МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри інтелектуальних інформаційних систем

_____Юрій КОНДРАТЕНКО

«____»_____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

ІНТЕЛЕКТУАЛЬНЕ ПРОГНОЗУВАННЯ НА ОСНОВІ БАГАТОШАРОВИХ АНСАМБЛЕВИХ СТРУКТУР

Спеціальність 122 Комп'ютерні науки

Освітня програма «Інтелектуальні інформаційні системи»

Здобувачка _____ Вікторія ПЕЩЕРЬОВА

«___»_____ 2024 р.

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

«___»_____ 2024 р.

Миколаїв – 2024

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

Завідувач кафедри інтелектуальних інформаційних систем

_____ Юрій КОНДРАТЕНКО

«____» _____ 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу здобувачки

Пещерьової Вікторії Сергіївни
(прізвище, ім’я, по батькові здобувачки)

1. Тема кваліфікаційної роботи: «Інтелектуальне прогнозування на основі багатошарових ансамблевих структур».

Керівник роботи: Калініна Ірина Олександрівна, в. о. професора кафедри ІІС, д-р техн. наук, доцент.

Затверджена наказом ЧНУ ім. Петра Могили від «03» червня 2024 р. № 140/1.

2. Строк представлення кваліфікаційної роботи «17» грудня 2024 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні: інтелектуальна система прогнозування на основі багатошарових ансамблевих структур; набір даних, що містить 9568 записів, зібраних з електростанції комбінованого циклу протягом 6 років, коли станція працювала на максимальному навантаженні.

4. Перелік питань, що підлягають розробці: проаналізувати поняття прогнозування, його типологію та процес; вивчити теоретичні основи інтелектуального прогнозування та сучасні підходи до підвищення точності прогнозів; поставити задачу інтелектуального прогнозування; проаналізувати регресійний аналіз як метод прогнозування, регресійні моделі, а також типологію ансамблевих структур; провести розвідувальний аналіз обраного набору даних та попередню обробку даних; побудувати базові моделі для прогнозування, сформувати багатошарові ансамблеві структури, оцінити та проаналізувати результати прогнозування.
5. Перелік графічних матеріалів: презентація.

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

Здобувачка _____ Вікторія ПЕЩЕРЬОВА
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата видачі завдання «07» червня 2024 р.

КАЛЕНДАРНИЙ ПЛАН
кваліфікаційної роботи

Тема: Інтелектуальне прогнозування на основі багатошарових ансамблевих структур

№	Найменування роботи	Початок	Закінчення	Примітки
1	Отримання завдання на виконання КР	03.06.2024	07.06.2024	Виконано
2	Аналіз предметної області та постановка задачі	10.06.2024	20.06.2024	Виконано
3	Огляд літературних джерел за темою кваліфікаційної роботи, зокрема аналіз поняття прогнозування як такого, інтелектуального прогнозування, актуальних підходів та досліджень в даних областях	21.06.2024	01.07.2024	Виконано
4	Детальний огляд поняття регресійного аналізу, регресійних моделей та типології ансамблевих структур	01.09.2024	25.10.2024	Виконано
5	Вибір набору даних, розвідувальний аналіз, попередня обробка, побудова базових моделей регресії, формування багатошарових ансамблевих структур, оцінка та аналіз результатів	26.10.2024	21.11.2024	Виконано
6	Перший попередній захист КР на засіданні комісії кафедри	22.11.2024	22.11.2024	Виконано
7	Корегування роботи за результатами попереднього захисту	23.11.2024	05.12.2024	Виконано
8	Другий попередній захист КР на засіданні комісії кафедри	06.12.2024	06.12.2024	Виконано
9	Доробка та остаточне оформлення КР	07.12.2024	10.12.2024	Виконано
10	Подання КР, її електронної копії та інших документів (відгуку, рецензії) до захисту	16.12.2024	17.12.2024	Виконано

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

Здобувачка _____ Вікторія ПЕЩЕРЬОВА
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата складання календарного плану
«19» червня 2024 р.

АНОТАЦІЯ

до кваліфікаційної роботи
здобувачки групи 601м ЧНУ ім. Петра Могили

Пещерьової Вікторії Сергіївни

на тему: **“ІНТЕЛЕКТУАЛЬНЕ ПРОГНОЗУВАННЯ НА ОСНОВІ БАГАТОШАРОВИХ АНСАМБЛЕВИХ СТРУКТУР”**

Актуальність. З розвитком новітніх технологій, як-от штучний інтелект та машинне навчання, питання інтелектуального прогнозування постає ще виразніше. Це пов’язано насамперед з тим, що точність і своєчасність прогнозів мають суттєве значення для прийняття рішень у численних сферах людської діяльності, наприклад, в економіці, медицині, екології, бізнесі, політиці тощо. А, отже, внаслідок такого прогресивного зростання попиту на інтелектуальне прогнозування, ця наукова галузь продовжує розвиватися, вдосконалюючись з кожною новою розробкою.

Однак інтелектуальне прогнозування далеко не бездоганне. Безумовно, успішні приклади застосування галузі існують, втім, завжди є місце для підвищення результатів, що вона демонструє. Одна з таких необхідностей в покращенні виникає при створенні інтелектуальних систем прогнозування. Ці системи часто вимагають оптимізації під актуальні дані, методи та моделі, а їхні прогнози, відповідно, потребують вдосконалення.

Потенційним розв’язком проблеми покращення таких систем можна вважати ансамблеві структури. Такі структури комбінують в собі декілька базових моделей прогнозування. Їхнє поєднання зменшує ризик недонавчання та перенавчання, пом’якшує компроміс між зміщенням та дисперсією. Чим кращий такий компроміс, тим стабільніше працює ансамбль, і тим точніше система прогнозує.

Об’єктом роботи є процес інтелектуального прогнозування на основі багатошарових ансамблевих структур.

Предмет роботи: методи, моделі інтелектуального аналізу даних, моделювання та прогнозування та їх комбінація в багатошарові ансамблеві структури в задачах прогнозування.

Мета: підвищення ефективності прогнозування за допомогою багатошарових ансамблевих структур.

В результаті виконання роботи було досліджено методи аналізу даних, регресійного та ансамблевого моделювання, прогнозування. Було проведено експериментальне дослідження їх ефективності на наборі даних, що дозволило визначити оптимальні комбінації моделей в ансамблях для підвищення точності прогнозів і пом'якшення компромісу між зміщенням та дисперсією.

Дана робота складається зі вступу, чотирьох розділів, висновків та додатків.

У першому розділі розглянуто теоретичні основи прогнозування.

У другому розділі описано регресійний аналіз і ансамблеві методи для підвищення точності.

У третьому розділі проведено розвідувальний аналіз даних і їх попередню обробку.

У четвертому розділі реалізовано базові регресійні моделі, багатошарові ансамблеві структури на їх основі та оцінено їх ефективність.

Загальний обсяг роботи – 114 сторінок.

Кваліфікаційна робота містить 1 додаток, 80 рисунків, 5 таблиць і 57 джерел посилання.

Ключові слова: *інтелектуальне прогнозування, базові прогнозні моделі, багатошарові ансамблеві структури, ефективність прогнозування, машинне навчання.*

ABSTRACT

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

Peshcherova Viktoriia

“INTELLIGENT FORECASTING BASED ON MULTILAYER ENSEMBLE STRUCTURES”

Relevance. With the development of new technologies, such as artificial intelligence and machine learning, the issue of intelligent forecasting is becoming even more important. This is primarily because the accuracy and timeliness of forecasts are essential for decision-making in numerous areas of human activity, such as economics, medicine, ecology, business, politics, etc. As a result of this progressive growth in demand for intelligent forecasting, this scientific field continues to evolve, improving with each new development.

However, intelligent forecasting is far from perfect. While there are certainly successful examples of the industry's applications, there is always room for improvement in the results it demonstrates. One such need for improvement arises when creating intelligent forecasting systems. These systems often need to be optimized for up-to-date data, methods, and models, and their forecasts must be improved accordingly.

Ensemble structures can be considered a potential solution to improving such systems. Such structures combine several basic forecasting models. Combining them reduces the risk of under- and overfitting and mitigates the trade-off between bias and variance. The better this trade-off is, the more stable the ensemble is and the more accurate the system is in its forecasting.

The object of the work is the process of intelligent forecasting based on multilayer ensemble structures.

The subject of the work: methods, data mining, modeling, forecasting models, and their combination into multilayer ensemble structures in forecasting tasks.

The purpose: is to improve the efficiency of forecasting using multilayer ensemble structures.

As a result of the work, the methods of data analysis, regression, ensemble modeling, and forecasting were investigated. An experimental study of their effectiveness on a dataset was conducted, which allowed to determine the optimal combinations of models in ensembles to improve forecast accuracy and mitigate the trade-off between bias and variance.

This paper includes an introduction, four chapters, conclusions, and appendices.

The first chapter discusses the theoretical foundations of forecasting.

The second chapter describes regression analysis and ensemble methods for improving accuracy.

The third chapter describes exploratory data analysis and data preprocessing.

The fourth chapter implements basic regression models, and multilayer ensemble structures based on them, and evaluates their effectiveness.

The total volume of the work is 114 pages.

The qualification work contains 1 appendix, 80 figures, 5 tables, and 57 references.

Keywords: *intelligent forecasting, basic forecasting models, multilayer ensemble structures, forecasting efficiency, machine learning.*

ЗМІСТ

ВСТУП

З розвитком новітніх технологій, як-от штучний інтелект та машинне навчання, питання інтелектуального прогнозування постає ще виразніше. Це пов’язано насамперед з тим, що точність і своєчасність прогнозів мають суттєве значення для прийняття рішень у численних сферах людської діяльності, наприклад, в економіці, медицині, екології, бізнесі, політиці тощо. А, отже, внаслідок такого прогресивного зростання попиту на інтелектуальне прогнозування, ця наукова галузь продовжує розвиватися, вдосконалюючись з кожною новою розробкою.

Однак інтелектуальне прогнозування далеко не бездоганне. Безумовно, успішні приклади застосування галузі існують, втім, завжди є місце для підвищення результатів, що вона демонструє. Одна з таких необхідностей в покращенні виникає при створенні інтелектуальних систем прогнозування. Ці системи часто вимагають оптимізації під актуальні дані, методи та моделі, а їхні прогнози, відповідно, потребують вдосконалення.

Потенційним розв’язком проблеми покращення таких систем можна вважати ансамблеві структури. Такі структури комбінують в собі декілька базових моделей прогнозування. Їхнє поєднання зменшує ризик недонавчання та перенавчання, пом’якшує компроміс між зміщенням та дисперсією. Чим кращий такий компроміс, тим стабільніше працює ансамбль, і тим точніше система прогнозує.

Тому дана кваліфікаційна робота присвячена дослідженню ефективності ансамблевих структур у контексті задачі інтелектуального прогнозування.

1 ПОНЯТТЯ ПРОГНОЗУВАННЯ. ІНТЕЛЕКТУАЛЬНЕ ПРОГНОЗУВАННЯ. АКТУАЛЬНІ ПІДХОДИ ТА ДОСЛІДЖЕННЯ. ПОСТАНОВКА ЗАДАЧІ

1.1 Поняття прогнозу. Загальна типологія. Прогнозування як процес

Терміни *«прогноз»*, *«передбачення»*, *«проєкція»* та *«прогнозування»* є взаємозамінними у загальному вжитку.

Перед тим, як прогнозувати, слід подумати, чи потрібно це робити. Прогнозування потрібне лише тоді, коли існує невизначеність; прогноз про те, що ситуація зміниться, не має жодної цінності. Прогнози також не потрібні, коли можна контролювати події. Наприклад, передбачення температури у домі не потребує прогнозування, оскільки людина сама може її контролювати. З усім тим, багато ситуацій є невизначеними, і належні процедури прогнозування можуть допомогти зменшити та оцінити невизначеність і, як підсумок, допомогти особі, яка приймає рішення (ОПР), приймати кращі рішення.

Прогноз, як зазначається в багатьох тлумаченнях, є передбаченням майбутніх подій або умов, що ґрунтується на аналізі історичних даних, поточних тенденцій та іншої релевантної інформації.

Типологія прогнозів визначається за різними критеріями, як-от масштаб, часові рамки, об’єкт дослідження, функції, методологія та інші характеристики.

За масштабом прогнозування виділяють:

– макроекономічні прогнози, що охоплюють загальнодержавні та національні економічні процеси;

– мікроекономічні прогнози, які стосуються діяльності окремих підприємств або організацій;

– галузеві, міжгалузеві та регіональні прогнози, що знаходяться між макро- та мікроекономічними рівнями.

За часовими рамками прогнози поділяються на:

– оперативні (до одного місяця), що застосовуються для швидкого реагування;

– короткострокові (до одного року), які слугують для квартального або річного планування;

– середньострокові (до трьох років), де враховуються як кількісні, так і якісні зміни;

– довгострокові (до п’яти років), орієнтовані на стратегічне планування;

– далекострокові (понад п’ять років), які враховують суттєві зміни в майбутньому.

За об’єктом прогнозування виділяють:

– науково-технічні;

– економічні;

– демографічні;

– методологічні;

– соціальні;

– політичні;

– військові прогнози та інші.

За функціонально-методологічною ознакою прогнози поділяються на:

– описувальні, які включають лише кількісні оцінки напрямів розвитку;

– досліджувальні (пошукові), побудовані на інерційних закономірностях розвитку;

– нормативні (цільові), спрямовані на досягнення конкретних результатів;

– комплексні, які об’єднують риси досліджувальних і нормативних прогнозів.

За частотою складання прогнози бувають:

– неперервні, що регулярно оновлюються;

– дискретні, розроблені на певні моменти часу.

За формою результатів розрізняють:

– детерміновані прогнози, які дають однозначне значення параметра;

– ймовірні прогнози, що враховують кілька можливих сценаріїв;

– змішані прогнози, які комбінують елементи детермінованих і ймовірних оцінок.

За ступенем точності прогнози поділяються на:

– точкові, що вказують одне конкретне значення;

– інтервальні, які дають діапазон можливих значень.

За можливістю впливу на процес прогнозування виділяють:

– активні прогнози, що передбачають можливість впливу на події;

– пасивні прогнози, які лише сприяють адаптації до бажаних умов.

Не можна не згадати поняття самого процесу прогнозування, адже прогноз є його таким собі очікуваним результатом.

Згідно з загальним трактуванням даного терму, ***прогнозування*** можна схарактеризувати як певний процес передбачення якоїсь майбутньої події за допомогою аналізу закономірностей і виявлення тенденцій як у минулих, так і у поточних даних. Іншими термами, що вживаються в даному контексті, є «метод прогнозування» та «модель прогнозування». Зрозуміло, що моделі та методи прогнозування пов’язані між собою, але мають ключову відмінність.

Термін *«метод прогнозування»* означає сукупність відповідних стратегій, які дозволяють робити певні прогнози щодо майбутнього розвитку процесу на основі аналізу інформації про нього. Як наслідок, завдяки методам, що є ширшим поняттям, можна визначити моделі прогнозування.

«Модель прогнозування» слугує основою процесу визначення майбутніх значень, і є його адекватним функціональним представленням.

На рис. 1.1 наведено загальну схему процесу прогнозування, що демонструє основні етапи та їхні взаємозв’язки у цій складній процедурі.

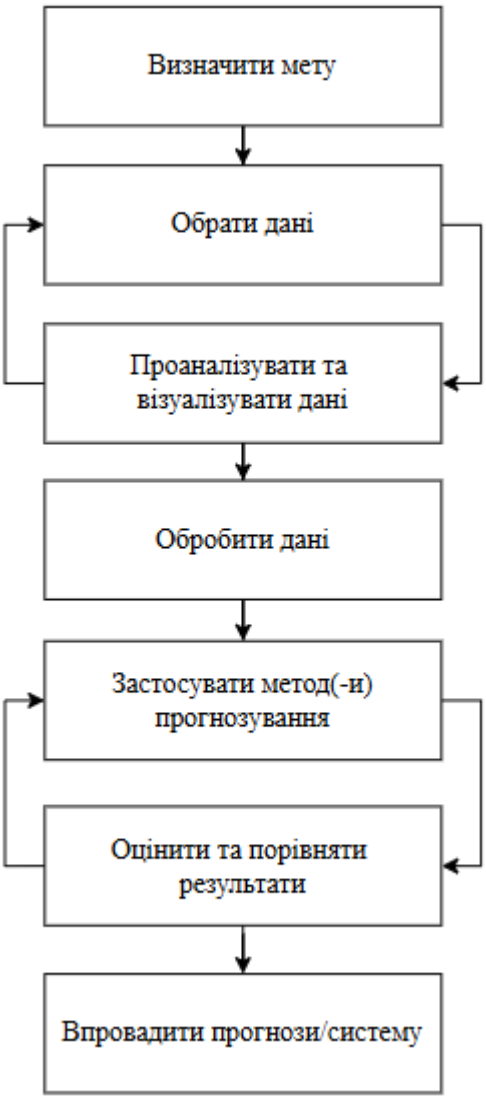

Рисунок 1.1 – Схема процесу прогнозування

Процес прогнозування завжди розпочинається з формулювання мети. Вона допомагає чітко зрозуміти, які питання потрібно вирішити, та які дані необхідні для цього. Треба враховувати, що саме слід прогнозувати: характеристики певного об’єкта, тенденції в даних або взаємозалежності між різними змінними. Тоді зроблені висновки дозволять обрати відповідні моделі та методи, налаштувати обробку даних відповідно до специфічних вимог. На додаток, наявність конкретної мети гарантує, що отримані результати внаслідок прогнозування будуть актуальними та корисними для кінцевих користувачів, компаній, організацій тощо.

На наступних етапах відбувається збір інформації, потім вона досліджується за допомогою інструментів візуалізації та обробляється.

По-перше, взагалі термін *«збір даних»* означає систематичне і структуроване збирання інформаційного матеріалу. Тут треба враховувати деякі аспекти. Важливо, щоб зібрані дані не лише відповідали меті дослідження, але й підтверджували свою якісність, що певним чином доводить їхню точність та правдивість. Збір даних охоплює використання джерел інформації, які містять уже наявні відомості, проведення опитувань для отримання первинних даних безпосередньо від людей, а також аналіз інших доступних матеріалів, як-от наявні статистичні показники з певних сфер діяльності, наукові статті, звіти та інтернет-ресурси.

По-друге, числова інформація не завжди дає повне уявлення про ключові взаємозв'язки в даних. Тут на допомогу приходять *інструменти візуалізації*, як-от графіки, діаграми та карти. Візуальне сприйняття часто призводить до розуміння даних загалом. Також воно сприяє глибшому усвідомленню структури й до того ж допомагає виявити потенційні проблеми, що потребують подальшої обробки, наприклад, викиди.

Нарешті, *обробка даних* є іншим важливим етапом у процесі підготовки інформації до моделювання. На цьому кроці дані піддаються перевірці на наявність помилок. Помилки зазвичай включають людський фактор, коли дані були введені некоректно; технічні збої, коли система неправильно спрацювала; або неякісні методи збору матеріалів. Процес очищення зазвичай охоплює кілька основних стадій. Спочатку відбувається заповнення відсутніх значень, усунення викидів, видалення дублікатів. Потім чиста інформація піддається стандартизації та нормалізації. Унаслідок цього, правильна обробка запобігає спотворенню результатів, адже невірогідна інформація призводить до хибних висновків і неправильних рішень.

Після етапу вище, який ще можна схарактеризувати як препроцесінг даних, на основі характеру даних обирається відповідний *набір потенційних методів прогнозування*. Дуже часто вибір алгоритму заснований на тому, наскільки добре

він підходить для набору даних і самого завдання прогнозування. Потім на основі обраного методу будуються відповідні моделі.

Етапи оцінки та порівняння результатів засновані на оцінках точності прогнозів з використанням показників якості моделі. Наприклад, це оцінки, як-от середня абсолютна похибка (англ. MAE, Mean Absolute Error), корінь середньоквадратичної похибки (англ. RMSE, Root Mean Square Error) тощо. Потім моделі, що показали непогані результати, порівнюються та аналізуються для того, щоб визначити ефективнішу.

Етап впровадження прогнозів передбачає практичне впровадження отриманих прогнозів кращою обраною моделлю на попередньому етапі. На цьому моменті модель прогнозування інтегрується в чинну систему прийняття рішень та використовується кінцевим користувачем.

Зі схеми видно, що деякі частини процесу є ітеративними. Наприклад, після дослідження даних можна визначити, чи дійсно отримана інформація призведе до успішного досягнення поставлених цілей. У випадку, якщо ні, то процес повторюється, збирається нова або додаткова інформація.

Другий ітеративний процес відбувається при виборі методів прогнозування, побудові моделей та доведенні їхньої ефективності. Останнє переважно призводить до доопрацювання, адаптації методів і моделей, або навіть до випробування інших підходів.

1.2 Теоретичне підґрунтя інтелектуального прогнозування

Десятиліттями прогнозування покладалось на статистичні моделі та аналіз часових рядів для передбачення результатів. Ці методи, попри їхню цінність, здебільшого обмежувалися історичними даними й часто не могли врахувати зовнішні фактори та складні взаємозв’язки. Поява великих даних і штучного інтелекту проклала шлях до нової ери прогнозування, яка заглиблюється більше, прогнозує точніше і відкриває неоціненні інсайти.

Інтелектуальне прогнозування зазвичай характеризує процес інтелектуального передбачення майбутніх подій за допомогою новітніх технологій ШІ та МН. Дана галузь використовує складні алгоритми, здатні виявляти приховані закономірності та зв'язки. Влучні приклади релевантності застосування інтелектуального прогнозування обґрунтовано нижче.

Наприклад, у науковому дослідженні [1] автори показали, що застосування інтелектуальних методів для прогнозування банкрутства має значну практичну цінність. Як зазначають дослідники, питання банкрутства завжди вважалось актуальним через таку собі наявність чисельних методів і технік. Такі підходи дають змогу з різним ступенем точності його спрогнозувати. Однак галузь дуже часто стикається з низкою проблем. Наприклад, браком методів, що здатні точно передбачати під час мінливого стану, – глобальної фінансової кризи. Втім, як довели автори статті, одним із перспективних підходів до розв'язання попередньо вказаної проблеми є розробка саме інтелектуальних моделей, які зможуть усунути недоліки наявних і врахувати їхню відповідну специфіку. Дослідниками було створено математичну модель нейронної мережі (НМ) для прогнозування банкрутства країн Азії на основі десяти вхідних факторів. Впроваджена НМ показала досить високі результати й підтвердила свою ефективність.

Інтелектуальне прогнозування докорінно змінює процеси ухвалення рішень та демонструє перевагу над традиційними методами прогнозування. Як приклад, дослідники Ілан Алон, Мін Ци та Роберт Садовський у своїй статті [2] досліджували результативність різних моделей прогнозування часових рядів. Сукупні роздрібні продажі прогнозувались поза вибіркою на щомісячній основі за допомогою штучної НМ та трьох традиційних статистичних методів. Для різних періодів та горизонтів прогнозування найкраще себе зарекомендував метод штучної НМ, за яким в значенні якості слідували експоненціальне згладжування Бокса-Дженкінса та Вінтерса. Множинна регресія з трендовими та сезонними фіктивними змінними показала найгірші результати. Штучна НМ перевершила традиційні статистичні методи в першому періоді, коли економічні умови були

відносно нестабільними. Коли макроекономічні умови були відносно стабільними, моделі експоненціального згладжування Бокса-Дженкінса та Вінтерса продемонстрували також більш-менш життєздатні результати. Автори вважають, що за нестабільних макроекономічних умов перевагу слід надавати інтелектуальним методам, оскільки нові дані можуть не додати багато корисної інформації до традиційної моделі прогнозування. Нарешті, продемонстровані графіки похідних показали, що штучна НМ своєю чергою здатна була вловити динамічний нелінійний тренд, сезонні закономірності та їхню взаємодію. Що стосується кількісних показників, то середня абсолютна похибка у відсотках (англ. MAPE, Mean Absolute Percentage Error) цієї нейронної мережі становила лише 1.50%.

Інтелектуальне прогнозування до того ж демонструє високу ефективність у роботі з великими даними (Big Data), які отримують із соціальних мереж, інтернету речей, фінансових транзакцій та інших джерел. У своєму дослідженні [3] Омар Саад, Юнфен Чен та ін. розробили систему прогнозування землетрусів у реальному часі й протестували її в сейсмічних регіонах на південному заході Китаю. Вхідними даними були характеристики, що надавались багатокомпонентною системою сейсмічного моніторингу акустико-електромагнітного штучного інтелекту AETA, в якій дані реєструвались за допомогою двох типів датчиків на кожній станції: електромагнітних (ЕМ) і геоакустичних (ГА). Загальною метою було прогнозування місця та магнітуди землетрусу, який може статися наступного тижня, враховуючи дані поточного тижня. Запропонований авторами метод базувався на зменшенні розмірності Big Data ЕМ і ГА з використанням аналізу головних компонент, за яким слідувала класифікація на основі випадкового лісу. Запропонований алгоритм навчався на доступних даних з 2016 по 2020 рік і оцінювався на Big Data в реальному часі протягом 2021 року. В результаті точність тестування досягла 70%. Середня абсолютна похибка відстані та прогнозованої величини за допомогою запропонованого методу становила 381 км та 0.49 відповідно, що є дуже гарним результатом.

У статті [4] автори показали, що точність методів на основі ШІ для прогнозування траєкторії поширення Covid-19 була доволі високою. Вони спрогнозували криві кумулятивних підтверджених випадків Covid-19 по всьому Китаю з 20 січня 2020 року по 20 квітня 2020 року. Використовуючи багатокрокове інтелектуальне прогнозування, дослідники отримали оцінювальні середні помилки 6-крокового, 7-крокового, 8-крокового, 9-крокового та 10-крокового прогнозування, що становили 1.64%, 2.27%, 2.14%, 2.08%, 0.73%, відповідно. Результати дійсно вражають.

Безперечно, доволі багато й інших досліджень у різноманітних сферах людської діяльності використовують методи інтелектуального прогнозування для покращення процесу прийняття рішень та оптимізації операцій. Приміром:

– у *фінансовому секторі* для оцінки кредитних ризиків, прогнозування фінансової волатильності [5-6] тощо;

– у *секторі ланцюгів постачання* для їх управління або прогнозування попиту [7-8];

– у *галузі охорони здоров'я* для передбачення поставлення ліків або бронювання медичних послуг [9-10];

– у *прогнозуванні погоди* [11-12], де інтелектуальні моделі вивчають супутникові дані, атмосферні умови та історичні погодні тенденції;

– в *енергетичному секторі* для оцінки споживання енергії [13], наприклад.

Отже, нині інтелектуальне прогнозування є потужним інструментом, який перевершує традиційні статистичні методи та революціонізує цілі сфери діяльності людини. Як доводять проаналізовані дослідження, його впровадження допомагає ефективно підвищувати точність прогнозів, що є справді критично важливим завданням у динамічних умовах світу технологічних відкриттів.

1.3 Сучасні підходи до підвищення ефективності прогнозування

Інтелектуальні методи прогнозування набувають все більшої популярності, як було доведено попередньо. Вони знаходять застосування у різних секторах, як-

от передбачення макроекономічних змінних [14], аналіз бухгалтерського балансу [15], прогнозування руху фондового ринку [16], а також фінансових часових рядів [17].

Однак, як і завжди, при такому великому колі застосування інтелектуального прогнозування, постає проблема повсякчасного покращення його якості. В цьому плані дослідники використовують ***низку методів*** [18] для підвищення точності вхідних даних моделі та результатів прогнозування загалом. Вони продемонстровані нижче.

1. Методи *коригування зсувів* вважаються доволі ефективним підходом покращення якості прогнозів, адже вони здатні усунути упередженість моделі. Наприклад, автори статті [19] оцінювали метод корекції похибок чисельного прогнозу погоди для короткострокових прогнозів опадів за допомогою моделі XGBoost. Дослідники порівняли три створені структури: EDCDFm (M1), базову модель XGBoost (M2) та XGBoost з багатофакторною корекцією (M3). Метод M3, як наслідок, показав найкращі результати, з найменшими значеннями RMSE. Своєю чергою Вей Чжан, Юеюе Цзян та ін. [20] продемонстрували власний модифікований метод коригування зсувів, який ґрунтувався на тому, що створена ними структура MT-DETrajGRU аналізувала і виправляла похибки у середньострокових прогнозах погоди, забезпечуючи більш точні прогнози під час різних погодних умов, включаючи нормальні умови та тайфуни.

2. *Злиття даних* – це ще один з методів підвищення ефективності прогнозування, він означає процес об’єднання різних джерел даних для створення інформації, яка є більш надійною, точною і практичною, ніж та, яку можна отримати лише з одного джерела даних. Принц Адуама та ін. [21] запропонували техніку прогнозування навантаження зарядних станцій для електромобілів з використанням мультифакторного злиття даних для покращення точності моделі глибокого навчання на основі довгої короткочасної пам’яті (англ. LSTM, Long Short-Term Memory). На відміну від традиційних моделей LSTM, які робили один прогноз, новий метод здійснював три прогнози на основі різних мультифакторних

входів (вітер, температура, вологість) та використовував злиття даних для їхньої оптимізації. Результати показали, що похибка прогнозування зменшилася до 3.29%, що є покращенням порівняно з початковими результатами простої моделі LSTM. Автори статті [22] виконали поєднання методів глибинної символьної регресії та ансамблевого оптимального інтерполяційного засвоєння даних для коригування похибок прогнозів числової моделі WaveWatch III. Злиття даних у цьому дослідженні полягало в інтеграції прогнозів із системи Global Forecast System, вимірювань із супутників Jason-2 та SARAL, а також даних від буйків. Це дозволило дослідникам створити точніші символічні рівняння для коригування прогнозів числової моделі, що своєю чергою призвело до зменшення середньоквадратичного відхилення для прогнозів до 42 годин при використанні 12-денного циклу засвоєння даних.

3. Хізер Вандер, Квінн Томас та ін. [23] вивчали вплив частоти *асиміляції даних,* як метода покращення ефективності прогнозування, на точність прогнозів температури води в евтрофному водосховищі. Використовуючи різні частоти асиміляції (щодня, щотижня, раз на два тижні й раз на місяць), дослідники порівняли ефективність прогнозів на 1-35 днів вперед. Виявилось, що щоденна асиміляція забезпечувала найкращі результати на коротших горизонтах прогнозування (1-7 днів); щотижнева асиміляція демонструвала кращі прогнози на довших горизонтах (8-35 днів). Загалом, точність прогнозів була високою з середніми значеннями RMSE 0.81°C на 1 день, 1.15°C на 7 днів і 1.94°C на 35 днів. Результати показали, що для деяких застосувань асиміляція даних з меншою частотою може бути достатньою, що дозволяє розробляти прогнози без необхідності у високочастотних сенсорах.

4. *Послідовні методи* для підвищення ефективності прогнозування базуються на найкращій лінійній незміщеній оцінці (англ. BLUE, Best Linear Unbiased Estimator), яка дозволяє знайти хороший компроміс між набором спостережень і попередньою інформацією про стан системи, наприклад, попереднім прогнозом. Такі методи підвищення точності передбачають оновлення

оцінок в міру надходження нових даних. Для прикладу, до них належать оптимальна інтерполяція, фільтр Калмана та його різновид, ансамблевий фільтр Калмана. У своєму дослідженні Мейлінг Ченг, Фансін Фан та ін. [24] використовували МН для прогнозування нелінійних динамічних систем на довгий термін. Для зменшення невизначеності в довгострокових прогнозах авторами було введено ансамблевий фільтр Калмана в моделі МН. Поєднуючи моделі з ним, дослідники змогли підвищити точність довгострокових прогнозів динамічних систем. Результати показали, що фільтр ефективно коригував похибки й значно покращував реальний час прогнозування динамічних систем на довгий термін.

5. Метод BLUE можна замінити іншим методом покращення якості прогнозів, *аналізом 3D-Var*. Це дозволяє зробити етап аналізу *варіаційним*, що забезпечує мінімізацію функції витрат із природним і точним врахуванням нелінійного оператора спостереження. У дослідженні Іоанніса Самоса, Петрули Луки та ін. [25] метод 3D-Var використовувався для інтеграції даних у модель, і різні конфігурації оцінювалися з метою підвищення точності прогнозів погоди, особливо в умовах складного рельєфу.

6. *Моделювання Монте-Карло (МК)* полягає у проведенні низки симуляцій зі збуреними вхідними даними. Збурення вибираються відповідно до розподілів ймовірностей, які задає дослідник, причому збурення, пов’язані з двома симуляціями, повинні бути незалежними. Луїс-Феліпе Дуке, Енда О’Коннел та ін. [26] досліджували переваги й недоліки ймовірнісних і детерміністичних стратегій попередження про повені. Використовуючи аналіз чутливості на основі методу МК, автори вивчили, як результати попереджень про рівень води під час повені змінюються при відхиленнях якості прогнозів. Було виявлено, що МК-стратегія мала перевагу над детерміністичною, коли прогнозна невизначеність висока, оскільки вона краще зберігала стабільність показників точності та кількість помилкових сигналів. Дослідники в статті [27] використовували метод МК для ймовірнісного прогнозування вітрової енергії. Оскільки швидкі та випадкові коливання швидкості вітру ускладнювали точне прогнозування, вони застосували

МК на завершальному етапі після створення та навчання шаблонів даних. Метод дозволив виконати велику кількість симуляцій для отримання розподілу можливих результатів, що допомогло оцінити не лише середнє значення прогнозованої потужності, але й імовірнісні межі. Це підвищило надійність і точність прогнозів, що є особливо важливим в умовах високої невизначеності генерованої енергії.

7. *Ансамблеве прогнозування* – це стратегія, яка поєднує в собі дві або більше моделей прогнозування для створення єдиного, найкращого прогнозу. Для прикладу, у дослідженні [28] автори використали ансамблі для підвищення точності короткострокових прогнозів температури. Структури ансамблів формувалися шляхом узагальнення прогнозів згорткових нейронних мереж, зворотних нейронних мереж і чисельних моделей. Завдяки інтеграції сильних сторін кожної з цих моделей, ансамблі демонстрували вищу точність прогнозів. Ансамблі в дослідженні Цзя Ван, Сюй Ван та ін. [29] використовувались для покращення прогнозування стоку через поєднання кількох різних моделей. Об’єднання НМ та методу опорних векторів, допомогло зменшити загальну похибку прогнозу та невизначеність, що зробило прогнозування якіснішим. Науково обґрунтованою причиною таких успішних результатів є загальна властивість ансамблів зменшувати ризик недонавчання та перенавчання, та пом’якшувати компроміс між зміщенням та дисперсією [30]. Не можна не згадати, що існує можливість поєднувати різні ансамблі моделей в шаруваті структури – багатошарові ансамблі. Наприклад, у статті [31] автори представили ансамбль з багатошаровою класифікацією з використанням посиленого бегінгу та оптимізованого зважування на різних рівнях для прогнозування серцевих хвороб. Аналіз результатів показав, що багатошаровий ансамбль досяг найвищої точності, чутливості та F-міри у порівнянні з окремими класифікаторами.

1.4 Постановка задачі інтелектуального прогнозування

Згідно з попереднім аналізом сучасного стану досліджень у сфері інтелектуального прогнозування, одним з ефективних способів покращення якості

прогнозів, який можна виділити, є використання ансамблевих структур. Такі структури комбінують в собі декілька базових моделей, поєднання яких зменшує ризик недонавчання та перенавчання, пом'якшує компроміс між зміщенням та дисперсією. Своєю чергою багатошарові ансамблеві структури передбачають організацію ансамблів у декілька рівнів, де результати моделей попереднього шару використовуються як вхідні дані для моделей наступного шару. Оскільки багато прикладів таких складних алгоритмів стосуються в основному задачі класифікації, цікаво прослідкувати їх поведінку в контексті регресійного прогнозування.

Тому, твердження вище дає можливість сформулювати, що ***об'єктом даної роботи*** є процес інтелектуального прогнозування на основі багатошарових ансамблевих структур.

Предмет роботи: методи, моделі інтелектуального аналізу даних, моделювання та прогнозування та їхня комбінація в багатошарові ансамблеві структури в задачах прогнозування.

Мета: підвищення ефективності прогнозування за допомогою багатошарових ансамблевих структур.

Для досягнення цієї мети було визначено ***низку етапів***, які необхідно виконати.

На *першому етапі* передбачається ретельний вибір і підготовка вхідних даних для моделювання. Важливо вибрати релевантний набір даних, що відповідає поточним проблемам певної сфери людської діяльності. Дані повинні пройти детальний розвідувальний аналіз, який включає перевірку якості, а також оцінку кореляцій між змінними. На кроці препроцесінгу здійснюється обробка пустих значень, викидів, дублікатів, та виконуються нормалізація, стандартизація, трансформація змінних і відбір ознак. Дані повинні бути коректно розподілені на тренувальні та тестові набори.

Другий етап полягає у виборі метрик для оцінки моделей. Їх правильний вибір дає змогу не лише порівняти моделі між собою, але й оптимізувати стратегії формування ансамблів.

На *третьому етапі* здійснюється вибір базових регресійних моделей, які стануть основою для побудови ансамблевих структур. Кожна модель проходить етапи тренування та тестування на попередньо підготовленому наборі даних. Моделі згодом оцінюються за обраними метриками.

Четвертий етап зосереджений на проєктуванні багатошарових ансамблевих структур. Цей крок є експериментальним, і поєднання моделей на різних рівнях підбирається згідно зі специфікою даних. Успішність ансамблю залежить від здатності моделей ефективно взаємодіяти та не корелювати між собою. Тренування таких архітектур проводиться з використанням тренувального набору даних, після чого тестується на незалежному наборі для оцінки узагальнювальної здатності ансамблю за відповідними метриками.

На *фінальному етапі* відбувається аналіз отриманих результатів. Особлива увага приділяється інтерпретації підсумків моделювання, виявленню сильних і слабких сторін підходу, а також пошуку можливостей для подальшого вдосконалення. Ретельний аналіз дозволяє сформулювати рекомендації щодо покращення якості прогнозування, зокрема через модифікацію архітектури або застосування нових методів обробки даних.

Як підсумок, задача інтелектуального прогнозування з використанням багатошарових ансамблевих структур успішно поставлена.

Висновки до розділу 1

У першому розділі було проведено всебічний аналіз інтелектуального прогнозування, дослідження та підходи в даній сфері.

Визначено, що процес прогнозування є багатогранним і складається з етапів, які включають збір даних, їх обробку, аналіз, побудову прогнозних моделей, оцінку результатів та впровадження прогнозів.

Детально розглянуто концепцію інтелектуального прогнозування, яка включає використання алгоритмів МН та ШІ для підвищення якості прогнозів.

Проведено огляд та аналіз сучасних підходів до підвищення точності інтелектуального прогнозування. В багатьох наукових роботах зустрічається низка таких методів: методи коригування зсувів, злиття даних, асиміляція даних, послідовні методи, 3D-Var, моделювання Монте-Карло та ансамблеве прогнозування.

Визначено основні аспекти задачі інтелектуального прогнозування на основі багатошарових ансамблевих структур та етапи для її успішного розв’язання.

Як підсумок, цей розділ закладає основу для майбутнього дослідження і розробки.

2 РЕГРЕСІЙНИЙ АНАЛІЗ. РЕГРЕСІЙНІ МОДЕЛІ В ЗАДАЧАХ ПРОГНОЗУВАННЯ. ТИПОЛОГІЯ АНСАМБЛЕВИХ СТРУКТУР

2.1 Поняття регресійного аналізу. Регресійні моделі в контексті задач прогнозування

Регресійний аналіз – це статистичний метод аналізу взаємозв'язку між двома або більше змінними так, що одна зі змінних може бути передбачена або пояснена за допомогою інформації про інші. Для успішного застосування регресійного аналізу необхідно чітко розрізняти ролі двох кількісних змінних. Та, яку необхідно передбачити, або яка зазнає впливу, називається залежною, змінною відгуку або результату, тоді як інша, яка використовується для прогнозування або викликає зміни, називається незалежною, пояснювальною або змінною-предиктором. Традиційно першу позначають як Y, а останню – X.

Основна ціль регресійного аналізу лежить саме в побудові функції, яка точно відображає залежність між змінними та дозволяє прогнозувати значення залежної на основі значень незалежних.

Регресійна модель є функціональним представленням регресійного аналізу. У контексті прогнозного моделювання регресійні моделі не лише забезпечують високу точність прогнозів, але й надають можливість визначити чинники, що мають найвагоміший вплив на прогнозовану змінну. Підтвердженням того, що такі моделі виступають важливим інструментом у прогнозуванні, є їх застосування в різноманітних сферах діяльності, зокрема в догляді за хворими [32], прогнозуванні захворювань [33], фінансах та діяльності фірми [34] тощо.

Однак перед побудовою регресійної моделі важливо впевнитися, що існує зв'язок між змінними, які є предметом дослідження. Це не обов'язково означає причинно-наслідковий зв'язок, але має бути наявна певна залежність між ними. Для візуалізації цього зв'язку часто використовують діаграму розсіювання, яка може вказати на наявність або відсутність тенденцій між змінними. Якщо діаграма не вказує на якусь тенденцію, ймовірно, модель не буде ефективною для опису

даних. Важливим числовим показником цього зв'язку є коефіцієнт кореляції, який набуває значень від -1 до 1 і вказує на силу зв'язку між змінними.

Після підтвердження наявності такого зв'язку основою для побудови регресійних моделей стають методи оцінки параметрів. Зазвичай використовують два таких способи. Перший – це метод найменших квадратів [35]. Інший метод оцінки параметрів – метод максимальної правдоподібності [36]. Перший зазначений підхід зводиться до мінімізації суми квадратів різниць між спостережуваними значеннями та передбаченими моделлю. Своєю чергою метод максимальної правдоподібності орієнтований на знаходження таких значень параметрів, за яких ймовірність отримання спостережуваних даних є максимальною.

Однією з найпростіших форм лінійних регресійних моделей є ***проста лінійна регресійна модель***. Вона базується на тому, що існує певна вхідна змінна *X*, яка впливає на вихідну *Y*, причому цей вплив є лінійним [37]. Проста лінійна регресійна модель описується даним рівнянням:

$$Y = \alpha_0 + \alpha_1 X + \varepsilon, \tag{2.1}$$

де Y – результативна змінна;

α_0 – постійний член або константа;

α_1 – коефіцієнт моделі;

X – незалежна змінна;

ε – випадкова похибка.

Дане рівняння моделі показує, як змінюється залежна змінна *Y* залежно від значення незалежної *X* в ній. Константа α_0 представляє початкове значення *Y*, коли *X* дорівнює нулю. Коефіцієнт α_1 показує, наскільки змінюється *Y* при зміні *X*. Останній компонент, ε, – це випадкова похибка, яка враховує вплив інших факторів, не включених у модель.

Інший вид лінійних регресійних моделей, ***множинна регресійна модель***, використовується, коли вихідна змінна *Y* піддається одночасному впливу кількох вхідних $X_1,...,X_n$. У таких випадках прогнозування здійснюється за допомогою рівняння, яке має наступний вигляд:

$$Y = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + ... + \alpha_n X_n + \varepsilon, \quad (2.2)$$

де Y – результативна змінна;

α_0 – постійний член або константа;

$\alpha_1,...,\alpha_n$ – своєрідні коефіцієнти моделі;

$X_1,...,X_n$ – незалежні змінні;

ε – випадкова похибка.

Це рівняння показує, що дійсно вихідна змінна *Y* залежить від кількох незалежних $X_1,...,X_n$. Константа α_0 відображає початкове значення *Y*, коли всі незалежні змінні дорівнюють нулю. Кожен коефіцієнт $\alpha_1,...,\alpha_n$ показує, як змінюється *Y* при зміні відповідної незалежної змінної $X_1,...,X_n$, при фіксованих значеннях інших. Випадкова похибка ε враховує вплив факторів, які не включені в модель.

Моделі множинної лінійної регресії часто використовуються як емпіричні моделі або апроксимаційні функції. Тобто справжня функціональна залежність між *Y* та $X_1,...,X_n$ невідома, але в певних діапазонах змінних регресорів лінійна регресійна модель є адекватним наближенням до справжньої невідомої функції.

Моделі, які мають складнішу структуру, ніж рівняння (2.2), можуть бути все одно проаналізовані з використанням методів множинної лінійної регресії. Для прикладу, ***множинна регресійна модель з ефектом взаємодії факторів*** – це статистична модель, яка дозволяє оцінити, як взаємодія між двома або більше незалежними змінними впливає на залежну змінну [38]. Така модель корисна, коли є потреба впевнитись, чи ефект одного фактора на результат змінюється залежно від рівня іншого.

Рівняння моделі таке:

$$Y = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \alpha_3 (X_1 \cdot X_2) + \varepsilon, \quad (2.3)$$

де Y – результативна змінна;

α_0 – постійний член або константа;

$\alpha_1,...,\alpha_n$ – своєрідні коефіцієнти моделі;

X_1, X_2 – незалежні змінні;

$X_1 \cdot X_2$ – взаємодія змінних;

ε – випадкова похибка.

У цій моделі додатковий компонент $X_1 \cdot X_2$ відображає взаємодію між двома факторами X_1, X_2. Тобто ефект одного фактора на залежну змінну *Y* може змінюватися відповідно до значення іншого фактора. Така взаємодія необхідна, якщо важливо врахувати не лише незалежний вплив кожної змінної на *Y*, але й те, як вони разом можуть змінювати результат.

У більшості реальних задач значення параметрів (коефіцієнти регресії $\alpha_1,...,\alpha_n$) невідомі, і їх потрібно оцінювати на основі вибіркових даних. Підібране рівняння регресії або модель зазвичай використовується для прогнозування майбутніх спостережень змінної відгуку *Y* або для оцінки його середнього значення при певних рівнях.

Нелінійні регресійні моделі базуються на тому, що існує певна функція, яка описує залежність між вихідною змінною *Y* і вхідною *X*.

Тобто рівняння таке:

$$Y = F(X, f), \quad (2.4)$$

де Y – результативна змінна;

F(X, f) – нелінійна функція;

X – незалежна змінна;

f – параметри функції *F*.

Отже, рівняння моделі точно показує залежність між змінною відгуку *Y* і незалежною *X* за допомогою нелінійної функції *F*. Зв'язок між *Y* і *X*, відповідно, має певну складну форму. Функція *F* своєю чергою визначає, як саме значення незалежної змінної *X* впливає на *Y*, при цьому використовуючи певні параметри *f*, які потрібно оцінити.

Проте на практиці рідко зустрічаються випадки, для яких форма функціональної залежності між вихідною змінною *Y* і вхідною *X* відома заздалегідь. Через це нелінійна регресія використовується нечасто.

Нелінійна регресія охоплює різні типи моделей, які відображають зв'язки між змінними нелінійним чином. Найпоширеніші види наведено нижче.

Поліноміальна регресійна модель – це модель, в якій зв'язок між незалежною змінною *X* і залежною *Y* моделюється у вигляді полінома *n*-го степеня від *X*:

$$Y = \alpha_0 + \alpha_1 X + \alpha_2 X^2 + \alpha_3 X^3 + ... + \alpha_n X^n + \varepsilon, \quad (2.5)$$

де Y – результативна змінна;

α_0 – постійний член або константа;

$\alpha_1,...,\alpha_n$ – своєрідні коефіцієнти моделі;

X – незалежна змінна;

ε – випадкова похибка.

Отже, рівняння поліноміальної регресії складається з константи α_0, що представляє початкове значення *Y* при $X = 0$, лінійного члена $\alpha_1 X$, квадратичного члена $\alpha_2 X$, кубічного $\alpha_3 X$ і так далі до *n*-го степеня. Кожен коефіцієнт показує, наскільки зміна відповідного степеня *X* впливає на результат *Y*. Похибка ε враховує випадкові або невраховані фактори.

Експоненціальна регресійна модель – це тип нелінійної регресійної моделі, який підганяє дані під експоненціальну функцію. Вона використовується при моделюванні сценаріїв, коли зростання функції починається повільно, а потім

набирає швидкість, або коли спад функції починається швидко, а потім наближається майже до нуля. Загальна форма моделі експоненціальної регресії виглядає наступним чином:

$$Y = \alpha e^{(\beta X)} + \varepsilon, \tag{2.6}$$

де Y – результативна змінна;

α, β – своєрідні коефіцієнти моделі;

X – незалежна змінна;

ε – випадкова похибка.

Логарифмічна регресійна модель – це тип нелінійної регресійної моделі, який підбирає логарифмічну функцію до даних. Це ситуації, коли зростання функції або її падіння спочатку швидко збільшується, а потім поступово сповільнюється. Загальною формою логарифмічної регресійної моделі є наведене нижче рівняння:

$$Y = \alpha + \beta \ln(X) + \varepsilon, \tag{2.7}$$

де Y – результативна змінна;

α, β – своєрідні коефіцієнти моделі;

X – незалежна змінна;

ε – випадкова похибка.

Степенева регресійна модель – це тип нелінійної регресійної моделі, яка підбирає степеневу функцію до ряду даних. Загальна форма моделі степеневої регресії така:

$$Y = \alpha X^{\beta} + \varepsilon, \tag{2.8}$$

де Y – результативна змінна;

α, β – своєрідні коефіцієнти моделі;

X – незалежна змінна;

ε – випадкова похибка.

Узагальнена адитивна модель – це тип нелінійної регресійної моделі, який поєднує декілька лінійних моделей для моделювання непростих взаємозв'язків між змінними. Узагальнені адитивні моделі дуже корисні при аналізі складних даних, які відображають нелінійні закономірності, як-от часові ряди та просторові дані, або коли зв'язки між предикторами та змінною відгуку важко описати простими лінійними функціями. Така модель є лінійною комбінацією одновимірних компонентних функцій [39]:

$$g(E(Y)) = \alpha_0 + f_1(X_1) + f_2(X_2) + ... + f_n(X_n), \quad (2.9)$$

де $g(E(Y))$ – функція зв'язку;

$E(Y)$ – середнє значення;

Y – результативна змінна;

α_0 – постійний член або константа;

$f_i(X_i)$ – функції із заданою параметричною формою;

$X_1,...,X_n$ – незалежні змінні.

Отже, модель пов'язує одновимірну залежну змінну, *Y*, з деякими предикторними змінними $X_1,...,X_n$. Для *Y* задається розподіл разом із функцією зв'язку *g()* (наприклад, тотожність або лог-функція), що пов'язує очікуване значення *Y* зі змінними предикторами [40].

Функції $f_i()$ представлені методами із заданою параметричною формою. Вони також можуть бути репрезентовані непараметрично чи напівпараметрично, тобто просто як «гладкі функції», які оцінюються непараметричними методами. Наприклад, адитивні моделі часто мають такі модифікації:

– *модель зі сплайнами* [41], що використовує сплайни для створення плавних кривих, які можуть адаптуватися до даних, зменшуючи відхилення від фактичних значень;

– *поліноміальна модель*, що моделює залежності через поліноміальні функції, підносячи предиктори до певних степенів;

– *модель з локальними регресійними функціями*, що використовує локальне згладжування для адаптації до відповідних структур у даних. Наприклад, функція LOESS [42].

Регресійна модель на основі дерева рішень менш популярна, ніж класифікаційні дерева, але, попри це, залишається надзвичайно конкурентоспроможною в порівнянні з іншими алгоритмами, та часто застосовується для розв'язання практичних задач у різних галузях [43-44].

Цей алгоритм працює шляхом поступового розподілу набору даних на підмножини залежно від значень вхідних параметрів. У результаті створюється ієрархічна структура у формі дерева: кожен внутрішній вузол відповідає за вибір, заснований на певній ознаці, що веде до подальшого розгалуження, а листові вузли містять прогнозовані числові результати.

Візуалізація структури дерева рішень – рис. 2.1.

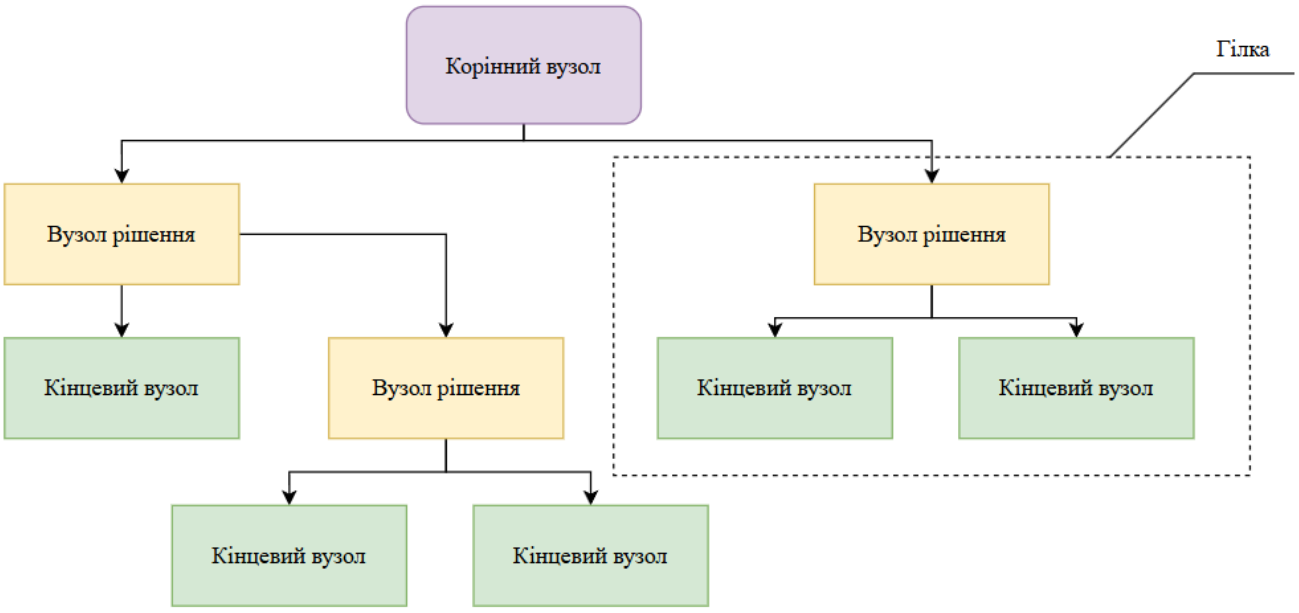

Рисунок 2.1 – Візуалізація структури дерева рішень

Даний алгоритм також спрямований на зменшення дисперсії цільової змінної в межах кожного поділу. У результаті формується модель, здатна точно прогнозувати безперервні величини. Регресія на основі дерева рішень відзначається прозорістю та простотою інтерпретації, адже правила, які вона використовує, зрозумілі й легко пояснюються. Такий підхід стає особливо корисним у випадках, коли важливо враховувати нелінійні взаємозв'язки та забезпечувати чітке обґрунтування отриманих прогнозів.

Регресійна модель на основі випадкового лісу є потужним інструментом з класу гомогенних ансамблів, зокрема тих, що базуються на деревах рішень. Її ключовою ідеєю є концепція «вибору найкращого з багатьох» [45-46]. Алгоритм, відповідно, такий: спочатку модель на основі випадкового лісу багаторазово генерує випадкові підвибірки з навчального набору даних для побудови численних дерев рішень (рис. 2.2). Потім для кожного дерева формується прогноз, і ці результати об'єднуються шляхом обчислення середнього значення прогнозів усіх дерев. Цей метод аналогічно призводить до зниження дисперсії та підвищення загальної стабільності моделі, що робить її менш схильною до перенавчання.

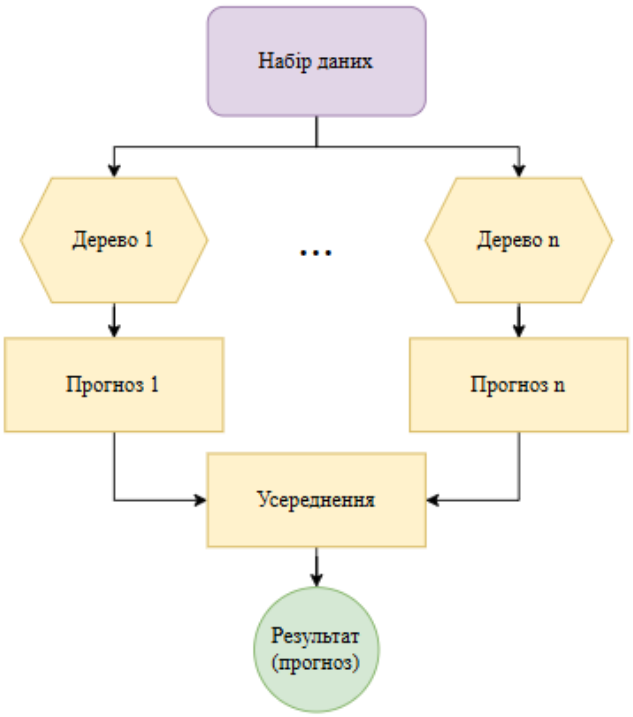

Рисунок 2.2 – Візуалізація алгоритму регресії на основі випадкового лісу

Регресійна модель на основі опорних векторів (англ. SVR, Support Vector Regression) – це розширення методу опорних векторів, яке вводить концепцію області, відомої як «трубка», навколо функції для оптимізації. Основна мета SVR полягає в пошуку трубки, яка найкраще апроксимує неперервну функцію, при цьому мінімізуючи похибку прогнозування, тобто різницю між передбаченими значеннями та істинними мітками класу.

SVR використовує нечутливу до відхилення функцію втрат, яка не штрафує прогнози, якщо вони знаходяться в межах цієї трубки. Значення ε визначає ширину трубки та, відповідно, впливає на кількість даних, які потрапляють у цю область. Чим менше ε, тим жорсткіші вимоги до моделі, і навпаки – велике значення ε дозволяє більшій кількості відхилень залишатися без штрафу. Прогнози, що виходять за межі ε, зазнають цього штрафу, що дозволяє моделі зосереджуватися на суттєвих похибках, замість спроби відтворити кожне спостереження з максимальною точністю.

Такий підхід робить модель SVR стійкою до шуму в даних, оскільки вона наголошує на конкретних важливих варіаціях, ігноруючи незначні деталі. Для SVR можна застосовувати різні функції втрат, як-от лінійну або квадратичну, залежно від специфіки завдання. Також SVR є потужним інструментом для розв’язання розсіяних задач, в яких локальні мінімуми не можуть бути знайдені іншими методами регресії [47].

Модель KNN-R є простим і ефективним непараметричним підходом у машинному навчанні, який прогнозує значення цільової змінної, базуючись на *k* найближчих сусідах у навчальному наборі даних.

Основна його ідея полягає в тому, що подібні дані зазвичай знаходяться близько один до одного в багатовимірному просторі. Модель використовує метрики відстані, як-от Евклідова або Мангеттенська, для знаходження сусідів, а для отримання прогнозу виконує агрегацію їх значень, наприклад, обчислюючи середнє. Хоча KNN-R має переваги, такі як простота реалізації та гнучкість, даний

метод також стикається з недоліками, зокрема високими обчислювальними витратами при великих наборах даних та чутливістю до шкалювання змінних.

Моделі на основі штучних НМ використовують подібну до роботи людського мозку структуру для обробки та аналізу даних.

Один зі способів концептуалізувати нейронну мережу – зазначити, що це мережа шаруватих нейронів. Предиктори (або входи) складають вхідний шар, а прогнози (або виходи) створюють вихідний.

Найпростіші нейронні мережі подібні до лінійних регресій і не містять жодних прихованих шарів. Наприклад, нейромережеве представлення лінійної регресії з трьома предикторами – рис. 2.3.

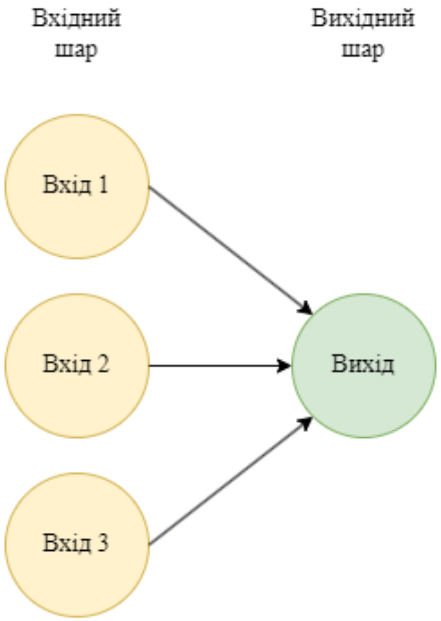

Рисунок 2.3 – Нейромережеве представлення лінійної регресії з трьома предикторами

Вагами НМ називаються коефіцієнти, які пов’язані з цими предикторами. Вхідні дані комбінуються лінійно, щоб отримати прогнози. У рамках нейронної мережі для вибору ваг використовується алгоритм навчання, який мінімізує функцію вартості, наприклад, середній квадрат похибки (англ. MSE, Mean Square Error).

Нейронна мережа стає нелінійною, коли додається проміжний шар, який містить приховані нейрони – рис. 2.4.

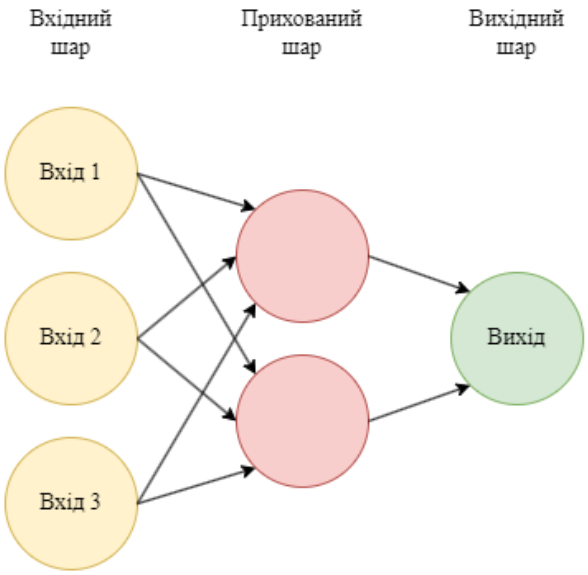

Рисунок 2.4 – Представлення нелінійної НМ

Досить багато досліджень, зокрема [48-50], показують, що моделі штучних нейронних мереж забезпечують більш точні та ефективні результати прогнозування порівняно з іншими регресійними методами. Це пов’язано з їх здатністю автоматично виявляти складні патерни й взаємозв’язки в даних завдяки своїй багатошаровій архітектурі.

2.2 Компроміс між зміщенням та дисперсією. Типи ансамблевої агрегації

Як відомо, ***математичне сподівання середньоквадратичної похибки*** складається з трьох основних компонентів: зміщення, дисперсії та шумового показника [51]. Формально дана похибка виглядає так:

$$E[y_0 - f(x_0)]^2 = Var[f(x_0)] + [Bias(f(x_0))]^2 + Var(\varepsilon), \quad (2.10)$$

де $E[y_0 - f(x_0)]^2$ – математичне сподівання середньоквадратичної похибки;

$Var[f(x_0)]$ – дисперсія;

$[Bias(f(x_0))]^2$ – зміщення;

$Var(\varepsilon)$ – шумовий показник.

Зміщення і дисперсія взаємозалежні. Іншими словами, зменшення зміщення моделі призводить до збільшення її дисперсії, і навпаки. Відповідно різні значення показників зміщення та дисперсії певним чином впливають на якість моделі прогнозування – рис. 2.5.

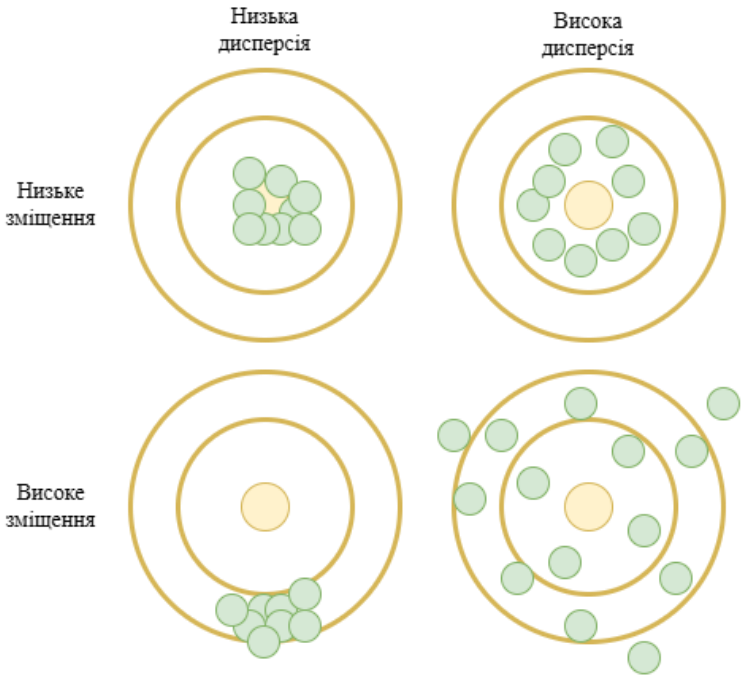

Рисунок 2.5 – Вплив показників зміщення та дисперсії на якість моделі прогнозування

Як можна побачити з візуалізації, модель МН з *низьким зміщенням* і *низькою дисперсією* вважається ідеальною. Однак на практиці вона зустрічається не часто.

Низьке зміщення та *висока дисперсія* своєю чергою призводять до перенавчання. Така комбінація дає непослідовні прогнози. Це зазвичай відбувається, коли модель має занадто багато параметрів і дуже сильно пристосована до навчальних даних.

Високе зміщення і *низька дисперсія* призводить до недонавчання. У цьому сценарії прогнози узгоджуються, але в середньому є неточними. Це трапляється,

коли модель погано навчається на тренувальних даних або має занадто мало параметрів.

Високе зміщення та *висока дисперсія*, звісно, призводить до неточних прогнозів.

Компроміс між зміщенням і дисперсією є фундаментальною концепцією МН, яка якраз-таки стосується балансу між ними. Простіше кажучи, це компроміс між здатністю моделі точно представляти основні закономірності даних і її чутливістю до коливань при зміні навчальних даних – рис. 2.6.

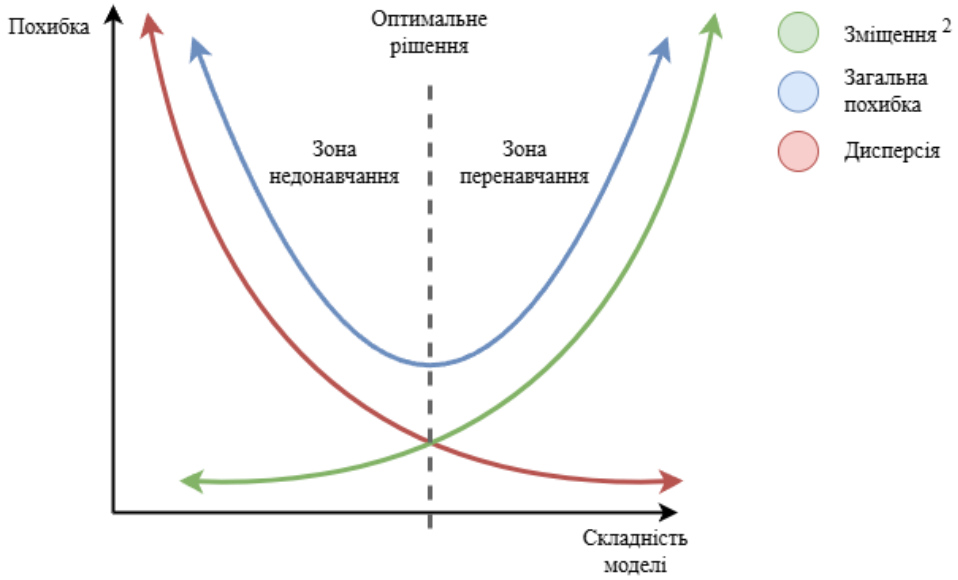

Рисунок 2.6 – Компроміс між зміщенням та дисперсією

Ансамблеві методи здатні впливати на попередньо описані компоненти: або на зміщення, або на дисперсію. Існує кілька технік створення ансамблів, їх можна поділити на три типи: бегінг, бустинг і стекінг.

Техніка, відома як ***бегінг,*** поєднує в собі ідеї агрегації та бутстрапінгу для створення цілісної ансамблевої моделі. За допомогою цього методу вибірка даних розбивається на багато підмножин, кожна з яких є бутстрапною реплікою.

Бегінг зазвичай підходить для моделей з низьким зміщенням і високою дисперсією. При його використанні дисперсія середнього значення прогнозів, $Var[y]$, зменшується пропорційно до кількості базових моделей, m:

$$Var[y] = Var[\frac{1}{m}\sum_{i=1}^{m} y_i] = \frac{1}{m^2}\sum_{i=1}^{m} Var[y_i] = \frac{1}{m} Var[y_i], \quad (2.11)$$

де $Var[y]$ – дисперсія середнього значення прогнозів;

y – підсумкове (усереднене) передбачення ансамблю моделей;

m – кількість базових моделей у складі ансамблю;

y_i – прогноз базової моделі i;

$Var[y_i]$ – дисперсія прогнозу базової моделі i.

У рівнянні показано, що дисперсія середнього значення прогнозів є $1/m$ від дисперсії кожної окремої моделі, $Var[y_i]$. Це означає, що при збільшенні кількості моделей в ансамблі зменшується загальна дисперсія прогнозу, що підвищує стабільність і надійність прогнозування [52]. Усереднений прогноз має однакове математичне сподівання. Навіть при створенні декількох бутстрапів, сподівання прогнозу залишиться однаковим:

$$E[y] = E[\frac{1}{m}\sum_{i=1}^{m} y_i] = E[y_i], \quad (2.12)$$

де $E[y]$ – математичне сподівання прогнозованого значення ансамблем моделей;

y – підсумкове (усереднене) передбачення ансамблю моделей;

m – кількість базових моделей у складі ансамблю;

y_i – прогноз базової моделі i;

$E[y_i]$ – математичне сподівання прогнозованого значення для однієї моделі.

Рівняння вище демонструє, що математичне сподівання ансамблевого прогнозу $E[y]$, який є середнім значенням прогнозів усіх моделей в ансамблі, дорівнює математичному сподіванню прогнозу будь-якої окремої моделі.

Формально даний ансамбль виглядає так – рис. 2.7.

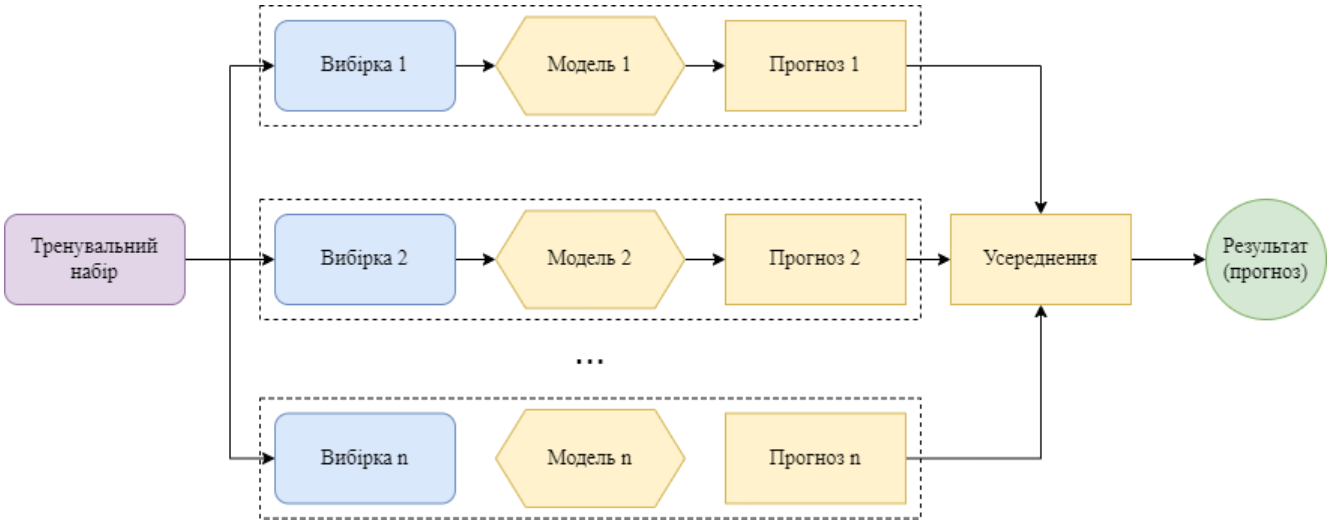

Рисунок 2.7 – Графічна інтерпретація ансамблю на основі бегінгу

Отже, бегінг ***зменшує дисперсію*** *моделі та допомагає запобігти надмірному пристосуванню через усереднення кількох прогнозів.*

Бустинг, навпаки, концентрується на поетапному навчанні моделей, коли кожна нова модель намагається виправити похибки, спричинені попередніми [53]. Методи бустингу включають такі алгоритми, як Gradient Boosting та AdaBoost. Для того, щоб поступово підвищити продуктивність, вони починають зі слабкої моделі, а потім неодноразово додають нові моделі, змінюючи вагу прикладів, які попередні моделі неправильно спрогнозували. *Бустинг часто підходить для моделей з низькою дисперсією і високим зміщенням.*

До успішного застосування на практиці бустинг існував лише теоретично. Алгоритм AdaBoost є його першим успішним застосуванням. З того часу бустинг змінився, і одним з найефективніших способів обробки структурованих даних зараз є градієнтний бустинг. Алгоритм наведено нижче.

Спочатку відбувається ініціалізація моделі з константним значенням:

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma), \tag{2.13}$$

де $F_0(x)$ – початкова модель з константним значенням;

x – вхідні дані;

γ – константа, яка мінімізує суму втрат для початкової моделі;

n – кількість спостережень у навчальній вибірці;

$L(y_i, \gamma)$ – функція втрат;

y_i – дійсне значення залежної змінної для спостереження *i*.

Потім відбувається ітеративне оновлення для *m=1* до *M*, що включає: обчислення залишків, навчання регресійного дерева з вхідними ознаками для прогнозування залишків, створення кінцевих вузлів, обчислення коефіцієнтів та оновлення моделі.

Обчислення залишків:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}, i = 1,...,n, \quad (2.14)$$

де r_{im} – залишки або похибки для кожного спостереження *i* на *m*-й ітерації;

$L(y_i, F(x_i))$ – функція втрат;

y_i – дійсне значення цільової змінної для спостереження *i*;

$F(x_i)$ – прогнозоване значення моделі для спостереження *i*;

x_i – вхідні характеристики для спостереження *i*;

$F(x)$ – накопичений прогноз моделі;

x – вхідні дані;

$F_{m-1}(x)$ – прогноз моделі на попередній ітерації *m-1*;

n – кількість спостережень у навчальному наборі даних.

Наступним проходить навчання регресійного дерева з вхідними ознаками *x* для прогнозування залишків *r* і створення кінцевих вузлів R_{jm} для $j = 1,..., J_m$.

Обчислення коефіцієнтів γ_{jm}:

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma), j = 1,..., J_m, \quad (2.15)$$

де γ_{jm} – оптимальне зміщення для вузла *j* на *m*-й ітерації, яке мінімізує функцію втрат;

γ – зміщення;

x_i – вхідні характеристики для спостереження *i*;

R_{jm} – кінцеві вузли (листки) регресійного дерева, побудованого на *m*-й ітерації;

$L(y_i, F_{m-1}(x_i) + \gamma)$ – функція втрат для спостереження *i*, з урахуванням зміщення γ;

y_i – дійсне значення цільової змінної для спостереження *i*;

$F_{m-1}(x_i)$ – прогноз моделі на попередній ітерації *m-1* для спостереження *i*;

J_m – кількість листків у регресійному дереві на *m*-й ітерації.

Оновлення моделі:

$$F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} 1(x \in R_{jm}), \tag{2.16}$$

де $F_m(x)$ – оновлена модель на *m*-й ітерації;

x – вхідні дані;

$F_{m-1}(x)$ – прогноз моделі на попередній ітерації *m-1*;

ν – коефіцієнт швидкості навчання, що контролює вплив кожного нового дерева на загальну модель;

J_m – кількість листків у регресійному дереві на *m*-й ітерації;

γ_{jm} – оптимальне зміщення для листка *j* на *m*-й ітерації;

$1(x \in R_{jm})$ – індикаторна функція, яка дорівнює 1, якщо *x* належить листку R_{jm}, і 0 в іншому випадку;

R_{jm} – кінцеві вузли (листки) регресійного дерева, побудованого на *m*-й ітерації.

Формально даний ансамбль виглядає так – рис. 2.8.

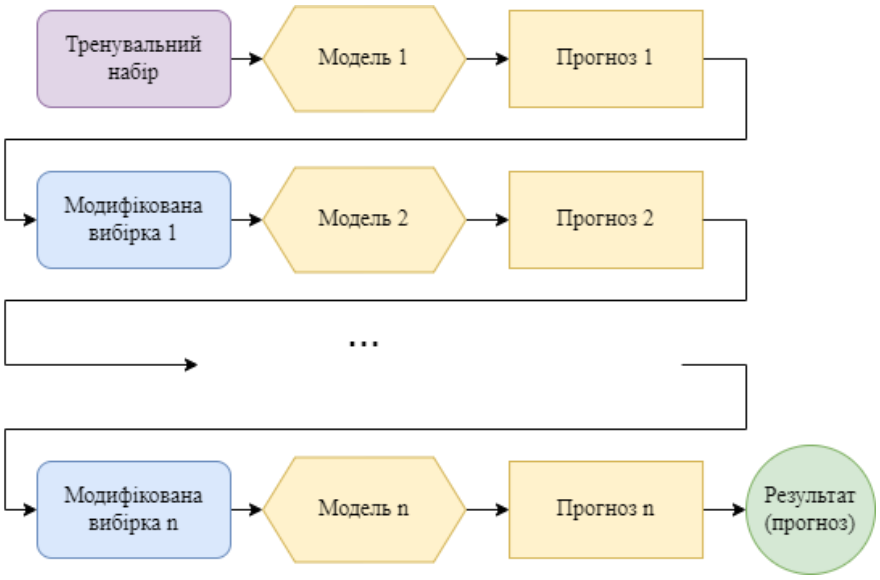

Рисунок 2.8 – Графічна інтерпретація ансамблю на основі бустингу

Отже, бустинг ***зменшує зміщення****, таким способом підвищуючи точність прогнозування.*

Іншим видом ансамблевого агрегування є ***стекінг***. Стекінг, також відомий як стекове узагальнення, є розширеною формою методу ансамблю ковзного середнього, в якому всі підмоделі беруть рівну участь відповідно до їхніх вагових коефіцієнтів і будують нову модель з кращими прогнозами. Ця нова модель накладається на інші моделі, саме тому така агрегація називається стекінгом.

Можна побачити (рис. 2.9), що архітектура ансамблю стекінгу розроблена так, що вона складається з двох або більше базових/навчальних моделей і метамоделі, яка об’єднує прогнози базових. Ці базові моделі називаються моделями рівня 0, а метамодель відома як модель рівня 1 [54].

Стекінг особливо підходить для моделей з низькою дисперсією і високим зміщенням.

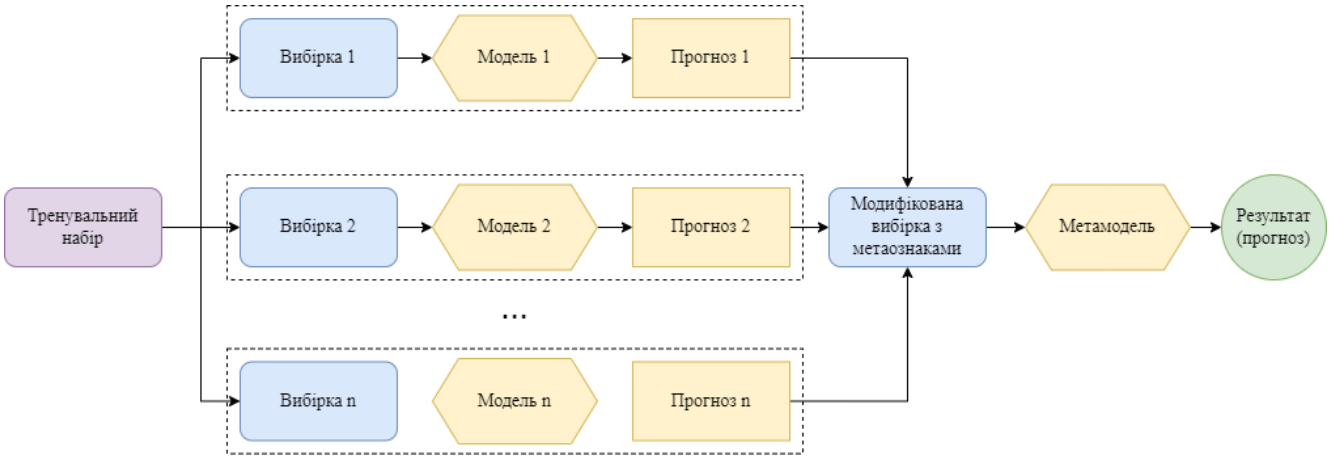

Рисунок 2.9 – Графічна інтерпретація ансамблю на основі стекінгу

Як підсумок, стекінг ***зменшує зміщення або дисперсію*** *залежно від типу метамоделі.*

Висновки до розділу 2

На початку розділу №2 розглянуто основи регресійних моделей, їх ключові особливості. Було детально вивчено, як саме регресійні моделі дозволяють враховувати взаємозв'язки між змінними, використовуючи ці зв'язки для прогнозування майбутніх значень.

Далі у розділі було представлено компроміс між зміщенням та дисперсією, типологію ансамблевих структур, як-от методи бегінгу, бустингу та стекінгу. Кожен з цих методів проаналізовано з погляду їхньої ефективності у балансуванні компромісу зміщення і дисперсії, та для загального покращення якості прогнозу порівняно з одиничними моделями.

У підсумку, дослідження регресійних моделей, проблеми компромісу складників похибки та типології ансамблевих структур в задачах прогнозування дозволило не лише сформувати чітке розуміння методології цих підходів, а й заклало базу для подальшого практичного впровадження.

3 РОЗВІДУВАЛЬНИЙ АНАЛІЗ ОБРАНОГО НАБОРУ ДАНИХ. ПРОЦЕС ПОПЕРЕДНЬОЇ ОБРОБКИ

3.1 Актуальність та опис обраного структурного набору

Прогнозування потужності на електростанції з комбінованим циклом є важливим фактором для забезпечення ефективності роботи газових турбін і мінімізації ризиків, пов’язаних із ручним керуванням.

Газові турбіни зазвичай досягають найкращих результатів при низьких температурах, оскільки щільне всмоктуване повітря покращує процес горіння. Проте підвищення температури зменшує щільність повітря, що негативно впливає на продуктивність і вихідну потужність. Завдяки точному прогнозуванню електростанція може оптимізувати такі процеси, забезпечуючи стабільність у межах максимального діапазону ефективності.

Крім технічних переваг, впровадження автоматизованої системи прогнозування потужності підвищує безпеку на виробництві. Робота в умовах електростанції пов’язана з численними ризиками, такими як екстремальна температура, високий тиск та високовольтне обладнання. Ручне коригування процесів у таких умовах наражає персонал на небезпеку. Натомість автоматизація не лише оптимізує робочі процеси, але й мінімізує потребу втручання людини, що знижує ризик травм. Як підсумок, надійна система прогнозування дозволяє операторам зосередитися на стратегічному управлінні, водночас забезпечуючи безпеку та ефективність експлуатації обладнання.

Тому згідно з наведеною вище проблематикою прогнозування потужності, було обрано відповідний набір даних [55]. Він містить ***9568 записів***, зібраних з електростанції комбінованого циклу протягом 6 років (2006-2011), коли станція працювала на максимальному навантаженні.

Як зазначається в інформації про датасет, характеристики містять середньогодинні змінні: температуру, атмосферний тиск, відносну вологість та розрідження на виході для прогнозування точного погодинного виходу

електроенергії.

Отже, можна вважати, що до ***предикторів*** відносяться:

– *температура* («AT» у файлі з даними): середня температура повітря, виміряна в градусах Цельсія;

– *атмосферний тиск* («AP» у файлі з даними): тиск повітря на рівні землі, виміряний в мілібарах;

– *відносна вологість* («RH» у файлі з даними): відсоткове відношення кількості водяної пари до максимально можливої кількості при даній температурі;

– *розрідження* («V» у файлі з даними): вимірювання вакууму в системі, що визначає кількість газу, що залишається в ній.

Залежна змінна – чиста погодинна електрична *потужність* («PE» у файлі з даними) – це кількість електричної енергії, що генерується за одну годину.

Для глибшого розуміння принципів роботи електростанції комбінованого циклу та впливу змінних на її продуктивність важливо не лише проаналізувати числові дані, але й візуалізувати сам процес. Це дозволяє краще уявити, як навколишні фактори впливають на ефективність роботи електростанції. Детальний та зрозумілий опис комбінованого циклу було представлено в роботі Ахмеда Аль Хашмі, Абделя Мохамеда та ін. [56 с. 35]. Процес схематично зображено на рис. 3.1.

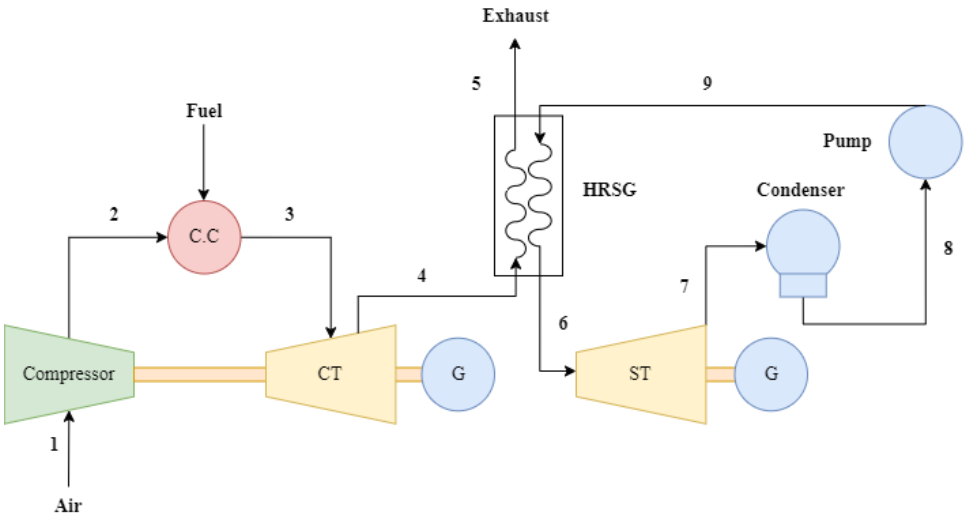

Рисунок 3.1 – Зображення роботи електростанції комбінованого циклу

Електростанція з парогазовим циклом поділяється на *сім різних контрольних об'ємів:* компресор (Compressor), пальник (C.C), турбогенератор (CTG), паровий турбогенератор (STG), котел-утилізатор (HRSG), конденсатор (Condenser) і помпа (Pump).

Технологічні етапи її роботи наступні.

1. Спочатку свіже повітря надходить у компресор.
2. Потім воно стискається до високого тиску.
3. Вийшовши з компресора, повітря потрапляє в систему згоряння, куди впорскується природний газ і відбувається процес горіння при постійному тиску.
4. Відпрацьовані гази залишають камеру згоряння і потрапляють до турбогенератора внутрішнього згоряння. У турбінній секції димові гази розширюються для виробництва електроенергії.
5. Димові гази залишають турбіну при високій температурі.

Вище наведений складник електростанції називається *газовою турбіною відкритого циклу*.

У *другій частині* гарячий потік з газової турбіни буде генерувати пару в котлі-утилізаторі. Етапи наведено нижче.

1. Паровий цикл складається з котла-утилізатора, парового турбогенератора, конденсатора і помпи, утворюючи цикл Ренкіна для виробництва електроенергії.
2. Вода надходить у котел-утилізатор під високим тиском, а утворена пара виробляє електроенергію в паровому турбогенераторі.
3. Насичена пара, що виходить з парової турбіни, спочатку конденсується, а потім її тиск підвищується перед поверненням до котла-утилізатора.

Отже, процес зрозуміло, тому можна переходити до розвідувального аналізу даних та обробки.

3.2 Розвідувальний аналіз

Для розробки програмних модулів була використана мова програмування R та середовище RStudio.

Спочатку встановлено робочий каталог за допомогою функції *setwd()*, що дозволяє працювати з файлами в поточній теці. Потім зчитано відповідні дані з Excel-файлу «powerplant_dataset.xlsx» за допомогою функції *read.xlsx()* і збережено їх у змінну *powerplant* – рис. 3.2.

```
> # Встановлення робочого каталогу
> setwd('C:\\Users\\peshe\\Desktop\\мкр')
>
> # Зчитування даних з файлу
> powerplant <- read.xlsx("powerplant_dataset.xlsx")
>
```

Рисунок 3.2 – Встановлення робочого каталогу, зчитування даних у змінну

Результатом роботи коду є датафрейм *powerplant*, збережений в робочому середовищі.

Наступним кроком здійснено виведення назв змінних за допомогою функції *names()* з датафрейму *powerplant*, щоб перевірити, які саме колонки містить набір даних, – рис. 3.3.

```
> # Виведення назв змінних
> names(powerplant)
[1] "AT" "V"  "AP" "RH" "PE"
>
```

Рисунок 3.3 – Виведення назв змінних

Дійсно, *всі предиктори*, які були зазначені в підрозділі 3.1, *присутні* в наборі:

– температура (AT);
– атмосферний тиск (AP);
– відносна вологість (RH);
– розрідження (V).

Залежна змінна – чиста погодинна електрична потужність (PE).

Виконано перейменування змінних у датафреймі *powerplant* за допомогою функції *colnames()*, тому що це покращує зрозумілість даних, запобігає плутанині та сприяє кращому документуванню, що спрощує подальшу роботу з ними, – рис. 3.4.

```
> # Перейменування змінних у датафреймі powerplant
> colnames(powerplant) <- c("Temperature", "Exhaust_Vacuum", "Ambient
_Pressure", "Relative_Humidity", "Energy_Output")
> names(powerplant)
[1] "Temperature"       "Exhaust_Vacuum"    "Ambient_Pressure"
[4] "Relative_Humidity" "Energy_Output"
```

Рисунок 3.4 – Перейменування змінних у датафреймі *powerplant*

В результаті отримано нові назви змінних: *«Temperature»*, *«Exhaust_Vacuum»*, *«Ambient_Pressure»*, *«Relative_Humidity»* та *«Energy_Output»*.

Виведено структуру набору даних за допомогою функції *str()* для вивчення типу кожної з ознак – рис. 3.5.

```
> # Виведення структури набору даних
> str(powerplant)
'data.frame':   9568 obs. of  5 variables:
 $ Temperature      : num  14.96 25.18 5.11 20.86 10.82 ...
 $ Exhaust_Vacuum   : num  41.8 63 39.4 57.3 37.5 ...
 $ Ambient_Pressure : num  1024 1020 1012 1010 1009 ...
 $ Relative_Humidity: num  73.2 59.1 92.1 76.6 96.6 ...
 $ Energy_Output    : num  463 444 489 446 474 ...
```

Рисунок 3.5 – Виведення структури набору даних

Результат виконання команди демонструє, що датафрейм дійсно містить 9568 спостережень та 5 змінних. *Усі змінні числові та безперервні. Потреби в їх перетворенні немає.*

Виконано виведення перших 6 записів з датафрейму *powerplant* за допомогою функції *head()*, що дозволяє швидко ознайомитися з початковими значеннями змінних. Також здійснено виведення статистичного підсумку даних за допомогою функції *summary()*, яка надає важливу інформацію про основні статистичні характеристики, як-от мінімум, максимум, середнє значення та квартилі. Вони допомагають оцінити розподіл та дисперсію у кожній змінній – рис. 3.6.

```
> head(powerplant)
  Temperature Exhaust_Vacuum Ambient_Pressure Relative_Humidity Energy_Output
1       14.96          41.76          1024.07             73.17        463.26
2       25.18          62.96          1020.04             59.08        444.37
3        5.11          39.40          1012.16             92.14        488.56
4       20.86          57.32          1010.24             76.64        446.48
5       10.82          37.50          1009.23             96.62        473.90
6       26.27          59.44          1012.23             58.77        443.67
>
> # Виведення статистичного підсумку
> summary(powerplant)
  Temperature    Exhaust_Vacuum  Ambient_Pressure Relative_Humidity
 Min.   : 1.81   Min.   :25.36   Min.   : 992.9   Min.   : 25.56
 1st Qu.:13.51   1st Qu.:41.74   1st Qu.:1009.1   1st Qu.: 63.33
 Median :20.34   Median :52.08   Median :1012.9   Median : 74.97
 Mean   :19.65   Mean   :54.31   Mean   :1013.3   Mean   : 73.31
 3rd Qu.:25.72   3rd Qu.:66.54   3rd Qu.:1017.3   3rd Qu.: 84.83
 Max.   :37.11   Max.   :81.56   Max.   :1033.3   Max.   :100.16
 Energy_Output
 Min.   :420.3
 1st Qu.:439.8
 Median :451.6
 Mean   :454.4
 3rd Qu.:468.4
 Max.   :495.8
```

Рисунок 3.6 – Виведення перших 6 записів та статистичного підсумку

Отже, згідно зі статистичним підсумком, температура варіюється від 1.81°C до 37.11°C, зі середнім значенням 19.65°C, що вказує на наявність як низьких, так і високих температур у даних. Вакуум має діапазон від 25.36 см рт. ст. до 81.56 см рт. ст., зі середнім значенням 54.31 см рт. ст.. Атмосферний тиск коливається між 992.9 мбар і 1033.3 мбар. Відносна вологість показує середнє значення 73.31% з мінімумом 25.56% та максимумом 100.16%. Вихід енергії варіюється від 420.3 МВт до 495.8 МВт, з середнім значенням 454.4 МВт.

Одним з найважливіших кроків в аналізі ознак є перевірка кореляції в даних. Такий підхід може допомогти у двох напрямках: по-перше, визначити, які ознаки мають високу кореляцію з цільовим показником. Другий напрямок – визначити, чи мають ознаки високу кореляцію між собою. Якщо дві ознаки мають високу кореляцію, можна вибрати лише одну з них для аналізу, що допоможе зменшити складність обчислень і можливе перенавчання.

Для цього було створено кореляційну матрицю за допомогою функції *cor()*, а також побудовано графік кореляцій за допомогою *corrplot.mixed()* (рис. 3.7 – рис. 3.8).

```
> # Кореляційний графік
> correlation_matrix <- cor(powerplant)
> correlation_matrix
                  Temperature Exhaust_Vacuum Ambient_Pressure Relative_Humidity
Temperature         1.0000000      0.8441067      -0.50754934       -0.54253465
Exhaust_Vacuum      0.8441067      1.0000000      -0.41350216       -0.31218728
Ambient_Pressure   -0.5075493     -0.4135022       1.00000000        0.09957432
Relative_Humidity  -0.5425347     -0.3121873       0.09957432        1.00000000
Energy_Output      -0.9481285     -0.8697803       0.51842903        0.38979410
                  Energy_Output
Temperature          -0.9481285
Exhaust_Vacuum       -0.8697803
Ambient_Pressure      0.5184290
Relative_Humidity     0.3897941
Energy_Output         1.0000000
> corrplot.mixed(correlation_matrix,order = 'AOE')
```

Рисунок 3.7 – Створення кореляційної матриці та побудова графіку кореляцій

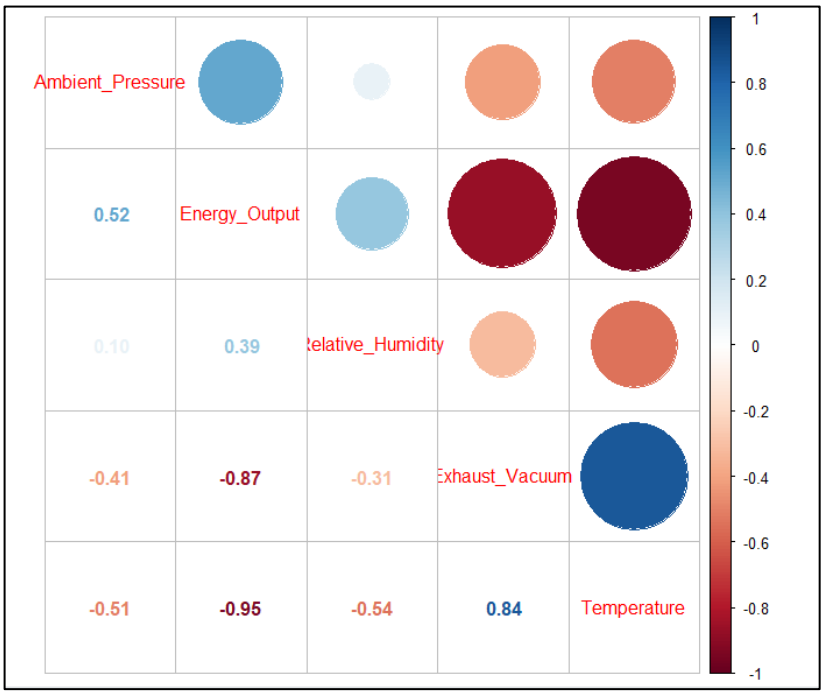

Рисунок 3.8 – Графік кореляцій

Аналіз матриці та графіку дозволяє зробити наступні висновки.

По-перше, між *«Temperature»* і *«Exhaust_Vacuum»* спостерігається сильна позитивна кореляція 0.84, що свідчить про підвищення вакууму вихлопних газів зі збільшенням температури. Тобто зміна температури впливає на умови роботи обладнання, змінюючи ефективність відведення вихлопних газів.

По-друге, *«Temperature»* і *«Energy_Output»* мають сильну негативну кореляцію -0.95, себто зростання температури навколишнього середовища

супроводжується зниженням виходу енергії. Вищі температури негативно впливають на ефективність роботи енергетичної установки, що призводить до скорочення обсягів виробленої електроенергії.

Нарешті, між *«Exhaust_Vacuum»* і *«Energy_Output»* також зафіксована сильна негативна кореляція -0.87. Це свідчить про те, що високий вакуум вихлопних газів супроводжується значним зниженням енергетичного виходу. Така залежність до того ж демонструє, що продуктивність генерації енергії зменшується, коли вихлопні гази не відводяться належним чином, що, зрештою, погіршує загальну ефективність установки.

Важливо додатково перевірити розподіли змінних, оскільки потрібно знати, чи не є якась з ознак сильно незбалансованою. У таких випадках можна вилучити ці ознаки або виконати певну інженерію ознак, щоб отримати кращий розподіл, залежно від їхньої кількості та доступних спостережень. Тому було побудовано графік парних відносин за допомогою функції *ggpairs()*, що дозволяє візуально дослідити взаємозв'язки між змінними в наборі даних, – рис. 3.9.

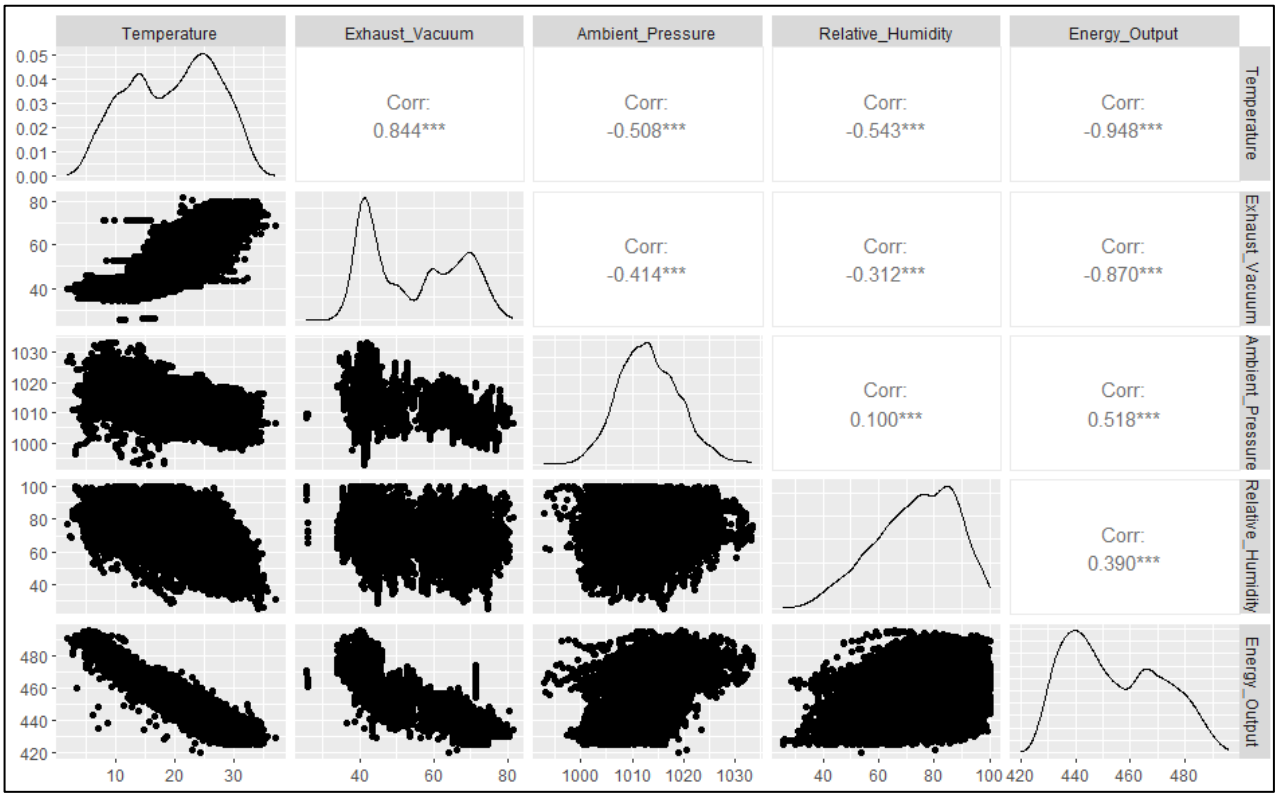

Рисунок 3.9 – Графік парних відносин змінних

На основі графіка можна зробити попередні *висновки щодо розподілів*. Лише *«Exhaust_Vacuum»* має розподіл, далекий від нормального. *«Ambient_Pressure»* є досить нормальним, *«Temperature»* та *«Energy_Output»* – близькі до нормального, а *«Relative_Humidity»* – зміщений вліво, однак значення даного предиктора можна нормалізувати на подальших етапах.

Викиди – це незвичайні значення у наборі даних, які можуть спотворювати статистичний аналіз і порушувати його припущення. Для попередньої їх ідентифікації було побудовано графіки boxplot за допомогою функції *boxplot()* – рис. 3.10.

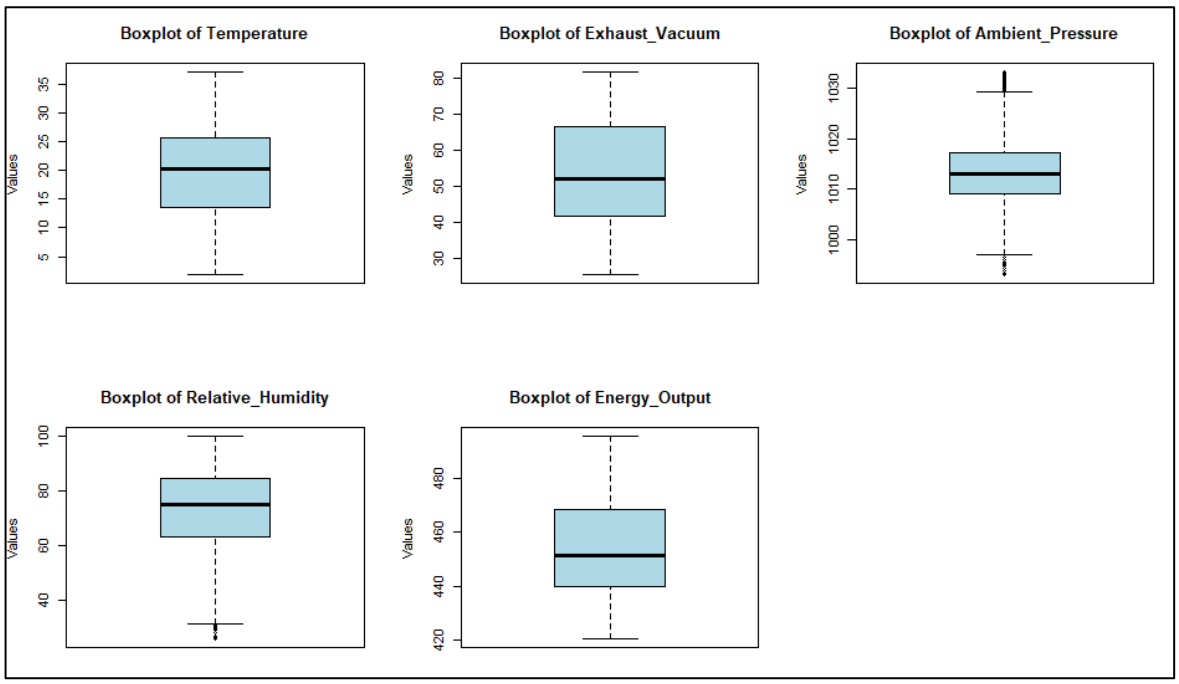

Рисунок 3.10 – Графіки boxplot змінних

Графіки boxplot показали, що *викиди присутні тільки у двох предикторах – «Ambient_Pressure»* та *«Relative_Humidity»*. Їх точніша ідентифікація та обробка буде описана в наступному підрозділі.

3.3 Попередня обробка даних

Оскільки обраний набір даних складається з числових ознак, згідно з розвідувальним аналізом, то ознаки не потребують додаткових трансформацій. Для

такого випадку, процес попередньої обробки даних включає *ідентифікацію й обробку пропусків, викидів та дублікатів, нормалізацію і відбір ознак*.

Детальна візуалізація процесу попередньої обробки обраного набору даних зображено на блок-схемі – рис. 3.11.

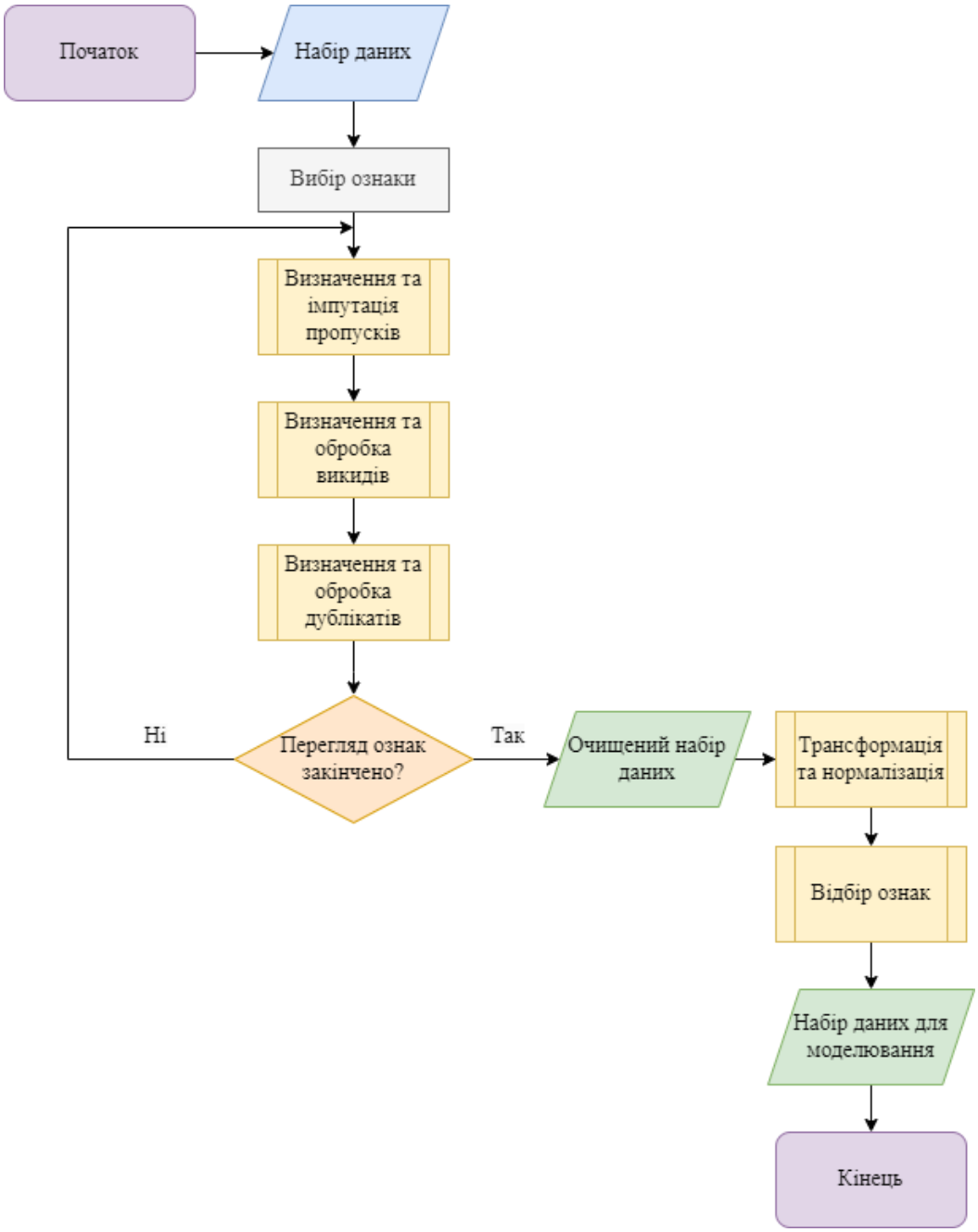

Рисунок 3.11 – Блок-схема процесу попередньої обробки даних

В ході перевірки на пропуски, було визначено, що *у наборі даних немає жодного відсутнього значення*, як показано нижче, – рис. 3.12.

```
> # Перевірка на пропуски
> sum(is.na(powerplant))
[1] 0
```

Рисунок 3.12 – Перевірка на пропуски

Обробка пропущених значень в такому випадку не потрібна, тому наступним кроком є ідентифікація та обробка викидів.

Першим етапом був вибір методу ідентифікації викидів. Найбільш простий в інтерпретації та ефективний в застосуванні є міжквартильний розмах або IQR, що є мірою статистичної дисперсії. Він являє собою діапазон, в який потрапляють середні 50% даних. Щоб розрахувати IQR, потрібно знайти різницю між 75-м квартилем (Q3) і 25-м квартилем (Q1).

Однією з переваг методу IQR є те, що він стійкий до викривлених розподілів даних. IQR до того ж визначає відхилення на основі процентилів, що робить його менш чутливим до екстремальних значень.

Тому за допомогою функції *sapply(),* відповідно, обчислюються перший і третій квартилі для кожного стовпчика з датафрейму *powerplant*. Результати зберігаються у змінних *q1* та *q3*. Потім обчислюється міжквартильний діапазон як різниця між Q3 і Q1 для кожного стовпчика. Нарешті, результати IQR виводяться на екран – рис. 3.13.

```
> # Обчислення квартилів та IQR для кожного стовпчика ознак
> q1 <- sapply(numeric_cols, function(col) quantile(powerplant[[col]], 0.25))
> q3 <- sapply(numeric_cols, function(col) quantile(powerplant[[col]], 0.75))
> IQR <- q3 - q1
>
> IQR
      Temperature.75%    Exhaust_Vacuum.75%
               12.2100               24.8000
 Ambient_Pressure.75% Relative_Humidity.75%
                8.1600               21.5025
    Energy_Output.75%
               28.6800
```

Рисунок 3.13 – Визначення IQR

Обчислені значення міжквартильного діапазону вказують на варіативність у даних. Найбільший IQR спостерігається для виходу енергії, що свідчить про її

значні коливання. Найменше значення IQR зафіксовано для атмосферного тиску, що означає більшу стабільність цього показника в наборі даних.

В попередньому підрозділі було зроблено припущення щодо наявності викидів згідно з графіками boxplot. Оскільки найкраще всього знати точну інформацію про відсоток викидів в тій чи іншій ознаці, то було виконано наступні маніпуляції. За допомогою функції *sapply()* визначаються викиди в кожному стовпчику. Для цього обчислюються індекси викидів, порівнюються значення в кожному стовпчику з відповідними межами. Результати зберігаються в матриці *powerplant_outliers*. Далі створюється датафрейм *powerplant_outlier*, у якому зберігається загальна кількість викидів *«Total_Outliers»* і відсоток викидів *«Percentage_Outliers»* для кожного зі стовпців. Відсоток викидів розраховується як відношення кількості викидів до загальної кількості рядків у датафреймі *powerplant*, помножене на 100%. Наприкінці виводиться інформація про викиди за допомогою функції *print()* – рис. 3.14.

```
> # Визначення викидів
> powerplant_outliers <- sapply(1:length(numeric_cols), function(i) {
+   col <- numeric_cols[i]
+   outlier_indices <- powerplant[[col]] < (q1[i] - 1.5 * IQR[i]) | powerplant[[c
ol]] > (q3[i] + 1.5 * IQR[i])
+   return(outlier_indices)
+ })
>
> colnames(powerplant_outliers) <- numeric_cols
>
> # Створення фрейму даних для зберігання інформації про викиди
> powerplant_outlier <- data.frame(
+   Total_Outliers = colSums(powerplant_outliers),
+   Percentage_Outliers = colSums(powerplant_outliers) / nrow(powerplant) * 100
+ )
>
> # Виведення інформації про викиди
> print(powerplant_outlier)
                 Total_Outliers Percentage_Outliers
Temperature                   0           0.0000000
Exhaust_Vacuum                0           0.0000000
Ambient_Pressure             88           0.9197324
Relative_Humidity            12           0.1254181
Energy_Output                 0           0.0000000
>
```

Рисунок 3.14 – Розрахунок кількості та відсотку викидів у змінних

Попередні припущення підтверджено. Виявлено *наявність викидів у двох предикторах*: *«Ambient_Pressure»* (0.92% – 88 викидів) та *«Relative_Humidity»* (0.13% – 12 викидів).

Для обробки викидів зазвичай використовують різні методи. Найбільш популярними є видалення або заміна викидів на пусті значення та імпутація. Зазвичай будь-яке видалення інформації з набору даних є небажаним рішенням, тому було вирішено зупинитись на імпутації значень.

Було створено функцію *impute_and_plot()*, яка замінює викиди на пропущені значення *NA* та виконує їх імпутацію за допомогою найбільш використовуваних методів: *missForest()* та *MICE()* (залежно від обраного підходу: *pmm, CART, lasso.norm*). Функція також підраховує кількість пропусків до і після імпутації, а для візуалізації результатів будує boxplot числових змінних для оцінки розподілу даних після імпутації – рис. 3.15.

```
> # Функція для виконання імпутації та побудови бокс-діаграм для різних методів
> impute_and_plot <- function(df, method, title_text) {
+
+   # Заміна пропусків на NA
+   for (col in numeric_cols) {
+     df[[col]][powerplant_outliers[, col]] <- NA
+   }
+
+   # Підрахунок пропущених значень перед інтерполяцією
+   cat("Missing values before imputation: ", sum(is.na(df)), "\n")
+
+   # Імпутація
+   if (method == "missForest") {
+     # Виконання імпутації з missForest
+     imputed_result <- missForest(df[, names(colSums(is.na(df)))])
+     df <- imputed_result$ximp
+   } else {
+     # Виконання імпутації MICE для інших методів
+     df <- complete(mice(df[, names(colSums(is.na(df)))], method = method))
+   }
+
+   # Підрахунок пропущених значень після імпутації
+   cat("Missing values after imputation: ", sum(is.na(df)), "\n")
+
+   # Бокс-діаграма
+   par(mfrow = c(2, 3), oma = c(0, 0, 2, 0))
+   for (var in numeric_cols) {
+     boxplot(df[[var]], main = paste("Boxplot of", var), ylab = "Values", col = "lightblue")
+   }
+   title(main = title_text, outer = TRUE)
+   par(mfrow = c(1, 1))
+ }
```

Рисунок 3.15 – Функція *impute_and_plot()*

Результати її роботи продемонстровані на рис. 3.16 – 3.19. Як видно з графіків, найкраще обробив викиди саме метод *MICE()* з підходом *CART*.

Результівний датасет, отриманий за даним методом, буде використано для подальших маніпуляцій з даними.

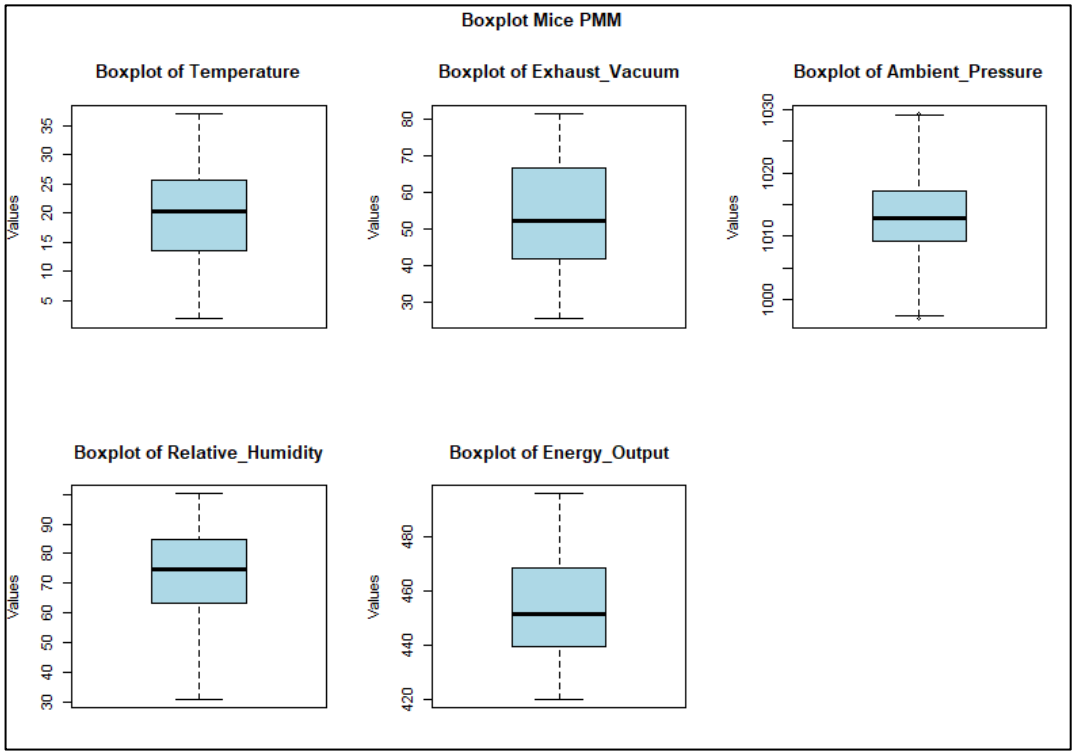

Рисунок 3.16 – Результат роботи методу *MICE()* з підходом *pmm*

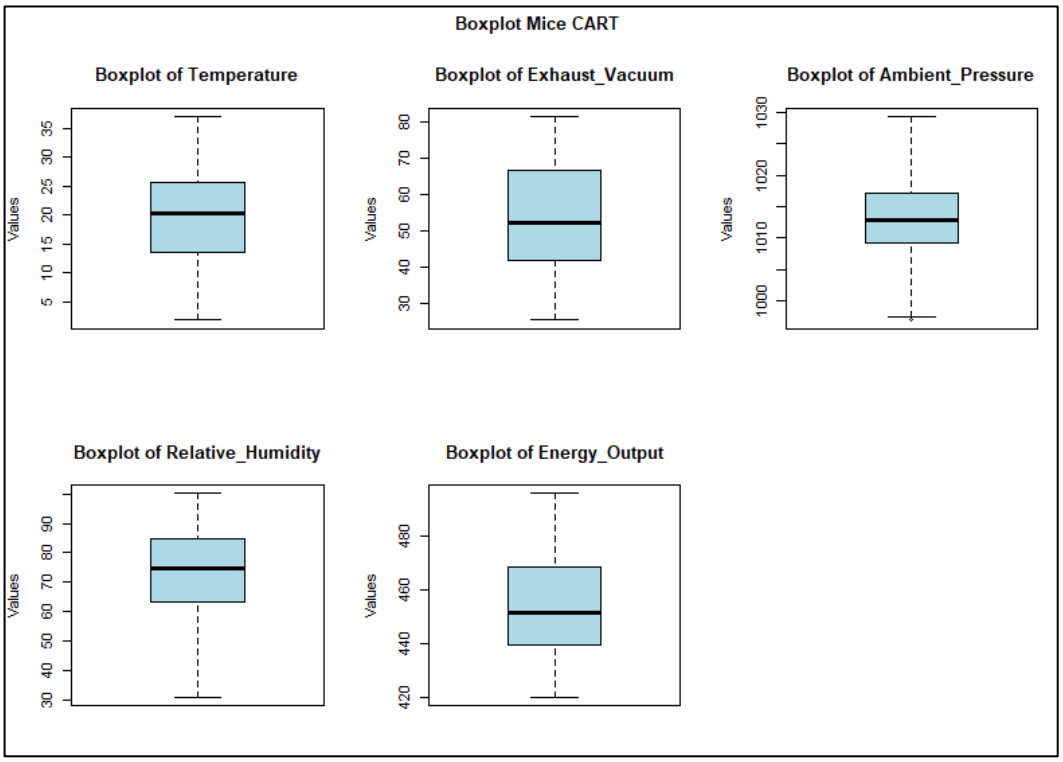

Рисунок 3.17 – Результат роботи методу *MICE()* з підходом *CART*

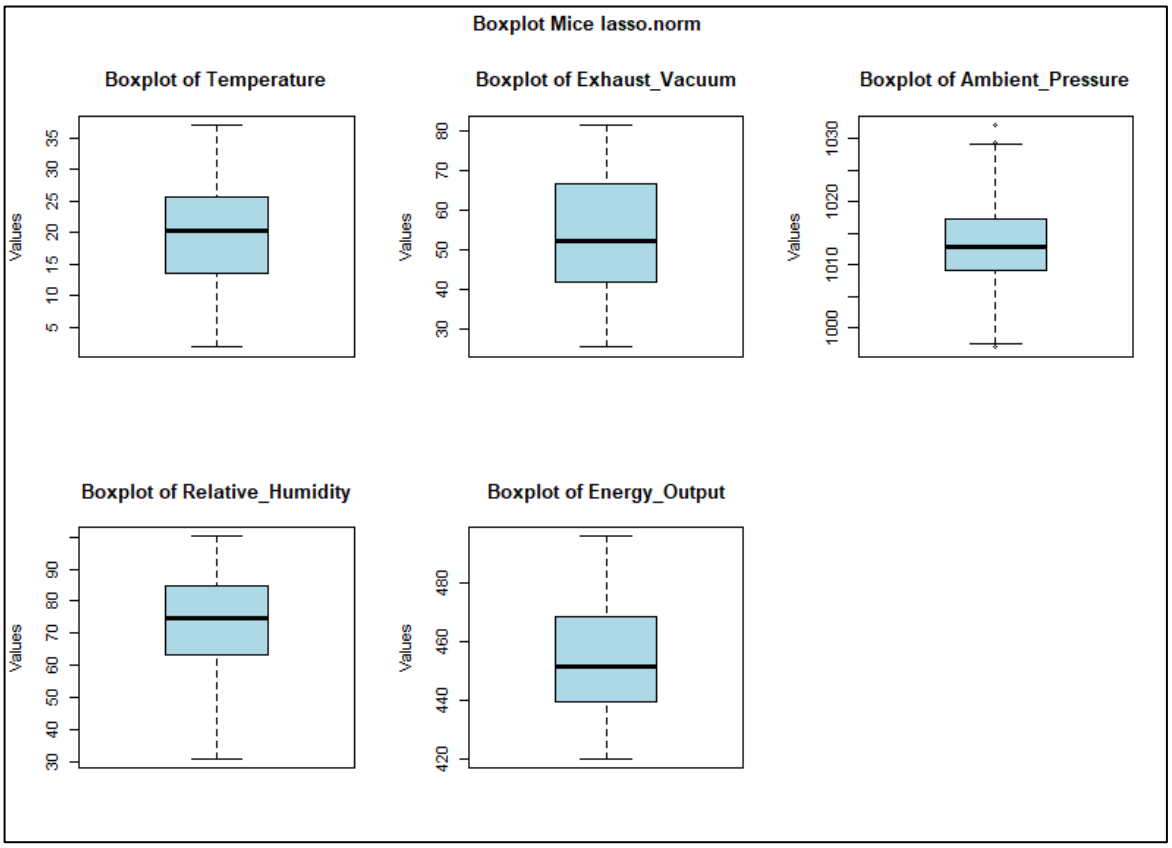

Рисунок 3.18 – Результат роботи методу *MICE()* з підходом *lasso.norm*

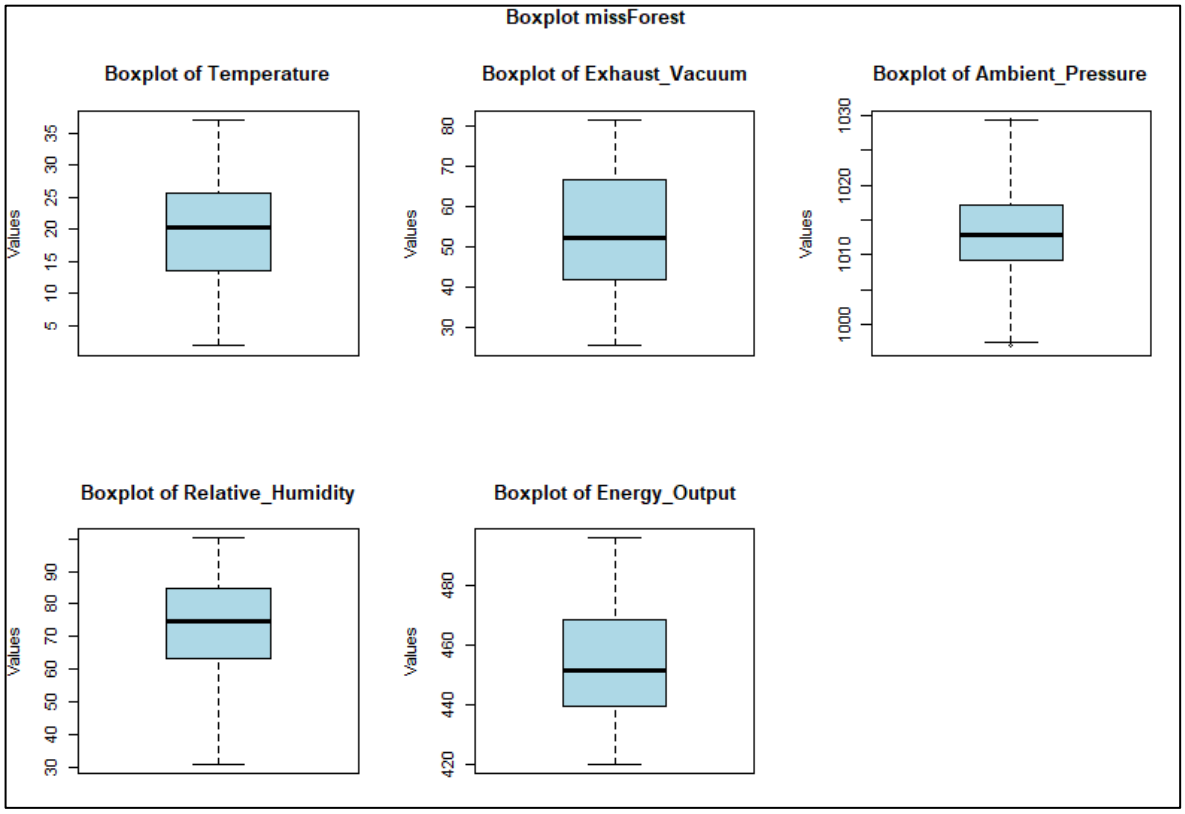

Рисунок 3.19 – Результат роботи методу *missForest()*

Останнім кроком очищення даних є ідентифікація та видалення дублікатів. Спочатку за допомогою функції *duplicated()* виявляються всі рядки, що повторюються у датафреймі *powerplant*, включаючи оригінали, які потім зберігаються у змінній *all_duplicates*. Згодом дублікатні рядки сортуються в порядку спадання за всіма стовпчиками за допомогою функції *order()*, і результат зберігається в *sorted_duplicates*. Далі за допомогою функції *head()* виконується перегляд перших шести рядків відсортованих дублікатів. На завершення, за допомогою *!duplicated()* з початкового датафрейму видаляються дублікати та зберігаються лише унікальні рядки – рис. 3.20.

```
> powerplant_clean <- powerplant_clean2
> # Виявлення всіх рядків, що повторюються, включно з оригінальними
> all_duplicates <- powerplant_clean[duplicated(powerplant_clean) | duplicated(powerplant_clean, fromLast
= TRUE), ]
>
> # Сортування повторюваних рядків у порядку спадання за всіма стовпчиками
> sorted_duplicates <- all_duplicates[do.call(order, c(all_duplicates, decreasing = TRUE)), ]
>
> # Перегляд відсортованих повторюваних рядків
> head(sorted_duplicates, n=6)
     Temperature Exhaust_Vacuum Ambient_Pressure Relative_Humidity Energy_Output
3815       29.51           75.6          1017.92             50.61        431.18
9202       29.51           75.6          1017.92             50.61        431.18
2650       29.45           75.6          1018.12             50.68        437.31
8883       29.45           75.6          1018.12             50.68        437.31
3246       29.23           75.6          1017.72             52.26        438.92
6228       29.23           75.6          1017.72             52.26        438.92
>
> # Видалення дубльованих рядків
> powerplant_clean <- powerplant_clean[!duplicated(powerplant_clean), ]
>
```

Рисунок 3.20 – Ідентифікація та видалення дублікатів

За результатами *дублікатів всього виявилось 82, включаючи оригінали*. Дублікати було успішно видалено.

Для визначення необхідності трансформації ознак знову-таки треба перевірити розподіли. Для цього було побудовано гістограми розподілів змінних – рис. 3.21. Як і було зазначено попередньо, *«Exhaust_Vacuum»* має розподіл, далекий від нормального. *«Ambient_Pressure»* є досить нормальним, *«Temperature»* та *«Energy_Output»* – близькі до нормального, а *«Relative_Humidity»* – зміщений вліво.

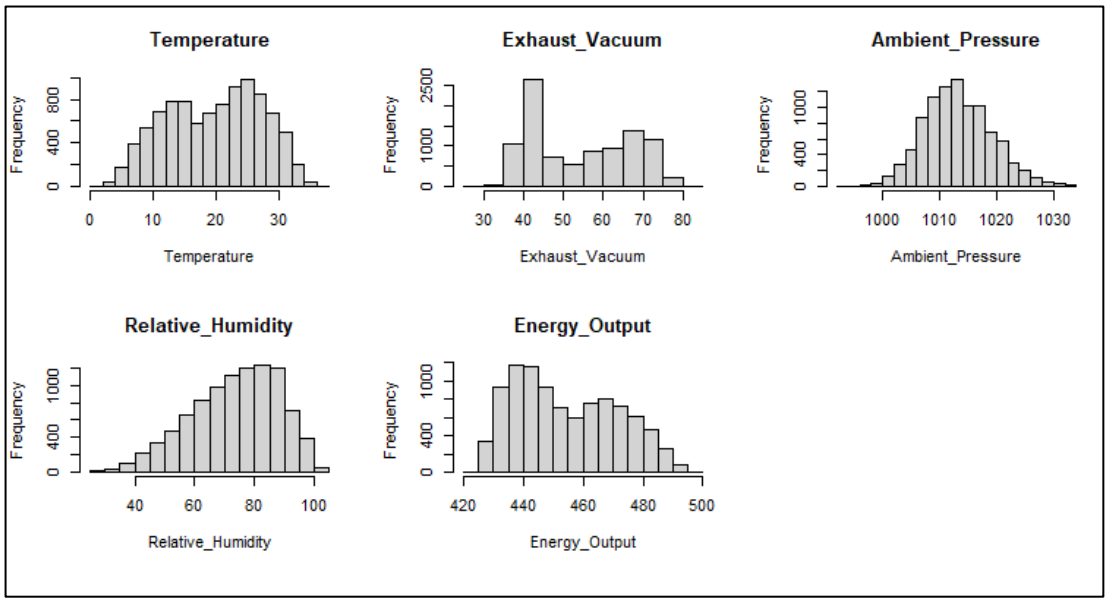

Рисунок 3.21 – Розподіл ознак в очищеному наборі даних

Оскільки деякі з ознак (*«Exhaust_Vacuum», «Relative_Humidity»)* мають відмінні від нормального розподіли, було вирішено виконати трансформацію Йео-Джонсона – рис. 3.22.

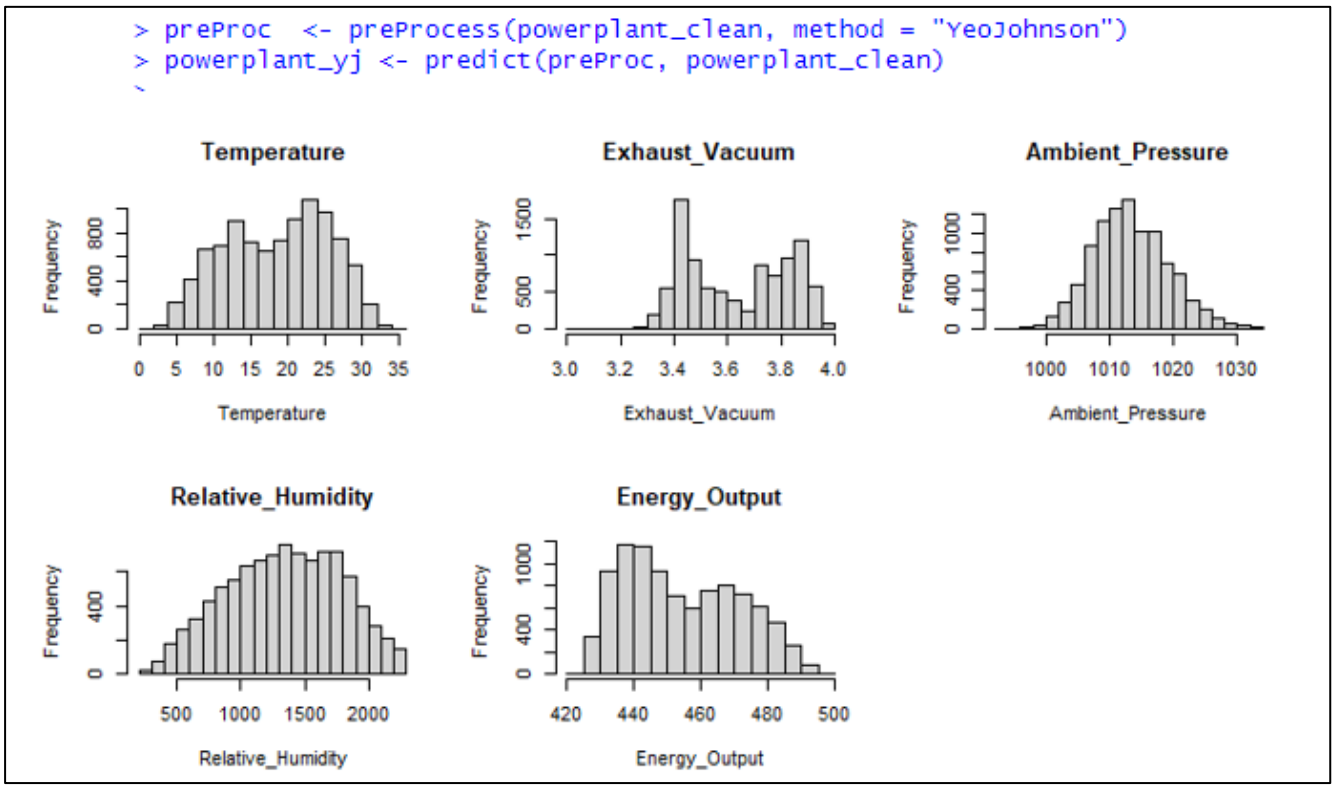

Рисунок 3.22 – Розподіл ознак в наборі даних після трансформації Йео-Джонсона

З графіків видно, що *дійсно ситуація покращилась для «Relative_Humidity».*

Для забезпечення швидкості та стабільності навчання було виконано min-max нормалізацію даних. Даний метод є одним із найбільш поширених та ефективних методів нормалізації. Він приводить ознаки до діапазону [0, 1], що усуває відмінності між шкалами. Min-max має переваги над іншими видами масштабування, коли розподіл ознаки (або її трансформації) не є нормальним – рис. 3.23.

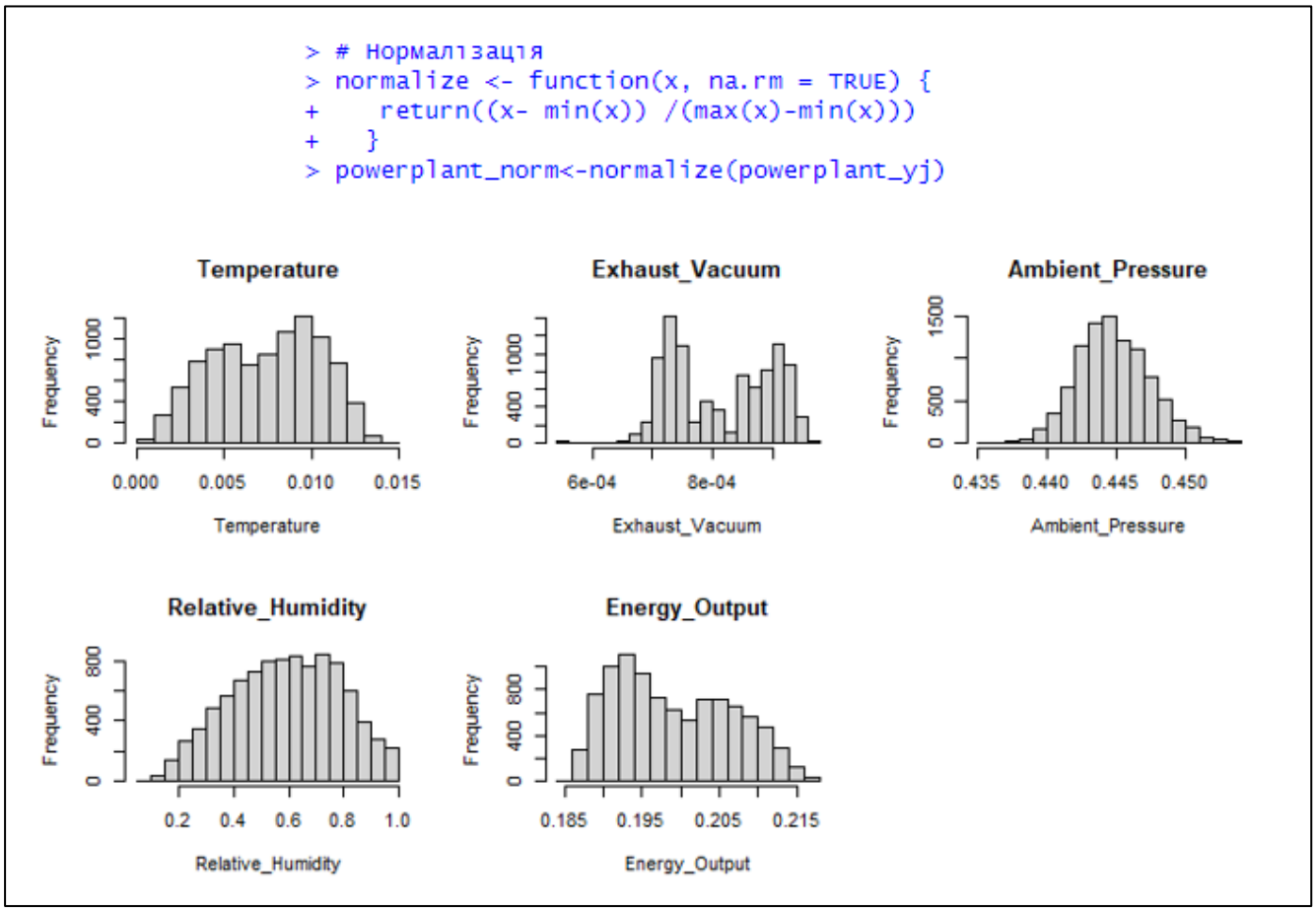

Рисунок 3.23 – Розподіл ознак в наборі даних після min-max нормалізації

Тепер *набір очищений та приведений до одного діапазону*.

Було вирішено також виконати *перехресну перевірку для відбору ознак*, з метою визначення кращої моделі з відповідними важливими ознаками. Метод обчислення MSE на вибірці перевірки використовується для оцінки точності моделі за допомогою крос-валідації. Спочатку відбираються ознаки моделі, а потім

розраховуються похибки для кожного блоку крос-валідації. Для цього використовується функція *predict.regsubsets()*, яка обчислює прогнози моделі та порівнює їх зі справжніми значеннями цільової змінної. Результатом є матриця похибок *cv.errors*, де кожен рядок відповідає окремому блоку крос-валідації, а кожний стовпчик – кількості ознак у моделі, – рис. 3.24.

```
> # Метод обчислення MSE на вибірці перевірки
> predict.regsubsets<-function(object,newdata,id,...){
+   form<-as.formula(object$call[[2]])
+   mat<-model.matrix(form,newdata)
+   coefi<-coef(object,id=id)
+   xvars<-names(coefi)
+   mat[,xvars]%*%coefi
+ }
```

Рисунок 3.24 – Функція *predict.regsubsets()*

За результатами відбору ознак видно, що найкраща модель включає всі 4 предиктори. *Похибки всіх моделей дуже низькі, що є хорошим показником* – рис. 3.25.

```
> # Відбір ознак
> k<-100
> set.seed(1)
> folds<-sample(1:k,nrow(powerplant_norm),replace=TRUE)
> cv.errors<-matrix(NA,k,4,dimnames=list(NULL,paste(1:4)))
>
> # Обчислення помилок на перевірочних вибірках для кожного блоку
> for (j in 1:k) {
+   best.fit <- regsubsets(Energy_Output ~ ., data = powerplant_norm[folds != j, ], nvmax = 4)
+   for (i in 1:4) {
+     pred <- predict.regsubsets(best.fit, powerplant_norm[folds == j, ], id = i)
+     cv.errors[j, i] <- mean((powerplant_norm$Energy_Output[folds == j] - pred)^2)
+   }
+ }
>
> # Розрахунок середніх значень по стовпцях матриці
> mean.cv.errors<-apply(cv.errors,2,mean)
> mean.cv.errors
           1            2            3            4
5.661753e-06 4.476888e-06 3.994487e-06 3.972898e-06
```

Рисунок 3.25 – Процес та результат відбору ознак

Графічна візуалізація MSE для моделей з різною кількістю предикторів – рис. 3.26.

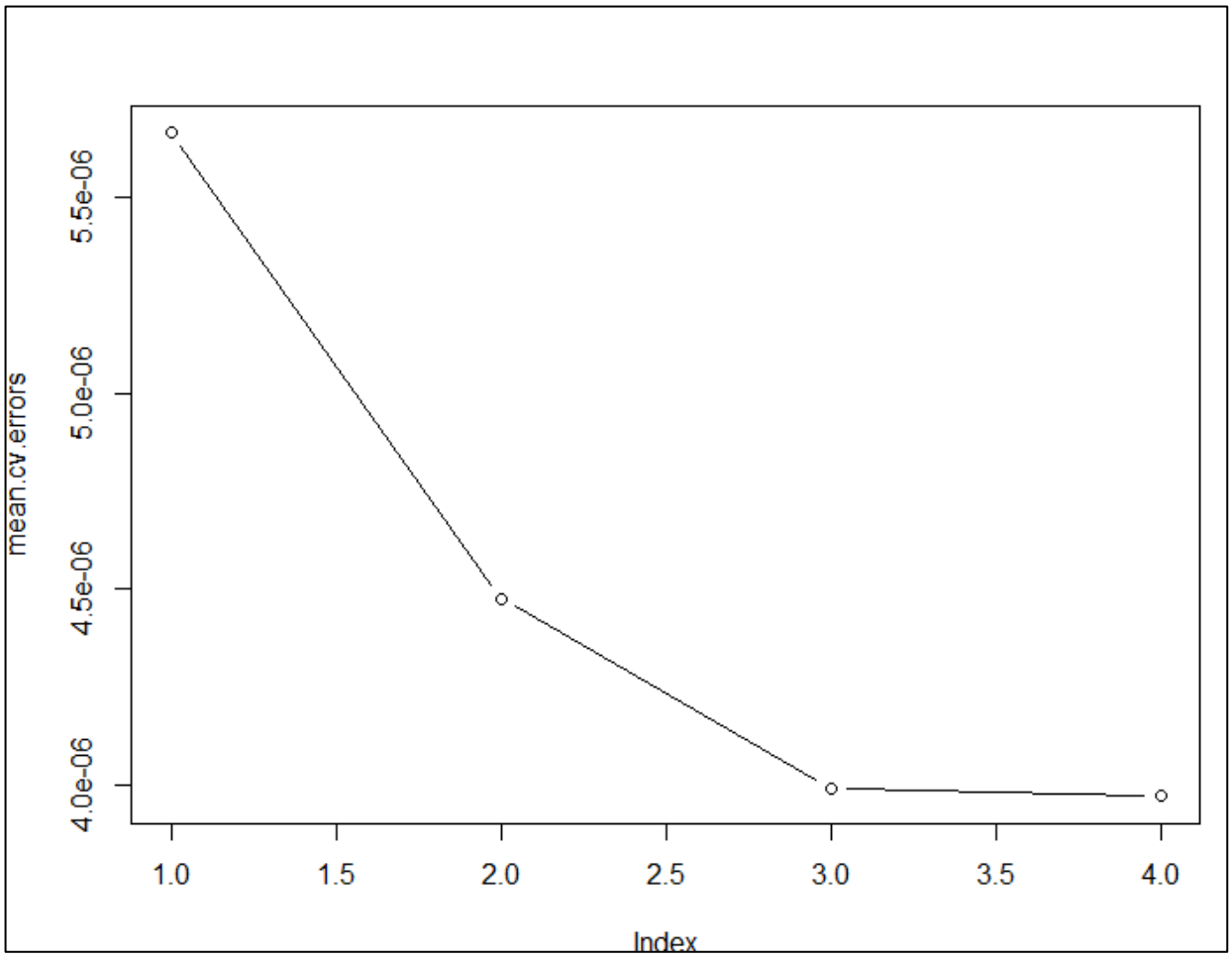

Рисунок 3.26 – Графічна візуалізація MSE для моделей з різною кількістю предикторів

Виконано відбір оптимальної підмножини змінних на повному наборі даних – рис. 3.27. Дійсно, *найкраща модель включає всі 4 предиктори.*

```
> # Відбір оптимальної підмножини змінних на повному наборі даних для отримання моделі
> reg.best<-regsubsets(Energy_Output~.,data=powerplant_norm,nvmax=4)
> coef(reg.best,which.min(mean.cv.errors))
      (Intercept)       Temperature    Exhaust_Vacuum  Ambient_Pressure Relative_Humidity
      0.199614693      -2.073933844     -17.093759338       0.070874124      -0.004855962
```

Рисунок 3.27 – Відбір оптимальної підмножини змінних

Етап попередньої обробки даних завершено. *Дані готові до моделювання.*

Висновки до розділу 3

У цьому розділі було проведено детальний розвідувальний аналіз обраного структурного набору даних, що дозволило виявити ключові характеристики та закономірності, які впливають на подальший етап моделювання.

У результаті проведеного аналізу даних роботи електростанції з комбінованим циклом було встановлено, що прогнозування потужності є критично важливим для оптимізації виробництва електроенергії та забезпечення надійності операцій. Систематичний моніторинг та автоматизація процесів контролю потужності сприяє підвищенню ефективності роботи газових турбін та зменшує ризики, пов’язані з ручним втручанням.

Аналіз набору даних показав діапазон змінних, як-от температура, атмосферний тиск, відносна вологість та розрідження, які впливають на продуктивність електростанції.

У процесі попередньої обробки даних були застосовані методи ідентифікації й обробки пропусків, викидів, дублікатів, нормалізації та відбору ознак.

Висновки, отримані в результаті цього розділу, демонструють, що проведений розвідувальний аналіз і етапи попередньої обробки даних суттєво сприяють підготовці набору даних для подальшого моделювання, створюючи підґрунтя для досягнення високих показників точності прогнозів на наступних етапах дослідження.

4 ПОБУДОВА БАЗОВИХ МОДЕЛЕЙ. ФОРМУВАННЯ БАГАТОШАРОВИХ АНСАМБЛЕВИХ СТРУКТУР. ОЦІНКА РЕЗУЛЬТАТІВ

4.1 Критерії оцінки якості прогнозів

Для оцінки ефективності майбутніх моделей та прогнозів були використані наступні чотири оцінювальні показники.

R2, або ***коефіцієнт детермінації***, – це такий певний статистичний показник у регресійній моделі, що характеризує відсоток дисперсії залежної змінної, який можна пояснити зміною незалежної змінної. Іншими словами, R^2 відображає, наскільки добре регресійна модель описує спостережувані дані. Обчислення R^2 відбувається за описаним нижче алгоритмом:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}, \tag{4.1}$$

де n – загальна кількість спостережень;

y_i – певне реальне значення для *i*-го спостереження;

$\hat{y}_i$ – передбачене значення для *i*-го спостереження;

$\bar{y}_i$ – усереднене значення.

Середньоквадратична похибка, ***RMSE***, є метрикою, яка оцінює, наскільки сильно прогнози моделі відрізняються від фактичних спостережуваних значень, використовуючи Евклідову відстань. Для її обчислення спочатку визначається залишок (відхилення між передбачуваним результатом і реальним значенням) кожної точки даних. Далі обчислюється квадрат кожного залишку, після чого визначається середнє значення квадратів залишків. На завершення береться квадратний корінь з цього середнього значення. Формула для розрахунку кореня середньоквадратичної похибки виглядає так:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{y}_i - y_i\right)^2}\,, \tag{4.2}$$

де n – загальна кількість спостережень;

$\hat{y}_i$ – передбачене значення для *i*-го спостереження;

y_i – певне реальне значення для *i*-го спостереження.

Середня абсолютна похибка (***MAE***) оцінює середній модуль відхилення між передбаченими значеннями та фактичними результатами. На відміну від інших метрик, MAE не підносить похибку до квадрата, що означає, що метрика не надає більшої ваги великим похибкам, як це робить, наприклад, середньоквадратична похибка. Замість цього, MAE надає однакову вагу всім похибкам незалежно від їхнього напрямку (переоцінки чи недооцінки). Ця властивість робить MAE особливо корисною, коли необхідно оцінити загальну величину похибок. Формула для розрахунку середньої абсолютної похибки має вигляд:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{y}_i - y_i\right)^2, \tag{4.3}$$

де n – загальна кількість спостережень;

$\hat{y}_i$ – передбачене значення для *i*-го спостереження;

y_i – певне реальне значення для *i*-го спостереження.

Середня абсолютна похибка у відсотках, ***MAPE***, вимірює середню величину відносної похибки, яку створює модель, і вказує, наскільки в середньому прогнози відхиляються від фактичних значень у відсотковому відношенні. Тобто цей показник дає змогу оцінити якість моделі як відсоткове відхилення. Формула для обчислення MAPE виглядає так:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \cdot 100\%, \tag{4.4}$$

де n – загальна кількість спостережень;

$\hat{y}_i$ – передбачене значення для *i*-го спостереження;

y_i – певне реальне значення для *i*-го спостереження.

Для трьох показників: RMSE, MAE та MAPE, чим менше значення, тим менша різниця між прогнозними даними та фактичними даними, та тим кращий прогнозний ефект моделі. Тобто чим менші ці показники, тим точнішими є прогнози моделі. Тобто чим менші ці показники, тим точнішими є прогнози моделі.

Щодо коефіцієнта детермінації R^2, то його значення лежить у діапазоні від 0 до 1. Значення R^2, близьке до 1, вказує на те, що модель добре пояснює дисперсію фактичних даних і має високу здатність відповідати їм. Навпаки, значення R^2, близьке до 0, вказує на те, що модель не може пояснити дисперсію даних і є менш ефективною.

4.2 Побудова та навчання базових моделей регресії. Оцінка результатів

Першим кроком перед початком моделювання є поділ набору даних на тестувальний та тренувальний. Обрано випадковий піднабір, що складається з 90% рядків очищеного набору даних *powerplant_norm*, який використовується як тренувальний набір для побудови моделей. Решта, що складає 10% рядків, залишається для тестування – рис. 4.1.

```
> # Генерація випадкового вибору рядків для тренувального набору даних (90% від усіх рядків)
> row.number <- sample(1:nrow(powerplant_norm), 0.9 * nrow(powerplant_norm))
> # Створення тренувального набору даних на основі випадково вибраних рядків
> traindata <- powerplant_norm[row.number,]
> # Створення тестового набору даних на основі решти рядків, які не ввійшли до тренувального набору
> testdata <- powerplant_norm[-row.number,]
```

Рисунок 4.1 – Поділ набору даних на тестувальний та тренувальний

4.2.1 Регресійна модель на основі множинної лінійної регресії

Оскільки результати крос-валідації показали, що *модель досягає кращих результатів при включенні всіх чотирьох предикторів*, тому було побудовано відповідну модель *lm.fit* на основі множинної лінійної регресії – рис. 4.2.

```
> # ========= Множинна лінійна регресія  =========
>
> # Побудова моделі
> lm.fit = lm(traindata$Energy_Output~., data = traindata)
> # Виведення статистичного резюме процедури навчання моделі
> summary(lm.fit)

Call:
lm(formula = traindata$Energy_Output ~ ., data = traindata)

Residuals:
       Min         1Q     Median         3Q        Max
-0.0193503 -0.0013866 -0.0000315  0.0013855  0.0079961

Coefficients:
                   Estimate Std. Error  t value Pr(>|t|)
(Intercept)        2.003e-01  4.429e-03   45.222  < 2e-16 ***
Temperature       -2.076e+00  1.767e-02 -117.469  < 2e-16 ***
Exhaust_Vacuum    -1.709e+01  5.150e-01  -33.188  < 2e-16 ***
Ambient_Pressure   6.946e-02  9.888e-03    7.025  2.3e-12 ***
Relative_Humidity -4.881e-03  1.429e-04  -34.157  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.001985 on 8569 degrees of freedom
Multiple R-squared:  0.9298,    Adjusted R-squared:  0.9297
F-statistic: 2.836e+04 on 4 and 8569 DF,  p-value: < 2.2e-16
```

Рисунок 4.2 – Модель *lm.fit* на основі множинної лінійної регресії

Згідно з результатами, значення коефіцієнта детермінації R^2 становить 0.9297, що означає, що модель *lm.fit* пояснює 93% дисперсії «*Energy_Output*». Залишкове стандартне відхилення є досить малим – 0.00199, що вказує на доволі непогану точність прогнозів.

Отже, нехай модель *lm.fit* попередньо вважається якісною. Для подальшої перевірки її ефективності було виконано прогнозування на тестових даних, а також обчислено показники RMSE, MAE, MAPE та R^2, – рис. 4.3.

```
> # Прогнозування
> test.lm <- predict(lm.fit, newdata = testdata)
>
> # Обчислення показників
> rmse.lm <- sqrt(sum((test.lm - testdata$Energy_Ou
> mae.lm <- sum(abs(test.lm - testdata$Energy_Outpu
> mape.lm <- sum(abs((test.lm - testdata$Energy_Out
> R2.lm <- summary(lm.fit)$r.squared
>
> # Результати для моделі
> c(R2=R2.lm, RMSE = rmse.lm, MAE = mae.lm, MAPE =
          R2         RMSE          MAE         MAPE
0.929778661 0.002043285 0.001572269 0.791299715
```

Рисунок 4.3 – Тестування та оцінка якості моделі *lm.fit*

Як підсумок, модель *lm.fit* демонструє високі результати: значення $R^2 = 0.93$ свідчить про значну здатність моделі пояснювати дисперсію даних, що вказує на хорошу узгодженість із реальними даними. Крім того, низькі значення показників похибки, як-от RMSE = 0.0020, MAE = 0.0016, та MAPE = 0.79%, підтверджують високу точність прогнозів моделі. Загалом, ці результати свідчать про високу якість прогнозування.

Графічно можна візуалізувати результати прогнозування моделі *lm.fit* – рис. 4.4.

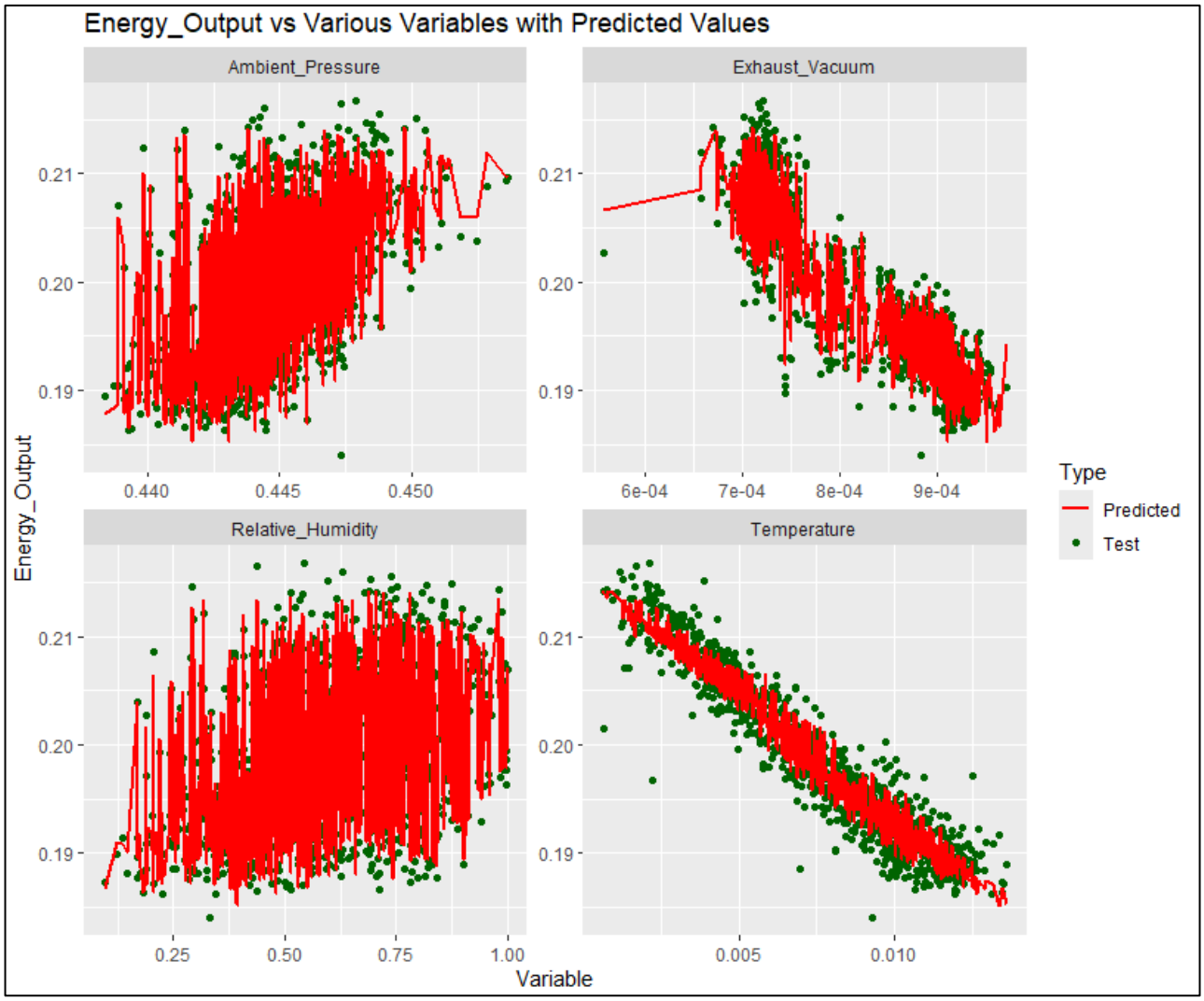

Рисунок 4.4 – Графічна демонстрація результатів прогнозування *lm.fit*

4.2.2 Регресійна модель на основі дерева рішень

Регресійна модель на основі дерева рішень була створена за допомогою функції *rpart()*. У процесі моделювання були включені всі предиктори з набору даних – рис. 4.5.

```
> # ========= Регресійна модель на основі дерева рішень  =========
>
> # Побудова моделі
> dt.fit <- rpart(traindata$Energy_Output~., data = traindata)
> plot(dt.fit, uniform = TRUE,
+      main = "Decision Tree Regression")
> text(dt.fit, use.n = TRUE, cex = .7)
> summary(dt.fit)
Call:
rpart(formula = traindata$Energy_Output ~ ., data = traindata)
  n= 8574

          CP nsplit rel error    xerror        xstd
1 0.71878417      0 1.0000000 1.0001751 0.010552180
2 0.07246354      1 0.2812158 0.2848891 0.004051563
3 0.07050222      2 0.2087523 0.2251676 0.003486563
4 0.02287851      3 0.1382501 0.1411920 0.002519312
5 0.01000000      4 0.1153716 0.1189669 0.002348827

Variable importance
      Temperature    Exhaust_Vacuum  Ambient_Pressure Relative_Humidity
               45                32                14                 9


     Node number 5: 1614 observations
       mean=0.1975586, MSE=8.222494e-06

     Node number 6: 1939 observations
       mean=0.2038162, MSE=6.897693e-06

     Node number 7: 1595 observations
       mean=0.2100389, MSE=8.266411e-06

     Node number 8: 2005 observations
       mean=0.1904159, MSE=4.262795e-06

     Node number 9: 1421 observations
       mean=0.1940523, MSE=4.980272e-06
```

Рисунок 4.5 – Регресійна модель *dt.fit* на основі дерева рішень

Видно, що з кожним рівнем дерева рішень похибка моделі зменшується, це свідчить про покращення точності прогнозів.

Побудоване дерево графічно виглядає так – рис. 4.6.

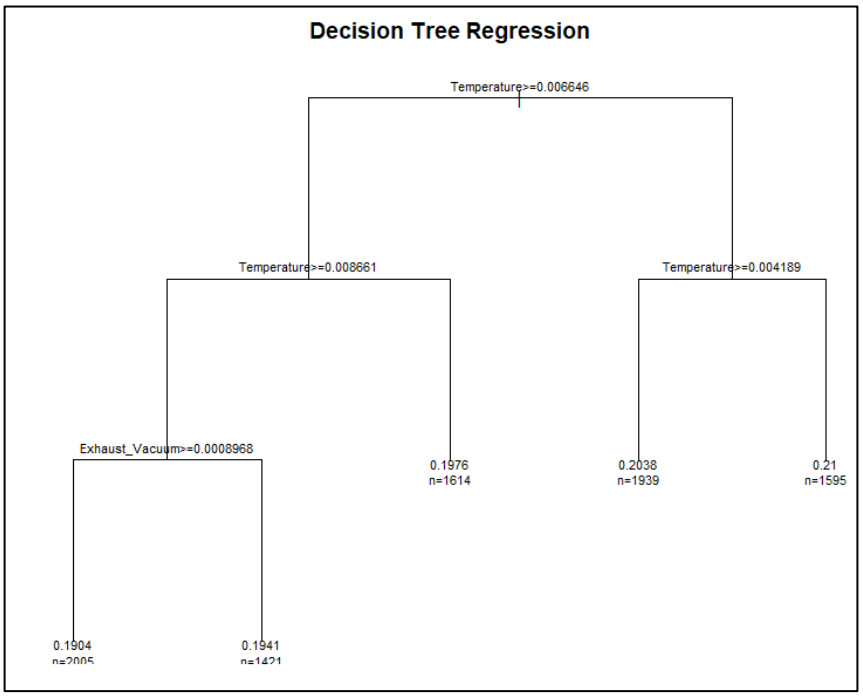

Рисунок 4.6 – Графік дерева рішень *dt.fit*

Нехай попередньо модель *dt.fit* вважається якісною. Для перевірки її ефективності було виконано прогнозування на тестових даних. Обчислено показники RMSE, MAE, MAPE та R^2 – рис. 4.7.

```
> # Прогнозування
> test.dt <- predict(dt.fit, newdata = testdata)
>
> # Обчислення показників
> rmse.dt <- sqrt(sum((test.dt - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
> mae.dt <- sum(abs(test.dt - testdata$Energy_Output))/length(testdata$Energy_Output)
> mape.dt <- sum(abs((test.dt - testdata$Energy_Output)/test.dt))/length(testdata$Energy_Output)*100
> residuals.dt <- testdata$Energy_Output - test.dt
> SST.dt <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
> SE.dt <- sum(residuals.dt^2)
> R2.dt <- 1 - (SE.dt / SST.dt)
>
> # Результати для моделі
> c(R2=R2.dt, RMSE = rmse.dt, MAE = mae.dt, MAPE = mape.dt)
        R2        RMSE         MAE        MAPE
0.878035145 0.002633456 0.002071454 1.037881538
```

Рисунок 4.7 – Тестування та оцінка якості моделі *dt.fit*

Отже, модель *dt.fit* демонструє непогану точність: коефіцієнт детермінації $R^2 = 0.8780$ вказує на те, що модель пояснює майже 88% варіативності даних. Середньоквадратична похибка і середня абсолютна похибка є дуже малими – 0.0026 та 0.0021 відповідно, що свідчить про хорошу точність прогнозів. Відносна

похибка прогнозу становить 1.04%, що додатково підтверджує ефективність моделі. Однак є місце для покращення, адже результати задовільні, але не високі.

Графічно можна візуалізувати результати прогнозування – рис. 4.8.

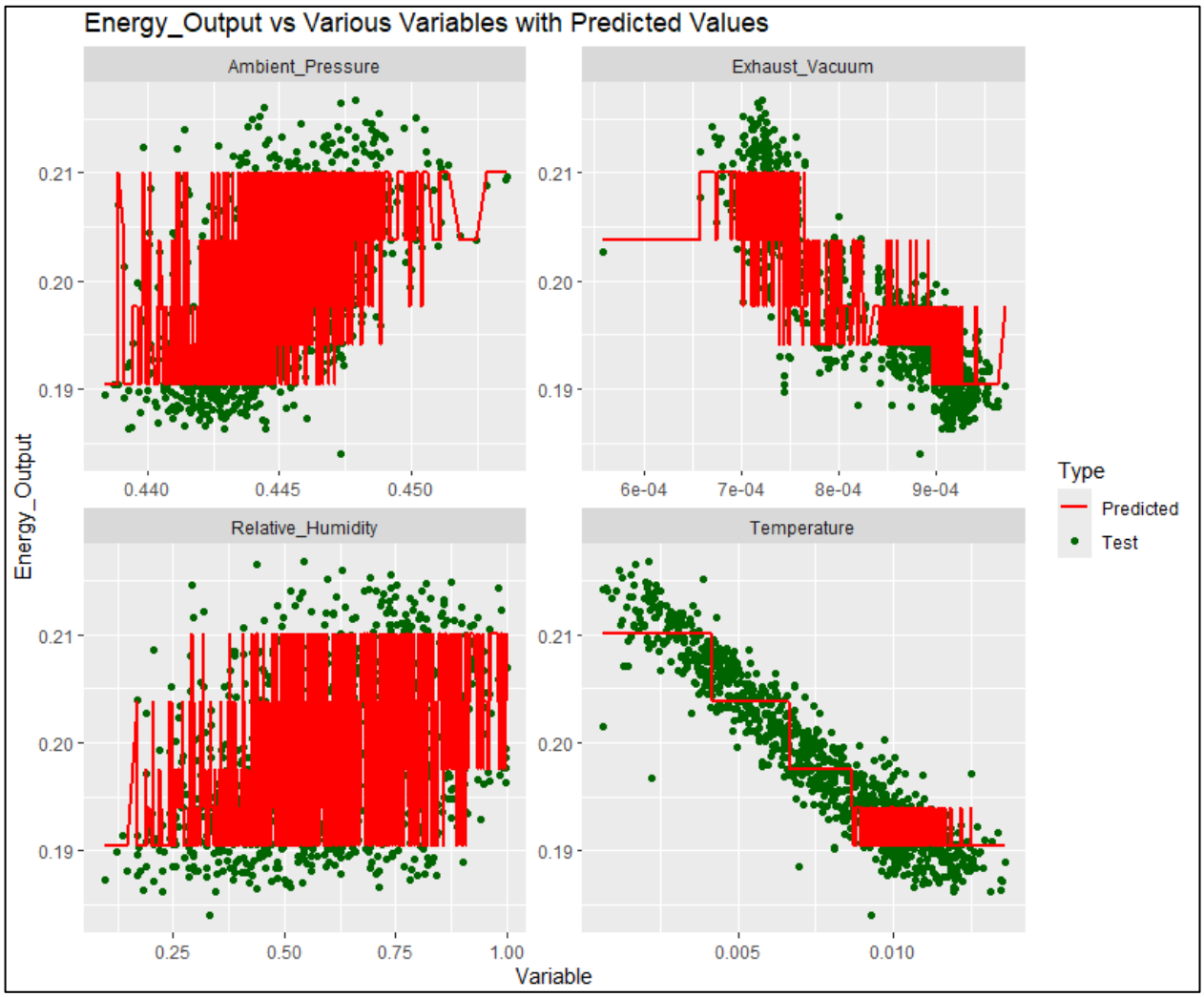

Рисунок 4.8 – Графічна демонстрація результатів прогнозування *dt.fit*

4.2.3 Регресійна модель на основі випадкового лісу

Створено модель випадкового лісу *rf.fit* за допомогою функції *randomForest()*. Модель складається з 500 дерев, при кожному розподілі вибирається 1 змінна. За результатами тренування вона пояснює 96% дисперсії даних, що свідчить про її високу ефективність, – рис. 4.9.

```
> # ========= Регресійна модель на основі випадкового лісу  =========
>
> # Побудова моделі
> rf.fit <- randomForest(traindata$Energy_Output~., data=traindata)
> print(rf.fit)

Call:
 randomForest(formula = traindata$Energy_Output ~ ., data = traindata)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 1

          Mean of squared residuals: 2.082323e-06
                    % Var explained: 96.29
```

Рисунок 4.9 – Регресійна модель *rf.fit* на основі випадкового лісу

Нехай попередньо модель *rf.fit* вважається якісною. Для перевірки її ефективності було виконано прогнозування на тестових даних. Обчислено показники RMSE, MAE, MAPE та R^2 для оцінки якості моделі – рис. 4.10.

```
> # Прогнозування
> test.rf <- predict(rf.fit, newdata = testdata)
>
> # Обчислення показників
> rmse.rf <- sqrt(sum((test.rf - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
> mae.rf <- sum(abs(test.rf - testdata$Energy_Output))/length(testdata$Energy_Output)
> mape.rf <- sum(abs((test.rf - testdata$Energy_Output)/test.rf))/length(testdata$Energy_Output)*100
> residuals.rf <- testdata$Energy_Output - test.rf
> SST.rf <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
> SE.rf <- sum(residuals.rf^2)
> R2.rf <- 1 - (SE.rf / SST.rf)
>
> # Результати для моделі
> c(R2=R2.rf, RMSE = rmse.rf, MAE = mae.rf, MAPE = mape.rf)
        R2        RMSE         MAE        MAPE
0.956638631 0.001570219 0.001077251 0.542219078
```

Рисунок 4.10 – Тестування та оцінка якості моделі *rf.fit*

Отже, модель випадкового лісу показала відмінні результати: коефіцієнт детермінації R^2 = 0.9566 свідчить про те, що модель пояснює понад 96% варіативності даних. Середньоквадратична похибка і середня абсолютна похибка є дуже низькими – 0.0016 і 0.0011 відповідно, що вказує на високу точність прогнозів. Відносна похибка прогнозу також дуже мала і становить лише 0.54%, що підтверджує високу якість моделі.

Графічно можна візуалізувати результати прогнозування – рис. 4.11.

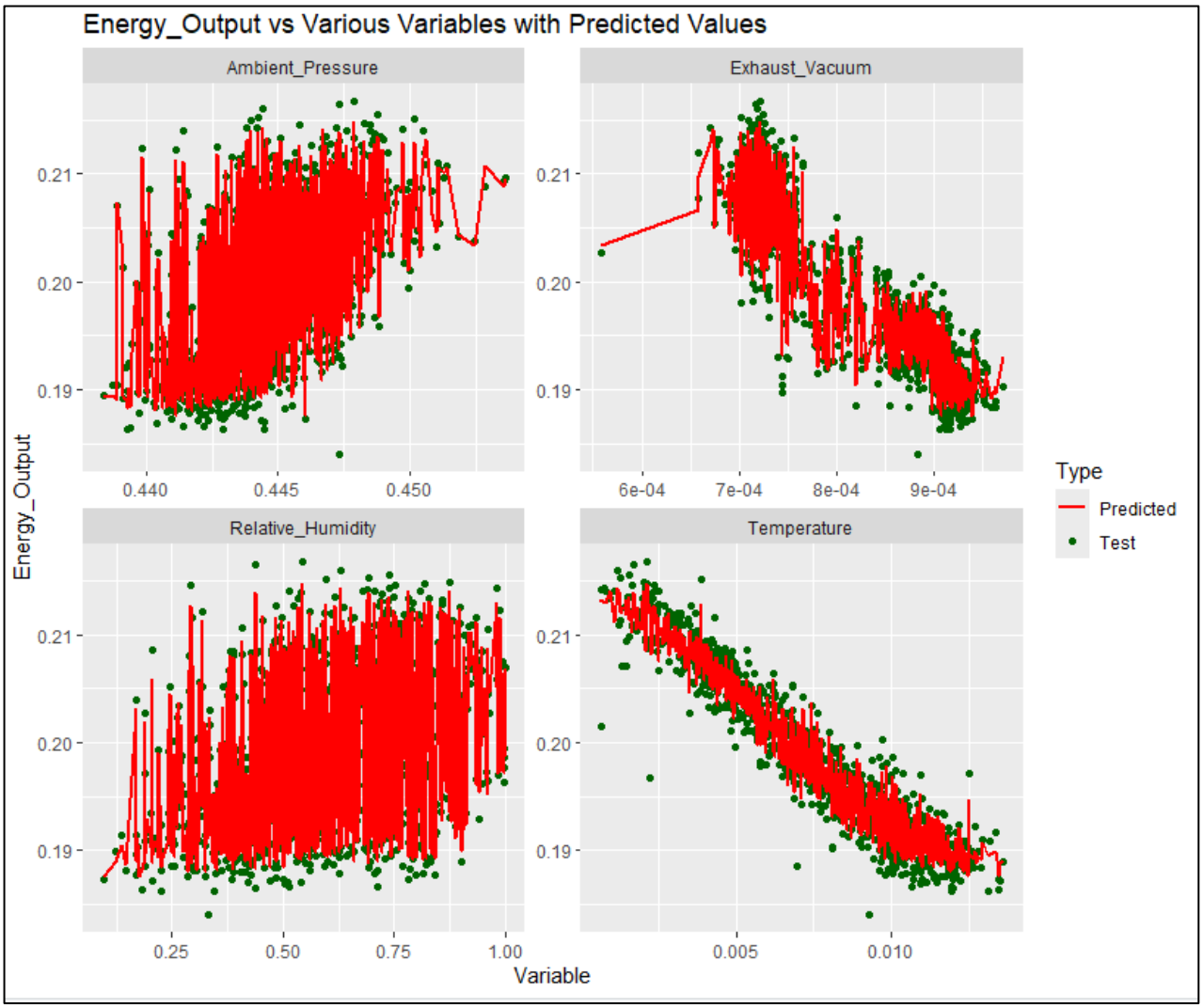

Рисунок 4.11 – Графічна демонстрація результатів прогнозування *rf.fit*

4.2.4 Регресійна модель на основі методу опорних векторів

Створено регресійну модель на основі опорних векторів для прогнозування «*Energy_Output*» на основі всіх інших змінних.

Модель використовує радіальну ядрову функцію з параметрами: *cost = 1*, *gamma = 0.25* та *epsilon = 0.1*. Кількість опорних векторів у моделі становить 5508 – рис. 4.12.

```
> # ========= Регресійна модель на основі опорних векторів   =========
>
> # Побудова моделі
> svr.fit <- svm(traindata$Energy_Output~., data=traindata)
> print(svr.fit)

Call:
svm(formula = traindata$Energy_Output ~ ., data = traindata)


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.25
    epsilon:  0.1


Number of Support Vectors:  5508
```

Рисунок 4.12 – Регресійна модель *svr.fit* на основі опорних векторів

Нехай попередньо модель вважається якісною. Було виконано прогнозування на тестових даних за допомогою *svr.fit*. Обчислено показники RMSE, MAE, MAPE, R^2 для її оцінки – рис. 4.13.

```
> # Прогнозування
> test.svr <- predict(svr.fit, newdata = testdata)
>
> # Обчислення показників
> rmse.svr <- sqrt(sum((test.svr - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
> mae.svr <- sum(abs(test.svr - testdata$Energy_Output))/length(testdata$Energy_Output)
> mape.svr <- sum(abs((test.svr - testdata$Energy_Output)/test.svr))/length(testdata$Energy_Output)*100
> residuals.svr <- testdata$Energy_Output - test.svr
> SST.svr <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
> SE.svr <- sum(residuals.svr^2)
> R2.svr <- 1 - (SE.svr / SST.svr)
>
> # Результати для моделі
> c(R2=R2.svr, RMSE = rmse.svr, MAE = mae.svr, MAPE = mape.svr)
         R2        RMSE         MAE        MAPE
0.937295466 0.001888244 0.001364954 0.686135584
```

Рисунок 4.13 – Тестування та оцінка якості моделі *svr.fit*

Отже, модель опорних векторів показала хороші результати: коефіцієнт детермінації R^2 = 0.9373 свідчить про те, що модель описує 94% варіативності даних. Середньоквадратична похибка і середня абсолютна похибка складають 0.0019 і 0.0014 відповідно, що свідчить про високу точність прогнозів. Відносна похибка прогнозу становить 0.69%, що також підтверджує ефективність моделі.

Графічно можна візуалізувати результати прогнозування – рис. 4.14.

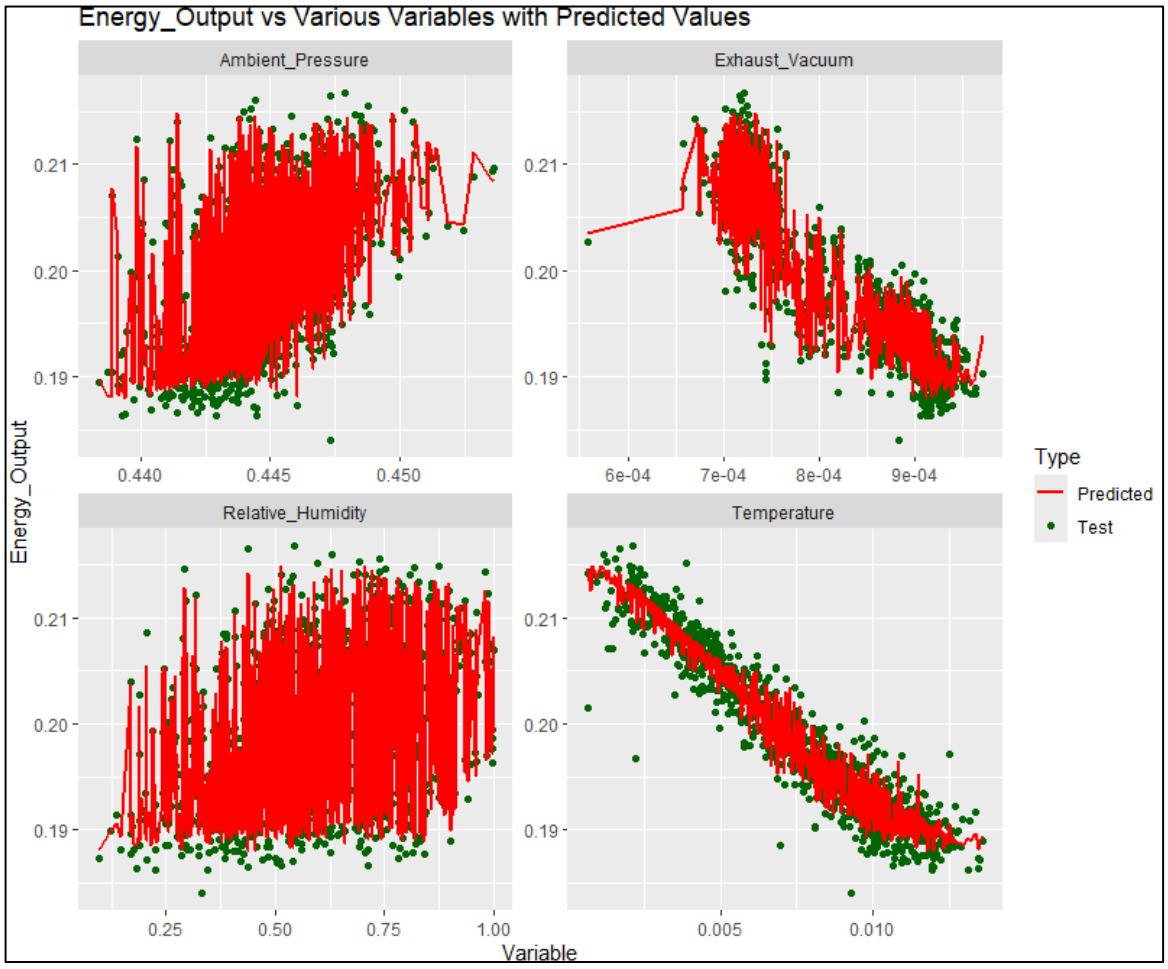

Рисунок 4.14 – Графічна демонстрація результатів прогнозування *svr.fit*

4.2.5 Регресійна модель на основі KNN-R

Створено модель регресії на основі 5 найближчих сусідів за замовчуванням для прогнозування «*Energy_Output*» на основі інших змінних – рис. 4.15.

```
> # ========== Модель KNN-R ==========
>
> # Побудова моделі
> knn.fit <- knnreg(traindata$Energy_Output~., data=traindata)
> knn.fit
5-nearest neighbor regression model
```

Рисунок 4.15 – Регресійна модель *knn.fit* на основі KNN-R

Нехай попередньо модель вважається якісною. Виконано прогнозування на тестових даних за допомогою *knn.fit*. Обчислено показники RMSE, MAE, MAPE, R^2 для її оцінки – рис. 4.16.

```
> # Прогнозування
> test.knn <- predict(knn.fit, newdata = testdata)
>
> # Обчислення показників
> rmse.knn <- sqrt(sum((test.knn - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
> mae.knn <- sum(abs(test.knn - testdata$Energy_Output))/length(testdata$Energy_Output)
> mape.knn <- sum(abs((test.knn - testdata$Energy_Output)/test.knn))/length(testdata$Energy_Output)*100
> residuals.knn <- testdata$Energy_Output - test.knn
> SST.knn <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
> SE.knn <- sum(residuals.knn^2)
> R2.knn <- 1 - (SE.knn / SST.knn)
>
> # Результати для моделі
> c(R2=R2.knn, RMSE = rmse.knn, MAE = mae.knn, MAPE = mape.knn)
        R2         RMSE          MAE         MAPE
0.879797949 0.002614355 0.002048412 1.030423445
```

Рисунок 4.16 – Тестування та оцінка якості моделі *knn.fit*

Отже, модель показала добрі результати: коефіцієнт детермінації становить 0.8798, що означає пояснення 88% варіативності. Середньоквадратична похибка і середня абсолютна похибка складають 0.0026 і 0.0020 відповідно, що свідчить про хорошу точність прогнозів. Відносна похибка прогнозу дорівнює 1.03%. Однак є місце для покращення, адже результати задовільні, але не високі.

Графічно можна візуалізувати результати прогнозування – рис. 4.17.

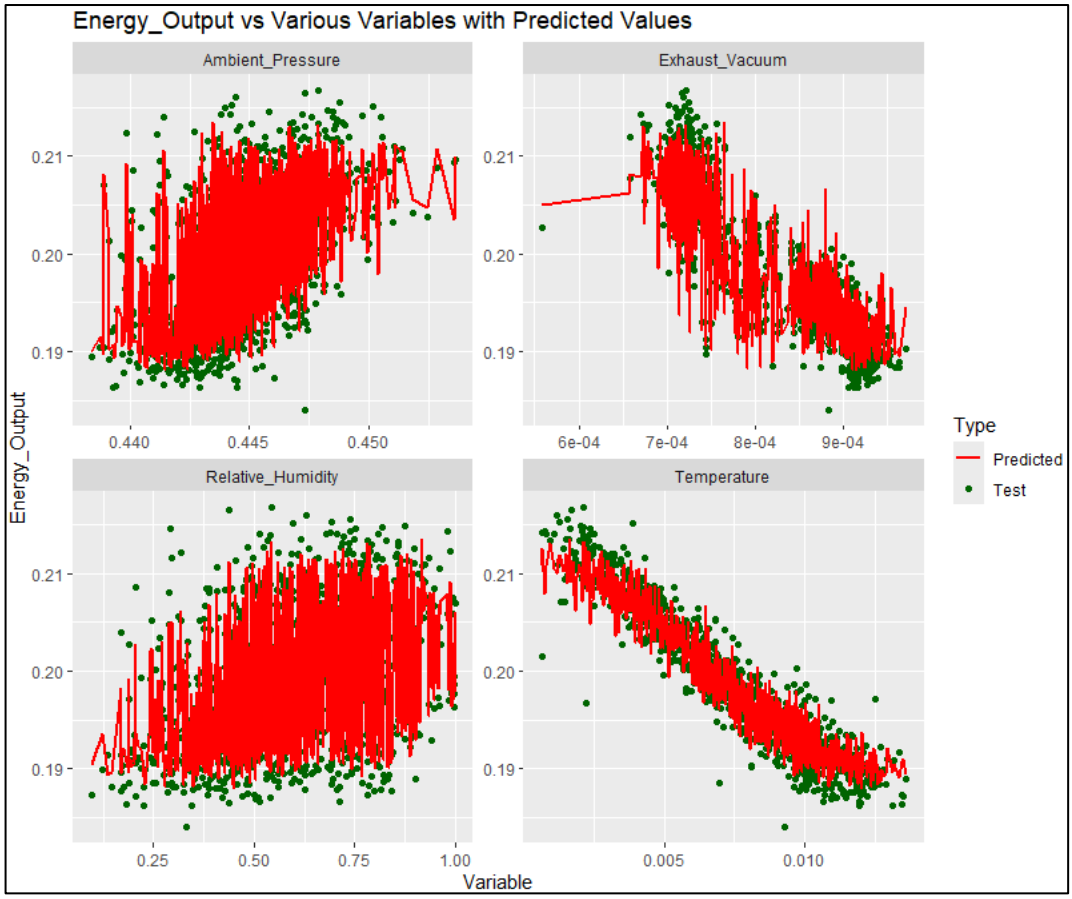

Рисунок 4.17 – Графічна демонстрація результатів прогнозування *knn.fit*

4.2.6 Регресійна модель на основі штучної НМ

Створено модель штучної нейронної мережі для прогнозування «*Energy_Output*» на основі інших змінних, що складається з двох прихованих шарів та згладжувального параметра. Модель була реалізована за допомогою функції *neuralnet()* – рис. 4.18.

```
> # ========= Модель на основі штучної НМ =========
>
> # Побудова моделі
> ann.fit <- neuralnet(traindata$Energy_Output~., hidden = 2, act.fct = "logistic", data=traindata)
> plot(ann.fit)
```

Рисунок 4.18 – Регресійна модель *ann.fit* на основі штучної НМ

Структура моделі штучної нейронної мережі – рис. 4.19.

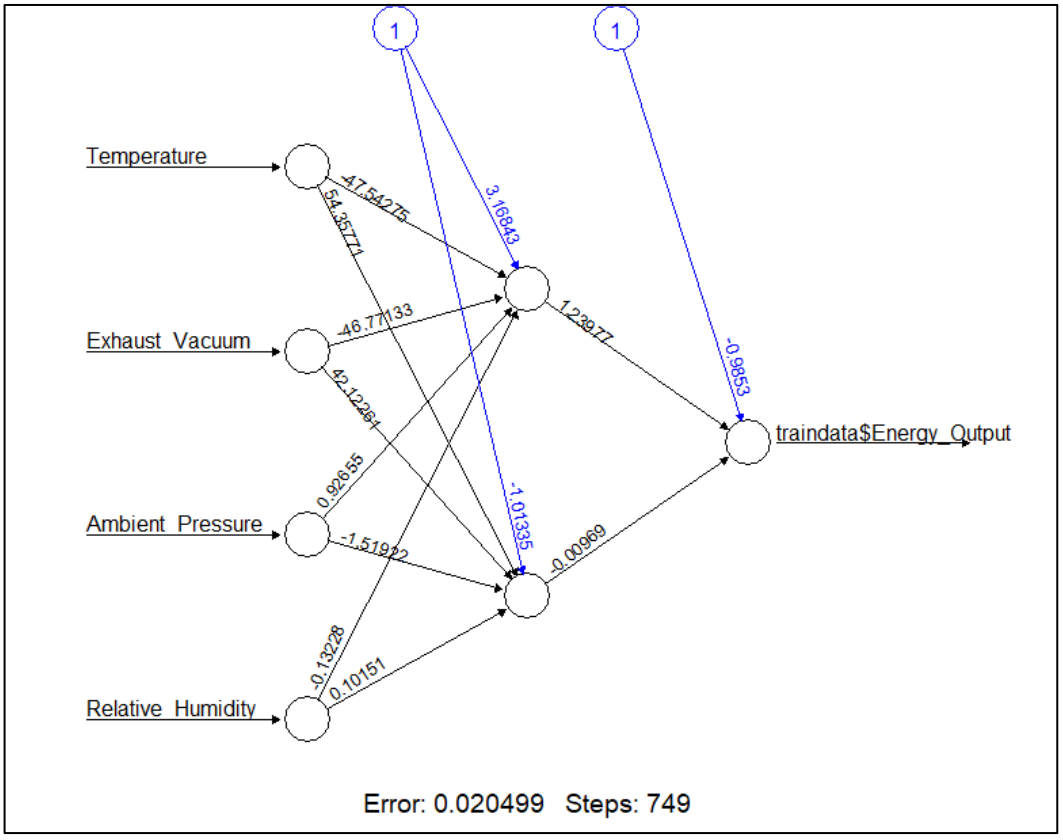

Рисунок 4.19 – Структура моделі *ann.fit*

Нехай попередньо модель вважається якісною. Виконано прогнозування на тестових даних за допомогою *ann.fit*. Обчислено показники RMSE, MAE, MAPE, R^2 для її оцінки – рис. 4.20.

```
> # Прогнозування
> test.ann <- predict(ann.fit, newdata = testdata)
>
> # Обчислення показників
> rmse.ann <- sqrt(sum((test.ann - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
> mae.ann <- sum(abs(test.ann - testdata$Energy_Output))/length(testdata$Energy_Output)
> mape.ann <- sum(abs((test.ann - testdata$Energy_Output)/test.ann))/length(testdata$Energy_Output)*100
> residuals.ann <- testdata$Energy_Output - test.ann
> SST.ann <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
> SE.ann <- sum(residuals.ann^2)
> R2.ann <- 1 - (SE.ann / SST.ann)
>
> # Результати для моделі
> c(R2=R2.ann, RMSE = rmse.ann, MAE = mae.ann, MAPE = mape.ann)
        R2       RMSE        MAE       MAPE
0.913416258 0.002218845 0.001706182 0.859698151
```

Рисунок 4.20 – Тестування та оцінка якості моделі *ann.fit*

Отже, модель продемонструвала високі результати: коефіцієнт детермінації становить 0.9134, що означає пояснення 91% варіативності. Середньоквадратична похибка і середня абсолютна похибка складають 0.0022 і 0.0017 відповідно, що свідчить про хорошу точність прогнозів. Відносна похибка прогнозу дорівнює 0.86%.

Графічно можна візуалізувати результати прогнозування – рис. 4.21.

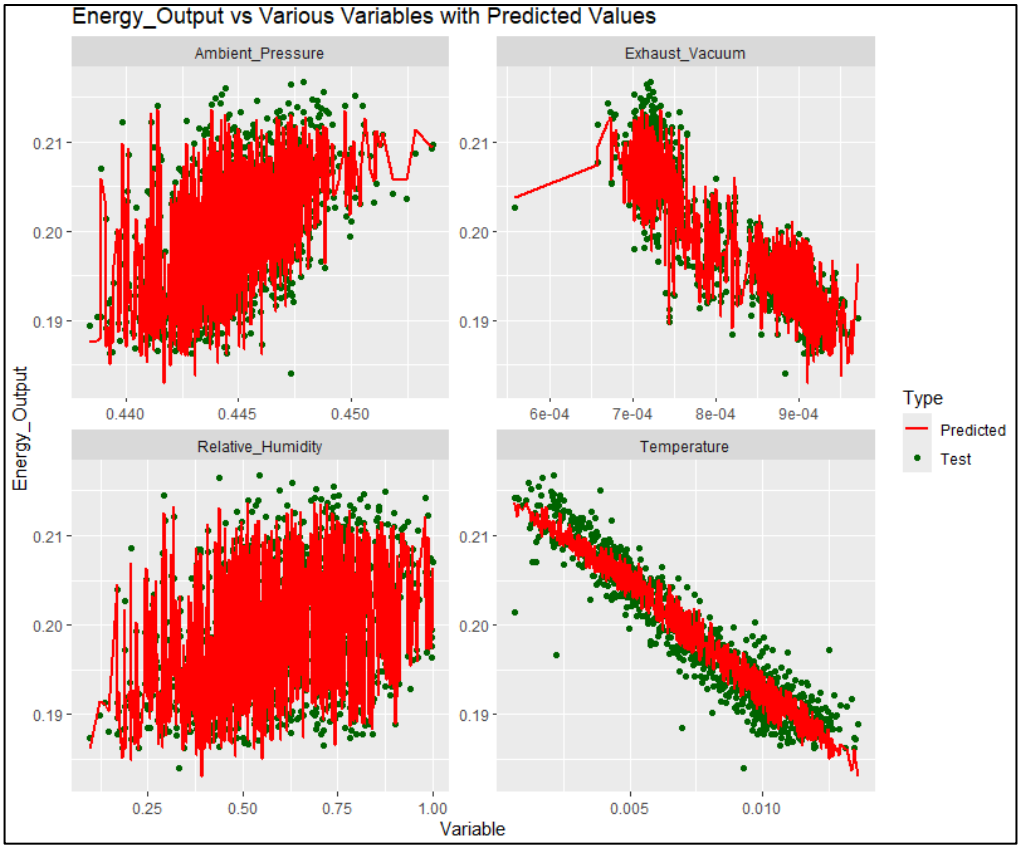

Рисунок 4.21 – Графічна демонстрація результатів прогнозування *ann.fit*

Результати прогнозування кожної з базових моделей регресії узагальнено в таблицю – табл. 4.1.

Таблиця 4.1 – Результати прогнозування кожної з базових моделей регресії

Тип моделі	**Якість моделі**	**Якість прогнозу**		
	R^2	**RMSE**	**MAE**	**MAPE (%)**
Регресійна модель на основі множинної лінійної регресії	0.93	0.0020	0.0016	0.79
Регресійна модель на основі дерева рішень	0.88	0.0026	0.0021	1.04
Регресійна модель на основі випадкового лісу	0.96	0.0016	0.0011	0.54
Регресійна модель на основі методу опорних векторів	0.94	0.0019	0.0014	0.69
Регресійна модель на основі KNN-R	0.88	0.0026	0.0020	1.03
Регресійна модель на основі штучної НМ	0.91	0.0022	0.0017	0.86

Отже, базові моделі показали непогані результати на обраному наборі даних. ***Найкраще*** відпрацювала *модель на основі випадкового лісу*. ***Найгірші результати*** в даному контексті показали регресійна *модель на основі дерева рішень* та регресійна *модель на основі KNN-R*.

4.3 Підбір оптимальної структури дворівневого ансамблю. Оцінка результатів

Як відомо, загальний підхід, що застосовується дослідниками для побудови гетерогенних ансамблів, тобто таких, що містять різнорідні базові моделі, передбачає застосування стекінгу до недонавчених моделей і бегінгу до перенавчених. Стекінг, як вже відомо, спрямований на мінімізацію зміщення у моделей із низькою дисперсією та високим зміщенням, а бегінг – на зменшення

дисперсії в моделях із високою дисперсією та низьким зміщенням. Багатошарові структури, що будуються на основі таких «класичних» гетерогенних ансамблів, мають можливість впливати на ці два складники похибки одночасно [57]. Сама архітектура інтелектуальної системи на основі такого «класичного» підходу, відповідно, виглядає так – рис. 4.22.

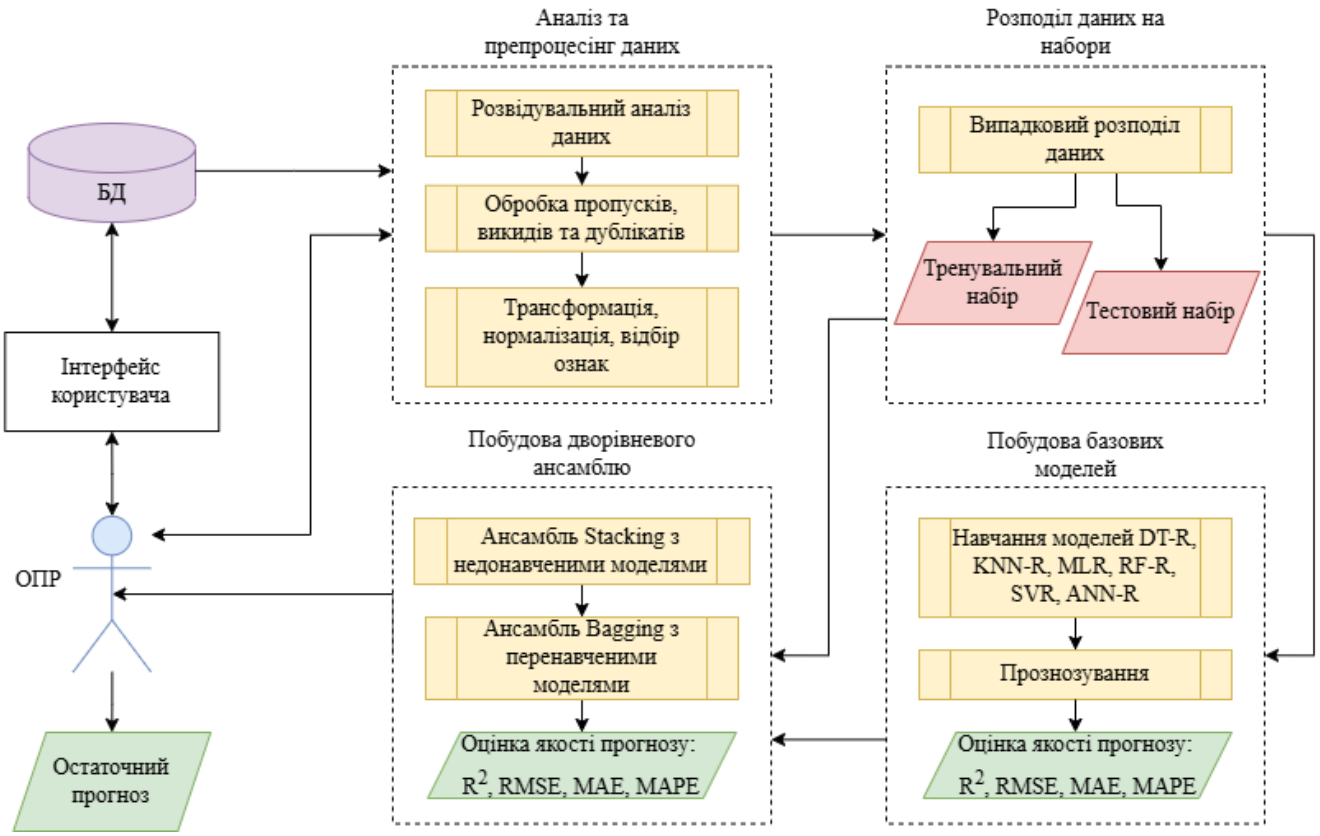

Рисунок 4.22 – Архітектура інтелектуальної системи прогнозування на основі «класичного» підходу

Такий варіант дворівневої ансамблевої структури в складі системи був реалізований програмно з метою оцінки ефективності цього підходу в задачі регресійного прогнозування потужності електростанції на основі власного набору даних.

Для визначення базових моделей було обчислено, відповідно, зміщення та дисперсію кожної з них – рис. 4.23.

```
> # Перетворення результатів у фрейм даних
> var.bias.resdf <- as.data.frame(t(var.bias.res))
> colnames(var.bias.resdf) <- c("Bias", "Variance")
> var.bias.resdf
             Bias     Variance
lm  -3.136184e-06 5.283102e-05
dt   9.487919e-05 4.974045e-05
rf  -9.609841e-06 5.355618e-05
svr  3.064479e-05 5.540902e-05
knn -1.382169e-04 4.767433e-05
ann  1.763085e-06 5.176174e-05
```

Рисунок 4.23 – Зміщення та дисперсія кожної базової моделі

Також важливо пам’ятати, аби моделі, які додаються в певний ансамбль, мали низьку кореляцію між собою, тобто, щоб кожна максимально використала свої прогностичні можливості.

Отже, в *перший шар, стекінг,* додаються недонавчені моделі з низькою дисперсією та високим зміщенням, а також не дуже високим значенням R^2, тобто:

– регресійна модель на основі дерева рішень;

– регресійна модель на основі KNN-R.

Результати першого рівня передаються як вхід для *другого шару, бегінгу*, де застосовуються перенавчені моделі з високою дисперсією та низьким зміщенням, а саме:

– регресійна модель на основі множинної лінійної регресії;

– регресійна модель на основі випадкового лісу;

– регресійна модель на основі методу опорних векторів;

– регресійна модель на основі штучної НМ.

Як підсумок, дворівнева ансамблева структура для прогнозування кількості електричної енергії, що генерується за одну годину електростанцією комбінованого циклу, в цьому випадку виглядає так – рис. 4.24.

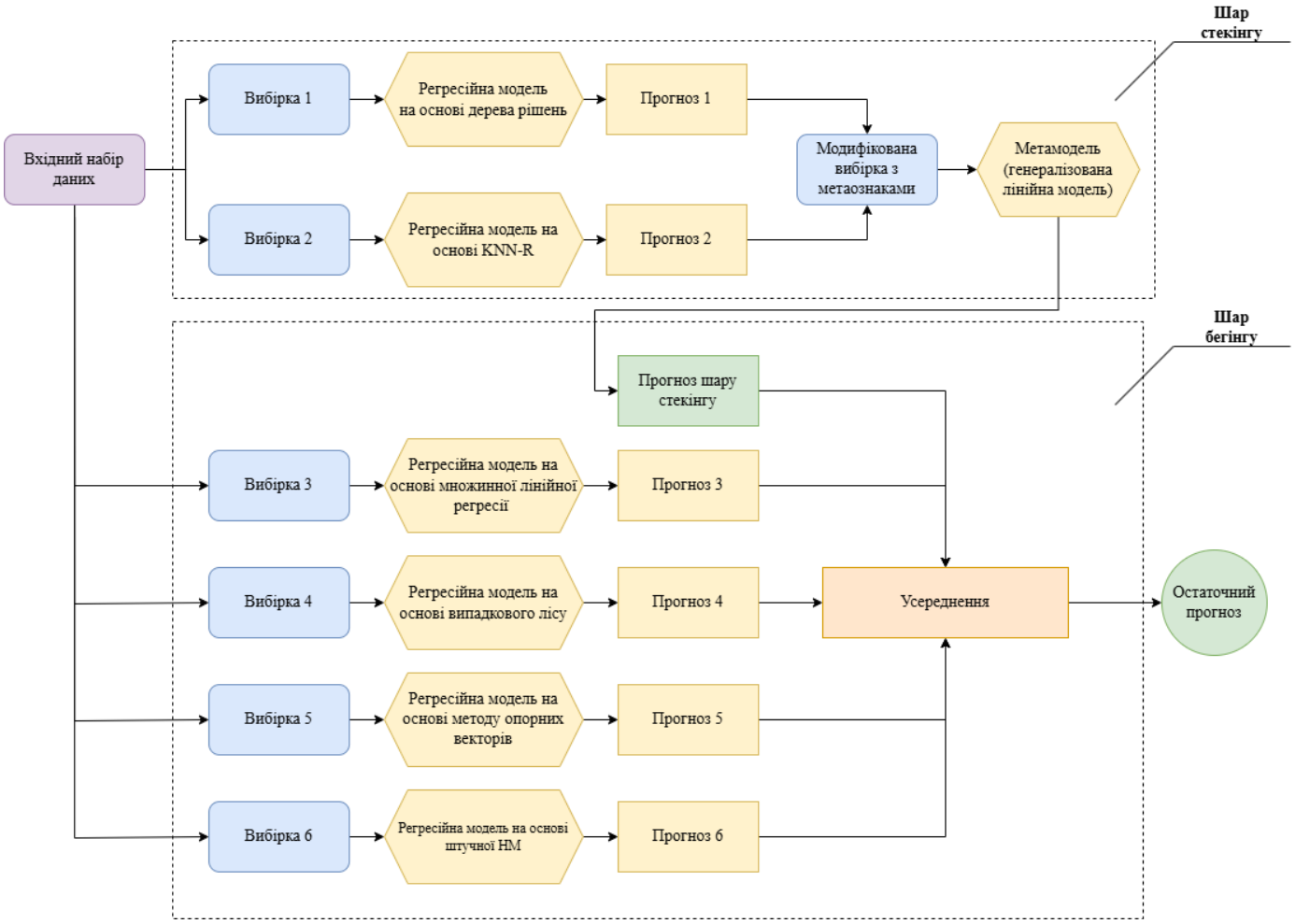

Рисунок 4.24 – Дворівнева ансамблева структура на основі «класичних» гетерогенних ансамблів

Програмно спочатку було виконано попереднє збереження чинних тестових даних та прогнозів базових моделей у змінну *ens1.preds* – рис. 4.25.

```
> # Збереження testdata в ens1.preds
> ens1.preds <- testdata
>
> # Додавання стовпців з прогнозами
> ens1.preds <- data.frame(
+   ens1.preds,                # Існуючі дані з testdata
+   test.lm = test.lm,       # Прогнози лінійної регресії
+   test.dt = test.dt,       # Прогнози дерев рішень
+   test.rf = test.rf,       # Прогнози випадкового лісу
+   test.svr = test.svr,     # Прогнози SVR
+   test.knn = test.knn,     # Прогнози KNN
+   test.ann = test.ann      # Прогнози нейронної мережі
+ )
```

Рисунок 4.25 – Додавання тестових даних та прогнозів у змінну *ens1.preds*

На першому етапі створюється стекінговий шар, який використовує недонавчені базові моделі.

Передусім визначається контрольна схема навчання для побудови моделі. Потім обираються моделі, які будуть використані для створення першого шару. Як вказано попередньо, для цього використовуються такі базові моделі, як регресійна модель на основі методу найближчих сусідів та регресійна модель на основі дерева рішень. Для налаштування самого стекінгу використовується контрольна схема з крос-валідацією – рис. 4.26.

```
> # ===== Перший шар =====
> ens1.control <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE)

> ens1.stack.layer.preds <-  c('test.knn','test.dt','test.ann')
> ens1.stack.layer.mods <- caretList(ens1.preds[,ens1.stack.layer.preds], ens1.preds$Energy_Output,
+                     trControl=ens1.control, methodList=c("glm"))
>
> # Налаштування ens1.stack.layer.control для регресії
> ens1.stack.layer.control <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE)
```

Рисунок 4.26 – Налаштування контрольних схем та вибір моделей

На основі отриманих прогнозів базових моделей створюється метамодель шару *ens1.stack.layer* з використанням генералізованої лінійної моделі – рис. 4.27.

```
> # Створення стекової моделі
> ens1.stack.layer <- caretStack(ens1.stack.layer.mods, method="glm", trControl=ens1.stack.layer.control)
> ens1.stack.layer
The following models were ensembled: glm

caret::train model:
Generalized Linear Model

953 samples
  1 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 858, 858, 858, 858, 857, 858, ...
Resampling results:

  RMSE         Rsquared   MAE
  0.002078893  0.9247595  0.001599073


Final model:

Call:  NULL

Coefficients:
(Intercept)          glm
  8.994e-05    9.996e-01

Degrees of Freedom: 952 Total (i.e. Null);  951 Residual
Null Deviance:      0.05419
Residual Deviance: 0.004145     AIC: -9055
```

Рисунок 4.27 – Створення генералізованої лінійної моделі як метамоделі

Отримані результати навчання шару стекінгу використовуються для прогнозування на тестовому наборі даних. Щоб оцінити точність отриманих прогнозів, обчислюються основні показники, як-от RMSE, MAE, MAPE та коефіцієнт детермінації, R^2, – рис. 4.28.

```
> # Прогнозування за допомогою стекової моделі
> ens1.stack.layer.test <- predict(ens1.stack.layer
>
> # Обчислення показників
> ens1.stack.layer.rmse <- sqrt(sum((ens1.stack.lay
Output))
> ens1.stack.layer.mae <- sum(abs(ens1.stack.layer.
t)
> ens1.stack.layer.mape <- sum(abs((ens1.stack.laye
gth(ens1.preds$Energy_Output) * 100
> ens1.stack.layer.residuals <- ens1.preds$Energy_O
> ens1.stack.layer.SST <- sum((ens1.preds$Energy_Ou
> ens1.stack.layer.SE <- sum(ens1.stack.layer.resid
> ens1.stack.layer.R2 <- 1 - (ens1.stack.layer.SE /
>
> # Результати для моделі
> c(R2 = ens1.stack.layer.R2, RMSE = ens1.stack.lay
        R2        RMSE         MAE        MAPE
0.924015021 0.002078608 0.001590870 0.799588227
```

Рисунок 4.28 – Прогнозування та оцінка роботи *ens1.stack.layer*

Отже, результати прогнозування першого шару *ens1.stack.layer* візуально показано в таблиці для зручності сприйняття – табл. 4.2.

Таблиця 4.2 – Результати прогнозування першого шару методом стекінгу

Тип моделі	**Якість моделі**	**Якість прогнозу**		
	R^2	**RMSE**	**MAE**	**MAPE (%)**
Регресійна модель на основі дерева рішень	0.88	0.0026	0.0021	1.04
Регресійна модель на основі KNN-R	0.88	0.0026	0.0020	1.03
Результівний шар стекінгу	0.92	0.0021	0.0016	0.80

Дійсно, застосування стекінгу призвело до покращення прогнозування. Регресійні моделі на основі дерева рішень і KNN-R мали однакову точність 0.88, але в стекінгу точність зросла до 0.92. Похибки RMSE і MAE також зменшились.

Значення MAPE знизилось до 0.80%, що свідчить про покращену стабільність і точність прогнозів.

Після цього, отримані прогнози шару стекінгу *ens1.stack.layer.test* додаються до основного тестового набору даних *ens1.preds* для подальшого використання в шарі №2, бегінгу, – рис. 4.29.

```
> ens1.preds$ens1.stack.layer.test <- ens1.stack.layer.test$pred
```

Рисунок 4.29 – Додавання прогнозів шару стекінгу до набору даних *ens1.preds*

Виконано створення другого шару «класичного» варіанту дворівневої ансамблевої структури – бегінгу, який включає прогнози перенавчених моделей. У цьому випадку, серед прогнозів присутні прогнози першого шару, стекінгу, (*ens1.stack.layer.test*), лінійної регресії (*test.lm*), нейронної мережі (*test.ann*), випадкового лісу (*test.rf*) та методу опорних векторів (*test.svr*).

Програмно було визначено моделі, які входять до другого шару ансамблю, та налаштовано контрольну схему навчання із застосуванням крос-валідації. На основі цього було створено шар бегінгу *ens1.bag.layer* – рис. 4.30.

```
> ens1.bag.layer
Bagged CART

953 samples
  4 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 857, 859, 859, 857, 857, 858, ...
Resampling results:

  RMSE         Rsquared   MAE
  0.001763644  0.9457767  0.001295697
```

Рисунок 4.30 – Другий шар *ens1.bag.layer* дворівневої ансамблевої структури

Шар бегінгу тренується на основі вибраних предикторів. Далі здійснюється прогнозування значень вихідної змінної «*Energy_Output*» на основі даних тестового набору. Після отримання прогнозів обчислено ключові показники точності, як-от RMSE, MAE, MAPE та R^2, – рис. 4.31.

```
> ens1.bag.layer.test<-predict(ens1.bag.layer,newdata= ens1.preds)

> # Обчислення показників
> ens1.bag.layer.rmse <- sqrt(sum((ens1.bag.layer.
t))
> ens1.bag.layer.mae <- sum(abs(ens1.bag.layer.tes
> ens1.bag.layer.mape <- sum(abs((ens1.bag.layer.t
reds$Energy_Output)*100
> ens1.bag.layer.residuals <- ens1.preds$Energy_Ou
> ens1.bag.layer.SST <- sum((ens1.preds$Energy_Out
> ens1.bag.layer.SE <- sum(ens1.bag.layer.residual
> ens1.bag.layer.R2 <- 1 - (ens1.bag.layer.SE / en
>
> # Результати для моделі
> c(R2=ens1.bag.layer.R2, RMSE = ens1.bag.layer.rm
          R2         RMSE          MAE         MAPE
0.951116723 0.001667204 0.001209867 0.608085831
```

Рисунок 4.31 – Тестування та оцінка ефективності шару *ens1.bag.layer*

Як підсумок, результати прогнозування «класичної» дворівневої ансамблевої структури – табл. 4.3.

Таблиця 4.3 – Результати прогнозування «класичної» дворівневої ансамблевої структури

	Тип моделі	**Якість моделі**	**Якість прогнозу**		
		R^2	**RMSE**	**MAE**	**MAPE (%)**
1	Регресійна модель на основі дерева рішень	0.88	0.0026	0.0021	1.04
	Регресійна модель на основі KNN-R	0.88	0.0026	0.0020	1.03
2	*Шар стекінгу*	0.92	0.0021	0.0016	0.80
	Регресійна модель на основі множинної лінійної регресії	0.93	0.0020	0.0016	0.79
	Регресійна модель на основі випадкового лісу	0.96	0.0016	0.0011	0.54
	Регресійна модель на основі методу опорних векторів	0.94	0.0019	0.0014	0.69
	Регресійна модель на основі штучної НМ	0.91	0.0022	0.0017	0.86
	Результівний шар бегінгу	0.95	0.0017	0.0012	0.61

На першому шарі використання стекінгу дозволило підвищити точність прогнозів до 0.92, що на 4% більше порівняно з окремими базовими моделями, як-

от дерево рішень і KNN-R. На другому шарі бегінг об'єднав прогнози першого шару, стекінгу, та перенавчених моделей, що дозволило досягти загальної точності в прогнозуванні. Всупереч цьому, окрема базова регресійна модель на основі випадкового лісу перевершила результати дворівневої структури. Причинами таких результатів можуть бути особливості задачі та набору даних в цілому.

Отже, «класична» дворівнева ансамблева структура забезпечила певне підвищення точності прогнозів, але результати свідчать про ***наявність потенціалу для випробування альтернативних комбінацій*** *базових моделей в гетерогенних рівнях багатошарової структури експериментальним шляхом.*

Для побудови експериментальної дворівневої ансамблевої структури було обрано нетрадиційний підхід, який передбачає використання недонавчених моделей на рівні бегінгу, та перенавчених моделей на рівні стекінгу. Архітектура інтелектуальної системи прогнозування тоді виглядатиме так – рис. 4.32.

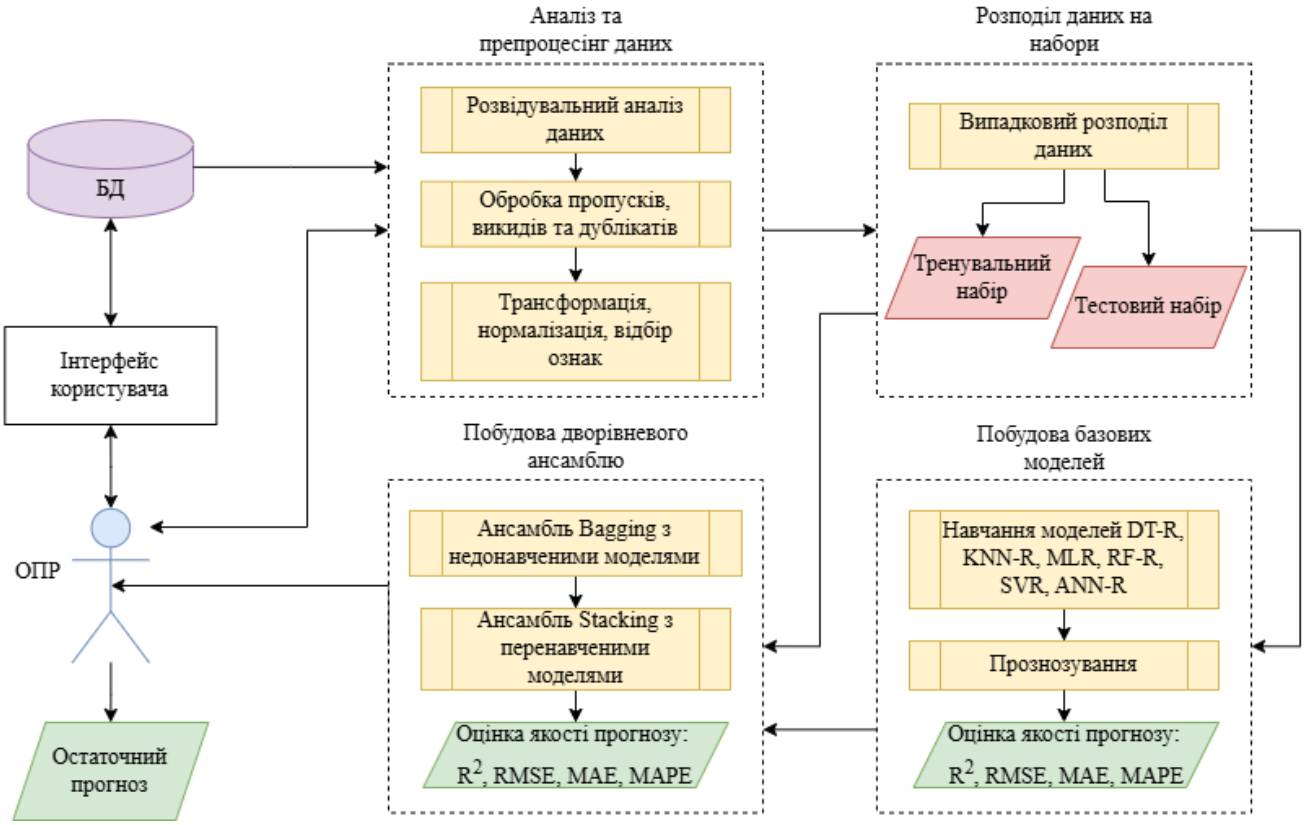

Рисунок 4.32 – Архітектура інтелектуальної системи прогнозування на основі експериментального підходу

В перший шар, бегінг, додаються моделі:

– регресійна модель на основі дерева рішень;

– регресійна модель на основі KNN-R.

Результати першого рівня передаються як вхід для *другого шару, стекінгу*, де застосовуються перенавчені моделі:

– регресійна модель на основі множинної лінійної регресії;

– регресійна модель на основі випадкового лісу;

– регресійна модель на основі методу опорних векторів;

– регресійна модель на основі штучної НМ.

Отже, сама експериментальна дворівнева структура виглядає так – рис. 4.33.

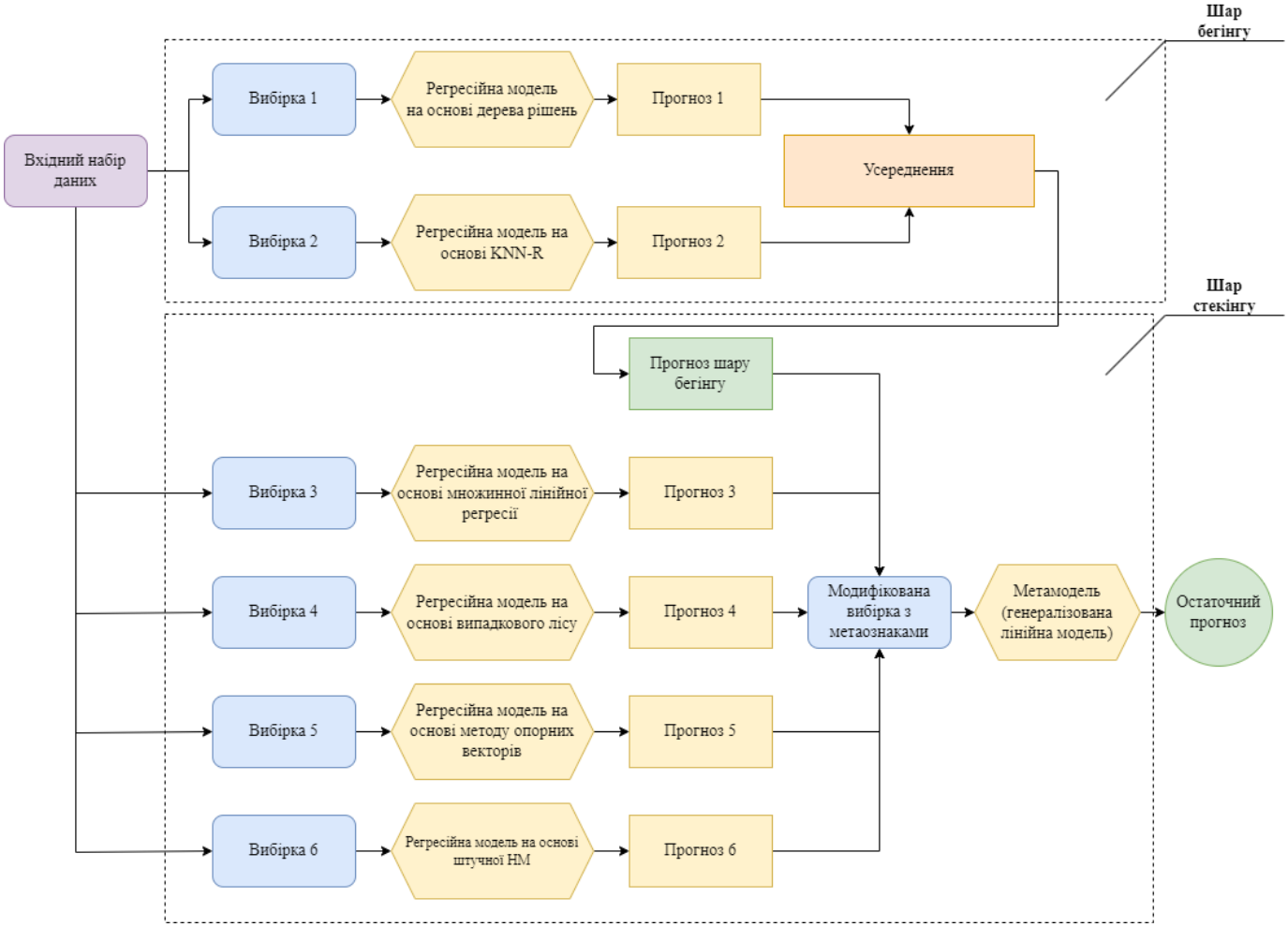

Рисунок 4.33 – Дворівнева ансамблева структура на основі експериментальних гетерогенних ансамблів

Знову-таки йде попереднє збереження чинних тестових даних та прогнозів базових моделей у змінну *ens2.pred* – рис. 4.34.

```
> # Збереження testdata в ens2.pred
> ens2.pred <- testdata
>
> # Додавання стовпців з прогнозами
> ens2.pred <- data.frame(
+   ens2.pred,                # Існуючі дані з testdata
+   test.lm = test.lm,       # Прогнози лінійної регресії
+   test.dt = test.dt,       # Прогнози дерев рішень
+   test.rf = test.rf,       # Прогнози випадкового лісу
+   test.svr = test.svr,     # Прогнози SVR
+   test.knn = test.knn,     # Прогнози KNN
+   test.ann = test.ann      # Прогнози нейронної мережі
+ )
```

Рисунок 4.34 – Додавання тестових даних та прогнозів у змінну *ens2.pred*

Виконано створення першого шару дворівневої ансамблевої структури – бегінгу, який включає недонавчені моделі.

Програмно було визначено моделі, які входять до першого шару ансамблю, та налаштовано контрольну схему навчання із застосуванням крос-валідації. На основі цього було створено шар бегінгу *ens2.bag.layer* – рис. 4.35.

```
> # ===== Перший шар =====
> ens2.bag.preds<-
+   c('test.knn','test.dt')
> ens2.bag.layer.ctrl <- trainControl(method="repeatedcv", number=10, repeats=3)
> ens2.bag.layer<-
+   train(ens2.pred[,ens2.bag.preds],ens2.pred$Energy_Output,method='treebag',trControl=ens2.bag.layer.ctrl)
>
> ens2.bag.layer
Bagged CART

953 samples
  2 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 859, 857, 858, 858, 859, 857, ...
Resampling results:

  RMSE         Rsquared   MAE
  0.002506783  0.8913398  0.001958383
```

Рисунок 4.35 – Перший шар *ens2.bag.layer* дворівневої ансамблевої структури

Шар бегінгу тренується на основі вибраних предикторів. Далі здійснюється прогнозування значень вихідної змінної «*Energy_Output*» на основі даних тестового набору – рис. 4.36.

```
> ens2.bag.layer.test<-predict(ens2.bag.layer,newdata= ens2.pred)
```

Рисунок 4.36 – Тестування шару бегінгу

Після отримання прогнозів, для оцінки ефективності першого шару *ens2.bag.layer* обчислено ключові показники точності, як-от RMSE, MAE, MAPE та R^2, – рис. 4.37.

```
> # Обчислення показників
> ens2.bag.layer.rmse <- sqrt(sum((ens2.bag.layer
> ens2.bag.layer.mae <- sum(abs(ens2.bag.layer.te
> ens2.bag.layer.mape <- sum(abs((ens2.bag.layer.
ed$Energy_Output)*100
> ens2.bag.layer.residuals <- ens2.pred$Energy_Ou
> ens2.bag.layer.SST <- sum((ens2.pred$Energy_Out
> ens2.bag.layer.SE <- sum(ens2.bag.layer.residua
> ens2.bag.layer.R2 <- 1 - (ens2.bag.layer.SE / e
>
> # Результати для моделі
> c(R2=ens2.bag.layer.R2, RMSE = ens2.bag.layer.r
        R2        RMSE         MAE        MAPE
0.893193198 0.002464383 0.001922559 0.963090726
```

Рисунок 4.37 – Оцінки ефективності першого шару *ens2.bag.layer*

Результати прогнозування першого шару візуально показано в таблиці для зручності сприйняття – табл. 4.4.

Таблиця 4.4 – Результати прогнозування першого шару методом бегінгу

Тип моделі	**Якість моделі**	**Якість прогнозу**		
	R^2	**RMSE**	**MAE**	**MAPE (%)**
Регресійна модель на основі дерева рішень	0.88	0.0026	0.0021	1.04
Регресійна модель на основі KNN-R	0.88	0.0026	0.0020	1.03
Результівний шар бегінгу	0.89	0.0025	0.0019	0.96

Отже, застосування бегінгу призвело до покращення прогнозування. Регресійні моделі на основі дерева рішень і KNN-R мали однакову точність 0.88, але в бегінгу точність зросла до 0.89. Похибки RMSE і MAE також зменшились. Значення MAPE знизилось до 0.96%, що свідчить про покращену стабільність і точність прогнозів.

Після цього, отримані прогнози шару бегінгу додаються до основного набору даних для подальшого використання в шарі №2, стекінгу, – рис. 4.38.

```
> ens2.pred$ens2.bag.layer.test <- ens2.bag.layer.test
```

Рисунок 4.38 – Додавання прогнозів шару бегінгу до основного набору даних

На другому етапі створюється стекінговий шар, який використовує прогнози з першого шару та перенавчені базові моделі для подальшого підвищення точності прогнозування.

Передусім визначається контрольна схема навчання для побудови моделі – рис. 4.39.

```
> # ===== Другий шар =====
> ens2.ctrl <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE)
```

Рисунок 4.39 – Налаштування контрольної схеми навчання

Потім обираються моделі, які будуть використані для створення другого шару. У цьому випадку, використовуються прогнози першого шару (*ens2.bag.layer.test*), лінійної регресії (*test.lm*), нейронної мережі (*test.ann*), випадкового лісу (*test.rf*) та методу опорних векторів (*test.svr*). Вони стають базовими моделями для метамоделі. Для налаштування стекінгу використовується контрольна схема з крос-валідацією – рис. 4.40.

```
> ens2.stack.layer.preds <- c('ens2.bag.layer.test','test.lm','test.ann','test.rf', 'test.svr')
> ens2.stack.layer.models <- caretList(ens2.pred[,ens2.stack.layer.preds], ens2.pred$Energy_Output,
+                     trControl=ens2.ctrl, methodList=c("glm"))
>
> # Налаштування ens2.stack.layer.ctrl для регресії
> ens2.stack.layer.ctrl <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE)
```

Рисунок 4.40 – Вибір моделей та налаштування контрольної схеми

На основі отриманих прогнозів базових моделей створюється метамодель *ens2.stack.layer* з використанням генералізованої лінійної моделі – рис. 4.41.

```
> # Створення стекової моделі
> ens2.stack.layer <- caretStack(ens2.stack.layer.models, method="glm", trControl=ens2.stack.layer.ctrl)
> ens2.stack.layer
The following models were ensembled: glm

caret::train model:
Generalized Linear Model

953 samples
  1 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 858, 858, 858, 857, 858, 857, ...
Resampling results:

  RMSE         Rsquared   MAE
  0.001526029  0.9586819  0.001029989


Final model:

Call:  NULL

Coefficients:
(Intercept)          glm
  8.106e-05    9.996e-01

Degrees of Freedom: 952 Total (i.e. Null);  951 Residual
Null Deviance:      0.05419
Residual Deviance: 0.002276     AIC: -9626
```

Рисунок 4.41 – Створення генералізованої лінійної моделі як метамоделі

Отримані результати навчання другого шару використовуються для прогнозування значення вихідної змінної на тестовому наборі даних. Щоб оцінити точність отриманих прогнозів, обчислюються основні показники, як-от RMSE, MAE, MAPE та коефіцієнт детермінації R^2, – рис. 4.42.

```
> # Прогнозування за допомогою стекової моделі
> ens2.stack.layer.test <- predict(ens2.stack.lay
>
> # Обчислення показників
> ens2.stack.layer.rmse <- sqrt(sum((ens2.stack.l
tput))
> ens2.stack.layer.mae <- sum(abs(ens2.stack.laye
> ens2.stack.layer.mape <- sum(abs((ens2.stack.la
th(ens2.pred$Energy_Output) * 100
> ens2.stack.layer.residuals <- ens2.pred$Energy_
> ens2.stack.layer.SST <- sum((ens2.pred$Energy_O
> ens2.stack.layer.SE <- sum(ens2.stack.layer.res
> ens2.stack.layer.R2 <- 1 - (ens2.stack.layer.SE
>
> # Результати для моделі
> c(R2 = ens2.stack.layer.R2, RMSE = ens2.stack.l
        R2        RMSE         MAE        MAPE
0.958683763 0.001532742 0.001019026 0.513034050
```

Рисунок 4.42 – Прогнозування та оцінка роботи

Результати прогнозування експериментальної дворівневої ансамблевої структури візуально показано в таблиці – табл. 4.5.

Таблиця 4.5 – Результати прогнозування експериментальної дворівневої ансамблевої структури

	Тип моделі	**Якість моделі**	**Якість прогнозу**		
		R^2	**RMSE**	**MAE**	**MAPE (%)**
1	Регресійна модель на основі дерева рішень	0.88	0.0026	0.0021	1.04
	Регресійна модель на основі KNN-R	0.88	0.0026	0.0020	1.03
2	*Шар бегінгу*	0.89	0.0025	0.0019	0.96
	Регресійна модель на основі множинної лінійної регресії	0.93	0.0020	0.0016	0.79
	Регресійна модель на основі випадкового лісу	0.96	0.0016	0.0011	0.54
	Регресійна модель на основі методу опорних векторів	0.94	0.0019	0.0014	0.69
	Регресійна модель на основі штучної НМ	0.91	0.0022	0.0017	0.86
	Результівний шар стекінгу	0.96	0.0015	0.0010	0.51

На першому шарі, де використовувався бегінг, точність прогнозів зросла до 0.89, порівняно з 0.88 для окремих моделей (дерево рішень і KNN-R).

На другому рівні, використовуючи метод стекінгу, вдалося отримати кращі результати порівняно з показниками кожної окремої базової моделі: ***R^2 = 0.96, RMSE = 0.0015, MAE = 0.0010, MAPE = 0.51%***, що підтверджує високу ефективність комбінування моделей у гетерогенних ансамблях структури.

Складена архітектура інтелектуальної системи прогнозування на основі обох дворівневих ансамблевих структур, в результаті, виглядає так – рис. 4.43.

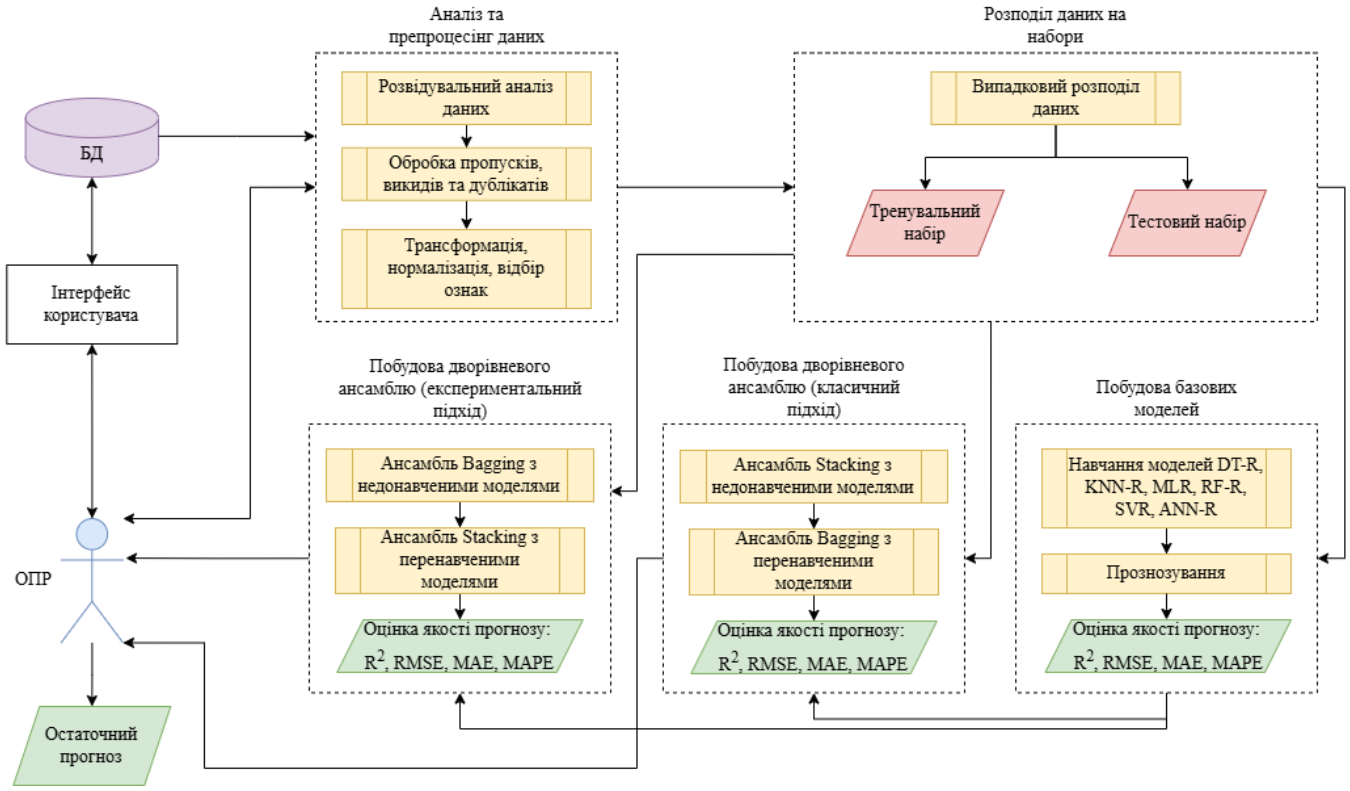

Рисунок 4.43 – Складена архітектура інтелектуальної системи прогнозування на основі дворівневих ансамблевих структур

У цьому випадку експериментальне поєднання моделей показало кращі результати. Такий варіант спрацював особливо ефективно завдяки специфіці набору даних і характеру залежностей у ньому, а також особливостям самого завдання регресії.

По-перше, набір даних містить різноманітні взаємозалежності між змінними, що створює складні, але значущі патерни, які потрібно належним чином виявити й врахувати для точної побудови регресійної моделі. Недонавчені моделі, такі як дерево рішень та KNN, недостатньо потужно розкривають ці залежності поодинці, однак їх об’єднання через бегінг допомогло краще узагальнити основні, стабільні патерни у даних, не схиляючись до перенавчання.

По-друге, перенавчені моделі завдяки своїм алгоритмам обробки можуть краще адаптуватися до складних нелінійних залежностей, які є в наборі даних, але схильні до підгонки під тренувальні дані. Стекінг в даному випадку допоміг

ефективно поєднати їх прогнози, згладжуючи похибки, спричинені перенавчанням, і збільшуючи узагальнювальну здатність ансамблю на тестових даних.

Як підсумок, результати експериментального варіанту дворівневої ансамблевої структури свідчать про ***значне покращення загальної якості прогнозування***.

Висновки до розділу 4

У цьому розділі було виконано побудову базових регресійних моделей для прогнозування, формування дворівневих ансамблевих структур та оцінку результатів їхнього прогнозування. Для оцінки якості моделей використовувалися такі критерії, як RMSE, MAE, MAPE та коефіцієнт детермінації, R^2.

При побудові базових моделей найкращі результати продемонструвала модель на основі випадкового лісу, тоді як найгірші показники мали регресійні моделі на основі дерева рішень і KNN-R.

Результати ансамблевого моделювання підтвердили ефективність використання комбінованого підходу. Спочатку було протестовано «класичний» підхід до побудови дворівневих ансамблевих структур, де на першому шарі використовувався стекінг для недонавчених моделей, а на другому – бегінг для перенавчених моделей. Це дало $R^2 = 0.95$, RMSE = 0.0017, MAE = 0.0012, MAPE = 0.61%, що перевищило ефективність майже всіх базових моделей, але залишило потенціал для подальшого вдосконалення.

У наступному експерименті було використано альтернативний підхід: бегінг на першому шарі для стабілізації недонавчених моделей і стекінг на другому шарі для корекції перенавчання. Цей варіант показав найкращі результати – $R^2 = 0.96$, RMSE = 0.0015, MAE = 0.0010, MAPE = 0.51%.

Як підсумок, експериментальна дворівнева ансамблева структура забезпечила вищу точність прогнозів, що вказує на ефективність нестандартної комбінації моделей в її гетерогенних складниках.

ВИСНОВКИ

У процесі дослідження було розглянуто основні теоретичні підходи до прогнозування, включаючи поняття прогнозу, його типологію та ключові аспекти процесу прогнозування. Вивчення теоретичних основ інтелектуального прогнозування та сучасних підходів до підвищення точності прогнозів стало основою для побудови ефективної методики.

Дослідження регресійного аналізу та регресійних моделей дозволило глибше зрозуміти їхні особливості та принципи функціонування. Розуміння компромісу між двома складниками похибки: зміщенням та дисперсією, довело ефективність застосування гетерогенних ансамблевих агрегацій для його пом’якшення.

Виконано розвідувальний аналіз обраного набору та його попередню обробку, що забезпечило якісну підготовку даних для моделювання. Завдяки належній обробці було досягнуто високого рівня узгодженості та чистоти даних.

Побудова базових моделей та дворівневих ансамблевих структур здійснювалась із застосуванням класичного підходу, при якому стекінг використовувався для недонавчених моделей на першому шарі, а бегінг – для перенавчених на другому. Однак експериментальна дворівнева структура, що передбачала використання бегінгу для недонавчених моделей на першому шарі та стекінгу для перенавчених на другому, продемонструвала кращі результати.

За результатами експериментального підходу отримано показники $R^2 = 0.96$, RMSE = 0.0015, MAE = 0.0010, MAPE = 0.51%, що свідчить про високу ефективність нестандартного варіанту.

Отже, мета дослідження досягнута, а отримані результати підтверджують успішне виконання поставлених у вступі завдань.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Zhailybayevich Z. K., Hamada M. A. Development of a Predictive Intellectual Model for Predicting the Financial Crisis in Banks. 2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC), Salem, India, 4–6 May 2023. 2023. DOI: https://doi.org/10.1109/icaaic56838.2023.10140628 (last accessed: 22.06.2024).

2. Alon I., Qi M., Sadowski R. J. Forecasting aggregate retail sales:. Journal of Retailing and Consumer Services. 2001. Vol. 8, no. 3. P. 147–156. DOI: https://doi.org/10.1016/s0969-6989(00)00011-4 (last accessed: 22.06.2024).

3. Earthquake Forecasting Using Big Data and Artificial Intelligence: A 30-Week Real-Time Case Study in China / O. M. Saad et al. Bulletin of the Seismological Society of America. 2023. DOI: https://doi.org/10.1785/0120230031 (last accessed: 22.06.2024).

4. Artificial Intelligence Forecasting of Covid-19 in China / Z. Hu et al. International Journal of Educational Excellence. 2020. Vol. 6, no. 1. P. 71–94. DOI: https://doi.org/10.18562/ijee.054 (last accessed: 23.06.2024).

5. Yu L., Wang S., Lai K. K. A neural-network-based nonlinear metamodeling approach to financial time series forecasting. Applied Soft Computing. 2009. Vol. 9, no. 2. P. 563–574. DOI: https://doi.org/10.1016/j.asoc.2008.08.001 (last accessed: 23.06.2024).

6. Neural Network–Based Financial Volatility Forecasting: A Systematic Review / W. Ge et al. ACM Computing Surveys. 2023. Vol. 55, no. 1. P. 1–30. DOI: https://doi.org/10.1145/3483596 (last accessed: 23.06.2024).

7. Kochak A., Sharma S. Demand forecasting using neural network for supply chain management. International journal of mechanical engineering and robotics research. 2015. Vol. 4, no. 1. P. 96–104. DOI: http://surl.li/xdggoe (last accessed: 23.06.2024).

8. Carbonneau R., Laframboise K., Vahidov R. Application of machine learning techniques for supply chain demand forecasting. European Journal of Operational

Research. 2008. Vol. 184, no. 3. P. 1140–1154. DOI: https://doi.org/10.1016/j.ejor.2006.12.004 (last accessed: 23.06.2024).

9. Artificial intelligence and healthcare: Forecasting of medical bookings through multi-source time-series fusion / F. Piccialli et al. Information Fusion. 2021. Vol. 74. P. 1–16. DOI: https://doi.org/10.1016/j.inffus.2021.03.004 (last accessed: 23.06.2024).

10. Forecasting sustainability of healthcare supply chains using deep learning and network data envelopment analysis / M. Azadi et al. Journal of Business Research. 2023. Vol. 154. P. 113357. DOI: https://doi.org/10.1016/j.jbusres.2022.113357 (last accessed: 24.06.2024).

11. Narvekar M., Fargose P. Daily Weather Forecasting using Artificial Neural Network. International Journal of Computer Applications. 2015. Vol. 121, no. 22. P. 9–13. DOI: https://doi.org/10.5120/21830-5088 (last accessed: 24.06.2024).

12. Weather Forecasting Model using Artificial Neural Network / K. Abhishek et al. Procedia Technology. 2012. Vol. 4. P. 311–318. DOI: https://doi.org/10.1016/j.protcy.2012.05.047 (last accessed: 25.06.2024).

13. Daş G. S. Forecasting the energy demand of Turkey with a NN based on an improved Particle Swarm Optimization. Neural Computing and Applications. 2016. Vol. 28, S1. P. 539–549. DOI: https://doi.org/10.1007/s00521-016-2367-8 (last accessed: 25.06.2024).

14. Bredahl Kock A., Teräsvirta T. Forecasting Macroeconomic Variables Using Neural Network Models and Three Automated Model Selection Techniques. Econometric Reviews. 2015. Vol. 35, no. 8-10. P. 1753–1779. DOI: https://doi.org/10.1080/07474938.2015.1035163 (last accessed: 25.06.2024).

15. Gabor M., Dorgo L. NEURAL NETWORKS VERSUS BOX-JENKINS METHOD FOR TURNOVER FORECASTING: A CASE STUDY ON THE ROMANIAN ORGANISATION. Transformations in Business & Economics. 2017. Vol. 16, no. 1. P. 42–59. DOI: https://rb.gy/a2h2z9 (last accessed: 25.06.2024).

16. Qiu M., Song Y. Predicting the Direction of Stock Market Index Movement Using an Optimized Artificial Neural Network Model. PLOS ONE. 2016. Vol. 11, no. 5.

P. e0155133. DOI: https://doi.org/10.1371/journal.pone.0155133 (last accessed: 25.06.2024).

17. Wang J., Wang J. Forecasting stochastic neural network based on financial empirical mode decomposition. Neural Networks. 2017. Vol. 90. P. 8–20. DOI: https://doi.org/10.1016/j.neunet.2017.03.004 (last accessed: 26.06.2024).

18. Real-time air quality forecasting, part II: State of the science, current research needs, and future prospects / Y. Zhang et al. Atmospheric Environment. 2012. Vol. 60. P. 656–676. DOI: https://doi.org/10.1016/j.atmosenv.2012.02.041 (last accessed: 26.06.2024).

19. Enhancing short-term forecasting of daily precipitation using numerical weather prediction bias correcting with XGBoost in different regions of China / J. Dong et al. Engineering Applications of Artificial Intelligence. 2023. Vol. 117. P. 105579. DOI: https://doi.org/10.1016/j.engappai.2022.105579 (last accessed: 26.06.2024).

20. A deep learning method for real-time bias correction of wind field forecasts in the Western North Pacific / W. Zhang et al. Atmospheric Research. 2022. P. 106586. DOI: https://doi.org/10.1016/j.atmosres.2022.106586 (last accessed: 26.06.2024).

21. Aduama P., Zhang Z., Al-Sumaiti A. S. Multi-Feature Data Fusion-Based Load Forecasting of Electric Vehicle Charging Stations Using a Deep Learning Model. Energies. 2023. Vol. 16, no. 3. P. 1309. DOI: https://doi.org/10.3390/en16031309 (last accessed: 26.06.2024).

22. Pokhrel P., Abdelguerfi M., Ioup E. A Machine Learning and Data Assimilation forecasting framework for surface waves. Quarterly Journal of the Royal Meteorological Society. 2023. DOI: https://doi.org/10.1002/qj.4631 (last accessed: 26.06.2024).

23. Wander H. L., Thomas R. Q. Data assimilation experiments inform monitoring needs for near-term ecological forecasts in a eutrophic reservoir. Ecosphere. 2024. Vol. 15, no. 2. DOI: https://doi.org/10.1002/ecs2.4752 (date of access: 27.06.2024).

24. Ensemble Kalman filter for GAN-ConvLSTM based long lead-time forecasting / M. Cheng et al. Journal of Computational Science. 2023. Vol. 69. P. 102024. DOI: https://doi.org/10.1016/j.jocs.2023.102024 (last accessed: 27.06.2024).

25. Samos I., Louka P., Flocas H. Assessing the Accuracy of 3D-VAR in Supercell Thunderstorm Forecasting: A Regional Background Error Covariance Study. Atmosphere. 2023. Vol. 14, no. 11. P. 1611. DOI: https://doi.org/10.3390/atmos14111611 (last accessed: 27.06.2024).

26. Duque L.-F., O'Connell E., O'Donnell G. A Monte Carlo simulation and sensitivity analysis framework demonstrating the advantages of probabilistic forecasting over deterministic forecasting in terms of flood warning reliability. Journal of Hydrology. 2023. Vol. 619. P. 129340. DOI: https://doi.org/10.1016/j.jhydrol.2023.129340 (last accessed: 27.06.2024).

27. An intelligent hybrid method based on Monte Carlo simulation for short-term probabilistic wind power prediction / A. A. Abdoos et al. Energy. 2023. P. 127914. DOI: https://doi.org/10.1016/j.energy.2023.127914 (last accessed: 27.06.2024).

28. Multi-Model Ensemble Forecasts of Surface Air Temperatures in Henan Province Based on Machine Learning / T. Wang et al. Atmosphere. 2023. Vol. 14, no. 3. P. 520. DOI: https://doi.org/10.3390/atmos14030520 (last accessed: 29.06.2024).

29. Wang J., Wang X., Thiam Khu S. A Decomposition-based Multi-model and Multi-parameter Ensemble Forecast Framework for Monthly Streamflow Forecasting. Journal of Hydrology. 2023. P. 129083. DOI: https://doi.org/10.1016/j.jhydrol.2023.129083 (last accessed: 29.06.2024).

30. Two-Stage Fine-Tuning for Improved Bias and Variance for Large Pretrained Language Models / L. Wang et al. Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Toronto, Canada. Stroudsburg, PA, USA, 2023. DOI: https://doi.org/10.18653/v1/2023.acl-long.877 (last accessed: 29.06.2024).

31. Bashir S., Qamar U., Khan F. H. IntelliHealth: A medical decision support application using a novel weighted multi-layer classifier ensemble framework. Journal of

Biomedical Informatics. 2016. Vol. 59. P. 185–200. DOI: https://doi.org/10.1016/j.jbi.2015.12.001 (last accessed: 29.06.2024).

32. Ali P., Younas A. Understanding and interpreting regression analysis. Evidence Based Nursing. 2021. P. ebnurs-2021-103425. DOI: https://doi.org/10.1136/ebnurs-2021-103425 (last accessed: 02.09.2024).

33. Leveraging Regression Analysis to Predict Overlapping Symptoms of Cardiovascular Diseases / S. Ghorashi et al. IEEE Access. 2023. P. 1. DOI: https://doi.org/10.1109/access.2023.3286311 (last accessed: 02.09.2024).

34. Ghardallou W. The heterogeneous effect of leverage on firm performance: a quantile regression analysis. International Journal of Islamic and Middle Eastern Finance and Management. 2023. DOI: https://doi.org/10.1108/imefm-12-2021-0490 (last accessed: 02.09.2024).

35. Abdi H. The method of least squares. Encyclopedia of measurement and statistics. 2007. Vol. 1. P. 530–532. DOI: https://personal.utdallas.edu/~herve/Abdi-LeastSquares06-pretty.pdf (last accessed: 03.09.2024).

36. Febrianti R., Widyaningsih Y., Soemartojo S. The parameter estimation of logistic regression with maximum likelihood method and score function modification. Journal of Physics: Conference Series. 2021. Vol. 1725. P. 012014. DOI: https://doi.org/10.1088/1742-6596/1725/1/012014 (last accessed: 03.09.2024).

37. Linear Regression / G. James et al. Springer Texts in Statistics. Cham, 2023. P. 69–134. DOI: https://doi.org/10.1007/978-3-031-38747-0_3 (last accessed: 03.09.2024).

38. Interaction between numerical variables in regression model, and its graphical interpretation / H. Ankarali et al. Bangladesh Journal of Medical Science. 2023. Vol. 22, no. 1. P. 189–194. DOI: https://doi.org/10.3329/bjms.v22i1.63078 (last accessed: 03.09.2024).

39. Interactive Generalized Additive Model and Its Applications in Electric Load Forecasting / L. Yang et al. KDD ‘23: The 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Long Beach CA USA. New York, NY, USA, 2023. DOI: https://doi.org/10.1145/3580305.3599848 (last accessed: 04.09.2024).

40. Relationships between total reserve and financial indicators of Bangladesh: Application of generalized additive model / M. S. A. Salan et al. PLOS ONE. 2023. Vol. 18, no. 4. P. e0284179. DOI: https://doi.org/10.1371/journal.pone.0284179 (last accessed: 04.09.2024).

41. Tu K., Yan Z., Qian C. Understanding seasonal cycle of daily extreme temperatures based on generalized additive model for location, scale and shape with smoothing spline. International Journal of Climatology. 2024. DOI: https://doi.org/10.1002/joc.8430 (last accessed: 09.09.2024).

42. Generalized Additive Models for Predicting Sea Level Rise in Coastal Florida / H. N. Vaidya et al. Geosciences. 2023. Vol. 13, no. 10. P. 310. DOI: https://doi.org/10.3390/geosciences13100310 (last accessed: 09.09.2024).

43. Czajkowski M., Kretowski M. The role of decision tree representation in regression problems – An evolutionary perspective. Applied Soft Computing. 2016. Vol. 48. P. 458–475. DOI: https://doi.org/10.1016/j.asoc.2016.07.007 (last accessed: 09.09.2024).

44. Prediction of higher heating value of coal based on gradient boosting regression tree model / N. Xu et al. International Journal of Coal Geology. 2023. P. 104293. DOI: https://doi.org/10.1016/j.coal.2023.104293 (last accessed: 09.09.2024).

45. Spatial heterogeneity modeling of water quality based on random forest regression and model interpretation / F. Wang et al. Environmental Research. 2021. Vol. 202. P. 111660. DOI: https://doi.org/10.1016/j.envres.2021.111660 (last accessed: 09.09.2024).

46. Desai S., Ouarda T. B. M. J. Regional Hydrological Frequency Analysis at Ungauged Sites with Random Forest Regression. Journal of Hydrology. 2020. P. 125861. DOI: https://doi.org/10.1016/j.jhydrol.2020.125861 (last accessed: 15.09.2024).

47. Safari M., Rabiee A. H., Joudaki J. Developing a Support Vector Regression (SVR) Model for Prediction of Main and Lateral Bending Angles in Laser Tube Bending Process. Materials. 2023. Vol. 16, no. 8. P. 3251. DOI: https://doi.org/10.3390/ma16083251 (last accessed: 15.09.2024).

48. Dynamic Neural Network Architecture Design for Predicting Remaining Useful Life of Dynamic Processes / S. Simani et al. Journal of Data Science and Intelligent Systems. 2023. DOI: https://doi.org/10.47852/bonviewjdsis3202967 (last accessed: 16.09.2024).

49. Zhou X., Zhou H., Long H. Forecasting the equity premium: Do deep neural network models work?. Modern Finance. 2023. Vol. 1, no. 1. P. 1–11. DOI: https://doi.org/10.61351/mf.v1i1.2 (last accessed: 17.09.2024).

50. Forecasting actual evapotranspiration without climate data based on stacked integration of DNN and meta-heuristic models across China from 1958 to 2021 / A. Elbeltagi et al. Journal of Environmental Management. 2023. Vol. 345. P. 118697. DOI: https://doi.org/10.1016/j.jenvman.2023.118697 (last accessed: 25.09.2024).

51. Geman S., Bienenstock E., Doursat R. Neural Networks and the Bias/Variance Dilemma. Neural Computation. 1992. Vol. 4, no. 1. P. 1–58. DOI: https://doi.org/10.1162/neco.1992.4.1.1 (last accessed: 25.09.2024).

52. Kumar B., Yadav N., Sunil. A Bagging Ensemble Algorithm for Seasonal Time Series Forecasting. SN Computer Science. 2024. Vol. 5, no. 3. DOI: https://doi.org/10.1007/s42979-024-02648-0 (last accessed: 25.09.2024).

53. Storylines for unprecedented heatwaves based on ensemble boosting / E. M. Fischer et al. Nature Communications. 2023. Vol. 14, no. 1. DOI: https://doi.org/10.1038/s41467-023-40112-4 (last accessed: 26.09.2024).

54. Zhao L., Lu S., Qi D. Improvement of Maximum Air Temperature Forecasts Using a Stacking Ensemble Technique. Atmosphere. 2023. Vol. 14, no. 3. P. 600. DOI: https://doi.org/10.3390/atmos14030600 (last accessed: 26.09.2024).

55. Tfekci P., Kaya H. Combined Cycle Power Plant [Dataset]. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5002N (last accessed: 26.10.2024).

56. Al Hashmi A. B., Mohamed A. A. A., Dadach Z. E. Process Simulation of a 620 Mw-Natural Gas Combined Cycle Power Plant with Optimum Flue Gas

Recirculation. Open Journal of Energy Efficiency. 2018. Vol. 07, no. 02. P. 33–52. DOI: https://doi.org/10.4236/ojee.2018.72003 (last accessed: 26.10.2024).

57. Intelligent classification system based on ensemble methods / O. P. Hozhyi et al. System technologies. 2023. Vol. 3, no. 146. P. 61–75. URL: https://doi.org/10.34185/1562-9945-3-146-2023-07 (last accessed: 28.10.2024).

ДОДАТОК А

Лістинг коду інтелектуальної системи прогнозування

```
# Набір пакетів для роботи з даними
library(openxlsx)  # Бібліотека для роботи з Excel-файлами
library(corrplot)  # Бібліотека для створення графіків кореляційних матриць
library(GGally)  # Бібліотека для розширеного створення графіків за допомогою ggplot2
library(ggplot2)  # Бібліотека для створення графіків
library(mice)  # Бібліотека для множинного заповнення пропущених даних
library(missForest)  # Бібліотека для заповнення пропущених значень за допомогою алгоритму random forest
library(car)  # Бібліотека для регресійного аналізу та інших статистичних моделей
library(bestNormalize)  # Бібліотека для нормалізації даних
library(nortest)  # Бібліотека для тестів на нормальність розподілу
library(caret)  # Бібліотека для тренування моделей машинного навчання
library(leaps) # Бібліотека для відбору ознак
library(tidyr)# Бібліотека для очищення та трансформації даних
library(dplyr)# Бібліотека для маніпуляцій з даними
library(rpart)# Бібліотека для побудови дерев рішень
library(randomForest)# Бібліотека для побудови випадкових лісів
library(e1071)# Бібліотека для роботи з SVM та іншими алгоритмами
library(neuralnet)# Бібліотека для створення нейронних мереж
library(caretEnsemble)# Бібліотека для створення ансамблів

# ================== Розвідувальний аналіз  ==================

# Встановлення робочого каталогу
setwd('C:\\Users\\peshe\\Desktop\\мкр')

# Зчитування даних з файлу
powerplant <- read.xlsx("powerplant_dataset.xlsx")

# Виведення назв змінних
names(powerplant)

# Перейменування змінних у датафреймі powerplant
colnames(powerplant) <- c("Temperature", "Exhaust_Vacuum", "Ambient_Pressure", "Relative_Humidity", "Energy_Output")
names(powerplant)

# Виведення структури набору даних
str(powerplant)

# Виведення перших 6 записів
head(powerplant)

# Виведення статистичного підсумку
summary(powerplant)

# Кореляційний графік
correlation_matrix <- cor(powerplant)
correlation_matrix
corrplot.mixed(correlation_matrix,order = 'AOE')

# Вибір колонок
numeric_cols <- c("Temperature", "Exhaust_Vacuum", "Ambient_Pressure", "Relative_Humidity", "Energy_Output")
```

```
# Графік парних відносин
ggpairs(powerplant)

# Графік boxplot для виявлення викидів
par(mfrow = c(2, 3))
for (var in numeric_cols) {
  boxplot(powerplant[[var]], main = paste("Boxplot of", var),  main = "Boxplot", ylab = "Values", col = "lightblue")
}

# ================== Обробка пропусків, викидів та дублікатів ==================

# Перевірка на пропуски
sum(is.na(powerplant))

# Обчислення квартилів та IQR для кожного стовпчика ознак
q1 <- sapply(numeric_cols, function(col) quantile(powerplant[[col]], 0.25))
q3 <- sapply(numeric_cols, function(col) quantile(powerplant[[col]], 0.75))
IQR <- q3 - q1
IQR

# Визначення викидів
powerplant_outliers <- sapply(1:length(numeric_cols), function(i) {
  col <- numeric_cols[i]
  outlier_indices <- powerplant[[col]] < (q1[i] - 1.5 * IQR[i]) | powerplant[[col]] > (q3[i] + 1.5 * IQR[i])
  return(outlier_indices)
})
colnames(powerplant_outliers) <- numeric_cols

# Створення фрейму даних для зберігання інформації про викиди
powerplant_outlier <- data.frame(
  Total_Outliers = colSums(powerplant_outliers),
  Percentage_Outliers = colSums(powerplant_outliers) / nrow(powerplant) * 100
)

# Виведення інформації про викиди
print(powerplant_outlier)

# Функція для виконання імпутації та побудови boxplot для різних методів
impute_and_plot <- function(df, method, title_text) {

  # Заміна пропусків на NA
  for (col in numeric_cols) {
    df[[col]][powerplant_outliers[, col]] <- NA
  }

  # Підрахунок пропущених значень перед інтерполяцією
  cat("Missing values before imputation: ", sum(is.na(df)), "\n")

  # Імпутація
  if (method == "missForest") {
    # Виконання імпутації з missForest
    imputed_result <- missForest(df[, names(colSums(is.na(df)))])
    df <- imputed_result$ximp
  } else {
    # Виконання імпутації MICE для інших методів
    df <- complete(mice(df[, names(colSums(is.na(df)))], method = method))
  }

  # Підрахунок пропущених значень після імпутації
```

```
  cat("Missing values after imputation: ", sum(is.na(df)), "\n")

  # Бокс-діаграма
  par(mfrow = c(2, 3), oma = c(0, 0, 2, 0))
  for (var in numeric_cols) {
    boxplot(df[[var]], main = paste("Boxplot of", var), ylab = "Values", col = "lightblue")
  }
  title(main = title_text, outer = TRUE)
  par(mfrow = c(1, 1))
}

# Імпутація та побудова графіків для PMM, CART, lasso.norm та missForest
powerplant_clean1 <- powerplant
impute_and_plot(powerplant_clean1, "pmm", "Boxplot Mice PMM")

powerplant_clean2 <- powerplant
impute_and_plot(powerplant_clean2, "cart", "Boxplot Mice CART")

powerplant_clean3 <- powerplant
impute_and_plot(powerplant_clean3, "lasso.norm", "Boxplot Mice lasso.norm")

powerplant_clean4 <- powerplant
impute_and_plot(powerplant_clean4, "missForest", "Boxplot missForest")


powerplant_clean <- powerplant_clean2

# Виявлення всіх рядків, що повторюються, включно з оригінальними
all_duplicates <- powerplant_clean[duplicated(powerplant_clean) | duplicated(powerplant_clean, fromLast =
TRUE), ]

# Сортування повторюваних рядків у порядку спадання за всіма стовпчиками
sorted_duplicates <- all_duplicates[do.call(order, c(all_duplicates, decreasing = TRUE)), ]

# Перегляд відсортованих повторюваних рядків
head(sorted_duplicates, n=6)

# Видалення дубльованих рядків
powerplant_clean <- powerplant_clean[!duplicated(powerplant_clean), ]

# ================== Трансформація та нормалізація  ==================

# Гістограми розподілів ознак
par(mfrow=c(3, 3))
for (feature in numeric_cols) {
  hist(powerplant_clean[[feature]], main=feature, xlab=feature)
}

# Трансформація YeoJohnson
preProc  <- preProcess(powerplant_clean, method = "YeoJohnson")
powerplant_yj <- predict(preProc, powerplant_clean)

# Гістограми розподілів ознак
par(mfrow=c(3, 3))
for (feature in numeric_cols) {
  hist(powerplant_yj[[feature]], main=feature, xlab=feature)
}

# Нормалізація
normalize <- function(x, na.rm = TRUE) {
```

```
  return((x- min(x)) /(max(x)-min(x)))
  }
powerplant_norm<-normalize(powerplant_yj)

# Гістограми розподілів ознак
par(mfrow=c(3, 3))
for (feature in numeric_cols) {
  hist(powerplant_norm[[feature]], main=feature, xlab=feature)
}

# ================== Відбір ознак  ==================

# Метод обчислення MSE на вибірці перевірки
predict.regsubsets<-function(object,newdata,id,...){
  form<-as.formula(object$call[[2]])
  mat<-model.matrix(form,newdata)
  coefi<-coef(object,id=id)
  xvars<-names(coefi)
  mat[,xvars]%*%coefi
}

# Відбір ознак
k<-100
set.seed(1)
folds<-sample(1:k,nrow(powerplant_norm),replace=TRUE)
cv.errors<-matrix(NA,k,4,dimnames=list(NULL,paste(1:4)))

# Обчислення помилок на перевірочних вибірках для кожного блоку
for (j in 1:k) {
  best.fit <- regsubsets(Energy_Output ~ ., data = powerplant_norm[folds != j, ], nvmax = 4)
  for (i in 1:4) {
    pred <- predict.regsubsets(best.fit, powerplant_norm[folds == j, ], id = i)
    cv.errors[j, i] <- mean((powerplant_norm$Energy_Output[folds == j] - pred)^2)
  }
}

# Розрахунок середніх значень по стовпцях матриці
mean.cv.errors<-apply(cv.errors,2,mean)
mean.cv.errors
par(mfrow=c(1,1))
plot(mean.cv.errors,typ='b')

# Відбір оптимальної підмножини змінних на повному наборі даних для отримання моделі
reg.best<-regsubsets(Energy_Output~.,data=powerplant_norm,nvmax=4)
coef(reg.best,which.min(mean.cv.errors))

# ================== Тренування та навчання базових моделей  ==================

# Генерація випадкового вибору рядків для тренувального набору даних (90% від усіх рядків)
row.number <- sample(1:nrow(powerplant_norm), 0.9 * nrow(powerplant_norm))
# Створення тренувального набору даних на основі випадково вибраних рядків
traindata <- powerplant_norm[row.number,]
# Створення тестового набору даних на основі решти рядків, які не ввійшли до тренувального набору
testdata <- powerplant_norm[-row.number,]

# ========= Множинна лінійна регресія  =========

# Побудова моделі
lm.fit = lm(traindata$Energy_Output~., data = traindata)
# Виведення статистичного резюме процедури навчання моделі
```

```
summary(lm.fit)

# Прогнозування
test.lm <- predict(lm.fit, newdata = testdata)

# Обчислення показників
rmse.lm <- sqrt(sum((test.lm - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
mae.lm <- sum(abs(test.lm - testdata$Energy_Output))/length(testdata$Energy_Output)
mape.lm <- sum(abs((test.lm - testdata$Energy_Output)/test.lm))/length(testdata$Energy_Output)*100
R2.lm <- summary(lm.fit)$r.squared

# Результати для моделі
c(R2=R2.lm, RMSE = rmse.lm, MAE = mae.lm, MAPE = mape.lm)

# Перетворення даних у довгий формат для фактичних даних
combined_testdata.lm <- testdata %>%
  gather(key = "Variable", value = "Value", Relative_Humidity, Exhaust_Vacuum, Temperature,
Ambient_Pressure) %>%
  mutate(Type = "Test")

# Створення окремого фрейму даних для прогнозів
# Об’єднання прогнозів зі змінними для рядків
predictions.lm <- data.frame(
  Variable = rep(c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure"), each =
nrow(testdata)),
  Value = unlist(testdata[, c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure")]),
  Predicted = rep(test.lm, times = 4)
)

# Графік
ggplot() +
  geom_point(data = combined_testdata.lm, aes(x = Value, y = Energy_Output, color = Type)) +
  geom_line(data = predictions.lm, aes(x = Value, y = Predicted, color = "Predicted"), size = 1) +
  facet_wrap(~ Variable, scales = "free") +
  ggtitle(‘Energy_Output vs Various Variables with Predicted Values’) +
  xlab(‘Variable’) +
  ylab(‘Energy_Output’) +
  scale_color_manual(values = c(‘red’, ‘dark green’, ‘blue’))

# ========= Регресійна модель на основі дерева рішень  =========

# Побудова моделі
dt.fit <- rpart(traindata$Energy_Output~., data = traindata)
plot(dt.fit, uniform = TRUE,
    main = "Decision Tree Regression")
text(dt.fit, use.n = TRUE, cex = .7)
summary(dt.fit)

# Прогнозування
test.dt <- predict(dt.fit, newdata = testdata)

# Обчислення показників
rmse.dt <- sqrt(sum((test.dt - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
mae.dt <- sum(abs(test.dt - testdata$Energy_Output))/length(testdata$Energy_Output)
mape.dt <- sum(abs((test.dt - testdata$Energy_Output)/test.dt))/length(testdata$Energy_Output)*100
residuals.dt <- testdata$Energy_Output - test.dt
SST.dt <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
SE.dt <- sum(residuals.dt^2)
R2.dt <- 1 - (SE.dt / SST.dt)
```

```
# Результати для моделі
c(R2=R2.dt, RMSE = rmse.dt, MAE = mae.dt, MAPE = mape.dt)

# Перетворення даних у довгий формат для фактичних даних
combined_testdata.dt <- testdata %>%
  gather(key = "Variable", value = "Value", Relative_Humidity, Exhaust_Vacuum, Temperature,
Ambient_Pressure) %>%
  mutate(Type = "Test")

# Створення окремого фрейму даних для прогнозів
# Об’єднання прогнозів зі змінними для рядків
predictions.dt <- data.frame(
  Variable = rep(c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure"), each =
nrow(testdata)),
  Value = unlist(testdata[, c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure")]),
  Predicted = rep(test.dt, times = 4)
)

# Графік
ggplot() +
  geom_point(data = combined_testdata.dt, aes(x = Value, y = Energy_Output, color = Type)) +
  geom_line(data = predictions.dt, aes(x = Value, y = Predicted, color = "Predicted"), size = 1) +
  facet_wrap(~ Variable, scales = "free") +
  ggtitle(‘Energy_Output vs Various Variables with Predicted Values’) +
  xlab(‘Variable’) +
  ylab(‘Energy_Output’) +
  scale_color_manual(values = c(‘red’, ‘dark green’, ‘blue’))

# ========== Регресійна модель на основі випадкового лісу  ==========

# Побудова моделі
rf.fit <- randomForest(traindata$Energy_Output~., data=traindata)
print(rf.fit)

# Прогнозування
test.rf <- predict(rf.fit, newdata = testdata)

# Обчислення показників
rmse.rf <- sqrt(sum((test.rf - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
mae.rf <- sum(abs(test.rf - testdata$Energy_Output))/length(testdata$Energy_Output)
mape.rf <- sum(abs((test.rf - testdata$Energy_Output)/test.rf))/length(testdata$Energy_Output)*100
residuals.rf <- testdata$Energy_Output - test.rf
SST.rf <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
SE.rf <- sum(residuals.rf^2)
R2.rf <- 1 - (SE.rf / SST.rf)

# Результати для моделі
c(R2=R2.rf, RMSE = rmse.rf, MAE = mae.rf, MAPE = mape.rf)

# Перетворення даних у довгий формат для фактичних даних
combined_testdata.rf <- testdata %>%
  gather(key = "Variable", value = "Value", Relative_Humidity, Exhaust_Vacuum, Temperature,
Ambient_Pressure) %>%
  mutate(Type = "Test")

# Створення окремого фрейму даних для прогнозів
# Об’єднання прогнозів зі змінними для рядків
predictions.rf <- data.frame(
  Variable = rep(c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure"), each =
nrow(testdata)),
```

```
  Value = unlist(testdata[, c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure")]),
  Predicted = rep(test.rf, times = 4)
)

# Графік
ggplot() +
  geom_point(data = combined_testdata.rf, aes(x = Value, y = Energy_Output, color = Type)) +
  geom_line(data = predictions.rf, aes(x = Value, y = Predicted, color = "Predicted"), size = 1) +
  facet_wrap(~ Variable, scales = "free") +
  ggtitle('Energy_Output vs Various Variables with Predicted Values') +
  xlab('Variable') +
  ylab('Energy_Output') +
  scale_color_manual(values = c('red', 'dark green', 'blue'))

# ========= Регресійна модель на основі опорних векторів  =========

# Побудова моделі
svr.fit <- svm(traindata$Energy_Output~., data=traindata)
print(svr.fit)

# Прогнозування
test.svr <- predict(svr.fit, newdata = testdata)

# Обчислення показників
rmse.svr <- sqrt(sum((test.svr - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
mae.svr <- sum(abs(test.svr - testdata$Energy_Output))/length(testdata$Energy_Output)
mape.svr <- sum(abs((test.svr - testdata$Energy_Output)/test.svr))/length(testdata$Energy_Output)*100
residuals.svr <- testdata$Energy_Output - test.svr
SST.svr <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
SE.svr <- sum(residuals.svr^2)
R2.svr <- 1 - (SE.svr / SST.svr)

# Результати для моделі
c(R2=R2.svr, RMSE = rmse.svr, MAE = mae.svr, MAPE = mape.svr)

# Перетворення даних у довгий формат для фактичних даних
combined_testdata.svr <- testdata %>%
  gather(key = "Variable", value = "Value", Relative_Humidity, Exhaust_Vacuum, Temperature,
Ambient_Pressure) %>%
  mutate(Type = "Test")

# Створення окремого фрейму даних для прогнозів
# Об'єднання прогнозів зі змінними для рядків
predictions.svr <- data.frame(
  Variable = rep(c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure"), each =
nrow(testdata)),
  Value = unlist(testdata[, c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure")]),
  Predicted = rep(test.svr, times = 4)
)

# Графік
ggplot() +
  geom_point(data = combined_testdata.svr, aes(x = Value, y = Energy_Output, color = Type)) +
  geom_line(data = predictions.svr, aes(x = Value, y = Predicted, color = "Predicted"), size = 1) +
  facet_wrap(~ Variable, scales = "free") +
  ggtitle('Energy_Output vs Various Variables with Predicted Values') +
  xlab('Variable') +
  ylab('Energy_Output') +
  scale_color_manual(values = c('red', 'dark green', 'blue'))
```

```
# ========= Модель KNN-R =========

# Побудова моделі
knn.fit <- knnreg(traindata$Energy_Output~., data=traindata)
knn.fit

# Прогнозування
test.knn <- predict(knn.fit, newdata = testdata)

# Обчислення показників
rmse.knn <- sqrt(sum((test.knn - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
mae.knn <- sum(abs(test.knn - testdata$Energy_Output))/length(testdata$Energy_Output)
mape.knn <- sum(abs((test.knn - testdata$Energy_Output)/test.knn))/length(testdata$Energy_Output)*100
residuals.knn <- testdata$Energy_Output - test.knn
SST.knn <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
SE.knn <- sum(residuals.knn^2)
R2.knn <- 1 - (SE.knn / SST.knn)

# Результати для моделі
c(R2=R2.knn, RMSE = rmse.knn, MAE = mae.knn, MAPE = mape.knn)

# Перетворення даних у довгий формат для фактичних даних
combined_testdata.knn <- testdata %>%
  gather(key = "Variable", value = "Value", Relative_Humidity, Exhaust_Vacuum, Temperature,
Ambient_Pressure) %>%
  mutate(Type = "Test")

# Створення окремого фрейму даних для прогнозів
# Об’єднання прогнозів зі змінними для рядків
predictions.knn <- data.frame(
  Variable = rep(c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure"), each =
nrow(testdata)),
  Value = unlist(testdata[, c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure")]),
  Predicted = rep(test.knn, times = 4)
)

# Графік
ggplot() +
  geom_point(data = combined_testdata.knn, aes(x = Value, y = Energy_Output, color = Type)) +
  geom_line(data = predictions.knn, aes(x = Value, y = Predicted, color = "Predicted"), size = 1) +
  facet_wrap(~ Variable, scales = "free") +
  ggtitle(‘Energy_Output vs Various Variables with Predicted Values’) +
  xlab(‘Variable’) +
  ylab(‘Energy_Output’) +
  scale_color_manual(values = c(‘red’, ‘dark green’, ‘blue’))

# ========= Модель на основі штучної НМ =========

# Побудова моделі
ann.fit <- neuralnet(traindata$Energy_Output~., hidden = 2, act.fct = "logistic", data=traindata)
plot(ann.fit)

# Прогнозування
test.ann <- predict(ann.fit, newdata = testdata)

# Обчислення показників
rmse.ann <- sqrt(sum((test.ann - testdata$Energy_Output)^2)/length(testdata$Energy_Output))
mae.ann <- sum(abs(test.ann - testdata$Energy_Output))/length(testdata$Energy_Output)
mape.ann <- sum(abs((test.ann - testdata$Energy_Output)/test.ann))/length(testdata$Energy_Output)*100
residuals.ann <- testdata$Energy_Output - test.ann
```

```
SST.ann <- sum((testdata$Energy_Output - mean(testdata$Energy_Output))^2)
SE.ann <- sum(residuals.ann^2)
R2.ann <- 1 - (SE.ann / SST.ann)

# Результати для моделі
c(R2=R2.ann, RMSE = rmse.ann, MAE = mae.ann, MAPE = mape.ann)

# Перетворення даних у довгий формат для фактичних даних
combined_testdata.ann <- testdata %>%
  gather(key = "Variable", value = "Value", Relative_Humidity, Exhaust_Vacuum, Temperature,
Ambient_Pressure) %>%
  mutate(Type = "Test")

# Створення окремого фрейму даних для прогнозів
# Об’єднання прогнозів зі змінними для рядків
predictions.ann <- data.frame(
  Variable = rep(c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure"), each =
nrow(testdata)),
  Value = unlist(testdata[, c("Relative_Humidity", "Exhaust_Vacuum", "Temperature", "Ambient_Pressure")]),
  Predicted = rep(test.ann, times = 4)
)

# Графік
ggplot() +
  geom_point(data = combined_testdata.ann, aes(x = Value, y = Energy_Output, color = Type)) +
  geom_line(data = predictions.ann, aes(x = Value, y = Predicted, color = "Predicted"), size = 1) +
  facet_wrap(~ Variable, scales = "free") +
  ggtitle(‘Energy_Output vs Various Variables with Predicted Values’) +
  xlab(‘Variable’) +
  ylab(‘Energy_Output’) +
  scale_color_manual(values = c(‘red’, ‘dark green’, ‘blue’))

# ==================== Побудова дворівневої ансамблевої структури (класичний варіант)
==================

# Список моделей та відповідних прогнозів
var.bias.models <- list(lm = test.lm, dt = test.dt, rf = test.rf, svr = test.svr, knn = test.knn, ann = test.ann)

# Обчислення зміщення та дисперсії для кожної моделі
var.bias.res <- sapply(var.bias.models, function(preds) {
  bias <- mean(preds - testdata$Energy_Output)
  variance <- var(preds)
  c(bias, variance)
})

# Перетворення результатів у фрейм даних
var.bias.resdf <- as.data.frame(t(var.bias.res))
colnames(var.bias.resdf) <- c("Bias", "Variance")
var.bias.resdf

# Збереження testdata в ens1.preds
ens1.preds <- testdata

# Додавання стовпців з прогнозами
ens1.preds <- data.frame(
  ens1.preds,              # Існуючі дані з testdata
  test.lm = test.lm,      # Прогнози лінійної регресії
  test.dt = test.dt,      # Прогнози дерев рішень
  test.rf = test.rf,      # Прогнози випадкового лісу
  test.svr = test.svr,    # Прогнози SVR
```

```
  test.knn = test.knn,    # Прогнози KNN
  test.ann = test.ann     # Прогнози нейронної мережі
)

# ===== Перший шар =====
ens1.control <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE)

ens1.stack.layer.preds <-  c('test.knn','test.dt','test.ann')
ens1.stack.layer.mods <- caretList(ens1.preds[,ens1.stack.layer.preds], ens1.preds$Energy_Output,
          trControl=ens1.control, methodList=c("glm"))

# Налаштування ens1.stack.layer.control для регресії
ens1.stack.layer.control <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE)

# Створення стекової моделі
ens1.stack.layer <- caretStack(ens1.stack.layer.mods, method="glm", trControl=ens1.stack.layer.control)
ens1.stack.layer

# Прогнозування за допомогою стекової моделі
ens1.stack.layer.test <- predict(ens1.stack.layer, newdata=ens1.preds)

# Обчислення показників
ens1.stack.layer.rmse <- sqrt(sum((ens1.stack.layer.test - ens1.preds$Energy_Output)^2) /
length(ens1.preds$Energy_Output))
ens1.stack.layer.mae <- sum(abs(ens1.stack.layer.test - ens1.preds$Energy_Output)) /
length(ens1.preds$Energy_Output)
ens1.stack.layer.mape <- sum(abs((ens1.stack.layer.test - ens1.preds$Energy_Output) / ens1.stack.layer.test)) /
length(ens1.preds$Energy_Output) * 100
ens1.stack.layer.residuals <- ens1.preds$Energy_Output - ens1.stack.layer.test
ens1.stack.layer.SST <- sum((ens1.preds$Energy_Output - mean(ens1.preds$Energy_Output))^2)
ens1.stack.layer.SE <- sum(ens1.stack.layer.residuals^2)
ens1.stack.layer.R2 <- 1 - (ens1.stack.layer.SE / ens1.stack.layer.SST)

# Результати для моделі
c(R2 = ens1.stack.layer.R2, RMSE = ens1.stack.layer.rmse, MAE = ens1.stack.layer.mae, MAPE =
ens1.stack.layer.mape)

ens1.preds$ens1.stack.layer.test <- ens1.stack.layer.test$pred

# ===== Другий шар =====
ens1.bag.layer.preds<-
  c('ens1.stack.layer.test','test.lm','test.rf', 'test.svr')
ens1.bag.layer.ctrl <- trainControl(method="repeatedcv", number=10, repeats=3)
ens1.bag.layer<-

train(ens1.preds[,ens1.bag.layer.preds],ens1.preds$Energy_Output,method='treebag',trControl=ens1.bag.layer.ctrl)

ens1.bag.layer

ens1.bag.layer.test<-predict(ens1.bag.layer,newdata= ens1.preds)

# Обчислення показників
ens1.bag.layer.rmse <- sqrt(sum((ens1.bag.layer.test -
ens1.preds$Energy_Output)^2)/length(ens1.preds$Energy_Output))
ens1.bag.layer.mae <- sum(abs(ens1.bag.layer.test -
ens1.preds$Energy_Output))/length(ens1.preds$Energy_Output)
ens1.bag.layer.mape <- sum(abs((ens1.bag.layer.test -
ens1.preds$Energy_Output)/ens1.bag.layer.test))/length(ens1.preds$Energy_Output)*100
ens1.bag.layer.residuals <- ens1.preds$Energy_Output - ens1.bag.layer.test
ens1.bag.layer.SST <- sum((ens1.preds$Energy_Output - mean(ens1.preds$Energy_Output))^2)
```

```
ens1.bag.layer.SE <- sum(ens1.bag.layer.residuals^2)
ens1.bag.layer.R2 <- 1 - (ens1.bag.layer.SE / ens1.bag.layer.SST)

# Результати для моделі
c(R2=ens1.bag.layer.R2, RMSE = ens1.bag.layer.rmse, MAE = ens1.bag.layer.mae, MAPE =
ens1.bag.layer.mape)


# ================== Побудова дворівневої ансамблевої структури (експериментальний варіант)
==================

# Збереження testdata в ens2.pred
ens2.pred <- testdata

# Додавання стовпців з прогнозами
ens2.pred <- data.frame(
  ens2.pred,              # Існуючі дані з testdata
  test.lm = test.lm,      # Прогнози лінійної регресії
  test.dt = test.dt,      # Прогнози дерев рішень
  test.rf = test.rf,      # Прогнози випадкового лісу
  test.svr = test.svr,    # Прогнози SVR
  test.knn = test.knn,    # Прогнози KNN
  test.ann = test.ann     # Прогнози нейронної мережі
)

# ===== Перший шар =====
ens2.bag.preds<-
  c('test.knn','test.dt')
ens2.bag.layer.ctrl <- trainControl(method="repeatedcv", number=10, repeats=3)
ens2.bag.layer<-
  train(ens2.pred[,ens2.bag.preds],ens2.pred$Energy_Output,method='treebag',trControl=ens2.bag.layer.ctrl)

ens2.bag.layer

ens2.bag.layer.test<-predict(ens2.bag.layer,newdata= ens2.pred)

# Обчислення показників
ens2.bag.layer.rmse <- sqrt(sum((ens2.bag.layer.test -
ens2.pred$Energy_Output)^2)/length(ens2.pred$Energy_Output))
ens2.bag.layer.mae <- sum(abs(ens2.bag.layer.test -
ens2.pred$Energy_Output))/length(ens2.pred$Energy_Output)
ens2.bag.layer.mape <- sum(abs((ens2.bag.layer.test -
ens2.pred$Energy_Output)/ens2.bag.layer.test))/length(ens2.pred$Energy_Output)*100
ens2.bag.layer.residuals <- ens2.pred$Energy_Output - ens2.bag.layer.test
ens2.bag.layer.SST <- sum((ens2.pred$Energy_Output - mean(ens2.pred$Energy_Output))^2)
ens2.bag.layer.SE <- sum(ens2.bag.layer.residuals^2)
ens2.bag.layer.R2 <- 1 - (ens2.bag.layer.SE / ens2.bag.layer.SST)

# Результати для моделі
c(R2=ens2.bag.layer.R2, RMSE = ens2.bag.layer.rmse, MAE = ens2.bag.layer.mae, MAPE =
ens2.bag.layer.mape)

ens2.pred$ens2.bag.layer.test <- ens2.bag.layer.test

# ===== Другий шар =====
ens2.ctrl <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE)

ens2.stack.layer.preds <- c('ens2.bag.layer.test','test.lm','test.ann','test.rf', 'test.svr')
ens2.stack.layer.models <- caretList(ens2.pred[,ens2.stack.layer.preds], ens2.pred$Energy_Output,
                trControl=ens2.ctrl, methodList=c("glm"))
```

```
# Налаштування ens2.stack.layer.ctrl для регресії
ens2.stack.layer.ctrl <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE)

# Створення стекової моделі
ens2.stack.layer <- caretStack(ens2.stack.layer.models, method="glm", trControl=ens2.stack.layer.ctrl)
ens2.stack.layer

# Прогнозування за допомогою стекової моделі
ens2.stack.layer.test <- predict(ens2.stack.layer, newdata=ens2.pred)

# Обчислення показників
ens2.stack.layer.rmse <- sqrt(sum((ens2.stack.layer.test - ens2.pred$Energy_Output)^2) /
length(ens2.pred$Energy_Output))
ens2.stack.layer.mae <- sum(abs(ens2.stack.layer.test - ens2.pred$Energy_Output)) /
length(ens2.pred$Energy_Output)
ens2.stack.layer.mape <- sum(abs((ens2.stack.layer.test - ens2.pred$Energy_Output) / ens2.stack.layer.test)) /
length(ens2.pred$Energy_Output) * 100
ens2.stack.layer.residuals <- ens2.pred$Energy_Output - ens2.stack.layer.test
ens2.stack.layer.SST <- sum((ens2.pred$Energy_Output - mean(ens2.pred$Energy_Output))^2)
ens2.stack.layer.SE <- sum(ens2.stack.layer.residuals^2)
ens2.stack.layer.R2 <- 1 - (ens2.stack.layer.SE / ens2.stack.layer.SST)

# Результати для моделі
c(R2 = ens2.stack.layer.R2, RMSE = ens2.stack.layer.rmse, MAE = ens2.stack.layer.mae, MAPE =
ens2.stack.layer.mape)
```