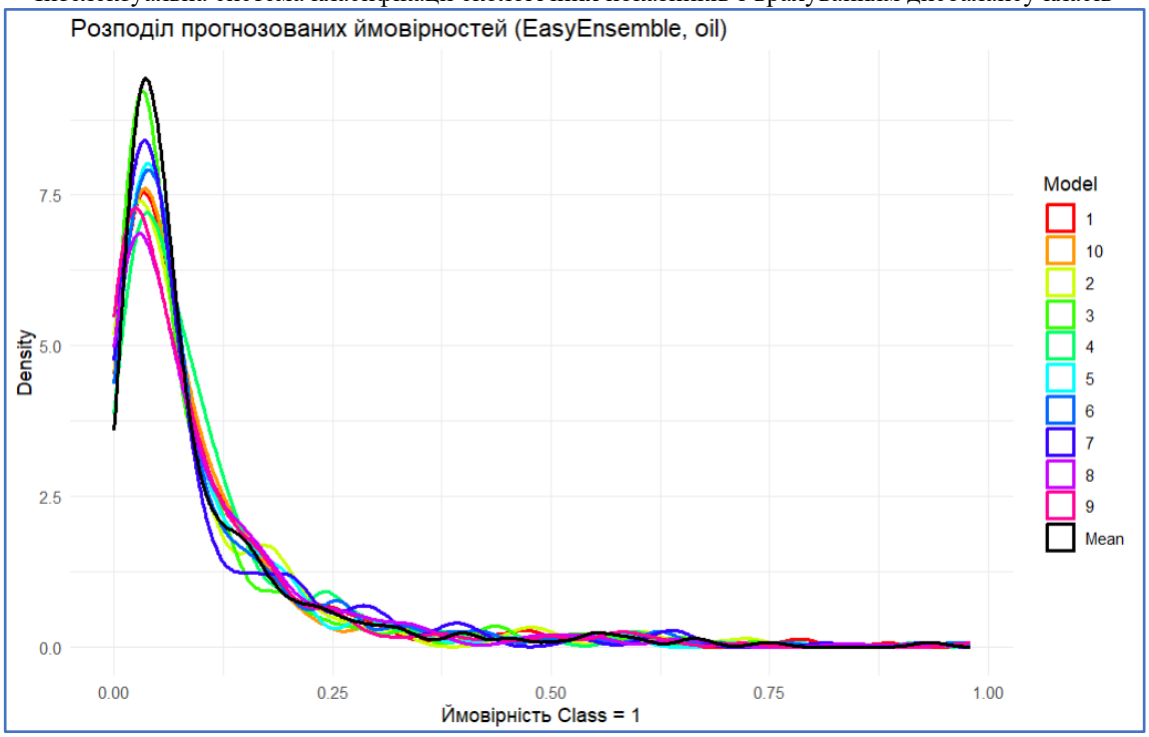

Рисунок 3.18 – Розподіл прогнозованих ймовірностей (другий набір даних)

BalancedBagging

BalancedBagging є модифікацією класичного бэггінгу, в якій кожна підвибірка формується так, щоб містити збалансоване представлення класів [31]. Бэггінг створює кілька вибірок із заміною та тренує окремі моделі, після чого усереднює їхні прогнози. BalancedBagging додає до цього процесу балансування класів шляхом undersampling або комбінованого sampling. Завдяки цьому кожна окрема модель отримує рівні умови для навчання та не зміщена в бік більшого класу. Обчислено підсумковий прогноз.

$$P(x) = \frac{1}{B}\sum_{b=1}^{B} P_b(x) \tag{3.11}$$

де $P(x)$ – прогноз ймовірності для об’єкта x від ансамблю дерев;

B – кількість дерев у ансамблі;

$P_b(x)$ – прогноз ймовірності об’єкта x від b-го дерева.

Для першого набору даних було створено 100 збалансованих підвибірок, кожна з яких використовувалася для тренування окремої моделі Random Forest із 50 дерев. Це дозволило отримати дуже стабільний ансамбль з великим охопленням прикладів.

На другому наборі даних балансування було виконано через ovun.sample(), що забезпечувало якісний undersampling та формування збалансованих навчальних підмножин. Після балансування також тренувалися моделі Random Forest із 50 дерев.

Кожна модель генерувала ймовірнісний прогноз, який надалі усереднювався між усіма моделями ансамблю. Balanced Bagging виявився ефективним через здатність охоплювати різні підмножини більшого класу та бачити усю структуру менших класів.

За результатами тестування для першого набору даних: AUC (BalancedBagging)=0,812.

На гістограмі (рис. 3.19) видно, що розподіли ймовірностей для обох класів значно перекриваються, що свідчить про помірну складність задачі для моделі. Загалом Balanced Bagging у цьому випадку забезпечує стабільні прогнози, але інформаційна структура даних не дозволяє досягти дуже високої роздільної здатності між класами.

За результатами тестування для другого набору даних: AUC (BalancedBagging)=0,921.

Гістограма на рис. 3.20 показує, що переважна більшість об’єктів класу 0 мають дуже низькі ймовірності належності до класу 1 – значення концентруються близько до нуля. Для класу 1 розподіл ширший: частина об’єктів також отримує низькі ймовірності, але присутні й спостереження з середніми та високими значеннями, включно з близькими до 1.

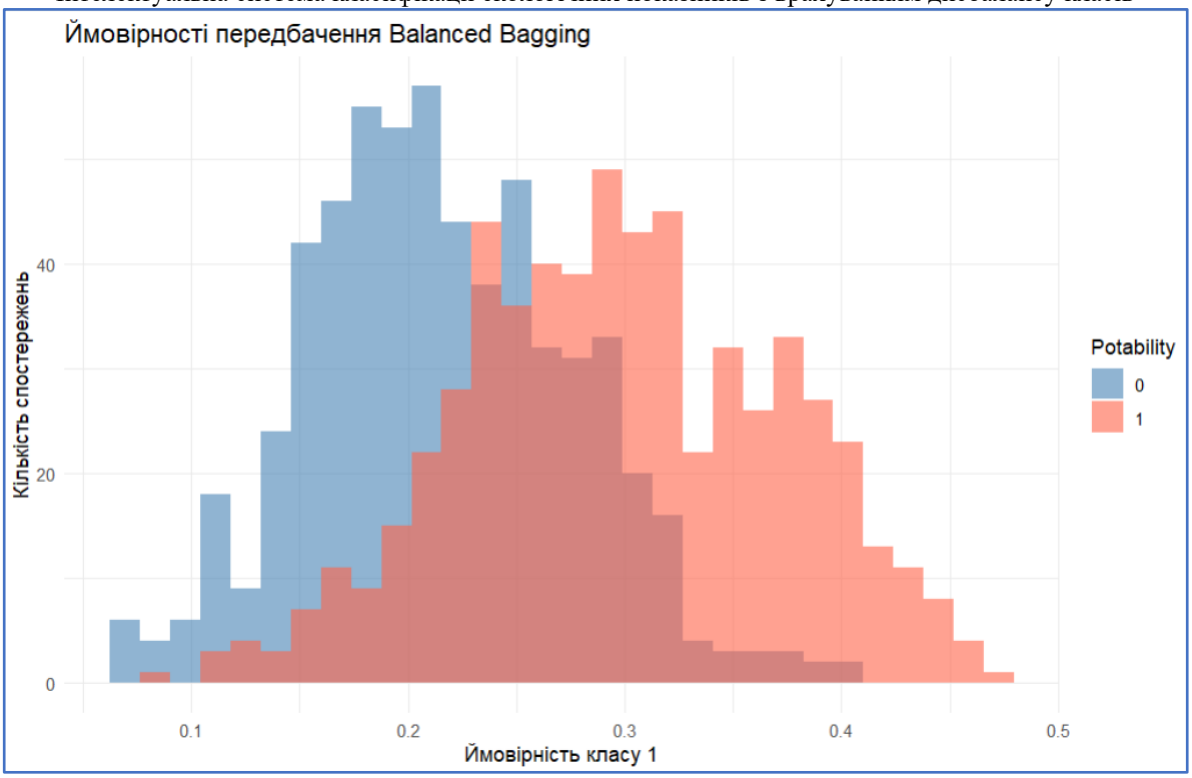

Рисунок 3.19 – Ймовірності передбачення (перший набір даних)

Графік демонструє здатність Balanced Bagging давати чіткі прогнози для основного класу та інколи високі ймовірності для дійсних позитивних прикладів.

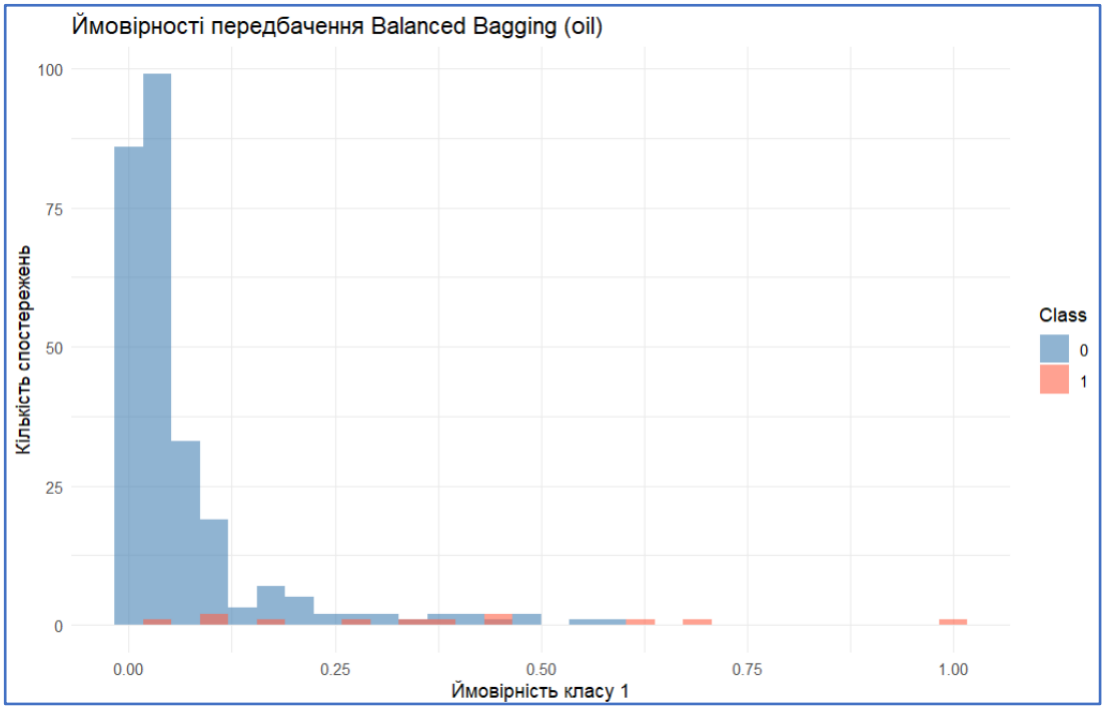

Рисунок 3.20 – Ймовірності передбачення (другий набір даних)

BalanceCascade

BalanceCascade є ансамблевим алгоритмом undersampling, який формує послідовність моделей, кожна з яких навчається на ускладнених даних. На відміну від випадкового undersampling, BalanceCascade не лише формує збалансовану підвибірку, але й видаляє правильно класифіковані приклади більшого класу після кожної ітерації. Це дозволяє наступним моделям концентруватися на більш «проблемних» та важких для класифікації прикладах [32]. Такий механізм поєднує ідею бустингу з ідеєю поступового ускладнення навчального набору.

У результаті кожна нова модель вдосконалює попередню, покращуючи якість класифікації у важких регіонах простору ознак. Каскадна структура дозволяє уникнути дублювання легко класифікованих прикладів, що зменшує обсяг даних без втрати важливої інформації. Балансування здійснюється на кожному етапі, що забезпечує стабільні умови для тренування моделі.

$$H(x) = \frac{1}{T}\sum_{t=1}^{T} f_t(x) \tag{3.12}$$

де $H(x)$ – прогноз ансамблю BalanceCascade для об’єкта x;

T – кількість моделей у каскаді;

$f_t(x)$ – прогноз t-ї моделі у каскаді.

Було виконано 5 каскадних ітерацій, на кожній з яких формувалася нова збалансована підвибірка. Для навчання використовувалися моделі Random Forest із 50 дерев, які забезпечували стабільність та здатність уловлювати складні нелінійні взаємозв’язки. Перший етап навчання на початковому збалансованому наборі, після чого всі правильно класифіковані приклади більшого класу видалялися. Процес повторювався, що дозволяло моделі фокусуватися на найважчих випадках. Після завершення каскаду прогнози окремих моделей агрегувалися для формування фінального прогнозу.

За результатами тестування для першого набору даних: AUC (BalanceCascade)=0,8.

Розподіли прогнозованих імовірностей розділені: для класу 0 більшість імовірностей нижче 0.5, а для класу 1 – вище 0.5. Медіани двох розподілів значно відрізняються, що свідчить про кращу здатність моделі відрізняти класи. Кількість перекриттів (чорних точок навколо порогу 0.5) зменшилася, тобто модель рідше плутає класи (див. рис. 3.21).

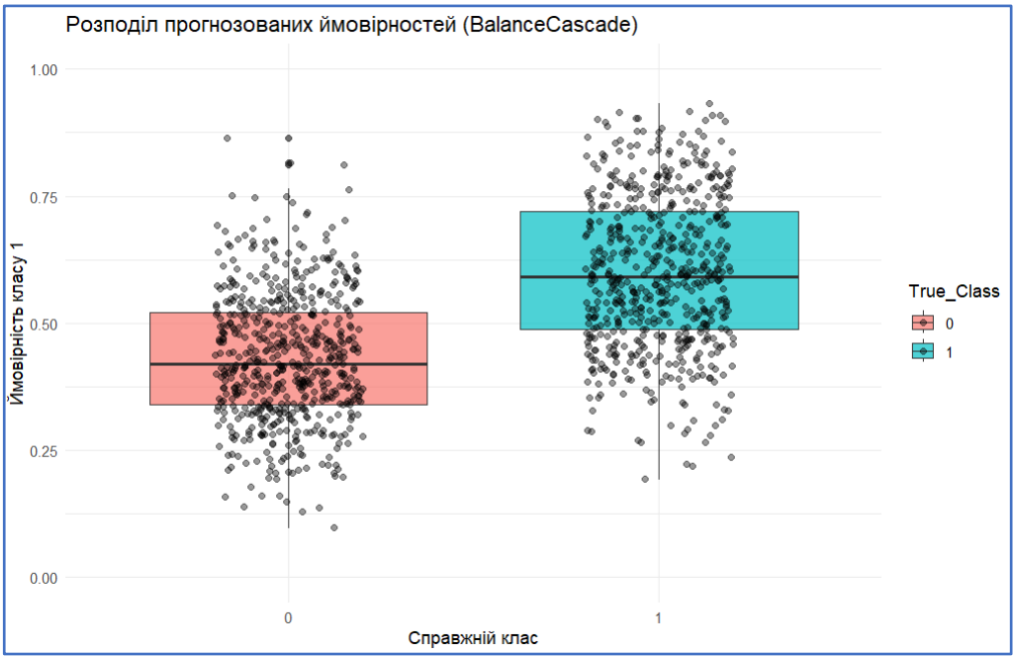

Рисунок 3.21 – Розподіл прогнозованих ймовірностей (перший набір даних)

За результатами тестування для другого набору даних: AUC (BalanceCascade)=0,94.

Графік на рис. 3.22 показує чітке розділення прогнозованих імовірностей між класами: для справжнього класу 0 модель здебільшого присвоює дуже низькі значення, близькі до нуля, що свідчить про високу точність у визначенні негативних прикладів. Попри певну варіативність, позитивний клас має чітко вищі значення, ніж негативний, і між групами практично немає серйозного перекриття.

Модель добре відокремлює класи, особливо ефективно розпізнаючи негативні приклади.

Рисунок 3.22 – Розподіл прогнозованих ймовірностей (другий набір даних)

Порівняння ROC-кривих моделей

На рис. 3.23 представлено результати навчання ансамблевих методів на першому наборі даних, де видно, що моделі демонструють помірну якість класифікації. Найкращий результат показує *BalancedBagging* з *AUC=0.805*, що свідчить про його здатність ефективно працювати з дисбалансом класів за рахунок поєднання бэггінгу та вибірки. Дуже близький показник має *BalanceCascade AUC=0.800*, який також формує збалансовані підвибірки на кожному етапі. Методи *RUSBoost* та *EasyEnsemble* демонструють середню продуктивність *AUC~0.74*, що є типовим для моделей, які використовують випадкове зменшення вибірки або одноразове ансамблювання. Найнижчий результат у *SMOTEBoost AUC=0.716*, імовірно через те, що синтетичні дані не покращили узагальнення моделі в умовах високої варіативності ознак.

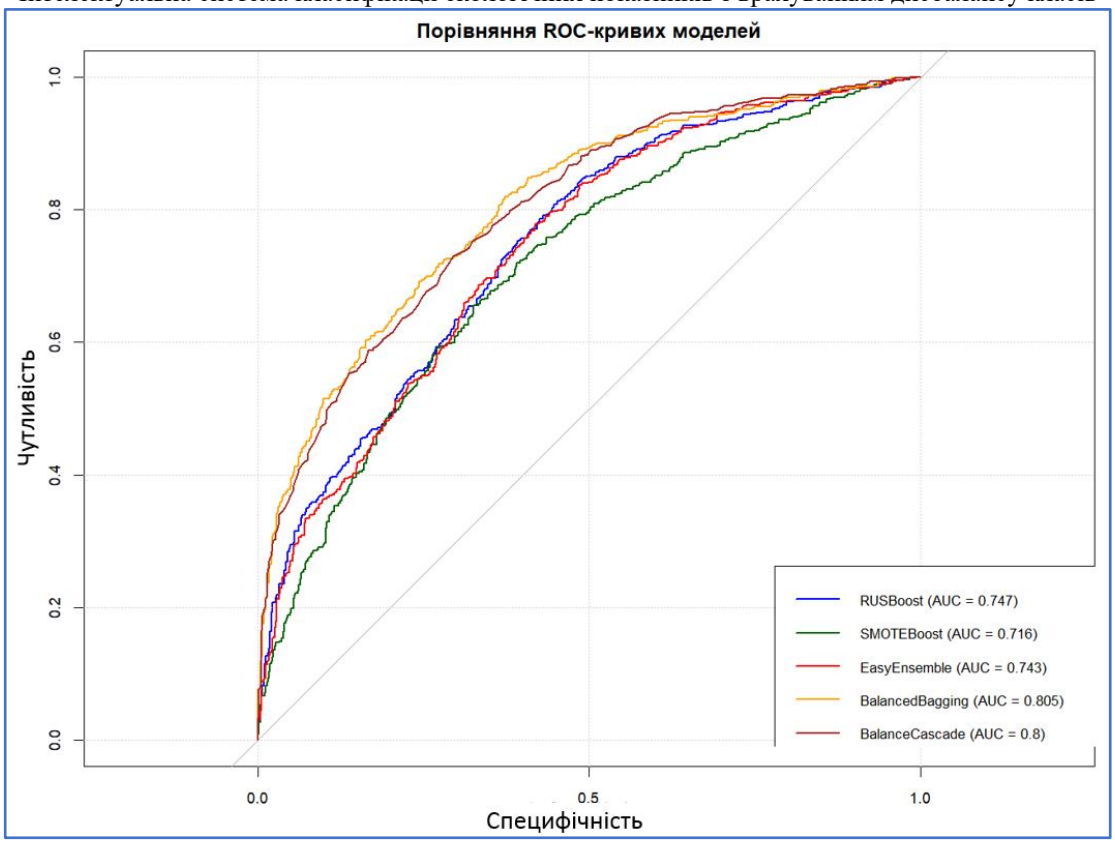

Рисунок 3.23 – Порівняння ROC-кривих моделей (перший набір даних)

На рис. 3.24 представлено результати на другому наборі даних, де загальний рівень AUC значно вищий. Тут усі моделі працюють краще, що свідчить про більшу стабільність даних або кращу роздільність класів.

Найвищі результати дає *BalanceCascade AUC=0.94* та *BalancedBagging AUC=0.921*, що знову підтверджує ефективність методів, які будують послідовно збалансовані підвибірки. *SMOTEBoost* показує вже суттєво кращий результат *AUC=0.899*, що означає, що у цьому наборі даних синтетичні приклади краще вписуються в структуру простору ознак. *RUSBoost* та *EasyEnsemble* мають трохи нижчі значення, але все одно демонструють хорошу стабільність.

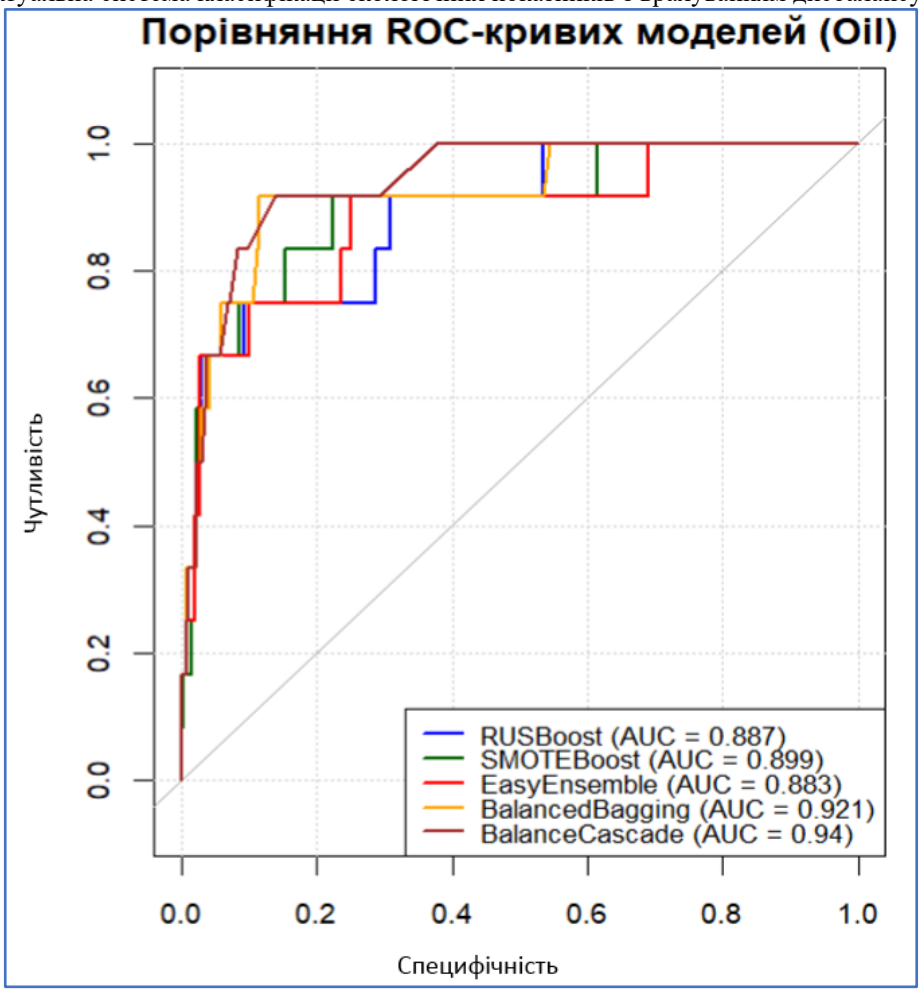

Рисунок 3.24 – Порівняння ROC-кривих моделей (другий набір даних)

Отже, у першому наборі даних найефективнішими виявилися *BalancedBagging* і *BalanceCascade*, тоді як *SMOTEBoost* показав найнижчу продуктивність. У другому наборі даних усі моделі значно покращилися, причому методи *BalancedBagging* і *BalanceCascade* знову стали лідерами.

Результати свідчать про те, що вибір методу залежить від структури конкретних даних, а каскадні та бэггінгові підходи виявляються найстійкішими.

3.4 Методи оцінювання моделей

У задачах МН правильна оцінка ефективності моделей є критично важливою для прийняття обґрунтованих рішень щодо їхнього застосування. Особливо складним є оцінювання моделей на наборах даних із дисбалансом класів, коли один

клас значно переважає над іншими. У таких випадках традиційна метрика загальної точності *accuracy* може давати оманливі результати, оскільки модель може демонструвати високу точність, просто правильно класифікуючи домінуючий клас, при цьому ігноруючи рідкісні, але важливі класи. Тому виникає необхідність застосовувати спеціалізовані метрики, які дозволяють оцінювати ефективність моделі більш комплексно та враховувати співвідношення між правильними та хибними прогнозами для кожного класу [33].

Однією з таких метрик є **Precision**, або точність, яка характеризує здатність моделі правильно визначати позитивні випадки серед усіх прогнозованих позитивних [34]. Вона розраховується як відношення кількості істинно позитивних прогнозів (TP) до суми істинно позитивних та хибно позитивних прогнозів (FP).

$$\text{Precision} = \frac{TP}{TP+FP} \tag{3.13}$$

де TP – кількість істинно позитивних прогнозів;

FP – кількість хибно позитивних прогнозів.

Precision дозволяє оцінити, наскільки модель надійна у виділенні позитивного класу та не робить зайвих хибних прогнозів.

Ще однією важливою метрикою є **Recall**, або повнота, яка визначає здатність моделі виявляти всі наявні позитивні випадки в наборі даних [35]. Recall обчислюється як відношення TP до суми TP та хибно негативних прогнозів (FN).

$$\text{Recall} = \frac{TP}{TP+FN} \tag{3.14}$$

де TP – кількість істинно позитивних прогнозів;

FN – кількість хибно негативних прогнозів.

Ця метрика дозволяє зрозуміти, наскільки модель не пропускає важливі позитивні об’єкти, що особливо важливо при роботі з рідкісними класами.

Для узагальненої оцінки точності та повноти часто використовується **F1-score**, яка являє собою гармонійне середнє Precision та Recall [35, 36].

$$F1 = \frac{2*Precision*Recall}{Precision+Recall} \quad (3.15)$$

де $Precision$ – точність;

$Recall$ – повнота.

F1-score є корисною метрикою у випадках дисбалансу класів, оскільки вона зменшує вплив надмірної точності на шкалі загальної ефективності моделі та відображає баланс між виявленням позитивних випадків і уникненням хибних сигналів.

Для оцінки здатності моделі правильно визначати негативний клас застосовується метрика **Specificity**, яка розраховується як відношення істинно негативних прогнозів (TN) до суми TN та FP [37].

$$\text{Specificity} = \frac{TN}{TN+FP} \quad (3.16)$$

де TN – кількість істинно негативних прогнозів;

FP – кількість хибно позитивних прогнозів.

Specificity дозволяє оцінити, наскільки модель надійна у визначенні негативного класу та не помиляється у класифікації негативних об’єктів як позитивних.

Ще однією метрикою, що враховує обидва класи одночасно, є **Balanced Accuracy**, яка визначається як середнє між Recall та Specificity [38].

$$\text{Balanced Accuracy} = \frac{Recall+Specificity}{2} \quad (3.17)$$

де $Recall$ – повнота;

$Specificity$ – специфічність.

Balanced Accuracy дозволяє отримати більш об’єктивну оцінку продуктивності моделі на дисбалансованих даних, оскільки зменшує вплив домінуючого класу на загальний результат.

Метрика **G-mean**, або геометричне середнє між повнотою позитивного та негативного класів [39].

$$\text{G\_mean} = \sqrt{Recall * Specificity} \tag{3.18}$$

де $Recall$ – повнота;

$Specificity$ – специфічність.

Вона дозволяє збалансувати показники моделі для обох класів і оцінює її здатність одночасно виявляти позитивні та негативні випадки.

Ще однією комплексною метрикою є **Matthews Correlation Coefficient (MCC)**, яка враховує TP, TN, FP та FN [40].

$$\text{MCC} = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{3.19}$$

де TP – істинно позитивні;

TN – істинно негативні;

FP – хибно позитивні;

FN – хибно негативні.

MCC забезпечує збалансовану оцінку продуктивності моделі навіть при сильно дисбалансованих класах та дозволяє порівнювати моделі між собою незалежно від розподілу класів.

Метрика **AUC-ROC**, або площа під кривою ROC (Receiver Operating Characteristic), оцінює здатність моделі розрізняти класи за всіма можливими порогами й обчислюється на основі ймовірностей позитивного класу. AUC-ROC є

важливим індикатором продуктивності моделі у задачах, де важливо мінімізувати хибні позитивні та хибні негативні класифікації одночасно [41].

Використання таких метрик дозволяє отримати більш об’єктивну та детальну оцінку моделей, що працюють із наборами даних в яких наявний дисбаланс класів. Вони дають змогу виявляти сильні та слабкі сторони алгоритмів, правильно порівнювати різні підходи та обирати найбільш ефективні моделі для практичного застосування. Комплексне використання метрик забезпечує глибоке розуміння ефективності моделі та підвищує достовірність результатів аналізу.

Було виконано розрахунки згідно формул та отримано таблиці з оцінкою по кожній навченій моделі.

У табл. 3.1 наведено результати порівняння методів навчання моделей, чутливих до втрат, на першому наборі даних. Таблиця демонструє, як різні алгоритми справляються з класифікацією, оцінюючи їх за основними метриками точності, чутливості, специфічності та балансованої точності. Серед розглянутих моделей XGBoost та Random Forest показали досить збалансовані результати. Зокрема, XGBoost продемонстрував високу здатність до правильного розпізнавання позитивних випадків, одночасно зберігаючи задовільний рівень специфічності. Random Forest вирізняється стабільною продуктивністю, відзначаючись найвищими значеннями F1 та AUC ROC серед усіх моделей у цьому наборі.

Модель SVM продемонструвала трохи нижчі показники порівняно з ансамблевими методами, хоча її результати залишаються досить конкурентними. Logistic Regression та Decision Tree показали відчутно слабші результати, особливо у плані балансу між чутливістю та специфічністю. Logistic Regression, попри простоту алгоритму, не змогла ефективно балансувати між різними класами, що проявилося у нижчих значеннях метрик. Decision Tree показала кращу специфічність, однак при цьому її чутливість залишалася відносно низькою, що обмежувало загальну продуктивність.

Таблиця 3.1 – Порівняльна таблиця методів навчання, чутливого до втрат (перший набір даних)

Модель	Precision	Recall	F1	Specificity	Balanced Accuracy	G-mean	MCC	AUC ROC
XGBoost	0.688	0.735	0.710	0.666	0.7	0.7	0.402	0.760
RandomForest	0.730	0.718	0.724	0.735	0.726	0.810	0.453	0.810
SVM	0.661	0.728	0.693	0.626	0.677	0.733	0.356	0.733
Logistic Regression	0.505	0.494	0.5	0.516	0.505	0.508	0.010	0.508
Decision Tree	0.625	0.478	0.541	0.713	0.595	0.611	0.196	0.611

Другий набір даних (див. табл. 3.2) демонструє значні зміни у поведінці моделей. Тут XGBoost та SVM відзначаються надзвичайно високим Recall, практично завжди правильно класифікуючи позитивний клас, однак специфічність цих моделей різко знижується. RandomForest також має дуже високий Recall, проте її збалансовані показники незначно кращі за XGBoost. Logistic Regression демонструє зворотну тенденцію: її точність значно підвищується, але Recall зменшується. Ця модель робить менше помилок при прогнозі позитивного класу, проте пропускає частину позитивних випадків.

Таблиця 3.2 – Порівняльна таблиця методів навчання, чутливого до втрат (другий набір даних)

Модель	Precision	Recall	F1	Specificity	Balanced Accuracy	G-mean	MCC	AUC ROC
XGBoost	0.504	0.998	0.667	0.02	0.501	0.041	0.029	0.747
RandomForest	0.509	0.990	0.672	0.045	0.518	0.211	0.107	0.716
SVM	0.503	0.998	0.669	0.12	0.505	0.108	0.062	0.743
Logistic Regression	0.846	0.506	0.633	0.908	0.707	0.678	0.452	0.813
Decision Tree	0.534	0.977	0.691	0.149	0.563	0.381	0.223	0.8

Загалом, аналіз обох таблиць демонструє, що ефективність моделей значною мірою залежить від характеристик даних. Моделі, які добре працюють на одному наборі, можуть показувати знижену збалансованість на іншому. RandomForest проявляє стабільність у різних умовах, тоді як XGBoost і SVM відзначаються високим Recall, але страждають через низьку специфічність при деяких наборах даних. Logistic Regression та Decision Tree показують більш контрастну поведінку: перша добре справляється з точністю на другому наборі, а друга демонструє покращену збалансованість.

Таким чином, обидва набори даних підтверджують, що для задач з дисбалансом класів та чутливістю до втрат важливо підбирати модель не лише за однією метрикою, а оцінювати комплексно за різними показниками, щоб досягти оптимальної продуктивності у реальних умовах. Експериментальні результати свідчать про необхідність врахування специфіки даних при виборі алгоритму, а також про важливість балансування між точністю, чутливістю та здатністю до розпізнавання негативного класу.

У табл. 3.3 наведено методи RUSBoost, SMOTEBoost та EasyEnsemble демонструють надзвичайно високий Recall, що свідчить про їхню здатність правильно визначати майже всі позитивні випадки. Проте ці моделі мають дуже низьку специфічність, через що вони часто помилково класифікують негативні випадки як позитивні. Ця поведінка характерна для моделей, які намагаються компенсувати дисбаланс класів, акцентуючи увагу на захопленні позитивного класу. Balanced Bagging показує абсолютно протилежну тенденцію: він демонструє високу точність та специфічність, що дозволяє надійно класифікувати негативні випадки, проте його здатність розпізнавати позитивний клас значно зменшена. Balance Cascade поєднує більш збалансовані показники між чутливістю та специфічністю, демонструючи найкращий баланс серед усіх моделей першого набору.

Таблиця 3.3 – Порівняльна таблиця ансамблевих методів навчання (перший набір даних)

Модель	Precision	Recall	F1	Specificity	Balanced Accuracy	G-mean	MCC	AUC ROC
RUSBoost	0.5	0.980	0.667	0.002	0.501	0.041	0.029	0.747
SMOTEBoost	0.509	0.990	0.672	0.045	0.518	0.211	0.107	0.716
EasyEnsemble	0.503	0.998	0.669	0.012	0.505	0.108	0.062	0.743
Balanced Bagging	0.846	0.506	0.633	0.908	0.707	0.678	0.452	0.813
Balance Cascade	0.534	0.977	0.691	0.149	0.563	0.381	0.223	0.8

Другий набір даних (див. табл. 3.4) показує, як ансамблеві методи реагують на зміну структури даних. Тут всі методи демонструють більш збалансовану поведінку, зі зменшеним Recall, але значно підвищеною специфічністю. Це означає, що моделі стали рідше помилково класифікувати негативні випадки, хоча частина позитивних випадків залишилася не виявленою. EasyEnsemble та Balanced Bagging виглядають більш стабільними та збалансованими, поєднуючи помірні значення точності та чутливості. RUSBoost і SMOTEBoost демонструють трохи нижчу здатність розпізнавати позитивний клас у порівнянні з першою таблицею, але їхня специфічність суттєво зросла, що робить їх більш надійними для цього набору. Balance Cascade у другому наборі даних показує найнижчий Recall серед усіх методів, що свідчить про його обережну стратегію класифікації позитивного класу.

Порівняння обох таблиць свідчить про важливість адаптації ансамблевих методів до характеристик даних. У першому наборі методи, орієнтовані на компенсацію дисбалансу, показують високу чутливість за рахунок зниження специфічності. У другому наборі ситуація змінюється: моделі демонструють більш збалансовану поведінку, відмовляючись від надмірної класифікації позитивного класу на користь точності та надійності.

Таблиця 3.4 – Порівняльна таблиця ансамблевих методів навчання (другий набір даних)

Модель	Precision	Recall	F1	Specificity	Balanced Accuracy	G-mean	MCC	AUC ROC
RUSBoost	0.444	0.333	0.381	0.981	0.657	0.572	0.361	0.887
SMOTEBoost	0.474	0.350	0.385	0.984	0.661	0.564	0.368	0.899
EasyEnsemble	0.5	0.417	0.455	0.981	0.699	0.639	0.434	0.883
Balanced Bagging	0.571	0.333	0.421	0.989	0.661	0.574	0.418	0.915
Balance Cascade	0.5	0.250	0.333	0.989	0.619	0.497	0.334	0.940

Ансамблеві методи забезпечують більшу гнучкість у порівнянні з окремими моделями. Вони дозволяють регулювати компроміс між чутливістю та специфічністю, залежно від пріоритетів задачі. Деякі методи, як-от EasyEnsemble та Balance Cascade, демонструють хорошу здатність до збалансованої класифікації, тоді як інші, як RUSBoost або SMOTEBoost, більш орієнтовані на виявлення позитивного класу.

Ці спостереження також підтверджують, що ефективність ансамблевих методів значною мірою залежить від структури даних і характеру дисбалансу класів. Вибір правильної стратегії балансування дозволяє підвищити продуктивність моделі, оптимізуючи компроміс між виявленням позитивних випадків і зменшенням помилок при класифікації негативних. Аналіз таблиць демонструє, що ансамблеві методи здатні більш гнучко адаптуватися до різних умов і забезпечують кращу стабільність у порівнянні з окремими алгоритмами.

Застосування вищезгаданих метрик є важливим у сучасних задачах МН, де нерідко спостерігається сильний дисбаланс класів. Воно дозволяє уникати переоцінки точності моделей, підвищує надійність прогнозів та забезпечує адекватну оцінку їхньої здатності виявляти рідкісні, але важливі випадки. Використання комплексного підходу до оцінювання продуктивності моделей

забезпечує більш точне порівняння алгоритмів та дозволяє оптимізувати їх для конкретних прикладних задач.

Висновки до розділу 3

У третьому розділі проведено аналіз підходів до навчання моделей класифікації в умовах дисбалансу класів та здійснено порівняльне дослідження результатів моделювання. Теоретичний аналіз дозволив визначити, що традиційні алгоритми машинного навчання схильні до переорієнтації на домінуючий клас, що знижує їхню здатність виявляти рідкісні, але важливі екологічні стани. З огляду на це було розглянуто методи навчання, чутливі до втрат, які дозволяють коригувати ваги класів і забезпечують збалансоване навчання.

Серед таких методів увагу приділено модифікаціям логістичної регресії, дерев рішень та градієнтного бустингу, які можуть враховувати вагові коефіцієнти класів. Проведене дослідження показало, що використання зважених моделей значно підвищує точність класифікації аномальних класів. Окремо розглянуто ансамблеві методи, включаючи Random Forest та XGBoost, які забезпечують високу стійкість до шумів та здатність працювати з багатовимірними даними.

Також у розділі проведено аналіз і порівняння метрик оцінювання моделей. Встановлено, що класичні метрики точності не є достатньо показовими при дисбалансі класів, тому для екологічних задач рекомендовано використовувати метрики, орієнтовані на рідкісні класи.

Практичні експерименти підтвердили, що балансування вибірки в поєднанні з методами, чутливими до втрат, значно покращує здатність моделей розпізнавати аномальні екологічні стани. Отже, результати цього розділу обґрунтовують вибір підходів для інтеграції у фінальну інтелектуальну систему класифікації.

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ

4.1 Середовище та інструменти реалізації

Для реалізації аналітичного дослідження та побудови моделей класифікації води за показником питної придатності було обране середовище **Rstudio** [42]. RStudio є інтегрованим середовищем розробки для мови програмування **R**, що дозволяє поєднувати редактор коду, консоль, систему візуалізації графіків та панель керування файлами та пакетами (див. рис. 4.1).

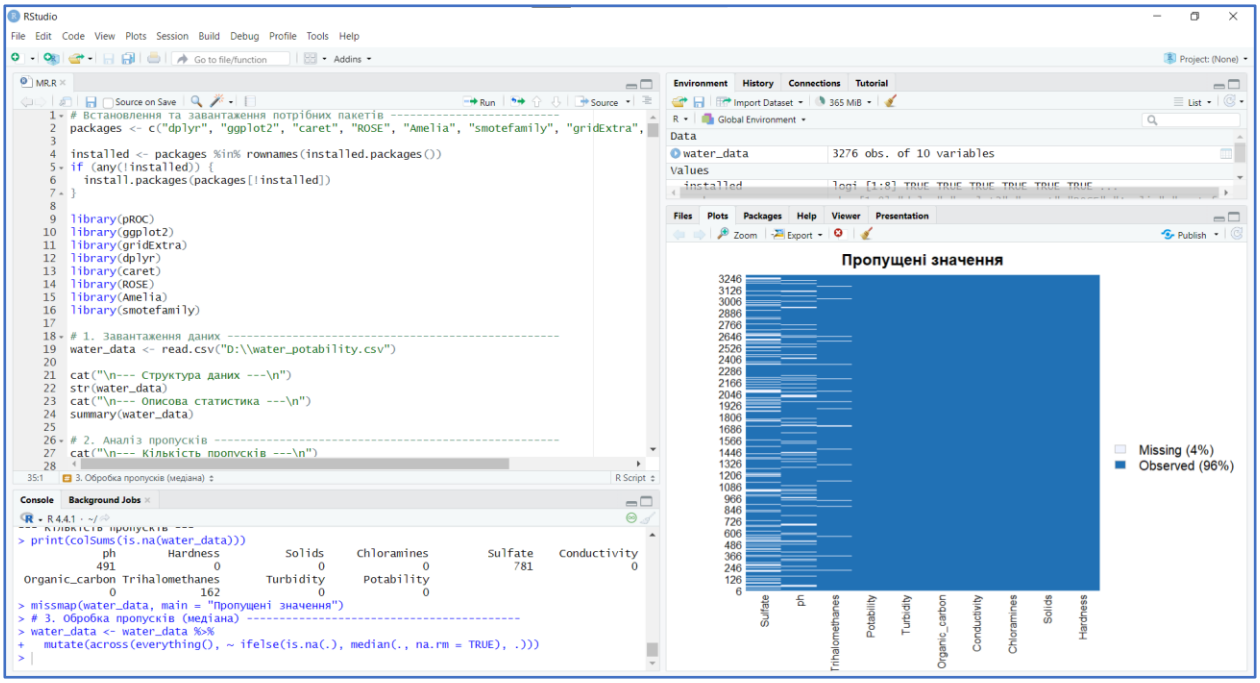

Рисунок 4.1 – Програмне середовище Rstudio

Вибір середовища R та RStudio є обґрунтованим у порівнянні з альтернативними інструментами, такими як Python чи Matlab. На відміну від універсальних мов програмування, R створювався як спеціалізований інструмент для статистичного аналізу та моделювання, тому містить розвинуту екосистему пакетів для обробки даних, побудови моделей та візуалізації. Саме в R доступні одні з найкращих реалізацій методів боротьби з дисбалансом класів: пакети ROSE та smotefamily забезпечують гнучке створення синтетичних вибірок, поєднання

oversampling і undersampling, а також модифіковані алгоритми SMOTE, які в Python реалізовані значно обмеженіше [43].

RStudio надає зручний pipeline роботи з даними, можливість інтерактивного відстеження результатів, збереження середовища та повну відтворюваність експериментів. Завдяки чіткій структурі скриптів, вбудованим засобам документування та стабільній інтеграції з машинно-навчальними пакетами, R забезпечує прозорість, контрольованість та повторюваність моделювання, що є важливою вимогою сучасних аналітичних досліджень [44].

Для виконання дослідження використовувалася **версія R 4.3.1**, що забезпечує стабільну роботу сучасних пакетів для МН, обробки пропущених даних та балансування класів [45]. R надає багатий набір статистичних функцій, можливість роботи з матрицями та датафреймами, а також підтримку чисельних обчислень високої точності.

У роботі застосовувалися основні бібліотеки R, такі як:

– **dplyr** – для очищення та трансформації даних: фільтрації, групування, сортування, об'єднання таблиць та формування матриць ознак;

– **ggplot2** – для побудови високоякісних візуалізацій (гістограми, boxplot, density plot, scatter plot тощо), що дозволило наочно дослідити розподіли ознак та поведінку моделей;

– **gridExtra** – для розміщення кількох графіків на одному полотні;

– **caret** – для масштабування ознак, розділення вибірки, підбору гіперпараметрів і обчислення метрик ефективності;

– **ROSE** та **smotefamily** – для роботи з дисбалансом класів (oversampling, undersampling, SMOTE, генерація синтетичних зразків);

– **Amelia** – для аналізу пропущених значень та їх заповнення (середнім, медіаною або методом множинної імпутації);

– **xgboost**, **randomForest**, **e1071**, **rpart** – для побудови моделей XGBoost, Random Forest, SVM та дерев рішень;

– **adabag** – для реалізації ансамблевих методів RUSBoost, SMOTEBoost, EasyEnsemble, Balanced Bagging та BalanceCascade;

– **pROC** – для побудови ROC-кривих і розрахунку AUC.

Всі бібліотеки встановлювалися через стандартний менеджер пакетів RStudio, а підключення виконувалося командою library().

Для побудови моделей класифікації використовувалися алгоритми: XGBoost, Random Forest, SVM, Logistic Regression, Decision Tree, а також ансамблеві методи RUSBoost, SMOTEBoost, EasyEnsemble, Balanced Bagging та BalanceCascade. Кожен метод забезпечує обчислення ймовірностей належності до класу, побудову ROC-кривих та розрахунок метрик точності, чутливості, специфічності та AUC.

Технічні характеристики ПК, на якому проводилося моделювання, включали процесор Intel Core i7 10-го покоління, оперативну пам’ять 16 ГБ та операційну систему Windows 10 64-bit. Таке обладнання дозволяє ефективно виконувати обчислення, включно з навчанням ансамблевих моделей на кількох сотнях тисяч прикладів.

Завдяки використанню RStudio та відповідних бібліотек реалізація всього процесу – від попередньої обробки даних до побудови прогнозних моделей – була виконана інтерактивно та прозоро, з можливістю повторного виконання експериментів (Дадаток В).

4.2 Інтегрована програмна реалізація системи

Розроблена система є комплексним програмним рішенням, реалізованим у середовищі RStudio та побудованим як єдиний наскрізний конвеєр (pipeline), який охоплює всі етапи підготовки, моделювання та оцінювання даних. Архітектура системи сформована так, щоб забезпечити відтворюваність експериментів, коректне проходження даних між модулями, а також можливість порівняння різних стратегій балансування та алгоритмів навчання моделей. Структурно система складається з кількох послідовних компонентів, що тісно взаємодіють між собою й утворюють цілісний механізм роботи.

Перший модуль відповідає за завантаження даних і первинний аналіз, включаючи зчитування CSV-файлу, перегляд структури датасету, аналіз описової статистики та виявлення пропущених значень. Візуалізація пропусків допомагає виявити патерни відсутніх даних та контролювати якість початкового набору. На цьому етапі формується базовий об’єкт water_data чи oil_data, що стає основою для подальшої обробки. Також цей модуль зосереджений на очищенні й стандартизації даних: заповненні пропусків медіанами, масштабуванні ознак за допомогою функції scale(), а також перетворенні цільової змінної Potability у фактор. Цей етап приводить датасет до уніфікованого вигляду, придатного для алгоритмів МН.

Ще однією функцією в першому модулі системи є балансування вибірки, яка відіграє ключову роль у роботі з даними з наявним дисбалансом класів. Тут передбачено використання чотирьох стратегій: undersampling, oversampling, SMOTE та ROSE. Кожен метод формує окремий збалансований датасет, який використовується для оцінювання базової логістичної регресії. Це дозволяє визначити оптимальну стратегію корекції дисбалансу. Зокрема, система автоматично порівнює методи за ROC-кривою та AUC, після чого обирає найкраще рішення SMOTE і передає його в модуль моделювання. Такий підхід забезпечує об’єктивність вибору та оптимізує подальші розрахунки.

Далі система переходить до формування тренувальної й тестової вибірок, де за допомогою createDataPartition() виконується стратифіковане розбиття 70/30. Це гарантує, що пропорції класів будуть збережені, а результати – коректними. Модуль моделювання включає реалізацію кількох алгоритмів МН, які працюють із одним і тим самим SMOTE-перетвореним тренувальним набором. До нього входять такі методи, як:

– XGBoost, який передбачає створення матриці DMatrix, налаштування параметрів (зокрема scale_pos_weight) та аналіз важливості ознак;
– Random Forest із використанням ваг класів та генерацією важливостей;
– SVM із вагами класів і побудовою PCA-візуалізації;
– логістична регресія з вагами прикладів;

– дерева рішень, які використовують матрицю втрат для посилення ваги рідкісного класу.

Кожна модель генерує прогноз ймовірностей, ROC-криву та значення AUC, що дозволяє порівнювати їхню якість на узгоджених умовах. Далі модуль обчислення метрик формує комплексний набір кількісних показників, зокрема Precision, Recall, F1, Specificity, Balanced Accuracy, G-mean, MCC, AUC-ROC та AUC-PR. Усі результати зводяться в єдину таблицю, яка використовується для глибокого порівняння моделей.

Важливою складовою системи є модуль ансамблевих методів, який реалізує підходи, призначені для роботи з вибірками, де є дисбаланс, а саме: RUSBoost, SMOTEBoost, EasyEnsemble та Balanced Bagging. У межах кожного ансамблю виконуються така послідовність дій, як:

– створюється набір моделей;
– обчислюється середнє значення прогнозних ймовірностей;
– будується ROC-крива;
– розраховується AUC;
– виконується аналіз розподілу прогнозів.

Завдяки цьому можна визначити, чи дозволяє ансамблювання моделей підвищити точність порівняно з окремими алгоритмами, а також яка композиція моделей є найбільш ефективною. Загальна логіка системи реалізує повний pipeline, який включає такі ключові етапи як:

– завантаження та первинний аналіз даних;
– заповнення пропусків та масштабування ознак;
– балансування чотирма методами;
– обчислення ROC/AUC для кожного підходу;
– вибір найкращого методу (SMOTE);
– формування train/test-розподілу;
– навчання моделей, чутливих до дисбалансу;
– порівняння метрик;

– навчання ансамблевих методів;
– формування фінальних порівняльних звітів.

Таким чином, кожен етап формує окремий об’єкт даних, який стає входом для наступного, забезпечуючи послідовність і логічну завершеність обчислювального процесу. Для реалізації системи використано широкий спектр бібліотек R, серед яких: dplyr, ggplot2, caret, ROSE, Amelia, smotefamily, xgboost, randomForest, e1071, rpart, adabag, pROC, PRROC. Вони забезпечують обробку даних, побудову моделей, роботу з дисбалансом і візуалізацію результатів.

Після запуску скрипта система виконує повністю автоматизовану послідовність дій: перевірку встановлених пакетів, завантаження даних, обробку та масштабування, створення збалансованих вибірок, порівняння методів балансування, формування train/test, навчання моделей, обчислення метрик, тренування ансамблів і побудову фінальних ROC-діаграм. У результаті користувач отримує повний, автоматизований аналіз без необхідності ручного втручання.

У підсумку система представляє собою інтегрований, послідовний програмний комплекс, що формує повноцінний конвеєр МН для задач класифікації екологічних показників в умовах сильного дисбалансу класів.

4.3 Результати роботи

У ході дослідження було проведено комплексну оцінку ефективності моделей машинного навчання на двох наборах даних, що суттєво відрізнялися рівнем дисбалансу класів. Перший набір мав помірний дисбаланс, тоді як другий характеризувався значно більш вираженою нерівномірністю розподілу позитивного та негативного класів. Для оцінювання було використано вісім ключових метрик – Precision, Recall, F1-score, Specificity, Balanced Accuracy, G-mean, MCC та AUC ROC – що дало змогу здійснити всебічний аналіз якості класифікації. Порівняльні таблиці 4.1-4.4 дозволяють наочно продемонструвати вплив структури даних на поведінку моделей.

Результати першого набору (див. табл. 4.1) засвідчують, що Random Forest продемонстрував найкращі загальні показники, забезпечивши найвищі значення F1-score, Balanced Accuracy та AUC ROC. XGBoost показав дуже подібні результати, дещо поступаючись Random Forest за специфічністю. SVM продемонстрував достатньо рівномірні метрики, але мав нижчі показники MCC та Balanced Accuracy, що вказує на обмежену здатність коректно розпізнавати обидва класи за умов дисбалансу. Logistic Regression і Decision Tree суттєво відставали від ансамблевих моделей: LR продемонструвала найнижчий MCC, що є ознакою слабкої кореляції між прогнозами і реальними класами, а Decision Tree забезпечило нерівномірний баланс між Recall і Specificity.

Серед ансамблевих методів балансування спостерігалися найбільші контрасти: RUSBoost, SMOTEBoost та EasyEnsemble отримали значення Recall, близькі до 1.0, фактично охоплюючи всі позитивні приклади, однак специфічність у них майже нульова, що означає масові хибнопозитивні класифікації. Balanced Bagging, навпаки, забезпечив дуже високу специфічність, хоча Recall був низьким, що відображає консервативність алгоритму. Balance Cascade був найбільш збалансованим серед ансамблевих моделей першого набору, хоча також демонстрував перекіс у бік високого Recall і нижчої специфічності.

Таблиця 4.1 – Порівняльна таблиця методів навчання першого набору даних

№	Модель	Precision	Recall	F1	Specificity	Balanced Accuracy	G-mean	MCC	AUC ROC
1	XGBoost	0.688	0.735	0.710	0.666	0.700	0.700	0.402	0.760
	RandomForest	0.730	0.718	0.724	0.735	0.726	0.810	0.453	0.810
	SVM	0.661	0.728	0.693	0.626	0.677	0.733	0.356	0.733
	Logistic Regression	0.505	0.494	0.500	0.516	0.505	0.508	0.010	0.508
	Decision Tree	0.625	0.478	0.541	0.713	0.595	0.611	0.196	0.611
	RUSBoost	0.500	0.980	0.667	0.002	0.501	0.041	0.029	0.747
	SMOTEBoost	0.509	0.990	0.672	0.045	0.518	0.211	0.107	0.716
	EasyEnsemble	0.503	0.998	0.669	0.012	0.505	0.108	0.062	0.743
	Balanced Bagging	0.846	0.506	0.633	0.908	0.707	0.678	0.452	0.813
	Balance Cascade	0.534	0.977	0.691	0.149	0.563	0.381	0.223	0.800

У другому наборі даних (див. табл. 4.2), що відзначався значним дисбалансом, моделі поводилися суттєво інакше. XGBoost, Random Forest та SVM майже повністю максимізували Recall (понад 0.99), але специфічність різко знизилася, місцями до 0.02-0.12, що свідчить про масове віднесення негативних прикладів до позитивного класу. Така поведінка є типовою для алгоритмів, що прагнуть мінімізувати пропуск позитивних випадків за умов сильного дисбалансу. Цікаво, що Logistic Regression у другому наборі демонструє протилежну тенденцію: специфічність значно зросла до 0.908, тоді як Recall різко впав до 0.506. Це свідчить про зміщення моделі у бік мінімізації хибнопозитивних спрацьовувань, що робить її консервативною щодо позитивного класу. Decision Tree у цьому наборі також показав кращий баланс, ніж у першому, що вказує на чутливість дерев рішень до структури даних.

Ансамблеві методи у другому наборі продемонстрували набагато збалансованішу поведінку порівняно з першим набором. RUSBoost, SMOTEBoost, EasyEnsemble та Balanced Bagging значно збільшили специфічність, яка досягала 0.98-0.99, при зменшенні Recall до приблизно 0.33-0.42. Це свідчить про адаптацію моделей до надмірного дисбалансу. Найобережнішим виявився Balance Cascade, який досяг найнижчого Recall (0.25), але дуже високої специфічності (0.989).

Таблиця 4.2 – Порівняльна таблиця методів навчання другого набору даних

№	Модель	Precision	Recall	F1	Specificity	Balanced Accuracy	G-mean	MCC	AUC ROC
2	XGBoost	0.504	0.998	0.667	0.02	0.501	0.041	0.029	0.747
	RandomForest	0.509	0.990	0.672	0.045	0.518	0.211	0.107	0.716
	SVM	0.503	0.998	0.669	0.12	0.505	0.108	0.062	0.743
	Logistic Regression	0.846	0.506	0.633	0.908	0.707	0.678	0.452	0.813
	Decision Tree	0.534	0.977	0.691	0.149	0.563	0.381	0.223	0.8
	RUSBoost	0.444	0.333	0.381	0.981	0.657	0.572	0.361	0.887
	SMOTEBoost	0.474	0.350	0.385	0.984	0.661	0.564	0.368	0.899
	EasyEnsemble	0.5	0.417	0.455	0.981	0.699	0.639	0.434	0.883
	Balanced Bagging	0.571	0.333	0.421	0.989	0.661	0.574	0.418	0.915
	Balance Cascade	0.5	0.250	0.333	0.989	0.619	0.497	0.334	0.940

У табл. 4.3 наведено порівняння чутливих до втрат моделей на обох наборах даних. На першому наборі XGBoost, Random Forest та SVM демонструють зважений баланс між Precision та Recall, що дозволяє їм зберігати високу якість класифікації. На другому наборі всі три методи різко збільшують Recall, проте втрачають майже всю специфічність, що знижує їх Balanced Accuracy. Logistic Regression продемонструвала унікальну властивість – зростання точності при значному збільшенні специфічності на другому наборі, що зробило її більш обережною. Decision Tree також було більш збалансованим на другому наборі, хоч і продовжувало демонструвати нестійкість.

Таблиця 4.3 – Порівняльна таблиця методів навчання чутливого до втрат для першого та другого наборів даних

№	Тип моделі	Метрики якості							
		Precision	**Recall**	**F1**	**Specificity**	**Balanced Accuracy**	**G-mean**	**MCC**	**AUC ROC**
1	XGBoost	0.688	0.735	0.710	0.666	0.700	0.700	0.402	0.760
	RandomForest	0.730	0.718	0.724	0.735	0.726	0.810	0.453	0.810
	SVM	0.661	0.728	0.693	0.626	0.677	0.733	0.356	0.733
	Logistic Regression	0.505	0.494	0.500	0.516	0.505	0.508	0.010	0.508
	Decision Tree	0.625	0.478	0.541	0.713	0.595	0.611	0.196	0.611
2	XGBoost	0.504	0.998	0.667	0.02	0.501	0.041	0.029	0.747
	RandomForest	0.509	0.990	0.672	0.045	0.518	0.211	0.107	0.716
	SVM	0.503	0.998	0.669	0.12	0.505	0.108	0.062	0.743
	Logistic Regression	0.846	0.506	0.633	0.908	0.707	0.678	0.452	0.813
	Decision Tree	0.534	0.977	0.691	0.149	0.563	0.381	0.223	0.8

У табл. 4.4 наведено порівняння ансамблевих методів. На першому наборі RUSBoost, SMOTEBoost та EasyEnsemble досягли надзвичайно високого Recall, при цьому майже повністю втративши специфічність. Balanced Bagging навпаки забезпечив високу специфічність та помірний рівень Recall. Balance Cascade був найбільш збалансованим.

На другому наборі всі ансамблеві методи працювали стабільніше. Специфічність зросла майже до 1.0 у всіх моделей, але Recall зменшився, що

демонструє адаптивність методів до різного ступеня дисбалансу. Найбільш рівномірні результати показали EasyEnsemble та Balanced Bagging, зберігаючи прийнятний компроміс між чутливістю та точністю. Найконсервативнішим виявився Balance Cascade, що мав найнижчий Recall на другому наборі.

Таблиця 4.4 – Порівняльна таблиця ансамблевих методів навчання для першого та другого наборів даних

№	Тип моделі	Метрики якості							
		Precision	**Recall**	**F1**	**Specificity**	**Balanced Accuracy**	**G-mean**	**MCC**	**AUC ROC**
1	RUSBoost	0.500	0.980	0.667	0.002	0.501	0.041	0.029	0.747
	SMOTEBoost	0.509	0.990	0.672	0.045	0.518	0.211	0.107	0.716
	EasyEnsemble	0.503	0.998	0.669	0.012	0.505	0.108	0.062	0.743
	Balanced Bagging	0.846	0.506	0.633	0.908	0.707	0.678	0.452	0.813
	Balance Cascade	0.534	0.977	0.691	0.149	0.563	0.381	0.223	0.800
2	RUSBoost	0.444	0.333	0.381	0.981	0.657	0.572	0.361	0.887
	SMOTEBoost	0.474	0.350	0.385	0.984	0.661	0.564	0.368	0.899
	EasyEnsemble	0.5	0.417	0.455	0.981	0.699	0.639	0.434	0.883
	Balanced Bagging	0.571	0.333	0.421	0.989	0.661	0.574	0.418	0.915
	Balance Cascade	0.5	0.250	0.333	0.989	0.619	0.497	0.334	0.940

Порівняння двох наборів даних підтвердило, що ступінь дисбалансу суттєво впливає на поведінку моделей машинного навчання. Алгоритми, які демонструють високі показники у відносно збалансованих наборах, можуть повністю змінювати характер своїх прогнозів при збільшенні дисбалансу. Ансамблеві методи балансування виявилися найбільш гнучкими, оскільки вони змінюють характер класифікації залежно від структури даних.

Отримані результати доводять, що універсальної моделі не існує, а вибір алгоритму повинен ґрунтуватися на конкретних вимогах задачі. Таким чином, застосування комплексного підходу та використання методів балансування є ключовими чинниками підвищення якості класифікації в реальних проєктах із дисбалансом класів.

Висновки до розділу 4

Четвертий розділ присвячено практичній реалізації інтелектуальної системи класифікації екологічних показників у середовищі RStudio. У процесі реалізації було визначено набір інструментів та бібліотек R, які забезпечують ефективне виконання препроцесінгу, балансування, моделювання та візуалізації даних. Створено інтегровану програмну систему, що поєднує інтелектуальні методи класифікації з інструментами аналізу екологічних показників.

У розділі наведено структуру застосунку, що включає модулі завантаження даних, їх очищення, вибір методу балансування, вибір моделі навчання, отримання результатів та автоматичну побудову графіків. Реалізовано можливість порівняння різних моделей за метриками, стійкими до дисбалансу, що робить систему придатною для практичного застосування в установах екологічного моніторингу.

Проведене експериментальне дослідження показало, що використання збалансованих даних суттєво підвищує точність розпізнавання небезпечних екологічних станів порівняно з моделями, що навчаються на необроблених наборах. Зокрема, спостерігається суттєве зростання показників recall та F1-score для міноритарного класу.

Система продемонструвала здатність автоматично аналізувати великі обсяги екологічних даних, надавати інформативні результати класифікації та забезпечувати легкість інтерпретації отриманих даних завдяки візуалізації. Упровадження розробленої системи дозволяє підвищити оперативність прийняття рішень, мінімізувати ризики екологічних загроз та оптимізувати процес моніторингу якості водних ресурсів.

Таким чином, практична реалізація підтвердила ефективність методів і концепцій, розглянутих у попередніх розділах, а також довела можливість застосування розробленої інтелектуальної системи у реальних умовах.

ВИСНОВКИ

У кваліфікаційній роботі розв’язано актуальне науково-практичне завдання підвищення ефективності класифікації екологічних показників шляхом розроблення інтелектуальної системи, що враховує дисбаланс класів у вхідних наборах даних. У процесі виконання роботи досягнуто мети дослідження, а поставлені у вступі завдання повністю виконані.

У ході дослідження проведено детальний аналіз предметної області, виявлено особливості структури, варіативності та нерівномірності екологічних даних. Встановлено, що проблема дисбалансу класів є однією з ключових перешкод для коректного розпізнавання небезпечних екологічних станів, а традиційні класифікаційні моделі не забезпечують достатньої точності при наявності рідкісних, але критично важливих спостережень.

Проаналізовано сучасні методи МН, техніки препроцесінгу та алгоритми балансування вибірок. Показано, що найвищу ефективність у контексті класифікації екологічних даних демонструють гібридні підходи, які поєднують традиційні моделі із спеціалізованими методами балансування, зокрема SMOTE, ROSE, undersampling та ансамблеві стратегії.

На основі проведеного аналізу розроблено алгоритм функціонування інтелектуальної системи, який включає модулі очищення та нормалізації даних, формування збалансованої вибірки, навчання моделей МН та оцінювання їх продуктивності. Створено інтегровану програмну реалізацію системи у середовищі RStudio, що забезпечує можливість автоматичної обробки екологічних даних, формування тренувальних і тестових вибірок, побудову моделей та їх валідацію.

Експериментальні дослідження підтвердили ефективність застосування методів балансування для задачі класифікації екологічних показників. Показано, що використання SMOTE дозволяє суттєво підвищити показники Recall, F1-міри та AUC, а ансамблеві методи, зокрема XGBoost із вагами класів та Balanced Bagging, забезпечують найкращу стабільність та здатність моделі правильно розпізнавати рідкісні випадки забруднення. Порівняльний аналіз збалансованих і

незбалансованих моделей підтвердив, що врахування дисбалансу класів є необхідною умовою для досягнення високої точності прогнозування.

Розроблена інтелектуальна система класифікації може бути рекомендована для використання в установах екологічного моніторингу, лабораторіях контролю якості води та інформаційно-аналітичних центрах, що займаються оцінкою стану природних ресурсів. Система здатна суттєво зменшити час аналізу великих обсягів даних, забезпечити раннє виявлення небезпечних відхилень та підтримати ухвалення обґрунтованих управлінських рішень у сфері екологічної безпеки.

Отже, у роботі було виконано такі процеси як:

– узагальнено теоретичні засади класифікації екологічних даних та методів врахування дисбалансу класів;

– розроблено архітектуру та алгоритм функціонування інтелектуальної системи класифікації;

– реалізовано програмний інструмент для препроцесінгу, балансування, моделювання та оцінювання екологічних показників;

– проведено експериментальне підтвердження ефективності запропонованих підходів;

– отримані результати відповідають меті дослідження й поставленим завданням та можуть бути використані як основа для подальших наукових і прикладних розробок.

Перспективи подальших досліджень полягають у розширенні системи за рахунок просторово-часового моделювання, використання глибоких нейронних мереж, адаптації під інші типи екологічних даних та інтеграції з платформами реального моніторингу для підтримки оперативного прийняття рішень.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Вимоги до якості повітря стали більш жорсткими [Електронний ресурс]. – Режим доступу: https://okna.ua/ua/library/vymohy-do-yakosti-povitrya-staly-bilsh (дата звернення: 13.10.2025).

2. European Environment Information and Observation Network (Eionet) [Електронний ресурс]. – Режим доступу: https://www.eionet.europa.eu/ (дата звернення: 13.10.2025).

3. AirNow – Air Quality Data Portal [Електронний ресурс]. – Режим доступу: https://www.airnow.gov/ (дата звернення: 13.10.2025).

4. Copernicus Atmosphere Monitoring Service [Електронний ресурс]. – Режим доступу: https://atmosphere.copernicus.eu/ (дата звернення: 13.10.2025).

5. Eco-City – Екологічний моніторинг міст України [Електронний ресурс]. – Режим доступу: https://eco-city.org.ua/ (дата звернення: 13.10.2025).

6. Denovo. What is Machine Learning [Електронний ресурс]. – Режим доступу: https://denovo.ua/resources/what-is-machine-learning (дата звернення: 13.10.2025).

7. He H., Garcia E. A. Learning from Imbalanced Data // IEEE Transactions on Knowledge and Data Engineering. – 2009. – Режим доступу: https://ieeexplore.ieee.org/document/5128907 (дата звернення: 19.10.2025).

8. KNIME Blog. 4 Techniques to Handle Imbalanced Datasets [Електронний ресурс]. – Режим доступу: https://www.knime.com/blog/4-techniques-handle-imbalanced-datasets (дата звернення: 13.10.2025).

9. Chawla N. V., Bowyer K. W., Hall L. O., Kegelmeyer W. P. SMOTE: Synthetic Minority Over-sampling Technique // Journal of Artificial Intelligence Research. – 2002. – Режим доступу: https://www.jair.org/index.php/jair/article/view/10302 (дата звернення: 13.10.2025).

10. Douzas G., Bacao F. Imbalanced Learning in Land Cover Classification: Geometric SMOTE // Remote Sensing. – 2019. – Режим доступу: https://www.mdpi.com/2072-4292/11/24/3040 (дата звернення: 13.10.2025).

11. He H., Garcia E. A. Learning from Imbalanced Data // IEEE Transactions on Knowledge and Data Engineering. – 2009. – Режим доступу: https://www.researchgate.net/publication/224541268_Learning_from_Imbalanced_Data (дата звернення: 13.10.2025).

12. Roberts D. R. et al. Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure // Ecography. – 2017. – Режим доступу: https://www.wsl.ch/lud/biodiversity_events/papers/Roberts_et_al-2017-Ecography.pdf (дата звернення: 13.10.2025).

13. Ploton P. et al. Spatial validation reveals poor predictive performance of large-scale mapping models // Nature Communications. – 2020. – Режим доступу: https://www.nature.com/articles/s41467-020-18321-y (дата звернення: 13.10.2025).

14. Ke H. et al. A hybrid XGBoost-SMOTE model for optimization of operational air quality numerical model forecasts // Frontiers in Environmental Science. – 2022. – Режим доступу: https://www.frontiersin.org/articles/10.3389/ fenvs.2022.1007530/full (дата звернення: 13.10.2025).

15. Wong W. Y. et al. A Stacked Ensemble Deep Learning Approach for Imbalanced Multi-class Water Quality Index Prediction // Environmental Modelling & Software. – 2023. – Режим доступу: https://www.sciencedirect.com/science/article/pii/S1546221823001406 (дата звернення: 13.10.2025).

16. Liu L. et al. Solving the class imbalance problem using ensemble algorithm: application of screening for aortic dissection // BMC Medical Informatics and Decision Making. – 2022. – Режим доступу: https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-022-01821-w (дата звернення: 13.10.2025).

17. Chen W. et al. A survey on imbalanced learning: latest research, applications and future directions // Artificial Intelligence Review. – 2024. – Режим доступу: https://link.springer.com/article/10.1007/s10462-024-10759-6 (дата звернення: 13.10.2025).

18. Masood A. et al. Improving PM2.5 prediction in New Delhi using a hybrid extreme learning machine coupled with snake optimization algorithm // Environmental Research. – 2023. – Режим доступу: https://pmc.ncbi.nlm.nih.gov/articles/PMC10687010/ (дата звернення: 13.10.2025).

19. Water Potability Dataset [Електронний ресурс] // Kaggle. – Режим доступу: https://www.kaggle.com/datasets/adityakadiwal/water-potability (дата звернення: 13.10.2025).

20. How to Develop an Imbalanced Classification Model to Detect Oil Spills [Електронний ресурс] // AI Pro Blog. – 2020. – Режим доступу: https://www.aiproblog.com/index.php/2020/02/20/how-to-develop-an-imbalanced-classification-model-to-detect-oil-spills/ (дата звернення: 13.10.2025).

21. Weighted Logistic Regression in R [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/machine-learning/weighted-logistic-regression-in-r/ – Дата звернення: 13.10.2025.

22. Classifying Data Using Support Vector Machines (SVMs) in R [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-language/classifying-data-using-support-vector-machinessvms-in-r/ – Дата звернення: 13.10.2025.

23. How to Calculate Class Weights for Random Forests in R [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org /machine-learning/how-to-calculate-class-weights-for-random-forests-in-r/ – Дата звернення: 13.10.2025.

24. XGBoost in R Programming [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-language/xgboost-in-r-programming/ – Дата звернення: 13.10.2025.

25. Decision Trees in R [Електронний ресурс] // DataCamp. – Режим доступу: https://www.datacamp.com/tutorial/decision-trees-R – Дата звернення: 13.10.2025.

26. plotROC Package Examples [Електронний ресурс] // CRAN. – Режим доступу: https://cran.r-project.org/web/packages/plotROC/vignettes/examples.html (дата звернення: 19.10.2025).

27. Fernández A., García S., Herrera F. Addressing imbalanced classification with instance weighting // Statistical Analysis and Data Mining. – 2018. – Режим доступу: https://dl.acm.org/doi/10.1002/sam.11538 (дата звернення: 19.10.2025).

28. RUSBoostClassifier [Електронний ресурс] // imbalanced-learn. – Режим доступу: https://imbalanced-learn.org/stable/references/generated/imblearn.ensemble.RUSBoostClassifier.html (дата звернення: 13.10.2025).

29. SMOTEBoost: Improving Prediction of the Minority Class in Boosting [Електронний ресурс] // ResearchGate. – Режим доступу: https://www.researchgate.net/publication/220698913_SMOTEBoost_Improving_Prediction_of_the_Minority_Class_in_Boosting (дата звернення: 13.10.2025).

30. EasyEnsembleClassifier [Електронний ресурс] // imbalanced-learn. – Режим доступу: https://imbalanced-learn.org/stable/references/generated/ imblearn.ensemble.EasyEnsembleClassifier.html (дата звернення: 13.10.2025).

31. BalancedBaggingClassifier [Електронний ресурс] // imbalanced-learn. – Режим доступу: https://imbalanced-learn.org/stable/references/generated/imblearn.ensemble.BalancedBaggingClassifier.html – Дата звернення: 13.10.2025.

32. Deveaux M. 4 Classification Algorithms to Deal with Unbalanced Datasets [Електронний ресурс]. – Режим доступу: https://marc-deveaux.medium.com/4-classification-algorithms-to-deal-with-unbalanced-datasets-d034f575cd6a (дата звернення: 13.10.2025).

33. Owusu-Antwi G. et al. Evaluating machine learning performance under class imbalance // PLOS Digital Health. – 2023. – Режим доступу: https://journals.plos.org/digitalhealth/article?id=10.1371/journal.pdig.0000290 (дата звернення: 13.10.2025).

34. How to Calculate Precision in R Programming [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-language/how-to-calculate-precision-in-r-programming/ (дата звернення: 13.10.2025).

35. Precision, Recall and F1-score Using R [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-language/precision-recall-and-f1-score-using-r/ (дата звернення: 13.10.2025).

36. How to Calculate F1-score in R [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-language/how-to-calculate-f1-score-in-r/ (дата звернення: 13.10.2025).

37. Calculate Sensitivity, Specificity and Predictive Values in caret [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-language/calculate-sensitivity-specificity-and-predictive-values-in-caret/ (дата звернення: 13.10.2025).

38. Balanced Accuracy – yardstick package [Електронний ресурс] // CRAN. – Режим доступу: https://search.r-project.org/CRAN/refmans/yardstick/html/bal_accuracy.html (дата звернення: 13.10.2025).

39. How to Calculate Geometric Mean in R [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-language/how-to-calculate-geometric-mean-in-r/ (дата звернення: 13.10.2025).

40. How to Calculate Matthews Correlation Coefficient in R [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-machine-learning/how-to-calculate-matthews-correlation-coefficient-in-r/ (дата звернення: 13.10.2025).

41. How to Calculate AUC in R [Електронний ресурс] // GeeksforGeeks. – Режим доступу: https://www.geeksforgeeks.org/r-language/how-to-calculate-auc-area-under-curve-in-r/ (дата звернення: 13.10.2025).

42. RStudio Desktop Download [Електронний ресурс] // Posit. – Режим доступу: https://posit.co/download/rstudio-desktop/ (дата звернення: 13.10.2025).

43. Cordon I. et al. Oversampling algorithms for imbalanced classification in R [Електронний ресурс]. – Режим доступу: https://sci2s.ugr.es/sites/default/files/bbvasoftware/publications/Imbalance.pdf (дата звернення: 13.10.2025).

44. Gandrud C. Reproducible Research with R and RStudio. 2nd edition [Електронний ресурс]. – Режим доступу: https://englianhu.wordpress.com/wp-content/uploads/2016/01/reproducible-research-with-r-and-studio-2nd-edition.pdf (дата звернення: 13.10.2025).

45. CRAN: The Comprehensive R Archive Network [Електронний ресурс]. – Режим доступу: https://cran.rstudio.com/ (дата звернення: 13.10.2025).

ДОДАТОК А

Фрагмент програмного коду

```
# Встановлення та завантаження потрібних пакетів -------------------------
packages <- c("dplyr", "ggplot2", "caret", "ROSE", "Amelia", "smotefamily", "gridExtra", "pROC")

installed <- packages %in% rownames(installed.packages())
if (any(!installed)) {
  install.packages(packages[!installed])
}

library(pROC)
library(ggplot2)
library(gridExtra)
library(dplyr)
library(caret)
library(ROSE)
library(Amelia)
library(smotefamily)

# 1. Завантаження даних ---------------------------------------------------
water_data <- read.csv("D:\\water_potability.csv")

cat("\n--- Структура даних ---\n")
str(water_data)
cat("\n--- Описова статистика ---\n")
summary(water_data)

# 2. Аналіз пропусків -----------------------------------------------------
cat("\n--- Кількість пропусків ---\n")
print(colSums(is.na(water_data)))
missmap(water_data, main = "Пропущені значення")

# 3. Обробка пропусків (медіана) ------------------------------------------
water_data <- water_data %>%
  mutate(across(everything(), ~ ifelse(is.na(.), median(., na.rm = TRUE), .)))

cat("\n--- Перевірка після заповнення ---\n")
print(colSums(is.na(water_data)))

# 4. Масштабування ознак --------------------------------------------------
data_scaled_water <- water_data %>%
  mutate(Potability = as.factor(Potability)) %>%
  mutate(across(-Potability, scale))

# Візуалізація розподілу ознак до/після масштабування (приклад для pH)
pH_raw_water <- data.frame(Value = water_data$ph, Type = "До масштабування")
pH_scaled_water <- data.frame(Value = data_scaled_water$ph, Type = "Після масштабування")
```

```
pH_compare_water <- rbind(pH_raw_water, pH_scaled_water)

ggplot(pH_compare_water, aes(x = Value, fill = Type)) +
  geom_histogram(alpha = 0.6, bins = 30, position = "identity") +
  facet_wrap(~Type, scales = "free") +
  labs(title = "Розподіл pH до та після масштабування",
       x = "Значення pH", y = "Кількість") +
  theme_minimal()

# 5. Аналіз дисбалансу ------------------------------------------------
cat("\n--- Розподіл класів (до балансування) ---\n")
print(table(data_scaled_water$Potability))

ggplot(data_scaled_water, aes(x = Potability)) +
  geom_bar(fill = "steelblue") +
  labs(title = "Розподіл класів (до балансування)")

# 6. Методи усунення дисбалансу ---------------------------------------
set.seed(123)

## 6.1. Undersampling
undersample_data_water <- downSample(
  x = data_scaled_water %>% select(-Potability),
  y = data_scaled_water$Potability
)
cat("\n--- Розподіл класів після undersampling ---\n")
print(table(undersample_data_water$Class))

## 6.2. Oversampling
oversample_data_water <- upSample(
  x = data_scaled_water %>% select(-Potability),
  y = data_scaled_water$Potability
)
cat("\n--- Розподіл класів після oversampling ---\n")
print(table(oversample_data_water$Class))

## 6.3. SMOTE
target_num_water <- as.numeric(as.character(data_scaled_water$Potability))

set.seed(123)
smote_out_water <- SMOTE(
  X = data_scaled_water %>% select(-Potability),
  target = target_num_water,
  K = 5,
  dup_size = 1
)

smote_data_water <- smote_out_water$data %>%
  rename(Potability = class) %>%
  mutate(Potability = as.factor(Potability))
```

```
# Робимо випадкове undersampling класу 1, щоб вирівняти до 1998
set.seed(123)
min_class_water <- smote_data_water %>% filter(Potability == "1") %>% sample_n(1998)
maj_class_water <- smote_data_water %>% filter(Potability == "0")

smote_balanced_water <- bind_rows(maj_class_water, min_class_water)

# Перевірка
table(smote_balanced_water$Potability)

## 6.4. ROSE
data_scaled_water <- water_data %>%
  mutate(Potability = as.factor(Potability)) %>%
  mutate(across(-Potability, ~ (scale(.) %>% as.numeric())))

set.seed(123)
rose_data_water <- ROSE(Potability ~ ., data = data_scaled_water, seed = 123)$data
cat("\n--- Розподіл класів після ROSE ---\n")
print(table(rose_data_water$Potability))

# 6.5. Функція для побудови ROC та обчислення AUC
get_roc_water <- function(data, target_col, model_label) {
  set.seed(123)
  # Навчання логістичної регресії
  formula <- as.formula(paste(target_col, "~ ."))
  model <- glm(formula, data = data, family = binomial)
  # Ймовірності позитивного класу ("1")
  probs <- predict(model, data, type = "response")
  # ROC-об'єкт
  roc_obj <- roc(data[[target_col]], probs, levels = rev(levels(data[[target_col]])))
  list(roc = roc_obj, auc = auc(roc_obj), label = model_label)
}

# Створюємо список ROC для всіх варіантів
roc_list_water <- list(
  get_roc_water(data_scaled_water, "Potability", "Первинні дані"),
  get_roc_water(undersample_data_water %>% rename(Potability = Class), "Potability",
"Undersampling"),
  get_roc_water(oversample_data_water %>% rename(Potability = Class), "Potability",
"Oversampling"),
  get_roc_water(smote_balanced_water, "Potability", "SMOTE"),
  get_roc_water(rose_data_water, "Potability", "ROSE")
)

# Побудова графіка
plot(roc_list_water[[1]]$roc, col = "black", lwd = 2, main = "ROC-криві для різних методів
балансування (Water)")
colors <- c("red", "blue", "green", "purple", "orange")
for (i in 2:length(roc_list_water)) {
  plot(roc_list_water[[i]]$roc, col = colors[i], lwd = 2, add = TRUE)
}
```

```
abline(a = 0, b = 1, lty = 2, col = "grey")

# Легенда
legend("bottomright",
       legend = sapply(roc_list_water, function(x) paste0(x$label, " (AUC=", round(x$auc,3), ")")),
       col = colors,
       lwd = 2)

# 6.6. Візуалізація розподілу класів після балансування
p1_water <- ggplot(undersample_data_water, aes(x = Class)) + geom_bar(fill = "red") +
ggtitle("Undersample")
p2_water <- ggplot(oversample_data_water, aes(x = Class)) + geom_bar(fill = "green") +
ggtitle("Oversample")
p3_water <- ggplot(smote_balanced_water, aes(x = Potability)) + geom_bar(fill = "blue") +
ggtitle("SMOTE")
p4_water <- ggplot(rose_data_water, aes(x = Potability)) + geom_bar(fill = "purple") +
ggtitle("ROSE")
grid.arrange(p1_water, p2_water, p3_water, p4_water, ncol = 2)

# 7. Поділ на тренувальну і тестову вибірки (для SMOTE) --------------------
set.seed(123)
train_index <- createDataPartition(smote_balanced_water$Potability, p = 0.7, list = FALSE)

train_data_water <- smote_balanced_water[train_index, ]
test_data_water  <- smote_balanced_water[-train_index, ]

cat("\n--- Розподіл у train ---\n")
print(table(train_data_water$Potability))

cat("\n--- Розподіл у test ---\n")
print(table(test_data_water$Potability))
```