

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,
Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
ПРОГНОЗУВАННЯ РЕЗУЛЬТАТІВ ПРОЦЕСУ ДЕСУЛЬФУРАЦІЇ
ДИМОВИХ ГАЗІВ ВІД ДІОКСИДУ СІРКИ
RESEARCH AND SOFTWARE IMPLEMENTATION OF METHODS
FOR PREDICTING THE RESULTS OF THE FLUE-GAS
DESULFURIZATION FROM SULFUR DIOXIDE

Виконав: студентка 2 курсу, групи КНТ-120м
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Харченко А.С.

(прізвище та ініціали)

Керівник Олійник А.О.

(прізвище та ініціали)

Рецензент Скрупський С.Ю.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
 Кафедра програмних засобів
 Ступінь вищої освіти магістр
 Спеціальність 121 Інженерія програмного забезпечення
 (код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
 “ ” 20__ року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТКИ

Харченко Анастасії Сергіївни

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів прогнозування результатів процесу десульфурзації димових газів від діоксиду сірки. Research and Software Implementation of Methods for Predicting the Results of the Flue-Gas Desulfurization from Sulfur Dioxide
 керівник проєкту (роботи) Олійник Андрій Олександрович, д.т.н., доцент,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
 затверджені наказом закладу вищої освіти від “05” листопада 2021 року
№ 435

2. Строк подання студентом проєкту (роботи) 15 грудня 2021 року
 3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Реферат. Перелік скорочень та умовних позначок. Вступ. 1. Аналіз предметної області. 2. Вибір та обґрунтування структури системи, яка проєктується. 3. Основні рішення щодо реалізації компонентів системи. 4. Керівництво програміста. 5. Керівництво оператора. 6. Експерименти та результати. Висновки. Перелік джерел посилань. Додатки.
 5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-6 Основна частина	Олійник А.О., професор		
Нормоконтролер	Камінська Ж.К., асистент		

7. Дата видачі завдання “6” вересня 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2–4 тижні	Розділ 1
3	Вибір засобів розробки програмного продукту.	5 тиждень	Розділ 2
4	Розробка архітектури програми.	6 тиждень	Розділ 3
5	Розробка програмного продукту	7-9 тижні	Розділ 4-5
6	Тестування та експериментальне дослідження програмного забезпечення.	10 тиждень	Розділ 6
7	Оформлення пояснювальної записки та документів до неї.	11 тиждень	Додатки
8	Нормоконтроль та рецензування.	12–13 тижні	
9	Захист роботи.	14 тиждень	

Студентка

(підпис) Харченко А.С.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Олійник А.О.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
148 с., 10 табл., 38 рис., 4 дод., 102 джерела.

ГЕНЕТИЧНИЙ АЛГОРИТМ, ДЕСУЛЬФУРАЦІЯ ДИМОВИХ ГАЗІВ,
НЕЙРОННА МЕРЕЖА, ПРОГНОЗУВАННЯ, PYTHON.

Об'єкт дослідження – процес прогнозування результатів очищення викидів димових газів в атмосферу.

Предмет дослідження – методи прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

Мета роботи – дослідження та програмна реалізація методів прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

Метод дослідження – традиційні моделі (метод найближчого сусіда, випадковий ліс, метод опорних векторів), нейронні мережі, комбіновані методи (нейронні мережі і генетичні алгоритми).

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мова програмування Python 3.7, середовище програмування Spyder, персональний комп'ютер з процесором Intel Core i5 під управлінням операційної системи Microsoft Windows 10.

Результати. Створено програмний Windows-застосунок, який здійснює прогноз вмісту діоксиду сірки у димовому газі на виході після процесу десульфурації.

Висновки. Створено програмний продукт для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

Галузь використання – прогнозування результатів та оптимізація експлуатаційних витрат процесу десульфурації димових газів від діоксиду сірки на вугільних електростанціях.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 148 pages, 38 figures, 10 tables, 4 appendixes, 102 sources.

DESULFURIZATION OF SMOKE GASES, FORECASTING, GENETIC ALGORITHM, NEURAL NETWORK, PYTHON.

The object of research is the process of predicting the results of cleaning flue gas emissions into the atmosphere.

The subject of research is methods for predicting the results of the process of desulfurization of flue gases from sulfur dioxide.

The purpose of this work is to study and implement methods for predicting the results of the process of desulfurization of flue gases from sulfur dioxide.

Research method – traditional models (nearest neighbor method, random forest, support vector machine), neural networks, combined methods (neural networks and genetic algorithms).

Materials, methods and technical means: object-oriented programming, the Python 3.7 programming language, the Spyder programming environment, a personal computer with an Intel Core i5 processor running the Microsoft Windows 10 operating system.

Results. A Windows software application has been created that predicts the content of sulfur dioxide in the flue gas at the outlet after the desulfurization process.

Conclusions. A software product has been created for predicting the results of flue gas desulfurization from sulfur dioxide.

The area of use is forecasting the results and optimizing the operating costs of the process of desulfurization of flue gases from sulfur dioxide at coal-fired power plants.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	11
1.1 Основні поняття	11
1.2 Огляд існуючих методів	13
1.3 Порівняльний аналіз існуючих методів вирішення завдання прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.....	32
1.4 Дослідження інформаційних систем.....	34
1.5 Порівняння існуючих систем дослідження, що можуть бути використані для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.....	39
1.6 Постановка завдання до роботи.....	41
1.7 Висновок	43
2 Вибір та обґрунтування структури системи, яка проєктується.....	44
2.1 Вибір мови програмування	44
2.2 Порівняння мов програмування	51
2.3 Вибір середовища розробки.....	52
2.4 Порівняння середовищ розробки	55
2.5 Огляд бібліотек машинного навчання	56
2.6 Структурна схема програмного продукту	61
2.7 Висновок	64
3 Основні рішення щодо реалізації компонентів системи.....	65
3.1 Попередня обробка даних	65
3.2 Основні відомі методи	66
3.3 Висновок	84
4 Керівництво програміста.....	85
4.1 Призначення й умови використання	85

4.2 Звертання до програмного продукту (файли проєкту)	86
4.3 Вхідні і вихідні дані	87
4.4 Повідомлення	89
4.5 Висновок	91
5 Керівництво оператора	92
5.1 Призначення й умови виконання програмного продукту.....	92
5.2 Виконання програмного продукту	93
5.3 Повідомлення оператору	97
5.4 Висновок	97
6 Експерименти та результати	98
6.1 Проведення експериментів	98
Висновки	100
Перелік джерел посилань	102
Додаток А Технічне завдання	114
Додаток Б Опис програмного продукту	121
Додаток В Лістинги файлів роботи	134
Додаток Д Слайди презентації.....	140

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
IDE	– Integrated Development Environment;
KNN	– <i>k</i> -nearest neighbor algorithm;
LSTM	– Long Short-term Memory;
MLP	– Multilayer Perceptron;
RBFN	– Radial Basis Function Network;
SO ₂	– діоксид сірки;
SVM	– Support Vector Machine;
ГА	– генетичний алгоритм;
ДДГ	– десульфурація димових газів;
НМ	– нейронна мережа;
ПП	– програмний продукт;

ВСТУП

За даними Nature Geoscience супутники NASA виявили 500 нових джерел забруднення повітря, близько 40 з них небезпечні діоксидом сірки (SO_2) [1]. Ця речовина вважається один з найчастіше зустрічаючих газів, який забруднює повітря та несе велику кількість небезпек для здоров'я людини і довкілля. Діоксид сірки пов'язана з різними респіраторними захворюваннями і підвищенням рівня смертності. Вдихання діоксиду сірки може викликати підвищений опір дихальних шляхів через звуження проходів у легенях [2].

Основними джерелами викидів двоокису сірки є теплові електростанції, що працюють на твердому і рідкому паливі, а також металургійні підприємства. Отже, контроль вмісту SO_2 в димових газах від спалювання вугілля є ефективним методом щодо скорочення викидів в атмосферу [2].

Викиди SO_2 можна зменшити за рахунок встановлення обладнання для сіркоочистки на новозбудованих та існуючих вугільних блоках, а також відповідних вимог до ефективності знесірчення. Щоб підвищити ефективність знесірчення і більш ефективно знизити викиди SO_2 , необхідно додатково оптимізувати систему управління знесірчення відповідно до потреб галузі [2].

Точне встановлення взаємозв'язку між параметрами процесу і фактичної ефективністю сіркоочистки є основою для оптимізації системи управління сіркоочистки. В даний час велика кількість локальних приладів для аналізу і моніторингу димових газів може безпосередньо контролювати концентрацію SO_2 в димових газах на вході і виході обладнання для десульфуризації і розраховувати її ефективність. Однак цей метод являє собою всього лише простий зворотний зв'язок за результатами процесу реакції знесірчення і не відображає результатів моніторингу. У той же час на обладнання для моніторингу сильно впливають зовнішні фактори, іноді

виникають збої і т. п., що призводить до неточних результатів вимірювань [2].

За останні роки були запропоновані численні підходи, включаючи експериментальні дослідження, математичні моделі та моделі машинного навчання, для прогнозування ефективності десульфурації. Серед них математичні моделі і моделі машинного навчання, а також їх комбінації викликали великий дослідницький інтерес. Однак прогнозування ефективності десульфурації все ж складно повністю змодельовати математично. Деякі дослідження спрощують систему за допомогою припущень, які вносять помилки в прогнози. Крім того, чисельні рішення, використовувані в цих математичних моделях, вимагають великих обчислювальних ресурсів. У цьому контексті моделі машинного навчання виявляються корисними інструментами для прогнозування [2].

Один з перспективних напрямків вирішення даної проблеми ґрунтується на застосуванні штучних нейронних мереж і генетичних алгоритмів, як найбільш прогресивних по відношенню проблем прогнозування ефективності десульфурації димових газів. Генетичні алгоритми (ГА) відносяться до класу методів пошуку, які ітераційно поліпшують якість варіантів вирішення процедурами рекомбінації і відбором для виживання. Завдяки універсальності обчислювальної схеми, можливостям паралельної реалізації та стійкості до шуму ГА знаходять успішне практичне застосування при вирішенні багатьох складних нелінійних багатовимірних задач оптимізації [3].

Отже, завдання прогнозування результатів процесу десульфурації димових газів від діоксиду сірки з використанням еволюційних генетичних алгоритмів та штучних нейронних мереж є актуальним завданням.

Для вирішення поставленого було розроблено модифікований генетичний метод для настройки вагових коефіцієнтів штучної нейронної мережі для підвищення точності прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Основні поняття

Десульфурація димових газів (ДДГ) – це набір технологій, які використовуються для видалення діоксиду сірки з димових газів, що утворюються при спалюванні викопного палива і багатьох промислових процесах, таких як очищення бензину, а також при виробництві цементу, паперу, скла, сталі, чавуну і міді. Системи ДДГ були розроблені як реакція на вихлопні гази, що утворюються на підприємствах, які працюють на викопному паливі. Викопні види палива, такі як вугілля і нафта, часто містять велику кількість сірки, і при спалюванні цього палива близько 95% або більше сірки перетворюється в діоксид сірки, який виділяється у вигляді димових газів [1]–[3].

Діоксид сірки – це кислий газ, який легко вступає в реакцію з іншими речовинами з утворенням зазвичай шкідливих сполук, таких як сірчана кислота, частки сульфату та сірчиста кислота. Діоксид сірки також є одним з основних інгредієнтів кислотних дощів. Кислотні дощі відбувається, коли діоксид сірки або інші газоподібні хімічні речовини, такі як оксиди азоту, викидаються в повітря. Ці гази піднімаються через атмосферу і, досягнувши верхніх рівнів хмар, вступають в реакцію з водою, киснем і сонячним світлом. Потім виробляються сірчані і азотні кислоти різних концентрацій. Отримані кислоти змішуються з конденсованою водяною парою у хмарах і падають на землю разом з водою у вигляді різних опадів. Такі опади з підвищеною кислотністю відомі як кислотні дощі. Кислотні дощі небезпечні для багатьох природних екосистем, вони можуть завдавати шкоди рослинам, піску, зробити воду непридатною для пиття і життя багатьох тварин [4]–[5].

Діоксид сірки пов'язана з різними респіраторними захворюваннями і підвищенням рівня смертності. Вдихання діоксиду сірки може викликати підвищений опір дихальних шляхів через звуження проходів у легенях [5].

Через руйнівну природу діоксиду сірки були введені суворі екологічні норми, що обмежують викиди SO_2 і спонукають промисловість вживати додаткових заходів з видалення діоксиду сірки з димових газів [5].

Найбільш поширеним методом ДДГ є метод мокрого знесірчення з використанням вапняку у якості абсорбенту через високу ефективність (більше 90%) абсорбції діоксиду сірки. Крім того, використання дешевого вапняку у якості реагенту і виробництво гіпсу є одними із значних переваг цього методу. ДДГ в системах мокрого знесірчення здійснюється в абсорбційних вежах (рис. 1.1) [6]–[8].

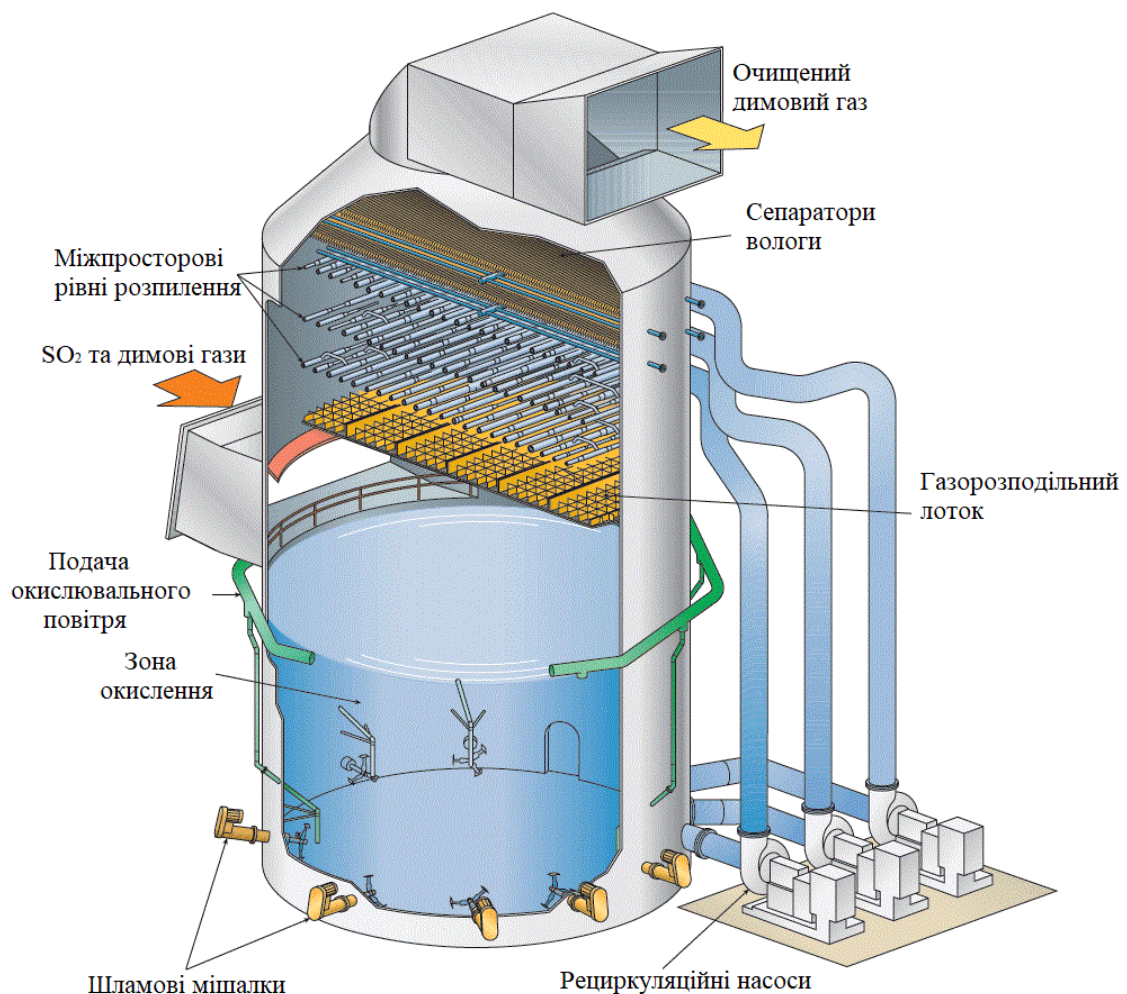


Рисунок 1.1 – Система мокрої десульфурації димових газів [8]

Основною частиною установки є абсорбер, в якому димові газ, що містить SO_2 , контактує з лужною суспензією. Перед тим, як лужна суспензія

потрапить в абсорбер, тверді частинки сорбенту подрібнюються (до бажаного розміру) на станції дроблення. Це неминучий крок, оскільки подрібнення твердих частинок збільшить швидкість розчинення в резервуарі для підготовки. Потім лужна суспензія подається в реакційний резервуар і транспортується в верхню частину абсорбера, зазвичай побудованого у вигляді розпилювального вежі або вежі з тарілками. Найбільш поширена конфігурація абсорбера – вертикальна з протитечією фаз, де димової газ, що містить SO_2 , тече вгору, а набір сопел розпорошує лужну суспензію (вапно / вапняк) вниз [9]–[11].

Коли теплий газ вступає в контакт з вапняковою суспензією, відбувається хімічна реакція між діоксидом сірки у газі та вапняком. Ця реакція видаляє діоксид сірки з димових газів і перетворює вапняк на сульфат кальцію. Отримана суспензія сульфату кальцію і вапняку подається на основу абсорбера, де її нагнітають стислим повітрям. Стисле повітря окислює сульфат кальцію і перетворює його на гіпс. Перш ніж димовий газ досягне вежі абсорбера, він проходить через теплообмінник. Призначення теплообмінника – охолодження димового газу у ДДГ до 90°C . Отримане тепло використовується для повторного нагріву охолодженого обробленого димового газу, коли він проходить назад через теплообмінник, де він знову нагрівається до температури 80°C , щоб надати йому плавучість, необхідну для проходження через димохід. Однак перед проходженням через теплообмінник газ проходить через сепаратори вологи для видалення шламу з вологого газу та для запобігання засмічень через перенесення вологи і гіпсу у випускний газопровід, а потім в теплообмінник [12]–[14].

1.2 Огляд існуючих методів

В останні роки з розвитком комп'ютерних технологій та штучного інтелекту можливості обробки нелінійних завдань поступово покращувалися, і було досягнуто значних успіхів у галузі моделювання та оптимізації

процесу десульфурації. Були запропоновані численні підходи, включаючи експериментальні дослідження, математичні моделі та моделі машинного навчання, для оцінки ефективності очищення димових газів від SO₂. В наш час найбільш поширеними методами прогнозування ефективності десульфурації димових газів від діоксиду сірки є метод найближчого сусіду, випадковий ліс, метод опорних векторів, генетичний алгоритм та штучні нейронні мережі [15]–[17].

1.2.1 Метод найближчого сусіду

Метод найближчого сусіду є одним з найбільш простих алгоритмів машинного навчання. Інша назва методу – алгоритм k -найближчих сусідів (k -nearest neighbor algorithm, KNN). Ідея полягає в тому, що використовується великий обсяг навчальних даних, де кожна точка даних характеризується набором змінних. Концептуально кожна точка відображається в багатовимірному просторі, де кожна вісь в просторі відповідає окремій змінній. Коли з'являється нова (тестова) точка даних, то необхідно знайти K найближчих сусідів, які є найближчими (тобто найбільш «схожими» на неї). Це приклад підходу «ледачого навчання», тому що він не вивчає розпізнавальну функцію з навчальних даних, а замість цього «запам'ятовує» навчальний набір даних [18].

Алгоритм k -найближчих сусідів обчислює відстань між точкою прогнозованих даних і відомої точкою даних, щоб обрати k -найближчих сусідів $(y_1, y_2, \dots, y_k)$, де y_1 є відомою точкою даних, найближчою до прогнозованої точки, а y_2 є відомою точкою даних, яка є другою найближчою до прогнозованої точки, і так далі. Отже, прогнозування може бути виконано за допомогою алгоритму k -найближчих сусідів за формулою (1.1):

$$S_i = \frac{1}{k} \cdot \sum_{j=1}^k S_{y_i}, \quad (1.1)$$

де S_i – прогнозоване значення;

S_{y_j} – є прогнозованим значенням j -й найближчої відомої точки даних y_j ;

k – кількість навчальних вибірок [19]–[20].

Перевагою методу найближчого сусіду є те, що цю модель дуже легко інтерпретувати і, як правило, цей метод дає прийнятну якість без необхідності використання великої кількості налаштувань. Він є хорошим базовим алгоритмом, який потрібно спробувати в першу чергу, перш ніж розглядати складніші методи. Як правило, побудова моделі найближчих сусідів відбувається дуже швидко, але, коли навчальний набір дуже великий (з точки зору кількості характеристик або кількості спостережень) отримання прогнозів може зайняти деякий час. При використанні алгоритму найближчого сусіда важливо виконати попередню обробку даних [21].

Недоліком методу найближчого сусіду є те, що він не створює будь-яких моделей або правил, які узагальнюють попередній досвід, – у виборі рішення вони ґрунтуються на всьому масиві доступних історичних даних, тому неможливо сказати, на якій підставі будуються відповіді. Існує складність вибору міри «близкості» (метрики). Від цієї міри головним чином залежить обсяг множини записів, які потрібно зберігати в пам'яті для досягнення задовільної класифікації або прогнозу. Також існує висока залежність результатів класифікації від обраної метрики. При використанні методу виникає необхідність повного перебору навчальної вибірки при розпізнаванні, наслідок цього – обчислювальна трудомісткість. Типові завдання даного методу – це завдання невеликої розмірності за кількістю класів і змінних. Даний метод не так добре працює, коли мова йде про набори даних з великою кількістю ознак (сотні і більше), і особливо погано працює в ситуації, коли переважна кількість ознак у більшій частині спостережень мають нульові значення [21].

1.2.2 Випадковий ліс

Випадковий ліс – це алгоритм машинного навчання з учителем, заснований на концепції ансамблевого навчання, яке являє собою процес об'єднання декількох класифікаторів для вирішення складної проблеми і підвищення продуктивності моделі. Алгоритм випадкового лісу складається з набору (ансамблів) дерев рішень, кожне з яких будується з вибірки даних, взятої з навчального набору із заміною. Цей крок вибір рядків із заміною називається беггінгом (bagging), також відомий як бутстреп агрегація (bootstrap aggregation). Беггінг – це ефективний ансамблевий алгоритм, оскільки кожне дерево рішень відповідає різному набору навчальних даних і, в свою чергу, має різну продуктивність [22]–[23].

На відміну від звичайних моделей дерев рішень, таких як дерева класифікації і регресії, дерева, які використовуються у ансамблі не обрізаються, що робить їх більш придатними для навчального набору даних. Це допомагає зробити кожне дерево більш різними і мати менш корельовані прогнози або помилки прогнозування. Потім кожна модель у ансамблі використовується для створення прогнозу нової вибірки, і ці N прогнозів усереднюються, щоб отримати прогноз для лісу. Прогноз для завдання регресії – це середнє значення прогнозу по деревах у ансамблі. Прогнозування проблеми класифікації – це більшість голосів за мітку класу по деревах у ансамблі [23].

Перевагою методу випадковий ліс є те, що він знижує ризик перенавчання, тому що класифікатор випадкового лісу бере середнє значення усіх прогнозів, усуваючи зміщення і, таким чином, усуваючи проблему перенавчання. У цьому методі масштабування об'єктів (стандартизація та нормалізація) не потрібна, оскільки він використовує підхід, заснований на правилах, замість обчислення відстані. Випадковий ліс ефективно обробляє нелінійні параметри, тому що вони не впливають на продуктивність випадкового лісу на відміну від алгоритмів на основі кривих. Таким чином,

якщо є висока нелінійність між незалежними змінними, випадковий ліс може перевершити інші алгоритми на основі кривих. Випадковий ліс є стабільним, наприклад, якщо в набір даних буде введена нова точка, то загальний алгоритм не сильно постраждає, оскільки нові дані можуть вплинути на одне дерево, але їм дуже важко вплинути на всі дерева [24].

Недоліком випадкового лісу є те що, для його навчання потрібно набагато більше часу у порівнянні з деревами рішень, оскільки він генерує багато дерев (замість одного дерева у разі дерева рішень) і приймає рішення більшістю голосів. Випадковий ліс вимагає більше обчислювальних ресурсів, оскільки він обробляє великі набори даних і йому потрібно більше ресурсів для зберігання цих даних. Модель випадкового лісу може бути досить складною для інтерпретації у порівнянні з деревом рішень, оскільки в дереві можна зробити вибір. Однак це неможливо у випадковому лісі, оскільки він має кілька дерев рішень [24].

1.2.3 Метод опорних векторів

Метод опорних векторів (Support Vector Machine – SVM) – це дуже потужний і універсальний метод машинного навчання, який здатний виконувати лінійну або нелінійну класифікацію, регресію і навіть виявлення викидів. Він є одним з найпопулярніших методів у машинному навчанні. Методи SVM особливо добре підходять для класифікації складних, але невеликих або середніх наборів даних [25].

Основна ідея, що лежить в основі методу SVM, полягає в тому, щоб побудувати $(n - 1)$ -мірну розділяючу гіперплощину, щоб розрізнити два класи в n -вимірному просторі. Точка даних розглядається як n -мірний вектор. Наприклад, дві змінні у наборі даних створюють двовимірний простір, то розділяюча гіперплощина буде прямою лінією (одновимірної), що розділяє простір навпіл. Коли задіяно більше вимірів, SVM шукає оптимальну розділяючу гіперплощину, яка називається розділяючою гіперплощиною з

максимальним запасом. Відстань між гіперплощиною і найближчою точкою даних з кожного боку (що називається опорними векторами) максимальна [25]–[26].

SVM буває двох типів: лінійний і не лінійний. Лінійний SVM використовується для даних, які лінійно розділяються, це означає, що набір даних можна класифікувати на два класи за допомогою однієї прямої лінії. Нелінійний SVM використовується для нелінійно розділених даних, це означає, що набір даних не може бути класифікований з використанням прямої лінії [26].

До переваг методу SVM можна віднести наступні фактори:

- теоретична і практична обґрунтованість методу;
- загальний підхід до багатьох завдань. Використовуючи різні функції, можна отримати рішення для різних завдань;
- автоматично обробляє пропущені значення;
- стійкі рішення, немає проблем з локальними мінімумами;
- не схильний до проблеми перенавчання;
- працює у будь-якій кількості вимірів [27].

Недоліками методу є:

- невисока продуктивність у порівнянні з більш простими методами;
- відсутність загальних рекомендацій по підбору параметрів і вибору ядра;
- побічні ефекти нелінійних перетворень;
- складності з інтерпретацією результату;
- для його реалізації потрібно більше ресурсів для обчислень;
- працює з невеликою кількістю векторів [27]–[28].

1.2.4 Генетичний алгоритм

Генетичний алгоритм (ГА) – це алгоритм адаптивного евристичного пошуку, який заснований на ідеях генетики і природного відбору, коли

сильніші особини з популяції переживають більш слабких і робляють наступне покоління особин. У ньому відбувається інтелектуальне використання випадкового пошуку на основі історичних даних для направлення пошуку в область більш високої продуктивності у просторі рішень. При використанні генетичного алгоритму не потрібно детально аналізувати і вивчати всі характеристики розв'язуваної проблеми, а тільки використовувати безперервне «природне успадкування» для вибору найбільш підходящого рішення різних складних проблем структури даних [29]–[30].

Основна особливість генетичного алгоритму полягає в тому, що йому не потрібно вказувати, що функція повинна мати безперервність або покладатися на інформацію про градієнт, він кодує модель і параметри конкретної проблеми, такі як перетворення проблеми в структурні об'єкти у вигляді послідовності, матриці і ланцюжка. Крім того, не обов'язково дотримуватися певних правил, але необхідно направляти пошук на основі певних ймовірнісних умов зміни. У своїй простій формі генетичний алгоритм включає в себе три типи операторів: селекція, кросовер і мутація [29]–[30].

Селекція – це процес вибору двох батьків з популяції для схрещування. Мета селекції полягає в тому, щоб виділити в популяції більш пристосованих особин в надії, що їх нащадки матимуть вищу пристосованість. Хромосоми відбираються з вихідної популяції, щоб стати батьками для відтворення. Відбір – це метод випадкового відбору хромосом з популяції відповідно до їх оцінної функції. Процес відбору ГА визначається як функція пристосованості, яка встановлює рівень успіху популяції, обчислення функції пристосованості дозволить популяції підтримувати і роботи наступне покоління. Цей процес являє собою серію етапів ітераційного процесу. Існують різні методи, які можна використовувати для вибору батьків. Використовувані методи вибору залежать від розв'язуваної задачі. Існує декілька методів батьківського вибору, таких як вибір на основі

рангів, вибір колеса рулетки, стохастична універсальна вибірка, вибір локального, вирізання вибору, турнір вибору [29]–[31].

Кросовер або кросинговер це оператор для формування нового індивіда з двох обраних батьків зі збереженням ознак обох батьків. Кросовер відбувається після вибору пар батьківських хромосом і допомагає в обміні інформацією між батьками для створення дітей. Під час кросовера батьківські хромосоми беруться у пару, і їх гени обмінюються в певному порядку для отримання потомства. Ці нащадки стають батьківськими хромосомами наступного покоління. Це виконується шляхом обміну алелями між двома обраними батьківськими хромосомами з метою дослідження нового простору рішень [29]–[31].

Мутація – це оператор, який підтримує генетичну різноманітність від однієї популяції до наступної. У той час як кросовер може створювати множину нових варіантів існуючих рішень, мутації можуть створювати абсолютно нові рішення. Вона застосовується випадковим чином після кросовера для одного або декількох генів у потомстві. ГА, що використовує двійкове кодування, просто повинен застосовувати логічний оператор «не» у випадково обраних бітових позиціях в рядку. Мутація випадково обраних генів, закодованих дійсними числами, виконується шляхом множення на випадковий коефіцієнт в межах певного інтервалу [29]–[31].

Якщо в одній хромосомі відбуваються два розриви, іноді область між розривами повертається на 180° перш ніж возз'єднатися з двома кінцевими фрагментами. Така подія викликає хромосомну мутацію, яка називається інверсією. Інверсії не змінюють загальної кількості генетичного матеріалу, тому інверсії, як правило, життєздатні і не показують особливих відхилень на фенотипічному рівні [29]–[31].

Елітарність дозволяє ГА зберігати одні з кращих хромосом на кожній ітерації. Такі хромосоми можуть бути втрачені, якщо вони не обрані для відтворення, або якщо вони зруйновані в результаті кросовера або мутації. Це значно покращує продуктивність ГА [30]–[32].

Загальна процедура функціонування класичного генетичного алгоритму представлена нижче [33].

Крок 1. Встановити лічильник ітерацій (часу): $t = 0$. Виконати ініціалізацію (initialization) початкової популяції особин: $P_t = \{H_1, H_2, \dots, H_N\}$.

Крок 2. Оцінити особини поточної популяції (evaluating) шляхом обчислення їх фітнес-функції $f(H_j)$, $j = 1, 2, \dots, N$.

Крок 3. Перевірити умови закінчення пошуку (termination criteria). Як такі умови можуть бути використані: досягнення максимально допустимого часу функціонування методу, числа ітерацій, значення функції пристосованості і т. ін. Якщо критерії закінчення пошуку задовільнено, тоді виконати перехід до кроку 12.

Крок 4. Збільшити лічильник ітерацій (часу): $t = t + 1$.

Крок 5. Вибрати частину популяції (батьківські особини) для схрещування (selection of parents) P' .

Крок 6. Сформувати батьківські пари (mating) з особин, що відібрані на попередньому кроці.

Крок 7. Схрестити (crossover) вибрані батьківські особини.

Крок 8. Застосувати оператор мутації (mutation) до особин P' .

Крок 9. Обчислити нову функцію пристосованості $f(H_j)$ особин, отриманих в результаті схрещування та мутації.

Крок 10. Сформувати нове покоління шляхом вибору особин, що вижили, виходячи з рівня їх пристосованості (replacing, selection of survivors).

Крок 11. Перейти до кроку 3.

Крок 12. Зупинення [33].

Важливим параметром ГА є визначення критеріїв зупинки. Генетичний алгоритм зазвичай завершується після певної кількості поколінь або при дотриманні деякого критерію зупинки (наприклад, придатність вища за деякий поріг, частота помилок нижча деякого порогу і т. д.). Потім найкраще рішення в сукупності повертається як загальне краще рішення. На

рис. 1.2 представлена узагальнена блок-схема роботи класичного генетичного алгоритму [33].

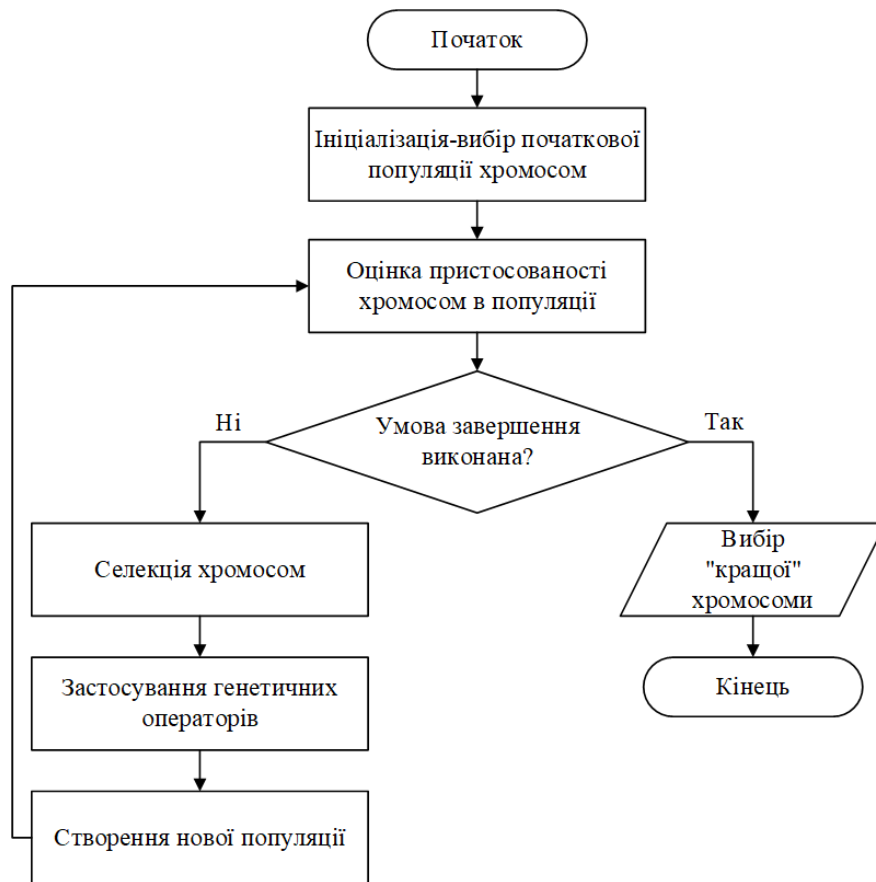


Рисунок 1.2 – Блок-схема класичного генетичного алгоритму [33]

До переваг генетичних алгоритмів можна віднести [34]:

- не вимагають максимум знань про вирішуване завдання та її рішення;
- швидше і ефективніше в порівнянні з традиційними методами;
- паралелізм, легко модифікується і адаптується до різних завдань;
- оптимізують безперервні і дискретні функції, а також багатокритеріальні задачі;
- використовують імовірнісні правила переходу, а не детерміновані правила;
- можливість пошуку великого і широкого простору рішень;

- відсутність обмежень на вид цільової функції (функції пристосованості);
 - використовують оцінку придатності, яка виходить з цільових функцій, без іншої похідної або допоміжної інформації;
 - працюють з хромосомою, яка представляє собою закодовану версію параметрів потенційних рішень, а не самих параметрів;
 - інтегрованість з штучними нейросетями та нечіткою логікою [34].
- До недоліків генетичного алгоритму відносять [34]–[35]:
- може не знайти найбільш оптимальне рішення певної проблеми у всіх випадках;
 - значення функції пристосованості оцінюється на множині поколінь, і це може бути досить коштовним процесом для певної кількості проблем;
 - передчасна збіжність, яка пов'язана з недостатньою різноманітністю хромосом в популяції;
 - проблема вибору різних параметрів, таких як розмір популяції, частота мутацій, частота кросовера, метод відбору;
 - існує велика ймовірність того, що навіть дуже маленькі зміни матимуть суттєвий вплив на придатність генерованого рішення [34]–[35].

1.2.5 Нейронні мережі

Нейронні мережі (НМ) – це набір алгоритмів, змодельованих за зразком людського мозку, призначених для розпізнавання закономірностей. Вони інтерпретують сенсорні дані у вигляді свого роду машинного сприйняття, маркування чи кластеризації необроблених вхідних даних. Вони розпізнають числові шаблони, що містяться у векторах, в які повинні бути переведені всі дані реального світу, будь то зображення, звук, текст або часові ряди [36].

НМ складаються з шарів вузлів, що містять вхідний шар, один або кілька прихованих шарів та вихідний шар. Кожен вузол або штучний нейрон

з'єднується з іншим і має пов'язану вагу та поріг. Якщо вихідний сигнал будь-якого окремого вузла перевищує задане порогове значення, цей вузол активується, надсилаючи дані на наступний рівень мережі. А якщо ні, то жодні дані не передаються на наступний рівень мережі [37].

Математичне представлення функціонування нейрона можна описати за допомогою формули (1.2) та (1.3):

$$y_k = \varphi(v), \quad (1.2)$$

$$v = \sum_{i=1}^m w_i u_i + b, \quad (1.3)$$

де y – вихідний сигнал нейрона;

φ – функція активації;

$w_1, w_2, \dots, w_m$ – значення ваг вхідних сигналів;

$u_1, u_2, \dots, u_m$ – компоненти вектора вхідних сигналів;

b – поріг активації [36]–[37].

Функція активації в нейронній мережі визначає, як зважена сума вхідних даних перетворюються на вихідний сигнал від вузла або вузлів у шарі мережі. Функція активації діє як функція стиснення, вихідний сигнал нейрона в нейронній мережі знаходиться між певними значеннями (зазвичай від 0 до 1 або може бути від -1 до 1) [38].

Класифікація НМ за характером навчання поділяється на:

- а) навчання з учителем (вихідний простір рішень НМ відомий);
- б) навчання без вчителя (вихідний простір рішень формується лише з урахуванням вхідних впливів);
- в) навчання з підкріпленням (використовується система призначення штрафів та заохочень, що отримуються у результаті взаємодії НМ з середовищем) [38].

Класифікація НМ з налаштування вагових коефіцієнтів поділяється на:

а) з фіксованими зв'язками – вагові коефіцієнти нейронної мережі обираються відразу, з умов завдання;

б) динамічними зв'язками – для них у процесі навчання відбувається налаштування синаптичних ваг [39].

Існують різні види нейронних мереж з різними функціями, які найкраще підходять для конкретних завдань, найбільш відомі архітектури:

- повнозв'язні мережі прямого розповсюдження;
- згорткові нейронні мережі;
- рекурентні нейронні мережі [39].

До переваг НМ можна віднести:

– НМ можуть навчатися на прикладах і застосовувати їх у разі виникнення аналогічної події, що дозволяє їм працювати з подіями у реальному часі;

– НМ здатні вивчати та моделювати нелінійні та складні відносини., що дійсно важливо, тому що в реальному житті багато відносин між входами та виходами є як нелінійними, так і складними;

– НМ можуть робити узагальнення - вивчивши вихідні дані та їх взаємозв'язки, вони можуть вивести неявні взаємозв'язки та на невидимі дані, тим самим змушуючи модель узагальнювати та прогнозувати на основі невидимих даних;

– на відміну від багатьох інших методів прогнозування, НМ не накладають жодних обмежень на вхідні змінні (наприклад, те, як вони повинні розподілятися). Крім того, багато досліджень показали, що НМ можуть краще моделювати гетероскедастичність, тобто дані з високою волатильністю і непостійною дисперсією, враховуючи її здатність вивчати приховані взаємозв'язки в даних без нав'язування будь-яких фіксованих взаємозв'язків у даних. Це дуже корисно при прогнозуванні фінансових часових рядів (наприклад, цін на акції), коли волатильність даних є дуже високою [39]–[41].

До недоліків НМ можна віднести:

- залежність від обладнання. НМ потребують процесорів з паралельною обчислювальною потужністю відповідно до їх структури. З цієї причини реалізація обладнання є залежною;
- незрозуміла поведінка мережі – це найважливіша проблема НМ, коли НМ видає пробне рішення, вона не дає зрозуміти, чому і як, це знижує довіру до мережі;
- визначення правильної структури мережі, не існує конкретного правила для визначення структури штучних нейронних мереж. Відповідна структура мережі досягається за рахунок досвіду, спроб та помилок;
- складність відображення проблеми в мережі, НМ можуть працювати з числовою інформацією. Проблеми повинні бути переведені в числові значення перед тим, як вони будуть представлені в НМ. Механізм відображення, який буде визначений, безпосередньо впливатиме на продуктивність мережі.
- тривалість мережі невідома, мережа зменшена до певного значення, помилка на вибірці означає, що навчання завершено, але це значення не дає оптимальних результатів [39]–[41].

1.2.6 Мережа радіально базисних функцій

Нейронна мережа з радіально базисними функціями (Radial Basis Function Network – RBFN) є однією з нейронних мереж, яка заснована на інтеграції базисної функції радіальної та спеціалізованої для виконання завдань, пов'язаних з нелінійною класифікацією [42].

RBFN відрізняються від багатошарових мереж персептронів, тому що вони не просто приймають вхідний вектор і множать його на коефіцієнт перед підсумовуванням результатів. Натомість у RBFN використовуються нейрони з радіальною базисною функцією, кожен з яких оцінює вхідний вектор, порівнюючи його збережене навчальне значення з вхідним і роблячи міру подібності. Потім значення подібності множиться на вагу та

підсумовується у вихідному шарі. Будь-яке нове введення можна легко обчислити за допомогою евклідової відстані між вхідними та навчальними даними [43].

Нейронна мережа з радіальною базовою функцією складається з вхідного шару (він утворений вихідними вузлами (сенсорними одиницями)), проміжного шару (прихований шар великого розміру, в якому одиниці (нейрони), що його формують, є базовими функціями для вхідних даних), та вихідного (відповідає в мережі за активацію шаблонів, які застосовуються на вхідному шарі) [43].

Радіальні базисні функції – це функції, які досягають рівня, близького до максимуму свого шляху, коли вхідний шаблон (X_n) знаходиться близько до центру нейрона. Якщо візерунок віддаляється від центру, то значення функції прагне до мінімального значення його шляху. На вихід мережі загалом впливає нелінійне перетворення, що відбувається у прихованому шарі через радіальну функцію та лінійне у вихідному шарі через безперервну лінійну функцію. Похідним від радіальних базисних моделей є використання стандартного відхилення для активації функції [44].

Перевагою RBF є її висока стійкість до вхідного шуму, а також її фаза навчання проходить швидше у порівнянні з багат шаровим персептроном, так як не потрібно навчання зі зворотним поширенням. Ролі інтерпретації вузлів прихованого шару прості, у порівнянні з багат шаровим персептроном. Кількість прихованих шарів та кількість вузлів у прихованому шарі визначаються у випадку мережі RBF, але у багат шаровому персептроні немає аналітичного підходу для визначення кількості вузлів у прихованому шарі або кількості прихованих шарів [42]–[45].

Хоча навчання в мережі RBF відбувається швидше, але класифікація повільніша порівняно з багат шаровим персептроном через те, що кожен вузол у прихованому шарі повинен обчислювати функцію RBF для вектора вхідної вибірки під час класифікації [42]–[45].

1.2.7 Довга короткочасна пам'ять (LSTM)

Довга короткочасна пам'ять (Long short-term memor – LSTM) – це варіант архітектури RNN, запропонованої Хохрайтером і Шмидхубером, який був розроблений для більш точного моделювання послідовностей і довгострокових залежностей, чим RNN. LSTM розширює архітектуру RNN автономним елементом пам'яті і механізмом стробування, який регулює потік інформації по мережі. Стробуючий механізм забезпечений трьома блоками, а саме: «вхідний», «забувальний» і «вихідний» вентиль. Цей механізм узгоджено визначає, яка інформація повинна зберігатися, як довго вона повинна зберігатися і коли повинна бути прочитана з комірки пам'яті. В результаті LSTM здатні зберігати ключову інформацію вхідних сигналів і ігнорувати менш важливі частини [46]–[47].

У результаті LSTM здатні зберігати ключову інформацію вхідних сигналів та ігнорувати менш важливі частини. Цей елемент пам'яті має періодично самозв'язаний лінійний блок, званий «карусель постійних помилок», який містить вектор стану (C_t), що дозволяє зберігати залежності впродовж тривалого часу. Отже, на відміну від звичайних RNN, LSTM зберігають інформацію і поширюють помилки для набагато більш довгого ланцюжка в мережі і долають проблему зникаючого градієнта [46]. На рис. 1.3 показана базова архітектура одного блоку мережі LSTM [47].

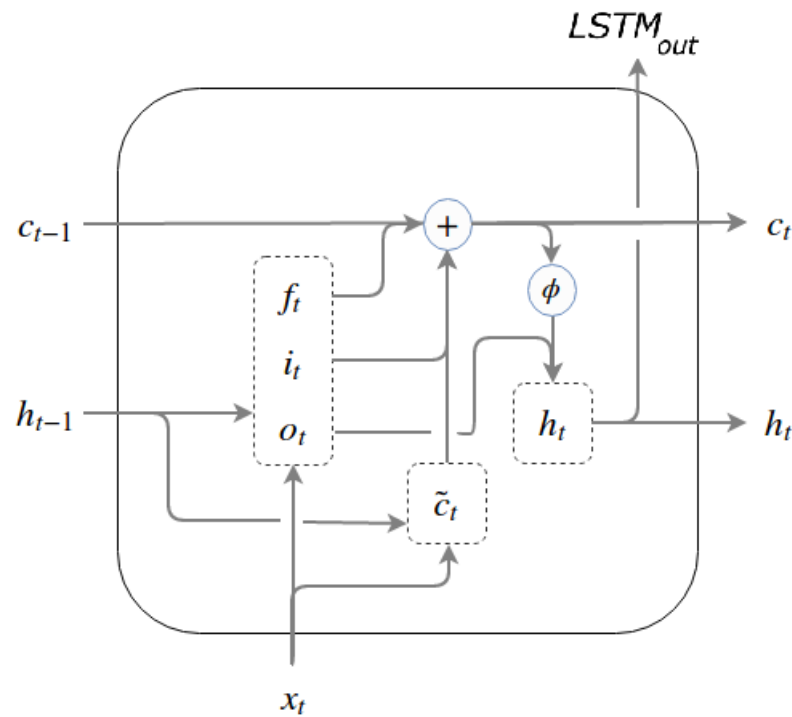


Рисунок 1.3 – Базова архітектура одного блоку мережі LSTM [47]

На рис. 1.3 $x_t \in R^m$ позначає вхід на тимчасовому кроці t , $C_{t-1} \in R^n$ та $C_t \in R^n$ позначають стан комірки на тимчасових кроках t і $(t-1)$, а $h_t \in R^n$ та $h_{t-1} \in R^n$ відповідають виходу на тимчасових кроках t та $(t-1)$ відповідно, де m представляє розмір вхідного вектора, а n означає розмір комірки пам'яті. Короткочасна пам'ять LSTM забезпечується станом h_t , в той час як стан C_t дозволяє зберігати довгострокові залежності. Елемент забування f_t приймає x_t і h_{t-1} як вхідні дані, щоб визначити, яка інформація повинна бути збережена в C_{t-1} . Функції активації затвора i_t , o_t і f_t зазвичай є сигмовидними шарами, отже на виході проектується значення від 0 до 1 для кожного значення в C_{t-1} , що описує масштаб збереження інформації, тобто 0 означає повне видалення значення з комірки пам'яті, у той час як 1 є повним збереженням цього значення в комірці пам'яті. Вхідний клапан i_t супроводжується сигмовидним шаром, який використовує x_t та h_{t-1} для визначення значень, які повинні бути підсумововані шляхом додавання C_t . Додатково, нелінійний шар ϕ (наприклад $\tanh$) також вводиться для генерації вектора значень-кандидатів, позначених як C'_t , для оновлення стану C_t . Вихідний клапан o_t регулює

вихідні значення комірки LSTM на основі оновленого стану C_t . Точно так, як у шлюзах забування та входу, вихідний клапан є сигмоїдальним шаром для фільтрації вихідних даних. Відповідно, оновлений стан комірки C_t подається в $\tanh$ -шар (ϕ), який масштабує вектор до значення між (-1) та $(+1)$. Потім поелементно помножується на вихідний сигнал сигмоїдального шару для обчислення кінцевого вихідного сигналу h_t комірки на тимчасовому кроці t [46]–[47].

Перевага мережі LSTM полягає у її здатності обробляти багатовимірні вектори ознак та можливості вибіркового запису історичної інформації [48].

Однак недоліком алгоритму LSTM є те, що навчання моделі займає багато часу і вимагає великої вибірки даних, а також складність налаштування її параметрів [49].

1.2.8 COGANN

Об'єднання генетичних алгоритмів і нейронних мереж називається COGANN (Combinations of Genetic Algorithms and Neural Networks), ці Об'єднання можна розділити на два загальні типи – допоміжні об'єднання, в яких НМ і ГА застосовуються послідовно, та рівноправні об'єднання, в яких вони застосовуються одночасно. В підтримуючі об'єднання одна з цих технологій ГА або НМ зазвичай відіграє роль основного вирішувача проблеми, в той час як інша грає допоміжну роль, або ставлячи завдання для вирішення, або аналізуючи рішення після того, як воно було досягнуто. Їх також можна використовувати незалежно для вирішення одного і того ж завдання. У рівноправному об'єднанні ГА і НМ працюють разом, щоб вирішити проблему. Серед спільних підходів дві основні групи – це ті, які використовують ГА як метод навчання для НМ, і ті, які використовують ГА для визначення топології НМ (яка, у свою чергу, часто використовує метод локального навчання, такий як зворотне поширення). В табл. 1.1 представлено класифікацію типів об'єднання ГА і НМ [50]–[51].

Таблиця 1.1 – Типи об'єднання генетичних алгоритмів і нейронних мереж

Вид об'єднання	Характеристика об'єднання	Приклади використання
Незалежне	Незалежне використання ГА та НМ для вирішення одного і того ж завдання	НМ прямого поширення, НМ з картою ознак Кохонена, що самоорганізуються, ГА із системою класифікаторів.
Допоміжне	Використання НМ для допомоги ГА.	Використання НМ для вибору початкової популяції ГА.
	Використання ГА для допомоги НМ.	Використання ГА для вибору ознак або перетворення простору ознак, що використовується НМ.
		Використання ГА для вибору правила навчання або параметрів, які керують навчанням у НМ.
		Використання ГА для аналізу НМ.
Рівноправне	ГА для навчання НМ	В НМ відбувається еволюція синаптичних ваг. ГА використовується як правило навчання НМ і оптимізує певні ваги НМ.
	ГА визначення топології НМ	Еволюція архітектури НМ, ГА застосовується у поєднанні з нейронним навчанням та оптимізує загальні структурні параметри НМ.
	Системи, які поєднуючи адаптивні стратегії ГА і ШНМ	ГА використовується як засіб для визначення синаптичних ваг НМ, які закодовані як двійкові чи дійсні числа у вигляді рядку.
		Побудова ГА з використанням НМ.
		Створення оператора схрещування в ГА за допомогою НМ.

1.3 Порівняльний аналіз існуючих методів вирішення завдання прогнозування результатів процесу десульфурації димових газів від діоксиду сірки

Проаналізувавши роботи [18]–[51] було проведено порівняльний аналіз досліджених методів для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки (табл.1.2).

Таблиця 1.2 – Порівняльна характеристика існуючих методів для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки

Можливості	Метод найближчого сусіду	Випадковий ліс	Метод опорних векторів	Генетичний алгоритм	Штучна нейронна мережа
Простота реалізації	Так	Ні	Ні	Так	Ні
Можливість розпаралелювання	Ні	Так	Так	Так	Так
Схильність до перенавчання	Ні	Ні	Ні	Ні	Так
Стійкість до шуму та викидів	Ні	Так	Ні	Ні	Так
Самонавчання	Ні	Ні	Ні	Так	Так
Обчислювально витратно для великих та складних наборів даних	Так	Так	Так	Ні	Ні
Легкість інтерпретації одержаних результатів	Так	Ні	Ні	Так	Ні
Адаптація до змін у вирішуваному завданні	Ні	Ні	Ні	Так	Так
Автоматично обробляє пропущені значення	Ні	Так	Ні	Ні	Ні
Обчислювально витратний зі збільшенням кількості атрибутів	Так	Так	Так	Ні	Ні

Провівши порівняльний аналіз досліджених методів для прогнозування результатів процесу десульфурзації (табл. 1.2) було встановлено, що математичні моделі зазвичай будуються у припущенні нормальних умов, точність прогнозів обмежена, що ускладнює адаптацію до змін навколишнього середовища та старіння обладнання. Можливість узагальнення моделей, керованих даними, обмежена діапазоном навчальних даних, що призводить до зниження можливості прогнозування, коли робочі умови виходять за межі навчального діапазону. Також в ході роботи було встановлено, що найбільш ефективним методом для вирішення поставленого завдання є використання нейронних мереж. Оскільки вони мають такі переваги як можливість адаптації до змін, яка дозволяє їм працювати навіть у критичній ситуації, самонавчання, швидкість роботи (складання прогнозу і планування може йти набагато швидше, ніж при використанні точних алгоритмів), а також високу стійкість до шумових даних та можливістю застосування необмеженої кількості незалежних змінних. При цьому, незважаючи на їх очевидні переваги, основним недоліком є складність вибору ефективної методики навчання та створення навчального масиву даних, а також складність правильного розподілу ваги матриці на початковому етапі, в умовах невизначеності. Для вирішення цієї проблеми добре підійде генетичний метод, перевага даного методу полягає в більшій точності та зниженні кількості часу, що витрачається на навчання. Він ефективний для навчання нейронної мережі типу «багатошаровий перцептрон», «ШНМ із загальною регресією», «мереж Кохонена» та ін. Застосування еволюційних генетичних алгоритмів на етапі навчання допомагає, скоротити кількість навчальних циклів за рахунок використання параметрів, що мають найбільшу вагу. Отже, підсумовуючи вищесказане, було прийнято рішення використовувати ГА та НМ для прогнозування результатів процесу десульфурзації димових газів від діоксиду сірки [48]–[51].

1.4 Дослідження інформаційних систем

1.4.1 RapidMiner

RapidMiner – це платформа аналізу даних, що дозволяє розгортати прогнозні моделі, моделі машинного навчання та ефективна при вирішенні різноманітних аналітичних завдань [52].

Платформа дослідження даних RapidMiner від однойменної компанії призначена для роботи для професійних аналітиків та дослідників даних при вирішенні завдань статистичного аналізу та аналізу даних із застосуванням технологій штучного інтелекту та машинного навчання. Система використовується для ділових та комерційних додатків, а також для наукових та аналітичних досліджень, освіти та навчання, швидкого прототипування та розробки додатків [52].

Програмний продукт RapidMiner є відкритою та розширюваною платформою аналізу даних дозволяє повноцінно впровадити методи машинного навчання та технології штучного інтелекту на підприємстві. Створений для аналітичних команд програмне забезпечення RapidMiner поєднує весь життєвий цикл науки про дані, включаючи підготовки даних до машинного навчання, візуалізацію результатів, валідацію моделей, розгортання прогнозних моделей та оптимізацію моделей [52].

Основна перевага RapidMiner – це відсутність необхідності в написанні робочих кодів, що суттєво підвищує швидкість обробки даних та зменшує кількість помилок. RapidMiner пропонує множину процедур, особливо в області вибору атрибутів та виявлення викидів. Програмний продукт має всі можливості для оцінки моделі з використанням наборів перехресної перевірки та незалежної перевірки [52].

Недоліком RapidMiner є його обмежені можливості поділу набору даних на набори для навчання та тестування, а також його обмежені варіанти використання набору процесорів та модулів, які він містить [52].

1.4.2 KNIME

KNIME – це масштабована платформа для аналізу даних, яка дозволяє фахівцям з обробки даних легко перемикатися на комп'ютер, передавати та обробляти великі дані. Вона пропонує статистичні та математичні функції, елементи управління робочим процесом, розширені алгоритми прогнозування, алгоритми машинного навчання та багато іншого, щоб оптимізувати завдання. Її інтуїтивно зрозумілий інтерфейс дозволяє легко орієнтуватися в програмному забезпеченні та допомогти у створенні потоків даних, реалізації певних тактик аналізу та легкому аналізі результатів, моделей та інтерактивних оглядів [53].

Платформа KNIME створена з використанням можливостей платформ Eclipse і використовує можливість розширення своїх модулів за допомогою модулів і з'єднувачів. Плагіни створені для підтримки інтеграції з використанням найсучасніших методів інтелектуального аналізу тексту, зображень та часових рядів. Разом із платформою Eclipse це програмне забезпечення включає понад 2000 модулів, які підтримують різні конектори даних для різних форматів файлів та основних баз даних. Ці модулі також підтримують алгоритми машинного навчання, різні типи даних (наприклад, зображення, документи, XML та JSON) та безліч статистичних робіт [53].

Платформа KNIME також інтегрована з різними проектами з відкритим вихідним кодом, такими як ScikitLearn та алгоритми машинного навчання Keras. Ці інтеграції підтримують глибоке навчання та оболонки для кодів викликів. KNIME надає вузли, які допомагають користувачам запускати Python, Java, Perl та інші сценарії кодування. Вона підтримує різні методи створення звітів через Інтернет та пропонує повну схему робочого процесу [53].

Основними перевагами KNIME є потужна аналітика, локальна автоматизація та можливість взаємодії з програмами, що дозволяють візуалізувати та аналізувати молекулярні дані. Вона також інтегрує всі

модулі аналізу, які пропонує доступ до широкої бібліотеки статистичних процедур [53].

Основними недоліками KNIME є обмежена можливість поділу набору даних, відсутність методів оболонки для вибору дескриптора, а також KNIME має лише обмежені методи вимірювання помилок [53].

1.4.3 Stata

Stata – це повноцінне інтегроване програмне рішення, що забезпечує потреби в галузі науки про дані – маніпулювання даними, візуалізацію, статистичний аналіз та автоматизовану звітність. Програмний продукт Stata від компанії StataCorp призначений для статистичних досліджень над різноманітними вибірками даних з різних предметних галузей та дисциплін [54].

Система надає сотні статистичних інструментів для керування даними, статистичного аналізу та інших завдань аналізу даних. Статистичний аналіз за допомогою програмного забезпечення Stata може бути виконаний за допомогою графічного інтерфейсу або за допомогою текстових програм на базі синтаксису Stata, який використовується як командний рядок. Інтерфейс користувача досить складний і підходить при проведенні глибоких систематичних досліджень. Користувачі можуть використовувати програмне забезпечення для збору даних, створення звітів, управління та обробки даних, а також для створення порівняльних звітів для своїх дослідних цілей [54].

Переваги ПП Stata:

- має широкий спектр функцій, включаючи майже всі відомі статистичні процедури;
- зручний і простий в освоєнні [54].

Недоліки ПП Stata:

- відсутність можливості створення власних функцій в Stata;

– його функції обмежені числовими чи категоріальними даними, Stata не може аналізувати просторові дані та інші типи [54].

1.4.4 Matlab

Matlab – це платформа для програмування, моделювання та симуляції, яка об'єднує середовище робочого столу, налаштоване для ітеративного аналізу та процесів проектування, з мовою програмування, яка безпосередньо виражає математику матриць та масивів [55].

Система автоматизованого проектування була створена компанією-розробником програмного забезпечення MathWorks, щоб допомогти користувачам аналізувати дані, розробляти алгоритми та створювати моделі. Він може похвалитися професійно розробленими інструментами, що пройшли ретельне тестування та повною документацією. Інтерактивні програми показують, як різні алгоритми працюють із даними користувачів. Технологія також масштабується, тому аналіз може виконуватися на кластерах, графічних процесорах і хмарах з мінімальними змінами коду, що усуває необхідність переписувати або вивчати програмування великих даних [55].

Matlab має великий набір бібліотек для машинного навчання. У ньому присутній набір інструментів – Deep Learning Toolbox, який забезпечує основу для проектування та реалізації глибоких нейронних мереж. Його можна використовувати для більш простого проектування складних нейронних архітектур. Він демонструє повний набір можливостей для глибокого навчання та забезпечує наскрізний інтегрований робочий процес від дослідження до прототипу. Matlab має Parallel Computing Toolbox для розподілу навчання між багатоядерними процесорами, графічними процесорами та кластерами комп'ютерів з декількома процесорами та графічними процесорами [55].

Переваги:

- не залежить від платформи, можна встановити на різних операційних системах, таких як Windows, Vista, Linux і Macintosh;
- наявність великої кількості бібліотеки машинного навчання
- має багато основних команд побудови графіків та зображень, за допомогою яких графіки та зображення можуть відображатися на будь-якому графічному пристрої виводу, який надається комп'ютеру, на якому працює Matlab [55].

Недоліки:

- мова, що інтерпретується і для її виконання потрібно більше часу, ніж для інших скомпільованих мов, таких як C, C++;
- висока вартість, користувачам необхідно отримати ліцензійну версію від MathWorks [55].

1.4.5 SAS Enterprise Miner

SAS Enterprise Miner – це програмний продукт для оптимізації процесу інтелектуального аналізу даних при розробці описових та прогнозних моделей з використанням структурованих алгоритмів та візуальних показників оцінки [56].

Програмний продукт SAS Enterprise Miner від компанії SAS призначений для інтелектуального аналізу даних. Програма дозволяє отримати інформацію, що сприяє кращому прийняттю рішень на основі даних та фактів, оптимізує процес інтелектуального аналізу даних для швидкої розробки моделей, розуміння ключових взаємозв'язків та пошуку найважливіших патернів у бізнесі [56].

Аналітична система SAS Enterprise Miner дозволяє користувачам створювати моделі за допомогою блок-схем, прогностичного моделера, дослідницьких алгоритмів та програмного коду. Система забезпечує розширену інтеграцію з системами управління даними та розгортання моделей, а завдяки широкому спектру вибору конфігурації пакета в

залежності від вимог бізнесу немає необхідності придбавати системи спеціалізованих рішень [56].

Важливою особливістю SAS Enterprise Miner є його повна інтеграція із програмним продуктом SAS Warehouse Administrator, призначеним для розробки та експлуатації інформаційних сховищ та іншими компонентами системи SAS. Розробка проектів Data Mining може виконуватися як локально, і у архітектурі клієнт-сервер [56].

Перевагою SAS Enterprise Miner є його зручний та зрозумілий інтерфейс, що дозволяє користувачам самостійно створювати моделі аналізу та прогнозування, а також автоматизований процес регламентного застосування моделей. У ньому є можливість пакетної обробки складних процесів, швидкий збір та підготовка даних, їх агрегація та дослідження. SAS Enterprise Miner має високу продуктивність навіть при роботі з великим масивом розрізнених даних [56].

Недоліком SAS Enterprise Miner є його висока вартість, тому що фізична особа або організація не зможуть використовувати всі додатки без належної ліцензії. SAS Enterprise Miner не є відкритим вихідним кодом, тому алгоритми, які використовуються в ньому, недоступні для загального використання, а також інтелектуальний аналіз тексту в ньому клопіткий та складний процес [56].

1.5 Порівняння існуючих систем дослідження, що можуть бути використані для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки

Проаналізувавши роботи [52]–[56] було створено порівняльну характеристику існуючих систем дослідження, що можуть бути використані для дослідження та створення прогнозування результатів процесу десульфурації димових газів від діоксиду сірки (талб. 1.3). Досліджувані

системи не є спеціальними програмними забезпеченнями і може бути використані у якості середовища для розробки спеціальних рішень [56].

Таблиця 1.3 – Порівняльна характеристика існуючих програмних засобів

Можливості	RapidMiner	KNIME	Stata	Matlab	SAS Enterprise Miner
Ліцензія	Proprietary	GNU, General Public Licence	Proprietary	Proprietary	Proprietary
Операційна система	Кросплат форменне	Windows, Linux, macOS	Windows, Linux, macOS	Windows, Linux, macOS	Windows, Linux
Доступність	Open Source	Open Source	Hi	Hi	Hi
Наявність пробної версії	Так	Так	Hi	Так	Так
Можливість аналізу великих даних	Так	Так	Hi	Так	Hi
Інтелектуальний аналіз даних	Так	Так	Так	Так	Так
Візуалізації даних	Так	Так	Так	Так	Так
Можливість машинного навчання	Так	Так	Hi	Так	Так
Можливість адміністрування	Hi	Так	Hi	Да	Так
Імпорт/експорт даних	Так	Так	Так	Так	Так
Інтерактивна аналітична обробка (OLAP)	Так	Так	Hi	Так	Hi
Наявність API	Так	Так	Hi	Так	Hi
Індикація трендів та проблем	Так	Так	Так	Так	Так
Прогнозування та передбачувана аналітика	Так	Так	Так	Так	Так
Статистичний аналіз	Так	Так	Так	Так	Так
Можливість доступу для багатьох користувачів	Hi	Так	Hi	Hi	Так

1.6 Постановка завдання до роботи

Метою роботи є дослідження та програмна реалізація методів прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

У роботі розглядається метод мокрого сіркоочищення димових газів від діоксиду сірки. В основі методу мокрого абсорбційного очищення димових газів лежить нейтралізація сірчистої кислоти, що виходить в результаті розчинення діоксиду сірки, що міститься в димових газах, гідратом оксиду кальцію $\text{Ca}(\text{OH})_2$ (вапном) або карбонатом кальцію CaCO_3 (вапняком). Вапно – більш дорожчий і дефіцитний матеріал, однак застосування вапна в якості сорбенту дозволяє майже в два рази скоротити витрати реагенту, знизити витрати електроенергії на приготування суспензії і зрошення абсорбера. Ефективність використання $\text{Ca}(\text{OH})_2$ для очищення газу залежить від температури газу, його вологості й часу перебування реагенту в системі. Для регулювання температури застосовується інжекція дрібнорозпиленої води під високим тиском, яка випаровується і прискорює реакції абсорбції [1]–[3].

Проаналізувавши принцип роботи методу мокрої абсорбційної очистки димових газів, було встановлено, що узагальнену модель залежності ступеня десульфурації від використовуваних реагентів з деяким припущенням, можна призвести у вигляд формули (1.4):

$$K_{des} = f(x_{con}, x_{first_stage}, x_{second_stage}, x_{water}, x_{lime}, x_{second_reagent}), \quad (1.4)$$

де x_{con} – вміст SO_2 на вході;

x_{first_stage} – об'єм газу на першу сходинку;

x_{second_stage} – об'єм газу на другу сходинку;

x_{water} – витрати води у абсорбері;

X_{lime} — витрати вапна;

X_{second_reagen} — витрати вторинного реагенту.

Отже, для вирішення даного завдання необхідно провести аналіз результатів роботи десульфуризації димових газів, встановити залежність між ступенем сіркоочищення від обсягу використовуваних реагентів та створити модель цієї залежності, яка надала б можливість прогнозувати ефективність знесірчення димових газів, внаслідок чого б знизилися викиди SO_2 та експлуатаційні витрати за рахунок оптимального використання реагентів у майбутньому [4]–[7].

Під час порівняльного аналізу існуючих методів було встановлено, що найбільш ефективним методом для вирішення поставленого завдання є використання НМ. Тому що вони мають такі переваги: високу адаптивність до зовнішніх змін, самонавчання, швидкість роботи (складання прогнозу й планування може йти набагато швидше, ніж при використанні точних алгоритмів), а також висока стійкість до шумів у вхідних сигналах та можливість застосування необмеженої кількості атрибутів. При цьому, незважаючи на їх очевидні переваги, основним недоліком є складність вибору ефективної методу навчання та складність правильного розподілу ваги матриці на початковому етапі, в умовах невизначеності. Для вирішення цієї проблеми добре підійде генетичний метод. Перевага даного методу полягає в більшій точності та зниженні кількості часу, що витрачається на навчання. Застосування еволюційних генетичних алгоритмів на етапі навчання допомагає скоротити кількість навчальних циклів за рахунок використання параметрів, що мають найбільшу вагу [30]–[34].

Отже, з метою вирішення проблем, що виникають при використанні досліджуваних методів, було прийнято рішення розробити модифікований генетичний метод для настройки вагових коефіцієнтів штучної нейронної мережі для підвищення точності прогнозування результатів процесу десульфуризації димових газів від діоксиду сірки.

1.7 Висновок

У цьому розділі було проведено аналіз предметної області, визначено основні поняття, пов'язані з предметною областю і об'єктом дослідження та розробки. Було розглянуто систему мокрої десульфурації димових газів та її принцип роботи.

Було досліджено основні методи прогнозування ефективності знесірчення димових газів та проведено порівняльний аналіз цих методів, який показав перевагу використання ГА та НМ для вирішення поставленого завдання.

Проведено порівняльну характеристику існуючих програмних засобів, що можуть бути використані для дослідження та створення прогнозування результатів процесу абсорбційної очистки димових газів від діоксиду сірки.

Було сформульовано і поставлено завдання до роботи.

2 ВИБІР ТА ОБҐРУНТУВАННЯ СТРУКТУРИ СИСТЕМИ, ЯКА ПРОЄКТУЄТЬСЯ

2.1 Вибір мови програмування

Існує декілька сотень мов програмування, кожна з мов має свої особливості, і та, яка з успіхом працює в одному проєкті, може зовсім не підходити для іншого, тому вибір мови програмування для розробки ПП є важливим етапом. Найчастіше використовувані мови програмування для штучного інтелекту та машинного навчання є R, C++, Python, Java, Julia [57].

2.1.1 R

R – це мова програмування для статистичної обробки даних та роботи з графікою, а також безкоштовне обчислювальне середовище з відкритим вихідним кодом у рамках проєкту GNU. Ця мова була створена як аналог S, розроблений у Bell Labs, і є її альтернативною реалізацією, хоча між мовами є суттєві відмінності, більша частина її коду мовою S виконується в середовищі R [57].

R була розроблена виключно для аналізу даних та розробки додатків та програмних рішень, які можуть виконувати статистичний та інтелектуальний аналіз даних. Популярна серед науковців та дослідників у галузі науки про дані мова R надає широкий спектр бібліотек та інструментів для наступного:

- очищення та підготовка даних;
- створення візуалізацій;
- навчання та оцінка алгоритмів машинного навчання та глибокого навчання [57].

Дослідники часто вважають за краще використовувати R, оскільки вона надає графіки, які можна відразу використовувати для публікації, враховуючи, що вона містить готові математичні формули та позначення. В

цілому, R також привертає увагу своєю візуалізацією даних, у тому числі графіками, діаграмами і т. д. Ці типи візуалізації полегшують інтерпретацію даних та ідентифікацію закономірностей, викидів (або аномалій) та тенденцій у наборах даних [57].

Переваги:

- мова з відкритим вихідним кодом;
- платформно-незалежна мова або кросплатформна мова програмування;
- дозволяє виконувати різні операції машинного навчання, такі як класифікація та регресія. З цією метою R надає різні пакети та функції для розробки штучної нейронної мережі;
- дозволяє користувачам або програмістам поєднувати мову R з іншими мовами програмування, такими як C, C++, Python та Java;
- надає безліч можливостей створення відмінних графіків для аналізу даних [57].

Недоліки:

- об'єкти зберігаються у фізичній пам'яті. R використовує більше пам'яті, ніж Python;
- дуже складна мова з крутою кривою навчання;
- відсутня базова безпека. Це невід'ємна частина більшості програмованих мов, таких як Python. Через це R має багато обмежень, так як вона не може бути вбудована у вебдодаток;
- не підтримує динамічну чи тривимірну графіку;
- набагато повільніше, ніж інші мови програмування, такі як MATLAB та Python. У порівнянні з іншими мовами програмування, пакети R працюють набагато повільніше.
- не так добре працює у тих галузях, де зазвичай використовуються мови загального призначення [57].

2.1.2 C++

C++ – це одна з найбільш адаптованих та гнучких мов програмування високого рівня, спочатку розроблена Б'ярном Страуструпом, який працював у Bell Labs. Страуструп створив C++ як розширення мови програмування C, шукаючи нову об'єктно-орієнтовану мову замість процедурно-орієнтованої. Це кросплатформна мова, яка використовується для створення всього, від веббраузерів до хмарних систем зберігання [58].

C++ – це скомпільована мова програмування, що означає, що користувач може писати код і організовувати дані зрозумілим для них способом. Після того, як код був написаний, платформа кодування або компілятор перекладає його мовою, якій комп'ютер може легко слідувати. Це робить C++ неймовірно швидкою мовою для роботи, що є однією з її ключових переваг [58].

Переваги:

- об'єктно-орієнтована мова програмування;
- може бути проблемно-орієнтованою мовою, яка набагато простіша у використанні, ніж інші низькорівневі мови, такі як двійкове кодування. Вона займає набагато більше місця, ніж мови низького рівня, але набагато легше зрозуміти і вивчити;
- використовує багатопарадигмальне програмування;
- забезпечує абстракцію високого рівня;
- допускає динамічний розподіл пам'яті, спрощуючи звільнення та виділення пам'яті, а також дозволяє звільняти блоки пам'яті, коли це необхідно;
- надає добрий набір вбудованих бібліотек;
- має величезну спільноту [58].

Недоліки:

- використання покажчиків. Покажчики як у C, так і в C++ зовсім не прості для розуміння та займають багато пам'яті. Коли відбувається

неправильне використання покажчиків, це може призвести до нестабільної поведінки чи збоїв. Пошкодження пам'яті також може статися, якщо вони неправильні;

- немає автоматичного керування пам'яттю;
- синтаксис дуже суворий і невелика помилка може призвести до серії помилок;
- не може підтримувати вбудовані потоки коду [58].

2.1.3 Python

Python – широко використовувана, інтерпретована, об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою, яка використовується для програмування загального призначення. Python є найпоширенішою мовою фахівців з обробки даних, що робить її невід'ємною частиною сучасних робочих навантажень у галузі науки про дані [59].

Більшість аналітиків даних використовують мову Python тому, що вона надає безліч бібліотек, таких як NumPy, SciPy, Matplotlib, pandas та Scikit-learn, щоб значно спростити завдання з обробки даних. Мова Python є також точною мовою, що дозволяє легко використовувати багатопроцесорну обробку великих наборів даних, скорочуючи час, необхідний для їхнього аналізу. Спільнота аналітиків даних розширила також свої можливості, використовуючи спеціалізовані IDE, такі як Anaconda, що реалізують концепцію Jupyter Notebook і значно спрощують роботу з обчисленнями в галузі науки про дані. Python – це відмінна мова для створення єднального коду з такими мовами, як C/C++ і Fortran [59].

Переваги Python:

- універсальна і гнучка, вона легко читається, має інтуїтивно зрозумілий синтаксис та форматування. Незважаючи на відносно спокійну криву навчання, Python залишається універсальною та

багатофункціональною. Вона став де-факто мовою програмування у багатьох галузях завдяки своїй універсальності та доступності;

- доступність бібліотек та фреймворків від статистики до науки про дані, операційні системи, вебінструменти та сервіси;

- Python інтегрується з корпоративними базами даних та GP через Enterprise Application Integration. Python можна викликати через Java, C++, C та Java через Jython. Вона легко інтегрується з PHP та .NET, тому може легко узгоджуватися з новими та старими протоколами, форматами та технологіями;

- Python є однією з найпродуктивніших мов програмування завдяки використанню виняткових фреймворків та бібліотек;

- незалежність від платформи. Python – це платформно-незалежна мова програмування, яка дозволяє розробникам розробляти та реалізовувати речі на одному пристрої та використовувати той самий код на будь-якій іншій платформі без зміни коду [59].

Недоліки Python:

- високе споживання пам'яті. Python споживає багато пам'яті і проблеми з пам'яттю можуть виникнути при великій кількості активних об'єктів в оперативно запам'ятовуючому пристрої;

- повільна швидкість виконання. Розробка на Python трохи повільніша порівняно з Java, C# або C/C++. Інтерпретований код – причина повільнішого коду;

- Python не підходить для мобільної розробки та розробки ігор через споживання пам'яті та швидкості [59].

2.1.4 Julia

Julia – це мова програмування, створена спеціально для обробки даних, складної лінійної алгебри, інтелектуального аналізу даних та машинного навчання. Творці цієї мови хотіли усунути недоліки Python та

інших мов програмування, запропонувавши більш зручний інструмент. Julia – це високорівнева, високопродуктивна мова динамічного програмування для технічних обчислень. Вона містить безліч математичних та статистичних бібліотек, які можна знайти у будь-якому високопродуктивному середовищі. Julia використовує JIT (Just in Time) – компілятор, який генерує машинний код. Компілятор JIT забезпечує стабільність за рахунок множинної диспетчеризації, що полегшує компіляцію коду. Julia підтримує паралельні, паралельні та розподілені обчислення, числову точність та велику бібліотеку математичних функцій. Julia використовує різні мови, від низькорівневих мов системного програмування, таких як C, до високорівневих мов динамічної типізації, таких як Python, R і MATLAB [60].

Переваги:

- підтримує метапрограмування;
- зручний для математики синтаксис;
- має чудовий паралелізм;
- автоматична генерація ефективного спеціалізованого коду для різних типів аргументів;
- автоматизоване керування пам'яттю. Julia не обтяжує користувача деталями виділення та звільнення пам'яті та забезпечує деяку міру ручного керування складанням сміття [60].

Недоліки:

- спільнота Julia невелика, це значно ускладнює пошук швидких відповідей на запитання та може змусити витратити більше часу на вирішення проблем порівняно з іншими мовами;
- у пакетах доволі часто зустрічаються помилки та є проблеми з надійністю;
- неможливо створити автономні двійкові файли або бібліотеки, такі як файли .exe або .dll [60].

2.1.5 Java

Java – це об'єктно-орієнтована мова програмування загального призначення, також вона використовується як обчислювальна платформа. Спочатку Java була розроблена для вбудованих мережесх програм, що працюють на декількох платформах. Java призначена для розробки портативних високопродуктивних програм для найширшого діапазону обчислювальних платформ, що дозволяє реалізувати фундаментальні принципи всеосяжної доступності, а також міжплатформної взаємодії. Один і той же додаток Java однаково працюватиме на будь-якому комп'ютері, незалежно від апаратних характеристик або операційної системи, якщо він має інтерпретатор Java. Крім переносимості, іншою ключовою перевагою Java є набір функцій безпеки, які захищають ПК, на якому виконується програма Java, не тільки від проблем, спричинених помилковим кодом, а й від шкідливих програм [61].

Переваги:

- розподілені обчислення включають спільну роботу декількох комп'ютерів в мережі;
- використовує об'єктно-орієнтовану парадигму, що робить її більш практичною;
- може працювати безпосередньо на декількох платформах, тобто не треба кожного разу компілювати код;
- у Java немає явного покажчика. Крім цього, вона має менеджер безпеки, який визначає доступ до класів;
- використовує багатопотокове середовище, в якому велика задача може бути перетворена в різні потоки і запущена окремо [61].

Недоліки:

- значно повільніше і споживає більше пам'яті, оскільки виконується поверх віртуальної машини Java;

– забезпечує автоматичне складання сміття, яким не може керувати програміст. Він не надає такі методи, як `delete()` та `free()` для звільнення пам'яті [61].

2.2 Порівняння мов програмування

Проаналізувавши роботи [57]–[61] було проведено порівняльний аналіз мов програмування, який наведена у табл. 2.1.

Таблиця 2.1 – Порівняльна характеристика мов програмування

Критерії порівняння	R	C++	Python	Java	Julia
Ліцензія	GNU GPL	–	Python Software Foundation License	GNU GPL	MIT
Простота навчання та написання коду	–	+	+	+	+
Кросплатформеність	+	+	+	+	+
Можливість обробляти операції з великим обсягом даних	+	+	+	+	+
Паралельне програмування	+	+	+	+	+
Наявність бібліотек для машинного навчання	+	±	+	±	+
Інтеграція з іншими мовами	+	+	+	+	±
Наявність зрозумілої та повної документації	+	+	+	+	+
Висока візуалізація даних	+	–	+	–	+
Підвищена продуктивність	±	+	+	+	+
Великий список предметно-орієнтованих пакетів з відкритим вихідним кодом	+	+	+	+	–
Сильна підтримка спільноти	+	+	+	+	–

Як видно з табл. 2.1 найбільші переваги має мова програмування Python, тому що вона легко адаптується і може ефективно працювати у будь-якому середовищі. Крім того, з незначними змінами вона може працювати в будь-якій операційній системі і може бути інтегрована з іншими мовами програмування. Python має багатий набір бібліотек та інструментів, що спрощує виконання завдань з аналізу даних, а також має широкий спектр

інструментів із відкритим вихідним кодом для візуалізації та машинного навчання. Крім того, у Python є величезна база спільноти, де інженери та фахівці за даними можуть запитувати свої запити та відповідати на запитання інших [59]. Отже, підбиваючи вищесказане для розробки ПП було обрано мову програмування Python.

2.3 Вибір середовища розробки

Середовище розробки дозволяє писати, тестувати та налагоджувати код більш простим способом, оскільки воно зазвичай надає автозавершення коду або розуміння коду шляхом виділення, керування ресурсами, інструментів налагодження. Середовище розробки заощаджує час при написанні додатків і дозволяє виправляти поширені помилки у коді, налагоджувати програми та розробляти великі проєкти. Найчастіше використовувані середовища розробки для створення ПП для аналіз даних та машинного навчання є PyCharm, Spyder, JupyterLab, Visual Studio Code [62].

2.3.1 PyCharm

PyCharm – повнофункціональне інтегроване середовище розробки (Integrated development environment – IDE), яке було створене спеціально для Python, воно доступне як у безкоштовній, так і в платній версіях, що отримали назву Community і Professional відповідно. PyCharm легко інтегрує такі інструменти, як Matplotlib та NumPy. Це означає, що можна легко працювати з програмами перегляду масивів та інтерактивними графіками, працюючи над проєктами в галузі науки про дані. Крім цього, IDE розширює підтримку JavaScript, Angular JS і т. д [62].

Переваги:

- активна підтримка спільноти;

- фактична розробка Python, як для проєктів, пов'язаних з наукою про дані, так і для проєктів, не пов'язаних з наукою про дані;
- швидша переіндексація;
- запускає, редагує та налагоджує код Python без будь-яких зовнішніх вимог [62].

Недоліки:

- може завантажуватись повільно;
- налаштування за замовчуванням може вимагати коригування, перш ніж можна буде використовувати існуючі проєкти [62].

2.3.2 Spyder

Spyder – це спеціалізована IDE з відкритим вихідним кодом для Python. Унікальність IDE полягає в тому, що воно оптимізовано для робочих процесів у галузі аналізу даних. Воно поставляється у комплекті з менеджером пакетів Anaconda, що є стандартним дистрибутивом мови програмування Python. Spyder має всі необхідні функції IDE, включаючи автозавершення коду та вбудований браузер документації. Створений спеціально для проєктів у галузі науки про дані, Spyder відрізняється плавною кривою навчання, це означає, що він досить простий для освоєння. Опція онлайн-довідки дозволяє шукати конкретну інформацію про бібліотеки при паралельній розробці проєкту [63].

Переваги:

- автозавершення коду та запам'ятовування змінних;
- легко використовувати;
- спеціально створений для проєктів у галузі науки про дані;
- акуратний інтерфейс;
- активна підтримка спільноти [63].

Недоліком Spyder є те, що не вистачає можливостей для проєктів, не пов'язаних із наукою про дані [63].

2.3.3 JupyterLab

JupyterLab створений на основі IPython у 2014 році, є вебдодатком, заснованим на структурі сервер-клієнт. Він дозволяє створювати та маніпулювати документами записників, званими записними книжками. JupyterLab пропонує одне з найбільш інтуїтивно зрозумілих та інтерактивних середовищ для аналізу даних. Крім роботи як IDE, JupyterLab також працює як інструмент навчання або презентації. Більше того, це хороший інструмент для тих, хто починає займатися наукою про дані. Використовуючи бібліотеки візуалізації, такі як Matplotlib та Seaborn, можна відображати графіки в тому ж документі, що й код. Крім того, можна експортувати всю свою роботу до PDF, HTML або .py файлу [64].

Переваги:

- дозволяє створювати блоги та презентації із записників;
- забезпечує відтворюваність досліджень;
- мінімалістичний інтерфейс;
- є хмарні альтернативи для спільної роботи [64].

Недоліки:

- нотатки JupyterLab не забезпечують зручного перегляду файлового менеджера при розробці коду;
- погано справляється із довгими асинхронними завданнями [64].

2.3.4 Visual Studio Code

Visual Studio Code – безкоштовна IDE з відкритим вихідним кодом, створена Microsoft для операційних систем Windows, Linux та macOS. Visual Studio Code забезпечує інтеграцію з локальними репозиторіями Git та віддаленими репозиторіями на GitHub та Azure Repos. Можна клонувати, фіксувати зміни та керувати гілками за допомогою цих інтеграцій. Visual Studio Code містить функцію Intelli-Sense, яка забезпечує доповнення коду на

основі типів змінних, функцій та імпортованих модулів. Воно також забезпечує підсвічування синтаксису та функцію автозаповнення. Visual Studio Code підтримує безліч мов програмування окрім Python, таких як C, C#, JavaScript, HTML, CSS і т.д. Це буде корисно під час експериментування з новим API або бібліотекою. Visual Studio Code має можливість виконувати модульні тести через фреймворки unittest або pytest [65].

Переваги:

- простий в управлінні весь життєвий цикл проекту в галузі науки про дані;
- широкі можливості налаштування;
- наявність Intelli-Sense, тобто можна визначити, чи залишився якийсь фрагмент коду незавершеним.
- легко створювати прототипи окремих компонентів, а потім об'єднувати їх у більші;
- кросплатформна підтримка, може працювати на всіх трьох платформах Windows, Linux та macOS [65].

Недоліки:

- урізана функціональність інтерфейсу користувача у порівнянні з Visual Studio;
- часто трапляється, що встановлення плагіна може призвести до збою Visual Studio Code;
- не має вбудованих функцій для перегляду файлу SVG [65].

2.4 Порівняння середовищ розробки

Проаналізувавши роботи [62]–[65] було проведено порівняльних аналіз середовищ розробки, який наведений у табл. 2.2.

Таблиця 2.2 – Порівняльна характеристика середовищ розробки

Критерії порівняння	PyCharm	Spyder	JupyterLab	Visual Studio Code
Ліцензія	Apache License 2.0	MIT	—	MIT
Кросплатформність	+	+	+	—
Автозавершення коду	+	+	+	+
Підтримка репозиторіїв	+	+	—	+
Простота встановлення	—	+	+	—
Перевірка та аналіз коду	+	+	+	+
Швидкий та безпечний рефакторинг коду	+	+	—	+
Наявність інтерактивної консолі	+	+	+	+
Наявність зручного користувацького інтерфейсу	+	+	—	+
Швидкий та зручний пошук	+	+	+	+
Підсвічування синтаксису	+	+	+	+
Підтримка наукових пакетів	+	+	+	+
Наявність функції статичного аналізу коду	+	+	+	+
Потребує малої кількості пам'яті для зберігання.	—	+	+	+

З табл. 2.2 можна побачити, що найбільші переваги має середовище розробки Spyder, тому що воно має такі переваги як простота в освоєнні, кросплатформне з відкритим вихідним кодом, яке було спеціально створене для аналізу даних. Spyder об'єднує основні бібліотеки для науки про дані, такі як NumPy, SciPy, Matplotlib та IPython, крім того, воно може бути розширене за допомогою плагінів. Spyder містить такі функції, як текстовий редактор з підсвічуванням синтаксису, автозавершення коду та вивчення змінних, значення яких можна редагувати за допомогою графічного інтерфейсу користувача [63]. Отже, підбиваючи вищесказане для розробки ПП було обрано середовище розробки Spyder.

2.5 Огляд бібліотек машинного навчання

Мова Python найбільш часто застосовується в науці про дані завдяки своєму інтуїтивно зрозумілому синтаксису, великій кількості ресурсів та

великому набору бібліотек для аналізу, візуалізації та машинного навчання. Бібліотеки та фреймворки Python допомагають скоротити часові витрати та підвищити ефективність розробки ПП, тому що вони надають набір готових рішень для завдань науки про дані, що охоплюють такі галузі, як обробка даних, моделювання та візуалізація. Для розробки ПП було обрано такі бібліотеки: NumPy, Scikit-learn, Pandas, Keras, Matplotlib [66].

2.5.1 Бібліотека NumPy

NumPy – одна з найважливіших бібліотек Python для наукових обчислень, яка активно використовується для додатків машинного навчання та глибокого навчання. Алгоритми машинного навчання складні в обчислювальному відношенні та вимагають операцій із багатовимірними масивами. NumPy забезпечує підтримку великих об'єктів багатовимірних масивів та різноманітних інструментів для роботи з ними. Завдяки цим масивам, а також функціям та операторам, які з ними працюють, NumPy частково вирішує проблему повільної роботи Python.

Переваги:

- висока ефективність обчислень, основаних на операціях з масивами;
- підтримка об'єктно-орієнтованого підходу;
- компактні та швидкі обчислення з використанням векторів.

Недоліком NumPy є те, що операції вставки та видалення вимагають великих витрат, оскільки дані зберігаються в суміжних осередках пам'яті та для їх переміщення потрібний зсув [66].

2.5.2 Бібліотека Scikit-learn

Scikit-learn – це модуль машинного навчання, побудований на основі NumPy та SciPy для використання з Python. Бібліотека Scikit-learn містить безліч ефективних інструментів для машинного навчання та статистичного

моделювання, включаючи класифікацію, регресію, кластеризацію та зменшення розмірності. Scikit-learn має сильну підтримку спільноти щодо документації, відстеження помилок та забезпечення якості. Спільнота Scikit-learn підтримує загальну документацію, єдиний процес відстеження/виправлення помилок на GitHub та сувору процедуру забезпечення якості, засновану на угоді в рамках усієї спільноти [67].

Основними перевагами Scikit-learn є безкоштовне використання, простота використання, універсальність, підтримка міжнародної спільноти та належна документація з API [67].

Основними недоліками Scikit-Learn є те, що вона не може використовувати категоріальні дані в алгоритмах та сильна залежність від стеку SciPy [67].

2.5.3 Pandas

Pandas – це бібліотека з відкритим вихідним кодом для машинного навчання. Pandas надає високорівневі структури даних та функції, призначені для швидкої, простої та виразної роботи зі структурованими або табличними даними. Основними об'єктами Pandas є DataFrame – таблична структура даних, орієнтована на стовпці, з мітками рядків та стовпців, і Series – одномірний об'єкт масиву з мітками. Pandas поєднує ідеї високопродуктивних обчислень масивів NumPy з гнучкими можливостями маніпулювання даними електронних таблиць і реляційних баз даних. Вона надає складні функції індексування, що дозволяють легко змінювати форму, виконувати агрегування та вибирати підмножини даних [68].

Переваги:

- багаті функціональні можливості, які дають свободу дій із відсутніми даними;
- дозволяє створити власну функцію і запустити її в серії даних;
- абстракція високого рівня;

– містить високорівневі структури даних та інструменти маніпулювання [68].

Недоліком є те, що Pandas менш зручна для кількісного моделювання та n -мірних масивів. Коли доводиться мати справу з кількісним або статистичним моделюванням, краще використовувати NumPy чи SciPy [68].

2.5.4 Keras

Keras – це компактна та проста в освоєнні бібліотека Python високого рівня для глибокого навчання, яка може працювати поверх TensorFlow (або Theano, або CNTK). Це дозволяє розробникам зосередитися на основних концепціях глибокого навчання, таких як створення шарів для нейронних мереж, приділяючи при цьому увагу найдрібнішим деталям. Оскільки вона написана на Python, вона має широкую спільноту користувачів і прихильників, і з неї дуже легко почати роботу. Простота Keras полягає в тому, що вона допомагає користувачам швидко розробляти моделі глибокого навчання та забезпечує більшу гнучкість, залишаючись при цьому високорівневим API. Більше того, враховуючи, що Keras підтримує декілька інших фреймворків як серверну частину, вона додає гнучкості для використання іншого низькорівневого API. Найбільш поширеним використанням Keras є TensorFlow як серверна частина (тобто Keras як високорівневий API глибокого навчання і TensorFlow як серверна частина низькорівневого API) [69].

Переваги:

- пропонує декілька моделей глибокого навчання із заздалегідь підготовленими вагами;
- пропонує легке представлення нейронних мереж;
- зручно використовувати для моделювання та візуалізації [69].

Недоліки:

- немає можливості створення динамічних діаграм;

- повільно навчає моделі глибокого навчання;
- має складну архітектуру.
- інструменти попередньої обробки даних Keras не так добре працюють, ніж інші пакети, такі як scikit-learn [69].

2.5.5 Matplotlib

Matplotlib – це бібліотека, яка використовується для візуалізації даних на Python. Вона вважається однією з невід'ємних бібліотек для візуалізації даних у машинному навчанні. Бібліотека Matplotlib працює зі структурами NumPy і високорівневими моделями, такими як Pandas. Matplotlib використовується для створення широкого спектру діаграм, таких як стовпчасті, лінійні, стовпці, кругові діаграми, діаграми розсіювання. Однією з найважливіших функцій Matplotlib є її здатність добре працювати з багатьма операційними системами та графічними процесорами. Вона також надає об'єктно-орієнтований API, який можна використовувати для вбудовування графіків у різні програми [70]

Переваги:

- надає можливість створення потужних і точних графіків;
- підтримка інструментів із графічним інтерфейсом, включаючи wxPython, Qt та Tkinter;
- кросплатформна бібліотека з відкритим вихідним кодом, яка добре працює з багатьма операційними системами та графічними серверами;
- низьке споживання пам'яті [71].

Недоліки:

- сильно залежність від NumPy та інших подібних бібліотек для стеку SciPy;
- легко заплутатися, тому що бібліотека надає дві окремі структури: об'єктно-орієнтовану та MATLAB;

– вона використовується тільки для візуалізації даних. Matplotlib для аналізу даних не підходить, тому її потрібно інтегрувати з іншими бібліотеками [71].

2.6 Структурна схема програмного продукту

Розроблюваний програмний продукт призначений для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Він являє собою набір пакетів, що містять скрипти за допомогою яких можна створювати власні моделі залежності ступеня десульфурації методу мокрої абсорбційної очистки димових газів від використовуваних реагентів. Програмний продукт представлений у вигляді окремого віконного додатку завдяки якому можна обчислити прогноз ефективності сіркоочистки від обраних параметрів процесу за допомогою методу найближчого сусіду, випадкового лісу та опорних векторів, а також можна здійснювати тренування моделі та прогноз на основі багатошарового перцептрону.

Для навчання на тестування моделей залежності ступеня сіркоочистки димових газів від використовуваних реагентів програмний продукт використовує експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів. Використовувані дані представляють собою файл у форматі csv, де кожний рядок файлу – це один проведений експеримент процесу десульфурації димових газів від діоксиду сірки, значення параметрів процесу розділені між собою комами.

У наборі даних є сім параметрів, останній з яких є цільовим:

- вміст SO_2 на вході (мг / м^3);
- first_stage – об'єм газу на першу сходинку (м^3 / год);
- second_stage – об'єм газу на другу сходинку (м^3 / год);
- water – витрати води у абсорбері (м^3 / год);
- lime – витрати вапна (т / год);
- second_reagen – витрати вторинного реагенту (т / год);

– вміст SO₂ на виході (мг /м³).

На рис. 2.1 представлена діаграма класів програмного продукту.

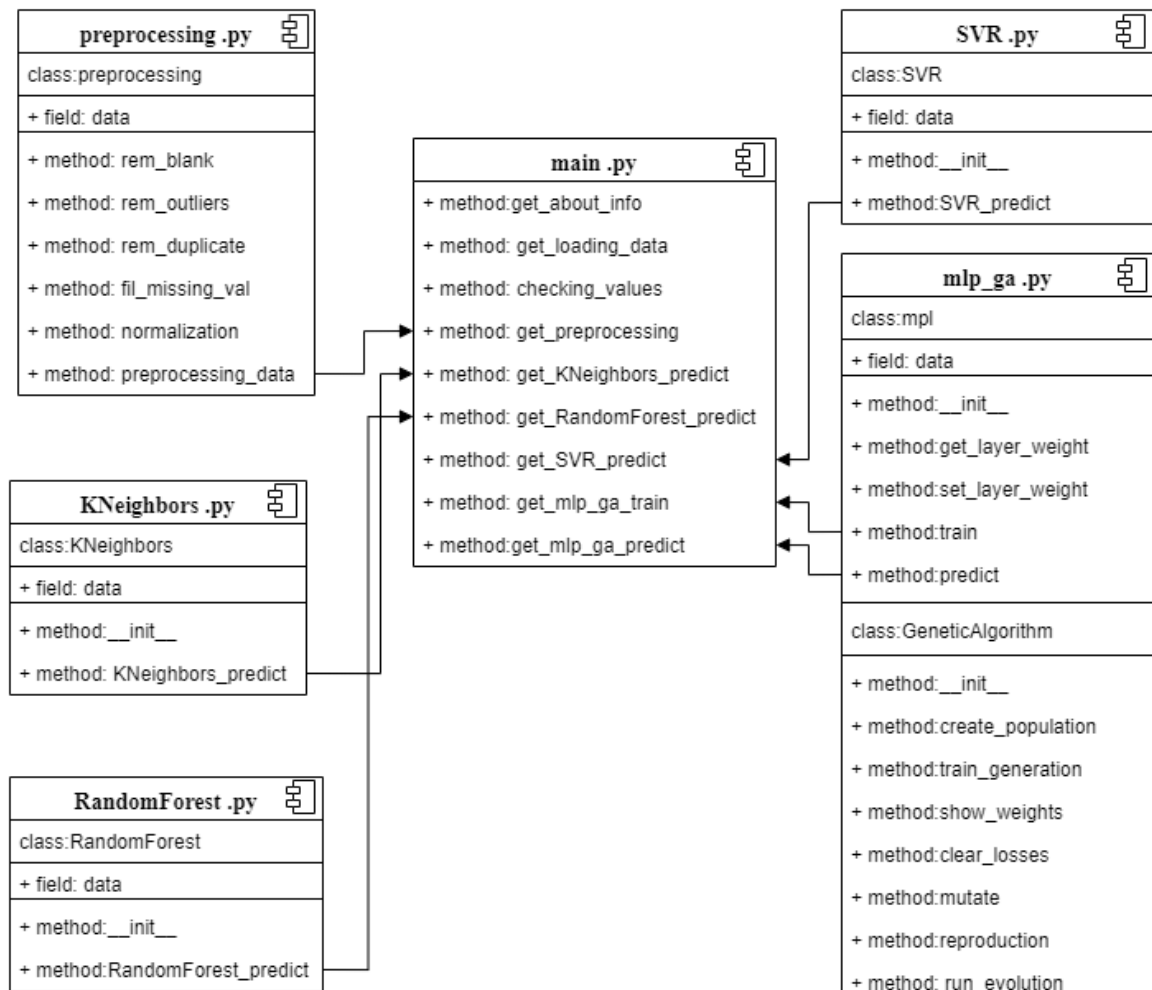


Рисунок 2.1 – Діаграма класів програмного продукту

Модуль **main.py** – головний модуль проєкту, в якому буде відбуватися ініціалізація програмного продукту та елементів інтерфейсу користувача, він має містити такі методи:

- **get_about_info** – призначений для виводу інформації про програмний продукт у окремому вікні;
- **checking values** – призначений для перевірки правильності значень введеним користувачем;
- **get_loading_data** – призначений для завантаження даних для навчання та тренування моделі;

- `get_preprocessing` – створює екземпляр класу `preprocessing` (модуля `preprocessing.py`) для здійснення попередньої обробки даних;
- `get_KNeighbors_predict` – створює екземпляр класу `KNeighbors` (модуля `KNeighbors.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу найближчого сусіда;
- `get_RandomForest_predict` – створює екземпляр класу `RandomForest` (модуля `RandomForest.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу випадковий ліс;
- `get_SVR_predict` – створює екземпляр класу `SVR` (модуля `SVR.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу опорних векторів;
- `get_mlp_ga_train` – створює екземпляр класу `mlp_ga` (модуля `mlp_ga.py`) для тренування моделі на основі багатoshарового перцептрону.
- `get_mlp_ga_predict` – створює екземпляр класу `mlp_ga` (модуля `mlp_ga.py`) для здійснення обчислення прогнозу з використанням моделі на основі багатoshарового перцептрону.

Модуль `preprocessing.py` – призначений для попередньої обробки даних, він виконує видалення порожніх рядків, усунення викидів, видалення рядків, що повторюються, заповнення пропущених значень та нормалізація даних.

Модуль `KNeighbors.py` – призначений для реалізації прогнозування за допомогою методу найближчого сусіду;

Модуль `RandomForest.py` – призначений для реалізації прогнозування за допомогою методу випадковий ліс.

Модуль `SVR.py` – призначений для реалізації прогнозування за допомогою методу опорних векторів.

Модуль `mlp_ga.py` – містить два класи `mpl` та `GeneticAlgorithm`, він призначений для реалізації прогнозування за допомогою моделі на основі багатoshарового перцептрону з використанням генетичного алгоритму для настройки вагових коефіцієнтів.

2.7 Висновок

У цьому розділі було проведено аналіз та вибір інструментарію розробки ПП та було спроектовано архітектуру майбутнього ПП. Було проведено порівняльний аналіз мов програмування для розробки ПП та встановлено, що найбільші переваги має мова програмування Python, тому що вона легко адаптується, має багатий набір бібліотек та інструментів, що спрощує виконання завдань з аналізу даних, а також має широкий спектр інструментів із відкритим вихідним кодом для візуалізації та машинного навчання.

Було проведено порівняльний аналіз середовищ розробки, в ході чого було встановлено, що найбільші переваги має середовище розробки Spyder, тому що воно має такі переваги як простота в освоєнні, кросплатформне з відкритим вихідним кодом, має великий набір бібліотеки для науки про дані.

Було проведено огляд бібліотек для розробки ПП. Для розробки нейронної мережі було обрано бібліотеку Keras. Для зручної роботи з масивами даних і формування датасетів використано бібліотеку NumPy та Pandas. Для реалізації алгоритмів машинного навчання було обрано бібліотеку Scikit-learn. Для побудови графіків та візуалізації даних було обрано бібліотеку Matplotlib. Було описано структурну схему розроблюваного продукту.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

Завданням роботи є програмна реалізація методів прогнозування результатів процесу десульфурзації димових газів від діоксиду сірки з використанням еволюційних генетичних алгоритмів та штучних нейронних мереж.

У роботі розглядається проблема складності вибору ефективної методу навчання та правильного розподілу ваги матриці на початковому етапі побудови нейронної мережі для прогнозування результатів процесу знесірчення димових газів.

Для досягнення поставленої мети необхідно розробити модифікований генетичний метод для настройки вагових коефіцієнтів штучної нейронної мережі для підвищення точності прогнозування результатів процесу десульфурзації димових газів від діоксиду сірки.

3.1 Попередня обробка даних

Для побудови та тестування моделей прогнозування ефективності процесу знесірчення димових газів використано експериментальні дані роботи абсорбційної вежі мокрої десульфурзації димових газів з 07:00 24 травня 2020 р. по 06:00 31 грудня 2020 р. Набір складається з 7 атрибутів та 5330 екземплярів:

- вміст SO_2 на вході ($\text{мг} / \text{м}^3$);
- об'єм газу на першу сходинку ($\text{м}^3 / \text{год}$);
- об'єм газу на другу сходинку ($\text{м}^3 / \text{год}$);
- витрата води в адсорбері ($\text{м}^3 / \text{год}$);
- витрата вапна ($\text{т} / \text{год}$);
- витрата вторинного реагенту ($\text{т} / \text{год}$);
- вміст SO_2 на виході ($\text{мг} / \text{м}^3$).

Продуктивність і якість знань, отриманих методом інтелектуального аналізу даних у будь-якій структурі, залежить не тільки від конструкції та продуктивності методу, але також дуже залежить від якості та придатності таких даних. На жаль, такі негативні чинники, як шум, недостатні значення, непослідовні та надлишкові дані, а також величезні розміри прикладів та функцій, сильно впливають на дані, що використовуються для вивчення та отримання знань. Добре відомо, що низька якість даних веде до низької якості знань. Таким чином, попередня обробка даних є основним і важливим етапом, основною метою якого є отримання остаточних наборів даних, які можна вважати правильними та корисними для подальших алгоритмів інтелектуального аналізу даних [72]–[73].

Попередня обробка даних у роботі складається з таких кроків:

- виявлення та обробка відсутніх значень;
- видалення рядків, що повторюються;
- видалення викидів та шумів;
- усунення невідповідностей;
- нормалізація [72]–[73].

Під час нормалізації дані зміщуються та масштабуються у в діапазоні від 0 до 1 значення змінної x розраховується за формулою (3.1):

$$x_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}, \quad (3.1)$$

де $x_{\min}$ – мінімальне значення ознаки;

$x_{\max}$ – максимальне значення ознаки [74]–[75].

3.2 Основні відомі методи

Спочатку були створено моделі прогнозування ефективності десульфурзації димових газів від діоксиду сірки на основі таких методів, як

метод найближчого сусіда, випадковий ліс та метод опорних векторів. Вхідними параметрами моделей є вміст SO₂ на вході, об'єм газу на першу сходинку, об'єм газу на другу сходинку, витрата води в адсорбері, витрата вапна, витрата вторинного реагенту. Вихідним параметрам моделей є вміст SO₂ на виході.

Для оцінки якості прогнозних моделей було використано коефіцієнт детермінації R², середньоквадратична помилка (Mean Square Error (MSE)) середня абсолютну помилку (Mean Absolute Error (MAE)).

Коефіцієнт детермінації R² – показник, що відображає міру якості регресійної моделі, яка описує зв'язок між залежною та незалежними змінними моделями. R² розраховується за формулою (3.2) [76]:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - y_i^*)^2}{\sum_{i=1}^n (y_i - y_{mean})^2}, \quad (3.2)$$

де y_i – фактичне значення i -го екземпляра;

y_i^* – прогнозне значення i -го екземпляра;

n – кількість спостережень;

y_{mean} – середнє фактичних значень, яке розраховується за формулою (3.3) [76]:

$$y_{mean} = \frac{1}{n} \sum_{i=1}^n y_i. \quad (3.3)$$

Значення коефіцієнта детермінації визначає частку змін, зумовлених впливом факторних ознак, у загальній мінливості результативної ознаки. Значення R² має бути в діапазоні від нуля до одиниці: $0 \leq R^2 \leq 1$. Модель вважається більш якісною, якщо значення коефіцієнта детермінації близько

до 1. Якщо $R^2 = 1$, то емпіричні точки лежать точно на лінії регресії і між залежними і незалежними змінними існує лінійна функціональна залежність. Якщо $R^2 = 0$, то варіація залежної змінної повністю обумовлена неврахованими в моделі чинниками [76].

Середньоквадратична помилка (MSE) відповідає середньому квадрату різниці між фактичним та прогнозованим значенням [77]. MSE розраховується за формулою (3.4):

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - y_i^*), \quad (3.4)$$

де y_i – фактичне значення i -го екземпляра;

y_i^* – прогнозне значення i -го екземпляра;

n – кількість спостережень [76]–[77].

Середня абсолютна помилка (MAE) – це показник оцінки моделі, що використовується з моделями регресії. Середня абсолютна помилка моделі по відношенню до набору тестів - це середнє абсолютних значень окремих помилок прогнозування по всіх екземплярах у наборі тестів. Кожна помилка прогнозування – це різниця між істинним значенням та прогнозованим значенням для екземпляра [17]. MAE розраховується за формулою (3.5):

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - y_i^*|, \quad (3.5)$$

де y_i – фактичне значення i -го екземпляра;

y_i^* – прогнозне значення i -го екземпляра;

n – кількість спостережень [76].

Невелике значення для критеріїв MSE та MAE означає, що оцінювана модель здатна передбачити значення досить точно, тобто найкращим вважається той прогноз, у якого MSE та MAE менше [17].

Результати випробування розроблених моделей прогнозування з використанням класичних методів (метод найближчого сусіду, випадковий ліс, метод опорних векторів) за вищезазначеними метриками представлені у табл. 3.1.

Таблиця 3.1 – Результати випробування розроблених моделей прогнозування з використанням класичних методів (метод найближчого сусіду, випадковий ліс, метод опорних векторів)

Назва методу	R^2	MSE	MAE
Метод найближчого сусіду	0,5453	0,0153	0,0897
Випадковий ліс	0,5482	0,0152	0,0954
Метод опорних векторів	0,6125	0,0131	0,0842

З табл. 3.1 можна побачити, що найкращі результати за метрикою R^2 були досягненні у моделі на основі методу опорних векторів, R^2 цієї моделі склала 0,6125, а найгірші результати були отримані у моделі з використанням методу найближчого сусіда, R^2 цієї моделі дорівнює 0,5453. Модель на основі методу опорних векторів показує більш успішні результати в порівнянні з методом найближчого сусіда та випадкового лісу, а саме за показником MAE (при використанні методу опорних векторів показник MAE дорівнює 0,0842 на відміну від метод найближчого сусіда та випадкового лісу, у яких ці показники дорівнюють 0,0897 та 0,0954 відповідно).

Для прогнозування результатів процесу десульфурзації димових газів від діоксиду сірки було використано архітектуру багатошарового перцептронну, оскільки він дає можливість будувати складніші структури, добре працює з великим набором даних, має адаптивний процес навчання, створює необхідну функцію прийняття рішень безпосередньо у процесі навчання із заданим набором даних [78]–[80].

Спочатку було створено модель нейронної мережу з одним прихованим шаром для прогнозування ефективності процесу знесірчення димових газів.

Визначення кількості нейронів у прихованих шарах – дуже важлива частина ухвалення рішення про загальну архітектуру нейронної мережі. Хоча ці шари не взаємодіють безпосередньо із зовнішнім середовищем, вони мають величезний вплив на кінцевий результат. Необхідно враховувати як кількість прихованих шарів, так і кількість нейронів у кожному з цих прихованих шарів. Використання занадто великої кількості нейронів у прихованих шарах може призвести до перенавчання або збільшення часу, необхідного для навчання мережі. Отже, має бути досягнутий деякий компроміс між занадто великою і занадто невеликою кількістю нейронів у прихованих шарах [81]. Вибір кількості нейронів у прихованих шарах персептрону визначався за допомогою запропонованих підходів Хехт-Нільсена, Лоуренса та Фредріксона. Хехт-Нельсон для визначення верхньої межі необхідної кількості прихованих нейронів використовував теорему Колмогорова, згідно з якою будь-яка функція від i змінних може бути представлена суперпозицією набору $2i + 1$ одновимірних функцій [82]–[83]. Отже, верхня межа hup кількості нейронів у одному прихованому шарі розраховувати за формулою (3.6):

$$hup = 2i + 1, \quad (3.6)$$

де i – кількість вхідних (параметрів) нейронів [82].

Лоуренс і Фредріксон припустили, що нижня межа low кількості нейронів у одному прихованому шарі – це половина суми вхідних та вихідних нейронів (3.7):

$$low = \frac{(i + o)}{2}, \quad (3.7)$$

де o – кількість вихідних нейронів [83].

Функцію активації вихідного шару було обрано функцію ReLU. Ця функція дозволяє обчислити вихідні значення за вхідними, при цьому значення на виході лежать у діапазоні від $[0; +\infty)$. ReLU виконується за формулою (3.8) [84]:

$$f(x) = \max(0, x). \quad (3.8)$$

Для оптимізації навчання нейронних мереж було обрано алгоритм Adam, це алгоритм оптимізації, який є подальшим розширенням стохастичного градієнтного спуску (SGD) для оновлення ваги мережі під час навчання. На відміну від підтримки єдиної швидкості навчання через навчання в SGD, оптимізатор Adam оновлює швидкість навчання для кожної ваги мережі індивідуально. Алгоритм Adam використовує переваги алгоритмів AdaGrad та RMSProp, які також є розширеннями алгоритмів стохастичного градієнтного спуску. Отже, оптимізатори Adam успадковують особливості алгоритмів Adagrad та RMSProp. Правило оновлення ваг для алгоритму Adam визначається на основі використання оцінок двох різних моментів за формулами (3.9) і (3.10), у першому з яких використовуються обчислені раніше значення часткових похідних (як у методі стохастичного градієнтного спуску), а у другому їх квадрати (як у RMSProp) [85]–[86].

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \quad (3.9)$$

$$v_t = \beta_2 m_{t-1} + (1 - \beta_2) g_t^2, \quad (3.10)$$

де m_t – експоненціальне рухоме середнє градієнта;

v_t – експоненціальне середнє квадрата градієнта;

β_1, β_2 – гіперпараметри, які визначають період усереднення;

g_t – градієнт поточного міні-пакету [85]–[86].

Як правило використовують $\beta_1 = 0,9$ та $\beta_2 = 0,999$, при цьому m_0 та v_0 приймають нульові значення. З такими параметрами формули (3.8) та (3.9), на початку навчання повертають значення близькі до нуля і, як наслідок, отримується низька швидкість навчання на початковому етапі [87]. Для прискорення навчання отримані моменти обраховуються за формулами (3.11) та (3.12):

$$\overline{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad (3.11)$$

$$\overline{v}_t = \frac{v_t}{1 - \beta_2^t}. \quad (3.12)$$

Перерахунок ваг проводиться за формулою (3.13):

$$w_{t+1} = w_t - \alpha \frac{\overline{m}_t}{\sqrt{\overline{v}_t} + \varepsilon}, \quad (3.13)$$

де ε – невелика константа, введена для чисельної стабільності (10^{-8});

α – коефіцієнт навчання, який виступає верхньою межею кроку навчання [87].

Було створено модель багатошарового персептрона з одним прихованим шаром, який складається з зовнішнього шару (6 нейронів), прихованого повнозв'язаного шару (13 нейронів) та вихідного (один нейрон). У якості функції активації використано ReLU. Для первинної ініціалізації ваг було обрано нормальний розподіл (середнє значення дорівнює 0 та дисперсія дорівнює 0.1). Для оптимізації навчання моделі було обрано алгоритм Adam. У якості функції помилки в алгоритмі adam використано функцію MSE. Для оцінювання моделі під час навчання та тестування було обрано метрики MSE та MAE.

Створена мережа проходила навчання в продовж 200 епох. Для навчання та оцінки якості створеної моделі було поділено вхідну вибірку даних на дві підвибірки у такій пропорції: навчальна вибірка – 75%, тестова вибірка – 25% спостережень.

На рис. 3.1 та на рис. 3.2 представлені результати роботи багатошарового персептрона с одним прихованим шаром.

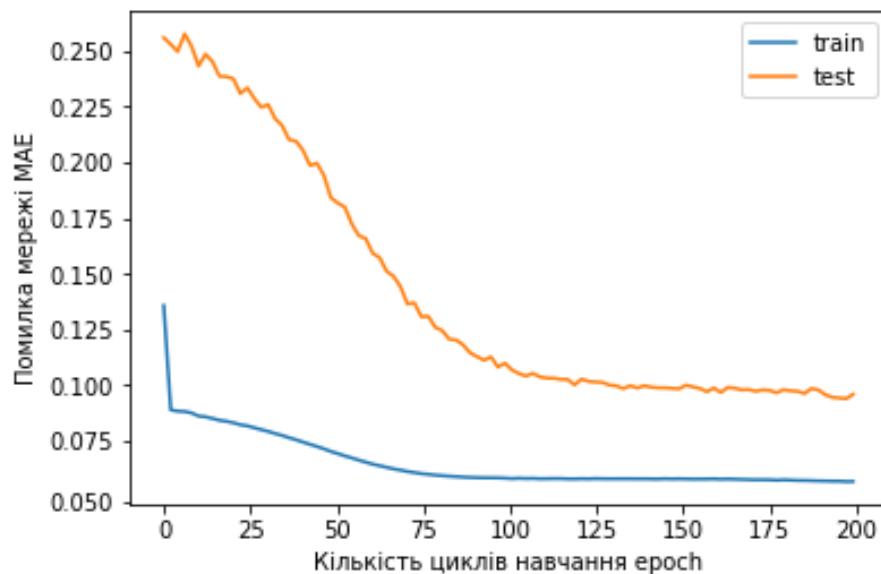


Рисунок 3.1 – Значення метрики MAE для багатошарового персептрона з одним прихованим шаром

На рис. 3.1 можна побачити, що впродовж навчання та тестування моделі спостерігається поступове зниження помилки MAE, а починаючи с 75 епохи значення помилки MAE на навчальній вибірці перестає змінюватися та дорівнює 0,0568, отже, розроблений багатошаровий персептрон с одним прихованим шаром стає тренованим і подальше навчання припиняється.

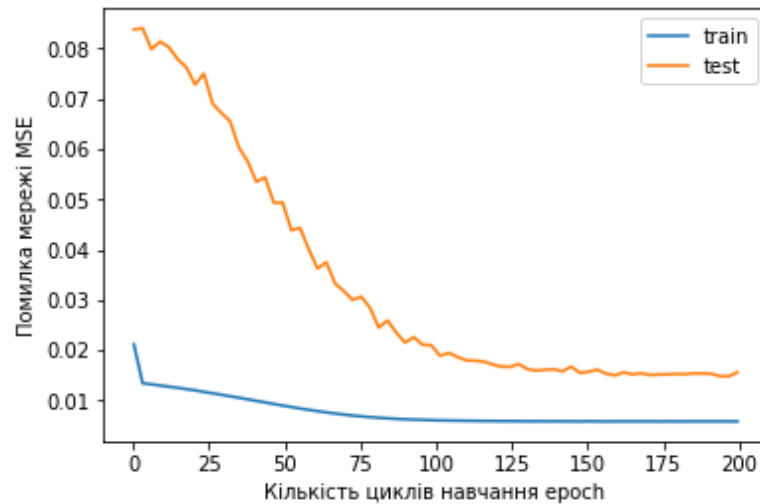


Рисунок 3.2 – Значення метрики MSE для багатошарового персептрона з одним прихованим шаром

На рис. 3.2 можна побачити поступове падіння помилки MSE під час навчання та тестування моделі багатошарового персептрона з одним прихованим шаром, а на 80 епосі значення помилки MSE на тестовій вибірці практично стає сталим значенням та дорівнює 0,01476.

У побудовано раніше модель багатошарового персептрона було додано ще один повнозв'язний прихований шар з 12 нейронів. Друга модель складається з зовнішнього шару (6 нейронів), прихованого повнозв'язаного шару (13 нейронів), прихованого повнозв'язаного шару (24 нейронів) та вихідного (один нейрон). Навчання мережі з двома прихованими шарами проводилося в продовж 100 епох.

На рис. 3.3 та на рис. 3.4 представлені результати роботи багатошарового персептрона з двома прихованими шарами.

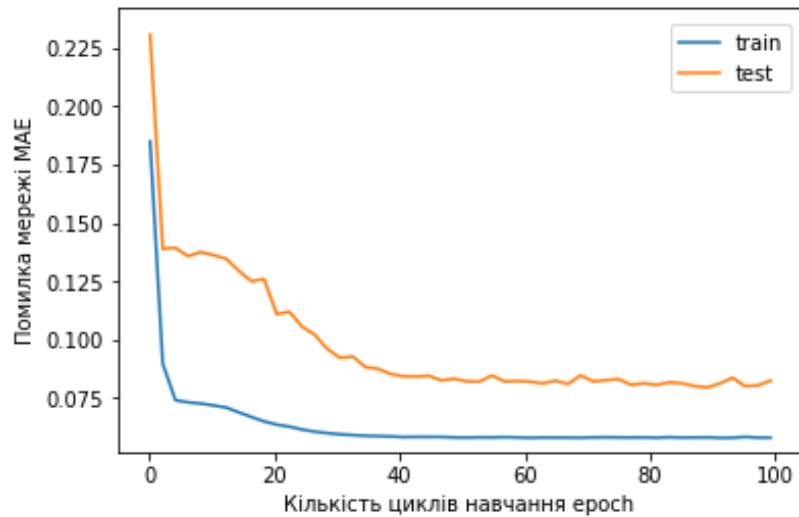


Рисунок 3.3 – Значення метрики MAE для багатошарового персептрона з двома прихованими шарами

На рис. 3.3 можна побачити, що впродовж навчання та тестування моделі спостерігається поступове зниження помилки MAE, а починаючи з 40 епохи значення помилки MAE на навчальній та тестовій вибірці перестає змінюватися та дорівнює 0,0605, отже, розроблений багатошаровий персептрон з двома прихованими шарами стає тренуваним і подальше навчання припиняється.

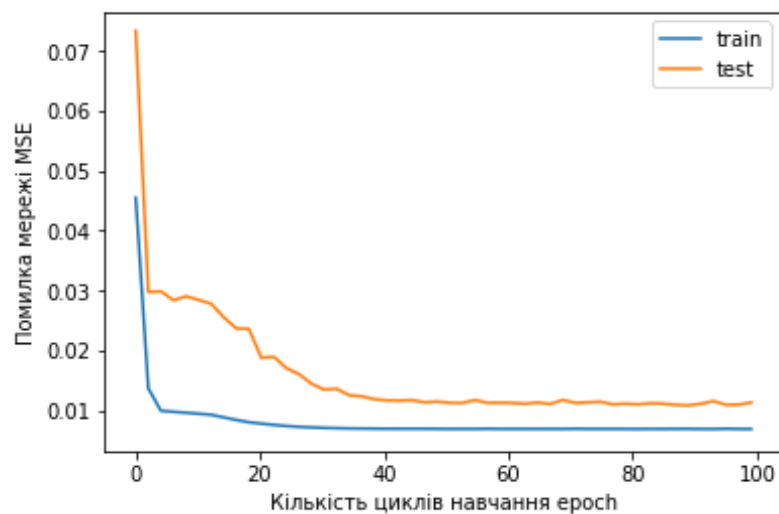


Рисунок 3.4 – Значення метрики MSE для багатошарового персептрона з двома прихованими шаром

На рис. 3.4 можна побачити поступове падіння помилки MSE під час навчання та тестування моделі багатошарового персептрона с двома прихованими шарами, а на 40 епосі значення помилки MSE на тестовій вибірці практично стає сталим значенням та дорівнює 0,0129.

3.2.1 Розробка модифікованого генетичного методу для настройки вагових коефіцієнтів нейронної мережі

Спочатку у запропонованому методі відбувається ініціалізація початкової популяції хромосомами, що містять інформацію про значення вагових коефіцієнтів мережі заданої структури. Хромосома складається з G генів, які містять значення ваг і зміщень всіх нейронів створеної мережі. У розроблюваному методі використовується дійсне кодування для представлення значень вагових коефіцієнтів у хромосомах. Найкращим способом кодування ваг при даному поданні топології є кодування за допомогою матриці, розмір якої еквівалентний розміру матриці топології НМ [88]–[90]. Елементи цієї матриці – вагові коефіцієнти відповідного зв'язку. Довжина хромосоми обчислюється за формулою (3.14):

$$\lambda = Q_1(T + 1) + \sum_{l=1}^L Q_l(Q_{l-1} + 1), \quad (3.14)$$

де Q_l – кількість нейронів на l -ому шарі;

T – кількість ознак у навчаючій вибірці;

L – кількість шарів нейромережі [88]–[90].

Після формування проводиться оцінювання хромосом поточної популяції, для цього спочатку здійснюється декодування кожної хромосоми популяції в набір вагових коефіцієнтів нейронної мережі. Далі виконується обчислення значення фітнес-функції, яка оцінює якість обраної архітектури

за значенням помилки навчання нейронної мережі за формулою (3.15) та (3.16):

$$F_{opt} = \min(F), \quad (3.15)$$

$$F = \frac{1}{2C_t} \sum_{i=1}^{C_t} (O_i - R_i)^2, \quad (3.16)$$

де C_t – потужність множини навчальних пар;

O_i – отримане за допомогою мережі значення вихідного нейрона при i -му наборі навчання;

R_i – необхідне значення вихідного нейрона при i -му наборі навчання [91]–[92].

Як бачимо, фітнес-функція буде оцінювати помилку, як різницю між отриманим виходом мережі і необхідним, яка визначає, наскільки отриманий вихід мережі відрізняється від потрібного. Отже, чим менше ця помилка тим вище значення придатності, тобто необхідно привести матрицю до такого виду, при якому помилка навчання нейронної мережі буде мінімальною [92].

Виходячи з отриманих значень фітнес-функції відбувається вибір особин для генерації нових рішень за допомогою рангової селекції. Для цього поточну популяцію було відсортовано відповідно до значень фітнес-функції, кожній хромосомі присвоюється число, що визначає її місце в ранзі відсортованої популяції, а не абсолютне її значення. Даний підхід дозволяє контролювати селективний тиск і обмежити кількість нащадків однієї хромосоми [93]–[95].

Лінійне ранжування всіх хромосом виконується за допомогою формулою (3.17):

$$P_{sl}(Pos) = 2 - SP + 2 \cdot (SP - 1) \cdot \frac{Pos - 1}{C - 1}, \quad (3.17)$$

де Pos – позиція хромосоми в популяції ($Pos = 1$ має хромосома з найменшим значенням фітнес-функції $F_{\min}$, а $Pos = C$ має хромосома з найбільшим значенням фітнес-функції $F_{\max}$);

C – кількість хромосом (особин) в популяції;

SP – коефіцієнт тиску селекції, який розраховується за формулою (3.18):

$$SP = \frac{F_{\max}}{F_{\text{avg}}}, \quad (3.18)$$

де $F_{\max}$ – хромосома з найбільшим значенням фітнес-функції;

F_{avg} – середнє значення фітнес-функції всієї популяції [93]–[95].

Перевага рангового методу полягає в можливості його застосування як для максимізації, так і для мінімізації функції. Він також не вимагає масштабування через проблеми передчасної збіжності, що є актуальною для методу рулетки [93]–[95].

Оператор схрещування в запропонованому генетичному методі, породжує двох нащадків від двох батьківських хромосом, тобто з двох векторів дійсних чисел здійснюється формування двох нових векторів. В якості основи використовується SBX – кросовер, який імітує роботу двійкового оператора схрещування [88]. Нехай $P^1 = (p_1^1, p_1^2, \dots, p_1^\lambda)$ та $P^2 = (p_1^2, p_2^2, \dots, p_\lambda^2)$ хромосоми двох предків, тоді гени хромосом-нащадків знаходяться за формулами (3.19) та (3.20):

$$c_j^1 = \frac{1}{2} \left((1 - \omega) \cdot p_j^1 + (1 + \omega) p_j^2 \right), \quad (3.19)$$

$$c_j^2 = \frac{1}{2} \left((1 + \omega) \cdot p_j^1 + (1 - \omega) p_j^2 \right), \quad (3.20)$$

де $j=1, 2, \dots, \lambda$;

λ – довжина хромосоми;

p_j^1, p_j^2 – гени першого і другого батьків відповідно;

ω – число, отримане за формулою (3.21):

$$\omega = \begin{cases} (2v)^{\frac{1}{b+1}}, v \leq 0,5 \\ \left(\frac{1}{2(1-v)}\right)^{\frac{1}{b+1}}, v > 0,5 \end{cases}, \quad (3.21)$$

де v – випадкове число, розподілене по рівномірному закону $v \in (0,1)$;

b – параметр, що впливає на ймовірність появи нащадків вдалині від батьківських особин $b \in [2,5]$. Під час дослідження було встановлено, що невеликі значення b дозволяють генерувати нащадків значно віддалених від батьків, тоді як збільшення b спричиняє появу нащадків поблизу батьківських пар [96]–[97].

У еволюційних алгоритмах оператор мутації відповідає за широкий пошук і захоплення нових областей пошукового простору. Для його налаштування користувачеві додатково доводиться приймати рішення про те, яке значення ймовірності мутації слід використовувати. При низькій мутації алгоритм буде рідко захоплювати нові області пошукового простору і швидше сходитися до локальних екстремумів. При високій мутації алгоритм буде частіше обстежувати різні області пошукового простору, але при цьому не буде локалізувати перспективні області простору пошуку. Таким чином, вибір оператора мутації безпосередньо впливає на якість роботи методу і одержуваних ним рішень [97]–[98].

Для вирішення цієї проблеми було запропоновано реалізувати механізм зміни ймовірності мутації. Спочатку на кожній ітерації знаходиться рівень розвитку популяції, тобто середнє значення фітнес-функції всіх особин, після чого отримане значення порівнюють із значенням функції придатності кожної особини. Якщо значення фітнес-функції індивіда менше середнього значення фітнес-функції всіх особин, то здійснюється підвищення ймовірності мутації, тобто чим гірше значення фітнес-функції, тим вище

ймовірність мутації. Ймовірність мутації хромосоми розраховується за формулою (3.22):

$$\beta_{mut} = \begin{cases} 0.5 \frac{F_{\max}(e) - F(P(e))}{F_{\max}(e) - \bar{F}(e)}, & \text{if } F(P(e)) \geq \bar{F}(e) \\ 0.5, & \text{if } F(P(e)) < \bar{F}(e) \end{cases}, \quad (3.22)$$

де $F_{\max}(e)$ – найбільше значення фітнес-функції поточної популяції;

$\bar{F}(e)$ – середнє значення фітнес-функції поточної популяції [98]–[100].

Якщо ген p_j піддається мутації, то його нове змінене значення p_j^{new} визначається за формулою (3.23):

$$p_j^{new} = p_j + b(o_j - p_j) \left(1 - \frac{e}{e_{\max}}\right)^{\xi}, \quad (3.23)$$

де b – випадкове згенероване число з інтервалу $[0,1]$;

o_j – випадковим чином генерується з множини $\{p_{\min}, p_{\max}\}$, де $p_{\min}$ і $p_{\max}$ – нижня і верхня межа можливої зміни значення змінної p_j ;

e – номер поточного покоління;

$e_{\max}$ – максимальне число поколінь;

ξ – параметр уточнення, який залежить від характеру збіжності ітераційного процесу, після досягнення стаціонарного стану, коли кращий з індивідуумів не змінився протягом останніх e поколінь, величина зменшується вдвічі, що призводить до розширення області пошуку і тим самим до подолання пасток локальних екстремумів [99]–[100].

Така адаптивна мутація дозволяє дотримуватися в процесі реалізації ГА (еволюції) необхідного балансу між двома різномасштабними змінами (мутаціями) генів, оскільки на первинних кроках методу в основному переважали великомасштабні зміни (що забезпечують широку область

пошуку), в той час як на заключному етапі (за рахунок зменшення масштабу мутацій) відбувалося уточнення рішення [98]–[101].

Застосування адаптивної мутації для кожного індивідууму необхідно, але не достатньо для запобігання збіжності популяції в локальному оптимумі. Допустимо i -та особина має ймовірністю мутації, яка становить 97%. З одного боку, настільки висока мутація при породженні від i -ої особини популяції рівносильна породженню хромосоми з повністю випадковими параметрами, які не несуть в собі корисної інформації про попередню еволюцію. З іншого боку, i -та особина залишається в популяції через високе значення функції пристосованості, витісняючи наступні покоління з меншими значеннями. Для того, щоб повністю уникнути збіжності, введемо селекцію яка буде проріджувати популяцію і відсівати особин з високою ймовірністю мутації за формулою (3.24):

$$\text{if } \beta_{mut}(P^i) > \delta : \text{removed } P^i, \quad (3.24)$$

де β_{mut} – ймовірність мутації P^i хромосоми;

δ – поріг селекції яка, що визначає ступінь мутацій та відсіює особини «несумісних з життям» [99]–[102].

Розроблюваний метод зупиняє свою роботу при досягненні максимального числа епох функціонування, яке задається користувачем.

Було створено модель багатошарового персептрона с двома прихованими шарами, яка складається з зовнішнього шару (6 нейронів), прихованого повнозв'язаного шару (13 нейронів), прихованого повнозв'язаного шару (27 нейронів) та вихідного (один нейрон). У якості функції активації використано ReLU. Для первинної ініціалізації ваг було обрано нормальний розподіл (середнє значення дорівнює 0 та дисперсія дорівнює 0,1). Для оптимізації навчання моделі було використано створений модифікований генетичний метод з адаптивною мутацією. Для оцінювання моделі під час навчання та тестування було обрано метрики MSE та MAE.

Створена мережа проходила навчання в продовж 200 епох. Для навчання та оцінки якості створеної моделі було поділено вхідну вибірку даних на дві підвибірки у такій пропорції: навчальна вибірка – 75%, тестова вибірка – 25% спостережень.

На рис. 3.5 та на рис. 3.6 представлені результати роботи багатошарового перцептрона с з двома прихованим шарами та генетичним методом з адаптивною мутацією.

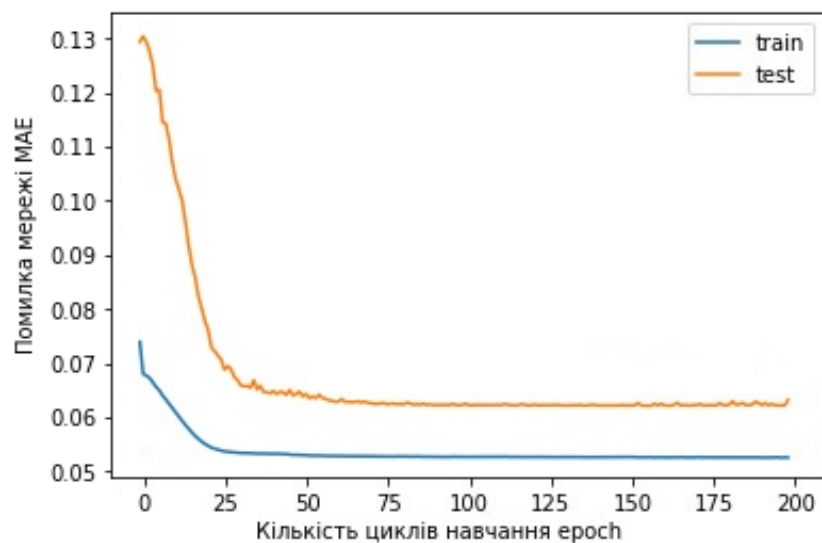


Рисунок 3.5 – Значення метрики MAE для багатошарового перцептрона з двома прихованим шарами та генетичним методом з адаптивною мутацією

На рис. 3.5 можна побачити, що впродовж навчання та тестування моделі спостерігається поступове зниження помилки MAE, а починаючи с 50 епохи значення помилки MAE на навчальній вибірці перестає змінюватися та дорівнює 0,0543.

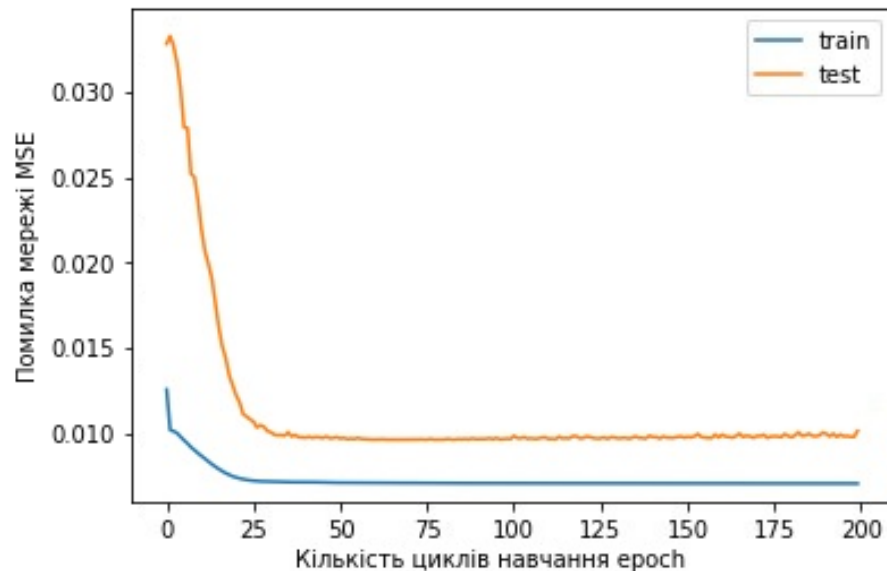


Рисунок 3.6 – Значення метрики MSE для багатошарового персептрона з двома прихованими шарами та генетичним методом з адаптивною мутацією

На рис. 3.6 можна побачити поступове падіння помилки MSE під час навчання та тестування моделі багатошарового персептрона з двома прихованими шарами та генетичним методом з адаптивною мутацією, а на 30 епосі значення помилки MSE на тестовій вибірці практично стає сталим значенням та дорівнює 0,0101.

Отже, запропоновано модифікований генетичний метод для настройки вагових коефіцієнтів штучної нейронної мережі для підвищення точності прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Принципова відмінність запропонованого генетичного методу від існуючих аналогів полягає у використанні адаптивної мутації, яка використовує в процесі своєї роботи рівень розвитку популяції. У ній найменш значущі гени з більшою ймовірністю будуть мутувати в хромосомах ніж з високою придатністю, це підвищує точність та збільшує їх роль в пошуку. Тобто чим гірше пристосована особина, тим вона далі від оптимуму, отже, їй надається підвищення ймовірності мутації, щоб вона могла відійти від поточного неоптимального положення.

3.3 Висновок

У цьому розділі було створено та проаналізовано моделі прогнозування ефективності десульфурації димових газів від діоксиду сірки на основі таких методів, як метод найближчого сусіда, випадковий ліс та метод опорних векторів. Було розроблено та проаналізовано моделі прогнозування ефективності процесу знесірчення димових газів з використанням штучної мережі типу багатошаровий перцептрон, а саме було створено багатошаровий перцептрон з одним прихованим шаром, багатошаровий перцептрон з двома прихованими шарами. Було розроблено генетичний метод для настройки вагових коефіцієнтів штучної нейронної мережі для підвищення точності прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Принципова відмінність запропонованого генетичного методу від існуючих аналогів полягає у використанні адаптивної мутації, яка використовує в процесі своєї роботи рівень розвитку популяції.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення й умови використання

Розроблюваний програмний продукт призначений для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Він являє собою набір бібліотек за допомогою яких можна створювати та досліджувати власні моделі залежності ступеня десульфурації методу мокрої абсорбційної очистки димових газів від використовуваних реагентів. Програмний продукт представлений у вигляді окремого віконного додатку завдяки якому можна обчислити прогноз вмісту діоксиду сірки у димовому газі після процесу десульфурації від обраних параметрів процесу за допомогою методу найближчого сусіду, випадкового лісу та опорних векторів, а також можна здійснювати тренування моделі та прогноз на основі багатопараметричного перцептрону, завантажуючи і застосовуючи власні експериментальні дані.

Для коректної роботи ПП необхідні такі мінімальні системні вимоги:

- платформа x64-based;
- операційна система Microsoft Windows 10/8/7 (рекомендується встановити всі останні оновлення);
- процесор з мінімальною тактовою частотою 1,8 ГГц.

Рекомендується використовувати як мінімум двоядерний процесор;

- оперативна пам'ять від 2 ГБ (рекомендується 4 ГБ та мінімум 2,5 ГБ при використанні на віртуальній машині);
- роздільна здатність екрану 1024x768 пікселів з 16-бітовим кольором;
- мінімум 2 ГБ вільного місця на жорсткому диску.

У своїй роботі ПП використовує такі необхідні компоненти:

- середовище Python 3.7;
- бібліотека NumPy та Pandas для роботи з масивами даних і формування датасетів;

- бібліотека Scikit-learn для реалізації алгоритмів машинного навчання;
- бібліотека Matplotlib для побудови графіків та візуалізації даних Matplotlib;
- бібліотека Tkinter для розробки інтерфейсу користувача.

4.2 Звертання до програмного продукту (файли проєкту)

Розроблюваний ПП складається з таких модулів:

Модуль `main.py` – головний модуль проєкту, що реалізує ініціалізацію ПП та елементів інтерфейсу користувача, він містить такі методи:

- `get_about_info` – здійснює вивід інформації про програмний продукт у окремому вікні;
- `checking values` – здійснює перевірку правильності значень введеним користувачем;
- `get_loading_data` – здійснює завантаження даних для навчання та тренування моделі;
- `get_preprocessing` – створює екземпляр класу `preprocessing` (модуля `preprocessing.py`) для здійснення попередньої обробки даних;
- `get_KNeighbors_predict` – створює екземпляр класу `KNeighbors` (модуля `KNeighbors.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу найближчого сусіда;
- `get_RandomForest_predict` – створює екземпляр класу `RandomForest` (модуля `RandomForest.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу випадковий ліс;
- `get_SVR_predict` – створює екземпляр класу `SVR` (модуля `SVR.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу опорних векторів;
- `get_mlp_ga_train` – створює екземпляр класу `mlp_ga` (модуля `mlp_ga.py`) для тренування моделі на основі багат шарового перцептрону.

– `get_mlp_ga_predict` – створює екземпляр класу `mlp_ga` (модуля `mlp_ga.py`) для здійснення обчислення прогнозу з використанням моделі на основі багат шарового перцептрону.

Модуль `preprocessing.py` – реалізує попередню обробку даних, він виконує видалення порожніх рядків, усунення викидів, видалення рядків, що повторюються, заповнення пропущених значень та нормалізацію даних.

Модуль `KNeighbors.py` – реалізує прогнозування за допомогою методу найближчого сусіду;

Модуль `RandomForest.py` – реалізує прогнозування за допомогою методу випадковий ліс.

Модуль `SVR.py` – реалізує прогнозування за допомогою методу опорних векторів.

Модуль `mlp_ga.py` – містить два класи `mpl` та `GeneticAlgorithm`, він реалізує прогнозування за допомогою моделі на основі багат шарового перцептрону з використанням генетичного алгоритму для настройки вагових коефіцієнтів.

4.3 Вхідні і вихідні дані

Для навчання на тестування моделей залежності ступеня сіркоочистки димових газів від використовуваних реагентів ПП використовує експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів та значення кількості епох навчання, введеним користувачем. Експериментальні дані представляють собою файл у форматі `csv`, де кожний рядок файлу – це один проведений експеримент процесу десульфурації димових газів від діоксиду сірки, значення параметрів процесу розділені між собою комами.

У наборі даних є сім параметрів, останній з яких є цільовим:

- `cont_entr` – вміст SO_2 на вході ($\text{м}^3 / \text{год}$);
- `first_stage` – об'єм газу на 1-у сходинку ($\text{м}^3 / \text{год}$);

- second_stage – об’єм газу на 2-у сходинку ($\text{м}^3 / \text{год}$);
- water – витрати води у абсорбері ($\text{м}^3 / \text{год}$);
- lime – витрати вапна ($\text{т} / \text{год}$);
- second_reagen – витрати вторинного реагенту ($\text{т} / \text{год}$);
- cont_out – вміст SO_2 на виході ($\text{м}^3 / \text{год}$).

На рис. 4.1 представлено приклад вхідних даних для навчання та тестування моделі на основі багатoshарового перцептрону.

	A	B	C	D	E	F	G
1	cont_entr;first_stage;second_stage;water;lime;second_reagen;cont_out						
2	1537;331;311;1.5;0.756;0;1169						
3	1501;367;329;1.5;0.651;0;881						
4	1571;365;328;1.5;0.48;;994						
5	1591;375;322;0.7;0;0.195;1180						
6	1554;380;322;1.6;0.457;0.212;980						
7	1445;373;319;1.5;0.766;0.254;816						
8	1472;359;311;1.6;0.776;0.361;728						
9	1534;385;320;2;0.765;0.134;806						
10	1391;372;314;1.7;0.42;0;812						
11	1553;376;321;1.7;0.486;0;946						
12							

Рисунок 4.1 – Експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів

Вихідними даними є значення середньої абсолютної і середньої квадратичної похибки моделі на навчальній і тестовій вибірці відповідно та час виконання, а також графік зміни значення цих похибок під час тренування цієї моделі.

Для здійснення прогнозу вмісту діоксиду сірки у димовому газі після процесу десульфурації ПП використовує в якості вхідних даних значення параметрів процесу десульфурації введеним користувачем (табл. 4.1):

Таблиця 4.1 – Вхідні дані для здійснення прогнозу вмісту діоксиду сірки у димовому газі після процесу десульфурації

Параметр	Тип даних
Вміст SO ₂ на вході (мг / м ³)	Ціле число
Об'єм газу на 1-у сходинку (м ³ / год)	Ціле число
Об'єм газу на 2-у сходинку (м ³ / год)	Ціле число
Витрати води у абсорбері (м ³ / год)	Число (крапка є роздільником десяткової частини)
Витрати вапна (т / год)	Число (крапка є роздільником десяткової частини)
Витрати вторинного реагенту (т / год)	Число (крапка є роздільником десяткової частини)

Вихідними даними прогнозу є значення вмісту діоксиду сірки (мг/м³, ціле число) на виході абсорбційної вежі у димовому газі після процесу десульфурації та час (секунди, число (крапка є роздільником десяткової частини)) виконання розрахунків.

4.4 Повідомлення

Під час роботи створений ПП виводить текстові повідомлення, які описані у табл. 4.2.

Таблиця 4.2 – Повідомлення під час роботи «Flue Gas Treatment 1.0»

Повідомлення	Опис
Вітаємо! Програмний продукт «Flue Gas Treatment 1.0» завантажений і готовий до роботи.	Повідомлення означає, що програмний продукт завантажений і готовий до роботи.
Будь ласка, заповніть порожнє поле.	Повідомлення появляється, коли поле порожнє і це означає, що потрібно обов'язково заповнити це поле для виконання прогнозування.

Продовження таблиці 4.2

Повідомлення	Опис
Будь ласка, виберіть метод для здійснення прогнозування.	Повідомлення появляється, коли користувач не вказав метод для прогнозування.
Використовуваний метод: Випадковий ліс Вміст SO ₂ на виході: 920 мг/м ³ Час виконання: 0.6593 с.	Повідомлення означає, що закінчено прогнозування з використанням методу випадковий ліс з такими результатами: прогнозований вміст діоксиду сірки після процесу десульфурації дорівнює 920 мг/м ³ та час виконання становить 0.6593 с.
Невірний формат файлу вибірки для тренування моделі.	Повідомлення появляється, коли користувач завантажив невірний формат файлу вибірки для навчання та тестування моделі багатошарового перцептрона, користувачу потрібно завантажити файл у форматі csv.
Помилка завантаження даних.	Повідомлення означає, що завантажені дані користувачем не є коректними.
Будь ласка, введіть кількість епох навчання в межах від 1 до 1000.	Повідомлення появляється, коли користувач ввів не валідне значення поля і це означає, що потрібно заповнити поле «Кількість епох навчання» в межах від одного до 1000.
Тренування моделі багатошаровий перцептрон Результати: MAE на навчальній вибірці: 0.0550797 MAE на тестовій вибірці: 0.0789698 MSE на навчальній вибірці: 0.0052979 MSE на тестовій вибірці: 0.0133135 Час виконання: 14.4692 с.	Повідомлення означає, що закінчено тренування моделі багатошарового перцептрона з такими результатами: середня абсолютна похибка на навчальній і тестовій вибірці склала 0.0550797 та 0.0789698 відповідно, а середньо квадратична похибка на навчальній і тестовій вибірці дорівнює 0.0052979 та 0.0133135 відповідно. Час виконання склав 14.4692 с.
Помилка розрахунку	Повідомлення означає, що під час роботи прогнозування або тренування моделі виникла помилка

4.5 Висновок

У цьому розділі було наведено призначення та функції розробленого ПП та мінімальні системні вимоги для його коректної роботи. Було наведено детальний опис модулів з яких складається ПП. Створений ПП має відкритий вихідний код, який доступний для перегляду, вивчення та зміни іншим розробникам. За допомогою його набору бібліотек можна створювати та досліджувати власні моделі залежності ступеня десульфурації методу мокрої абсорбційної очистки димових газів від використовуваних реагентів. Також у цьому розділі було наведено описання вхідних і вихідних даних та список всіх текстових повідомлень, які можуть з'являтися під час роботи оператора з ПП.

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення й умови виконання програмного продукту

Розроблюваний програмний продукт призначений для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Програмний продукт представлений у вигляді окремого віконного додатку (Windows-застосунок) завдяки якому можна обчислити прогноз вмісту діоксиду сірки у димовому газі після процесу десульфурації від обраних параметрів процесу за допомогою методу найближчого сусіду, випадкового лісу та опорних векторів, а також можна здійснювати тренування моделі та прогноз на основі багат шарового перцептрону, завантажуючи і застосовуючи власні експериментальні дані.

Для коректної роботи ПП необхідні такі мінімальні системні вимоги:

- платформа x64-based;
 - операційна система Microsoft Windows 10/8/7 (рекомендується встановити всі останні оновлення);
 - процесор з мінімальною тактовою частотою 1,8 ГГц.
- Рекомендується використовувати як мінімум двоядерний процесор;
- оперативна пам'ять від 2 ГБ (рекомендується 4 ГБ та мінімум 2,5 ГБ при використанні на віртуальній машині);
 - роздільна здатність екрану 1024x768 пікселів з 16-бітовим кольором;
 - мінімум 2 ГБ вільного місця на жорсткому диску.

У своїй роботі ПП використовує такі необхідні компоненти:

- середовище Python 3.7;
- бібліотека NumPy та Pandas для роботи з масивами даних і формування датасетів;
- бібліотека Scikit-learn для реалізації алгоритмів машинного навчання;
- бібліотека Matplotlib для побудови графіків та візуалізації даних Matplotlib;

– бібліотека Tkinter для розробки інтерфейсу користувача.

5.2 Виконання програмного продукту

Після вдалого запуску ПП відкривається головне вікно, яке представлено на рис. 5.1, у консолі виводу інформації буде відображено повідомлення «Вітаємо! Програмний продукт «Flue Gas Treatment 1.0» завантажений і готовий до роботи».

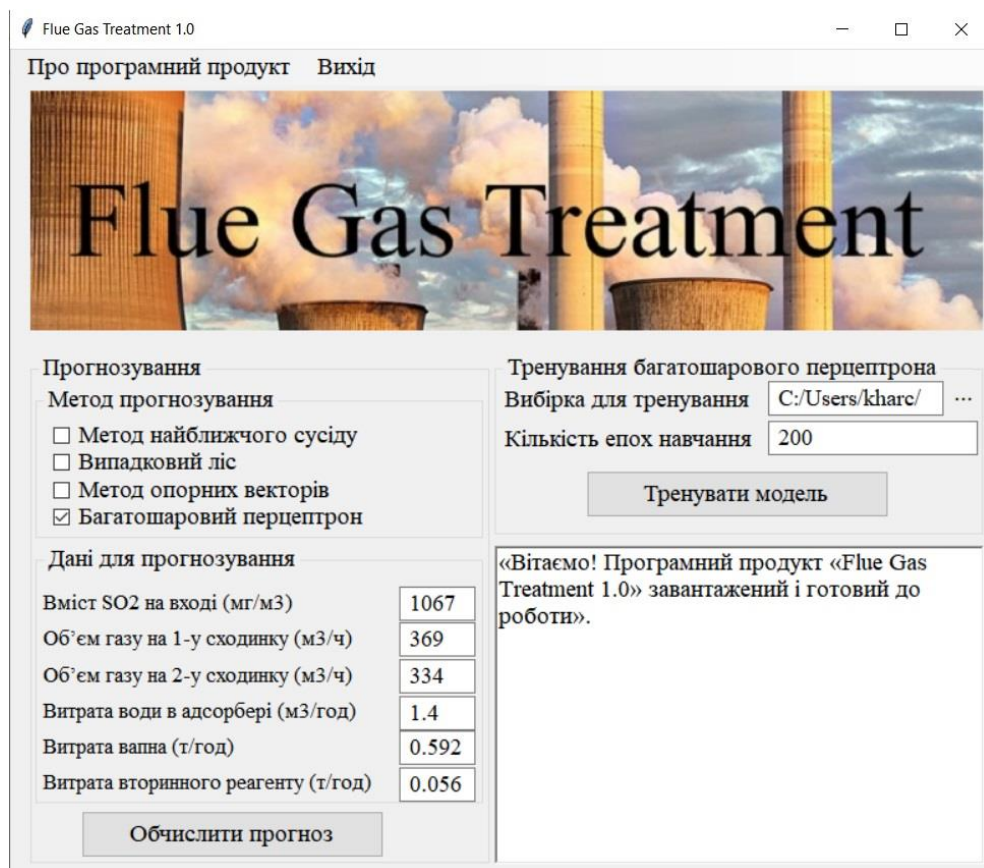


Рисунок 5.1 – Головне вікно ПП

Для того, щоб здійснити прогнозування результатів процесу десульфурації димових газів від діоксиду сірки, спочатку необхідно обрати один або декілька методів із запропонованих (метод найближчого сусіду, випадковий лісу, метод опорних векторів та багатошаровий перцептрон). Потім необхідно ввести значення параметрів процесу, а саме:

- вміст SO_2 на вході ($\text{мг} / \text{м}^3$);
- об'єм газу на 1-у сходинку ($\text{м}^3 / \text{год}$);
- об'єм газу на 2-у сходинку ($\text{м}^3 / \text{год}$);
- витрати води у абсорбері ($\text{м}^3 / \text{год}$);
- витрати вапна ($\text{т} / \text{год}$);
- витрати вторинного реагенту ($\text{т} / \text{год}$).

Результатами прогнозування є значення вмісту діоксиду сірки ($\text{мг} / \text{м}^3$, ціле число) на виході абсорбційної вежі у димовому газі після процесу десульфурації та час (секунди, число (крапка є роздільником десяткової частини)) виконання розрахунків. Результати прогнозування виводяться у консоль виводу інформації. Приклад виведення результатів прогнозування ПП приведено на рис. 5.2.

The screenshot shows the 'Flue Gas Treatment 1.0' application window. It has a title bar with standard Windows controls. Below the title bar is a menu bar with 'Про програмний продукт' and 'Вихід'. The main area features a large banner image of industrial smokestacks with the text 'Flue Gas Treatment' overlaid. Below the banner, there are two main sections: 'Прогнозування' (Prediction) on the left and 'Тренування багатошарового перцептрона' (Training of a multilayer perceptron) on the right. The 'Прогнозування' section includes a 'Метод прогнозування' (Prediction method) dropdown with radio buttons for 'Метод найближчого сусіду' (Nearest neighbor method), 'Випадковий ліс' (Random forest), 'Метод опорних векторів' (Support vector method), and 'Багатошаровий перцептрон' (Multilayer perceptron). Below this is a 'Дані для прогнозування' (Prediction data) table with input fields for various parameters. At the bottom of this section is an 'Обчислити прогноз' (Calculate prediction) button. The 'Тренування' section includes a 'Вибірка для тренування' (Training data selection) dropdown, a 'Кількість епох навчання' (Number of training epochs) input field, and a 'Тренувати модель' (Train model) button. Below these are the results of the training process, including the used method and the resulting SO_2 concentration and execution time.

Дані для прогнозування	
Вміст SO_2 на вході ($\text{мг}/\text{м}^3$)	1663
Об'єм газу на 1-у сходинку ($\text{м}^3/\text{ч}$)	344
Об'єм газу на 2-у сходинку ($\text{м}^3/\text{ч}$)	321
Витрата води в адсорбері ($\text{м}^3/\text{год}$)	1
Витрата вапна ($\text{т}/\text{год}$)	0.618
Витрата вторинного реагенту ($\text{т}/\text{год}$)	0.134

Тренування багатошарового перцептрона	
Вибірка для тренування	C:/Users/kharc/...
Кількість епох навчання	100
Тренувати модель	
Використовуваний метод: Випадковий ліс	
Вміст SO_2 на виході: 920 $\text{мг}/\text{м}^3$	
Використовуваний метод: Метод опорних векторів	
Вміст SO_2 на виході: 935 $\text{мг}/\text{м}^3$	
Час виконання: 0.6593 сек.	

Рисунок 5.2 – Вікно виводу результатів прогнозування

Для того, щоб здійснити навчання на тестування моделі залежності ступеня сіркоочистки димових газів від використовуваних реагентів на основі багатошарового перцептрону необхідно завантажити експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів у форматі csv та вказати кількість епох навчання в межах від одного до 1000.

Результатами навчання на тестування моделі багатошарового перцептрона є значення середньої абсолютної і середньої квадратичної похибки моделі на навчальній і тестовій вибірці відповідно та час виконання, а також графік зміни значення цих похибок під час тренування цієї моделі. Результати навчання на тестування моделі багатошарового перцептрона виводяться у консоль виводу інформації. Приклад виведення результатів прогнозування ПП приведено на рис. 5.3–5.5

The screenshot shows the 'Flue Gas Treatment 1.0' application window. It features a title bar, a menu bar with 'Про програмний продукт' and 'Вихід', and a main content area with a background image of industrial chimneys. The interface is divided into several sections:

- Прогнозування (Forecasting):** Includes a dropdown for 'Метод прогнозування' (Forecasting method) with options: 'Метод найближчого сусіду' (Nearest neighbor method), 'Випадковий ліс' (Random forest), 'Метод опорних векторів' (Support vector method), and 'Багатошаровий перцептрон' (Neural network) which is selected.
- Дані для прогнозування (Data for forecasting):** A table with input parameters and their values:

Вміст SO ₂ на вході (мг/м ³)	794
Об'єм газу на 1-у сходинку (м ³ /ч)	360
Об'єм газу на 2-у сходинку (м ³ /ч)	307
Витрата води в адсорбері (м ³ /год)	1.3
Витрата вапна (т/год)	0.09
Витрата вторинного реагенту (т/год)	0.08
- Тренування багатошарового перцептрона (Training the neural network):** Includes a dropdown for 'Вибірка для тренування' (Training sample) set to 'C:/Users/kharc/' and a text box for 'Кількість епох навчання' (Number of training epochs) set to 50. A 'Тренувати модель' (Train model) button is present.
- Тренування моделі багатошаровий перцептрон (Training the neural network model):** Displays the results:

Результати:

MAE на навчальній вибірці: 0.0550797

MAE на тестовій вибірці: 0.0789698

MSE на навчальній вибірці: 0.0052979

MSE на тестовій вибірці: 0.0133135

Час виконання: 14.4692 сек.

Buttons for 'Обчислити прогноз' (Calculate forecast) and 'Тренувати модель' (Train model) are located at the bottom of their respective sections.

Рисунок 5.3 – Вікно виводу результатів навчання на тестування моделі багатошарового перцептрона

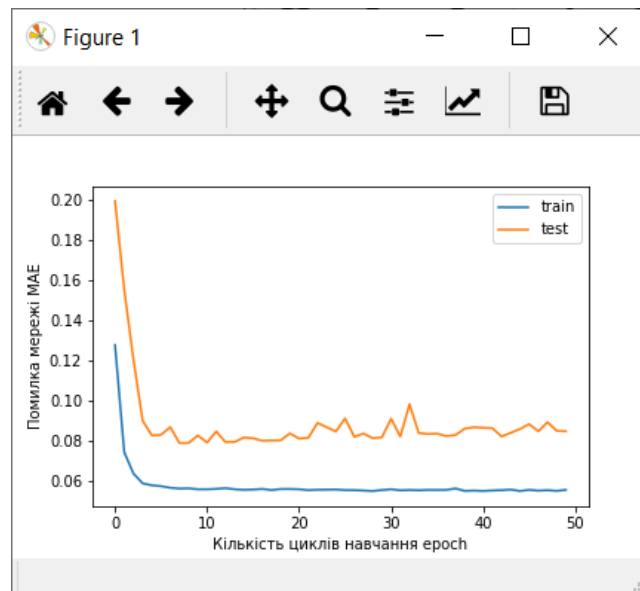


Рисунок 5.4 – Вікно виводу графіка зміни значення метрики MAE на навчальній і тестовій вибірці під час навчання

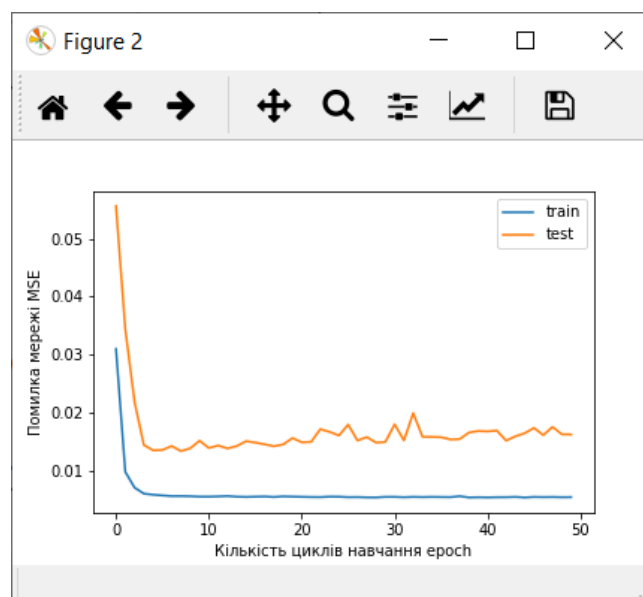


Рисунок 5.5 – Вікно виводу графіка зміни значення метрики MSE на навчальній і тестовій вибірці під час навчання

Для того щоб отримати керівництво по роботі з ПП, користувачу необхідно у верхньому лівому кутку у меню обрати «Про програмний продукт» і відкриється вікно з інформацією про ПП та довідка як з ним працювати.

Архітектура моделі багатошарового перцептрона та метод навчання не можуть бути змінені оператором через Windows-застосунок.

5.3 Повідомлення оператору

Під час роботи оператора з розробленим ПП виводяться різні текстові повідомлення, вони інформують оператора про помилки або його неправильні дії, а також про закінчення розрахунків прогнозування. Список всіх текстових повідомлень, описання їх змісту та дії оператора, які він повинен виконати при отриманні цих повідомлень представлені у розділі «Керівництво програміста» в таблиці 4.2.

5.4 Висновок

У цьому розділі було наведено призначення та функції розробленого ПП та мінімальні системні вимоги для його коректної роботи. Було описана покрокова інструкція роботи оператора з розробленим ПП, а саме було зазначено як виконувати прогнозування результатів процесу десульфурації димових газів від діоксиду сірки або здійснювати навчання та тестування моделі багатошарового перцептрону. Було перераховано список всіх текстових повідомлень, які можуть з'являтися під час роботи оператора з ПП. Була наведена детальна інструкція по роботі оператора з ПП, а саме приклад пошуку і вибір місць розміщення ДЖ та одночасного закріплення за ними споживачів. Розроблений інтерфейс ПП є інтуїтивно зрозумілий та простим і додаткове навчання оператора не потребується, якщо виникають питання по роботі з ПП присутня довідка.

6 ЕКСПЕРИМЕНТИ ТА РЕЗУЛЬТАТИ

6.1 Проведення експериментів

Спроектувавши та побудувавши моделі прогнозування ефективності процесу знесірчення димових газів від діоксиду сірки було проведено оцінку ефективності цих моделі за допомогою спеціалізованих метрик та тестової вибірки даних. Проведено порівняльний аналіз результатів роботи розроблених моделей (багатошаровий перцептрон з одним прихованим шаром, багатошаровий перцептрон з двома прихованим шаром та багатошаровий перцептрон з двома прихованим шаром з розробленим ГА) з такими методами як: метод найближчого сусіду, випадковий ліс, метод опорних векторів. Критеріям оцінювання були значення MAE, MSE та час виконання (табл. 6.1). Для тестування розроблених моделей прогнозування ефективності процесу знесірчення димових газів використано експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів з 07:00 24 травня 2020 р. по 06:00 31 грудня 2020 р., набір складається з 3300 екземплярів.

Таблиця 6.1 – Порівняльний аналіз результатів розроблених моделей прогнозування результатів процесу десульфурації димових газів від діоксиду сірки

Назва методу	MSE	MAE	Час виконання, с
Метод найближчого сусіду	0,0153	0,0897	362
Випадковий ліс	0,0152	0,0954	315
Метод опорних векторів	0,0131	0,0842	374
Багатошаровий перцептрон з одним прихованим шаром	0,0149	0,0940	764
Багатошаровий перцептрон з двома прихованими шарами	0,0129	0,0794	867
Багатошаровий перцептрон з двома прихованими шарами та розробленим ГА	0,0101	0,0677	489

З табл. 6.1 можна побачити, що застосування нейронних мереж у порівнянні з класичними методами дає вигравш у точності, так наприклад,

значення MSE у багатошарового перцептрона з одним прихованим шаром або двома прихованими шарами відповідно склало 0,0149 та 0,0129, а значення MSE прогнозування метод найближчого сусіду та випадковий ліс відповідно дорівнює 0,0153 та 0,0152. Отже, нейронні мережі можуть досягати більш високої точності прогнозування, ніж класичні моделі, оскільки вони набагато краще здатні обробляти нелінійну поведінку, але нейронні мережі поступаються у часі виконанні. Час виконання багатошарового перцептрона з одним прихованим шаром складає 764 с, а методу найближчого сусіду дорівнює 362 с. Для подолання цієї проблеми було розроблено модифікований генетичний метод з адаптивною мутацією, який дозволяє підвищити точність прогнозування та зменшити час навчання нейронної мережі. Під час тестування він показав найменше значення MAE прогнозування, вона склала 0,0677, це на 29% менше ніж у метода випадковий ліс та на 14% ніж у багатошарового перцептрона з двома прихованими шарами. Також час виконання багатошарового перцептрона з двома прихованими шарами та розробленим ГА дорівнює 489 с, що є на 35% менше ніж у багатошарового перцептрона з одним прихованим шаром та на 43% ніж у багатошарового перцептрона з двома прихованими шарами. На підставі проведеної роботи можна зробити висновок, що для навчання НМ є перспективним використання модифікованого генетичного алгоритму з адаптивною мутацією, оскільки його застосування дозволяє скоротити час навчання і досягти більш глибокого мінімуму помилки навчання НМ.

У цьому розділі було описано методику проведення експериментів розроблених моделей. Було проведено порівняльний аналіз розробленого генетичного методу та відомих, в результаті чого було встановлено, що він дає найменшу помилку MSE та MAE, а також дозволяє зменшити час виконання прогнозування ефективності десульфуризації димових газів.

ВИСНОВКИ

Під час виконання роботи було проведено аналіз предметної області, визначено основні поняття, пов'язані з предметною областю і об'єктом дослідження та розробки, визначено проблематику поставленого завдання, наведено її актуальність та доцільність вирішення.

Було досліджено та проведено порівняльну характеристику існуючих програмних засобів, що можуть бути використані для дослідження та створення прогнозування результатів процесу абсорбційної очистки димових газів від діоксиду сірки.

Було досліджено основні методи прогнозування ефективності знесірчення димових газів та проведено порівняльний аналіз цих методів, який показав перевагу використання ГА та НМ для вирішення поставленого завдання.

Наукова новизна роботи полягає у тому, що запропоновано модифікований генетичний метод для настройки вагових коефіцієнтів штучної нейронної мережі для прогнозування результатів процесу десульфурзації димових газів від діоксиду сірки, який раніше не використовувався. Принципова відмінність запропонованого генетичного методу від існуючих аналогів полягає у використанні адаптивної мутації, яка використовує в процесі своєї роботи рівень розвитку популяції. У ній найменш значущі гени з більшою ймовірністю будуть мутувати в хромосомах ніж з високою придатністю, це підвищує точність та збільшує їх роль в пошуку. Тобто чим гірше пристосована особина, тим вона далі від оптимуму, отже, їй надається підвищення ймовірності мутації, щоб вона могла відійти від поточного неоптимального положення. Було проведено порівняльний аналіз розробленого генетичного методу та відомих, в результаті чого було встановлено, що він дає найменшу помилку MAE, вона склала 0.677, це на 29% менше ніж у метода випадковий ліс та на 14% ніж у багатошарового перцептрона з двома прихованими шарами. Також

встановлено, що час виконання багатошарового перцептрона з двома прихованими шарами та розробленим ГА дорівнює 489с, що є на 35% менше ніж у багатошарового перцептрона з одним прихованим шаром та на 43% ніж у багатошарового перцептрона з двома прихованими шарами. На підставі проведеного дослідження створеного модифікованого генетичного методу можна зробити висновок, що використання його дає підвищення точності прогнозування результатів процесу десульфурації димових газів.

Розроблено ПП для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Він являє собою набір бібліотек за допомогою яких можна створювати та досліджувати власні моделі залежності ступеня десульфурації методу мокрої абсорбційної очистки димових газів від використовуваних реагентів. Програмний продукт представлений у вигляді окремого віконного додатку (Windows-застосунок) завдяки якому можна обчислити прогноз вмісту діоксиду сірки у димовому газі після процесу десульфурації від обраних параметрів процесу за допомогою методу найближчого сусіду, випадкового лісу та опорних векторів, а також можна здійснювати тренування моделі та прогноз на основі багатошарового перцептрону, завантажуючи і застосовуючи власні експериментальні дані.

Отримані результати дозволяють запропонувати ефективний метод для підвищення точності прогнозування та оптимізації експлуатаційних витрат процесу десульфурації димових газів від діоксиду сірки, внаслідок чого знизити викиди SO_2 в атмосферу від вугільних електростанцій.

За результатами досліджень було опубліковано 15 наукових праць, з яких 3 статті та 8 доповідей на конференціях, що індексується в міжнародній наукометричній базі Scopus, та 4 статті у наукових фахових журналах України.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Operation strategy optimization of desulfurization system based on data mining / [S. Liu, L. Sun, S. Zhu et al.] // *Applied Mathematical Modelling*. – 2020. – Vol. 81. – P. 144–158.
2. Prediction Model of Desulfurization Efficiency of Coal-Fired Power Plants Based on Long Short-Term Memory Neural Network / [J. Fu, H. Xiao, T. Wang et al.] // 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) : Second international conference, Atlanta, 14-17 July 2019 : proceedings. – Atlanta: IEEE, 2019. – P. 40–45.
3. Multimodel Anomaly Identification and Control in Wet Limestone-Gypsum Flue Gas Desulphurization System / [X. Li, Q. Liu, K. Wang et al.] // *Complexity*. – 2020. – Vol. 2020. – P. 1–17.
4. Global Sulfur Dioxide Emissions and the Driving Forces / [Q. Zhong, H. Shen, X. Yun et al.] // *Environmental Science & Technology*. – 2020. – Vol. 54. – P. 6508–6517.
5. Predictive scheduling of wet flue gas desulfurization system based on reinforcement learning / [Z. Shao, F. Si, D. Kudenko et al.] // *Computers & Chemical Engineering*. – 2020. – Vol. 141. – P. 1–13.
6. An investigation on data mining and operating optimization for wet flue gas desulfurization systems / [Z. Qiao, X. Wang, H. Gu et al.] // *Fuel*. – 2019. – Vol. 258. – P. 1–10.
7. Modeling of the wet flue gas desulfurization system to utilize low-grade limestone / [J. Lim, Y. Choi, G. Kim et al.] // *Korean Journal of Chemical Engineering*. – 2020. – Vol. 37. – P. 2085–2093.
8. Wet Scrubber - Flue Gas Desulfurization (FGD) [Electronic resource]. – Access mode: <https://www.babcock.com/home/products/wet-scrubbers-fgd>
9. Kiil, S. Experimental Investigation and Modeling of a Wet Flue Gas

Desulfurization Pilot Plant / S. Kiil, M. Michelsen, K. Dam-Johansen // Industrial & Engineering Chemistry Research. – 1998. – Vol. 37, Issue 7. – P. 2792–2806.

10. A model for performance optimization of wet flue gas desulfurization systems of power plants / [Y. Zhong, X. Gao, W. Huo et al.] // Fuel Processing Technology. – 2008. – Vol. 89. – P. 1025-1032.

11. Modeling Study of Selenium Migration Behavior in Wet Flue Gas Desulfurization Spray Towers / [R. Zou, G. Luo, C. Fang et al.] // Environmental Science & Technology. – 2020. – Vol. 54. – P. 16128–16137.

12. Wet flue gas desulfurization using micro vortex flow scrubber: Characteristics, modeling and simulation / [Y. Guo, Z. Xu, C. Zheng et al.] // Separation and Purification Technology. – 2020. – Vol. 141. – P. 1–8.

13. Simulation of SO₂ absorption and performance enhancement of wet flue gas desulfurization system / [Z. Shao, Y. Zhang, W. Gao et al.] // Process Safety and Environmental Protection. – 2021. – Vol. 150. – P. 453–463.

14. The evaluation on artificial neural networks (ANN) and multiple linear regressions (MLR) models for predicting SO₂ concentration / [S. Shams, A. Jahani, S. Kalantary et al.] // Urban Climate. – 2021. – Vol. 37. – P. 1–10.

15. Genetic-Algorithm-Optimized Sequential Model for Water Temperature Prediction / [S. Stajkowski, D. Kumar, P. Samui et al.] // Sustainability. – 2020. – Vol. 12. – P. 5374–5391.

16. Prediction of SO_x–NO_x emission from a coal-fired CFB power plant with machine learning: Plant data learned by deep neural network and least square support vector machine / [D. Adams, D. Oh, D. Kim et al.] // Journal of Cleaner Production. – 2020. – Vol. 270. – P. 1–16.

17. Application of the Weighted K-Nearest Neighbor Algorithm for Short-Term Load Forecasting / [G. Fan, Y. Guo, J. Zheng et al.] // Energies. – 2019. – Vol. 12. – P. 916–934.

18. An efficient instance selection algorithm for k nearest neighbor regression / [Y. Song, J. Liang, J. Lu et al.] // Neurocomputing. – 2017. – Vol. 251. – P. 26–34.

19. Magnussen, S. Thek-nearest neighbor technique with local linear regression / S. Magnussen, E. Tomppo // *Scandinavian Journal of Forest Research*. – 2013. – Vol. 29, Issue 2. – P. 120–131.
20. Мюллер, А. О. Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными / А. Мюллер, С. Гвидо. – Москва : Вильямс, 2016. – 393 с.
21. Regression Methods Based on Nearest Neighbors with Adaptive Distance Metrics Applied to a Polymerization Process / [S. Curteanu, F. Leon, A. Mircea-Vicoveanu et al.] // *Mathematics*. – 2021. – Vol. 9. – P. 547–567.
22. Schonlau, M. The random forest algorithm for statistical learning / M. Schonlau, R. Zou // *The Stata Journal: Promoting communications on statistics and Stata*. – 2020. – Vol. 20, Issue 1. – P. 3–29.
23. A Machine Learning Approach for Heart Rate Estimation from PPG Signal using Random Forest Regression Algorithm / [S. Bashar, M. Miah, A. Karim et al.] // *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE) : Second international conference, Cox'sBazar, 7–9 February 2019 : proceedings*. – Cox'sBazar: IEEE, 2019. – P. 1–5.
24. Couronne, R. Random forest versus logistic regression: a large-scale benchmark experiment / R. Couronne, P. Probst, A. Boulesteix // *BMC Bioinformatics*. – 2018. – Vol. 19, Issue 1. – P. 2–14.
25. Flach, P. Machine Learning: The Art and Science of Algorithms that Make Sense of Data / P. Flach. – Cambridge : Cambridge University Press, 2017. – 409 p.
26. Gao, X. Optimal Parameter Selection for Support Vector Machine Based on Artificial Bee Colony Algorithm: A Case Study of Grid-Connected PV System Power Prediction / X. Gao, S. Yang, S. Pan // *Computational Intelligence and Neuroscience*. – 2017. – Vol. 2017. – P. 1–14.
27. Comparing different supervised machine learning algorithms for disease prediction / [S. Uddin, A. Khan, M. Hossain et al.] // *BMC Medical Informatics and Decision Making*. – 2019. – Vol. 19. – P. 1–16.

28. Smart grid load forecasting using online support vector regression / [P. Vrablecova, A. Bou Ezzeddine, V. Rozinajova et al.] // Computers & Electrical Engineering. – 2018. – Vol. 65. – P. 102–117.
29. Prediction and optimization of a desulphurization system using cmac neural network and genetic algorithm / [Z. Kong, Y. Zhang, X. Wang et al.] // Journal of Environmental Engineering and Landscape Management. – 2020. – Vol. 28, – P. 74–87.
30. Xu, J. Improved Genetic Algorithm to Solve the Scheduling Problem of College English Courses / J. Xu // Complexity. – 2021. – Vol. 2021. – P. 1–11.
31. Олійник, А. О. Інтелектуальний аналіз даних / А. О. Олійник, С. О. Субботін, О. О. Олійник. – Запоріжжя : ЗНТУ, 2012. – 278 с.
32. Subbotin, S. A. Methods of sampling based on exhaustive and evolutionary search / S. A. Subbotin // Automatic Control and Computer Sciences. – 2013. – Vol. 47, № 3. – P. 113–121.
33. Субботін, С. О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей: монографія / С. О. Субботін, А. О. Олійник, О. О. Олійник ; під заг. ред. С.О. Субботіна. – Запоріжжя : ЗНТУ, 2009. – 375 с.
34. Xu, M. Optimization of Multiple Traveling Salesman Problem Based on Simulated Annealing Genetic Algorithm / M. Xu, S. Li, J. Guo // MATEC Web of Conferences. – 2017. – Vol. 100. – P. 1–8.
35. Katoch, S. A review on genetic algorithm: past, present, and future / S. Katoch, S. Chauhan, V. Kumar // Multimedia Tools and Applications. – 2020. – Vol. 80. – P. 8091–8126.
36. Субботін, С. О. Нейронні мережі : Навчальний посібник / С. О. Субботін, А. О. Олійник ; під заг. ред. проф. С. О. Субботіна. – Запоріжжя : ЗНТУ, 2014. – 135 с.
37. Subbotin, S. A. The training set quality measures for neural network learning / S. A. Subbotin // Optical Memory and Neural Networks (Information Optics). – 2010. – Vol. 19, № 2. – P. 126–139.

38. Park, Y. Artificial Neural Networks: Multilayer Perceptron for Ecological Modeling / Y. Park, S. Lek // *Developments in Environmental Modelling*. – 2016. – Vol. 28. – P. 123–140.

39. Artificial Neural Networks to Estimate the Influence of Vehicular Emission Variables on Morbidity and Mortality in the Largest Metropolis in South America / Y. Kachba, D. Chiroli, J. T. Belotti // *Sustainability*. – 2020. – Vol. 12. – P. 2621–2636.

40. Субботин, С. А. Методы эволюционного отбора комбинаций признаков с использованием априорной информации об их индивидуальной значимости / С. А. Субботин, А. А. Олейник // *Нейрокомпьютеры: разработка, применение*. – 2007. – № 7. – С. 8–13.

41. Субботін, С. О. Нейронні мережі : теорія та практика: навч. посіб. / С. О. Субботіна. – Житомир : Вид. О. О. Євенок, 2020. – 184 с.

42. Faris, H. Evolving Radial Basis Function Networks Using Moth–Flame Optimizer / H. Faris, I. Aljarah, S. Mirjalili // *Expert Systems with Applications*. – 2017. – Vol. 1. – P. 537–550.

43. Subbotin, S. Radial-Basis Function Neural Network Synthesis on the Basis of Decision Tree / S. Subbotin // *Optical Memory and Neural Networks*. – 2020. – Vol. 29. – P. 7–18.

44. Homero De Jesus, D. Radial Basis Function Network versus Regression Model in Manufacturing Processes Prediction / D. Homero De Jesus // *Expert Systems with Applications*. – 2018. – Vol. 1. – P. 1–4.

45. Multivariate statistical inference in a radial basis function neural network / [H. de Leon-Delgado, R. Praga-Alejo, D. Gonzalez-Gonzalez et al.] // *Expert Systems with Applications*. – 2018. – Vol. 93. – P. 313–321.

46. Bandara, K. Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach / K. Bandara, C. Bergmeir, S. Smyl // *Expert Systems with Applications*. – 2020. – Vol. 140. – P. 1–16.

47. Stacked bidirectional and unidirectional LSTM recurrent neural

network for forecasting network-wide traffic state with missing values / [Z. Cui, R. Ke, Z. Pu et al.] // *Transportation Research Part C: Emerging Technologies*. – 2020. – Vol. 118. – P. 1–11.

48. Dong, J. Cost Index Predictions for Construction Engineering Based on LSTM Neural Networks / J. Dong, Y. Chen, G. Guan // *Advances in Civil Engineering*. – 2020. – Vol. 2020. – P. 1–14.

49. Qian, F. Stock Prediction Based on LSTM under Different Stability / F. Qian, X. Chen // *2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, Chengdu, 12–15 April 2019 : proceedings. – Los Alamitos: IEEE, 2019. – P. 1–4.

50. EvoDeep: A new evolutionary approach for automatic Deep Neural Networks parametrisation / [A. Martin, R. Lara-Cabrera, F. Fuentes-Hurtado et al.] // *Journal of Parallel and Distributed Computing*. – 2018. – Vol. 117. – P. 180–191.

51. Designing neural networks through neuroevolution / [K. Stanley, J. Clune, J. Lehman et al.] // *Nature Machine Intelligence*. – 2019. – Vol. 1. – P. 24–35.

52. Chauhan, N. Parametric comparison of data mining tools / N. Chauhan, N. Gautam // *International Journal of Advanced Technology in Engineering and Science*. – 2015. – Vol. 3. – P. 291–298.

53. Latest Tools for Data Mining and Machine Learning / [K. Verma, S. Bhardwaj, R. Arya et al.] // *International Journal of Innovative Technology and Exploring Engineering*. – 2019. – Vol. 8. – P. 18–23.

54. Cleves, M. An introduction to survival analysis using Stata / M. Cleves, W. Gould. – College Station : Stata Press, 2016. – 428 p.

55. Performance Assessment and Translation of Physiologically Based Pharmacokinetic Models From acslX to Berkeley Madonna, MATLAB, and R Language: Oxytetracycline and Gold Nanoparticles As Case Examples / [Z. Lin, M. Jaber-Douraki, C. He et al.] // *Toxicological Sciences*. – 2017. – Vol. 158. – P. 23–35.

56. Jotsov, V. Applications of Advanced Analytics Methods in Sas

Enterprise Miner / V. Jotsov, E. Iliev // *Advances in Intelligent Systems and Computing*. – 2015. – Vol. 14. – P. 413–429.

57. Wickham, H. *R for Data Science: Visualize, Model, Transform, Tidy, and Import Data* / H. Wickham, G. Grolemund. – Sebastopol : O'Reilly Media, 2016. – 520 p.

58. Прата, С. *Язык программирования C++. Лекции и упражнения* / С. Прата. – М. : Диалектика-Вильямс, 2016. – 1248 с.

59. Lee, W. *Python machine learning* / W. Lee. – London : Wiley, 2019. – 320 p.

60. McNicholas, P. *Data science with Julia* / P. McNicholas, P. Tait. – London : Chapman and Hall, 2019. – 240 p.

61. Сьерра, К. *Изучаем Java. Лекции и упражнения* / К. Сьерра, Б. Бэйтс. – М. : Эксмо, 2017. – 720 с.

62. Nguyen, Q. *Hands-On Application Development with PyCharm* / Q. Nguyen. – Birmingham : Packt Publishing Ltd, 2019. – 494 p.

63. Linge, S. *Programming for computations - Python* / S. Linge, H. Langtangen. – New York : Springer, 2016. – 232 p.

64. Galea, A. *Beginning data analysis with Python and Jupyter* / A. Galea. – Birmingham : Packt Publishing Ltd, 2018. – 194 p.

65. Stetsyuk, V. *Method of choosing the programming environment for software* / V. Stetsyuk, T. Hovorushchenko // *International Scientific Journal Computer Systems and Information Technologies*. – 2021. – Vol. 2. – P. 6–10.

66. Johansson, R. *Numerical Python: Scientific Computing and Data Science Applications with Numpy, SciPy and Matplotlib* / R. Johansson. – Berkeley : Apress, 2019. – 723 p.

67. Hao, J. *Machine Learning Made Easy: A Review of Scikit-learn Package in Python Programming Language* / J. Hao, T. Ho // *Journal of Educational and Behavioral Statistics*. – 2019. – Vol. 44. – P. 348–361.

68. McKinney, W. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython* / W. McKinney. – Beijing : O'Reilly Media, 2017. –

550 p.

69. Manaswi, N. Understanding and Working with Keras / N. Manaswi // Deep Learning with Applications Using Python. – 2018. – Vol. 1. – P. 31–43.

70. Yim, A. Matplotlib for Python Developers: Effective techniques for data visualization with Python / A. Yim, C. Chung, A. Yu. – Birmingham : Packt Publishing, 2018. – 300 p.

71. Parial, P. Python the game changer in the field of Machine Learning, Data Science and IoT: A Review / P. Parial // International Journal for Research in Applied Science and Engineering Technology. – 2021. – Vol. 9. – P. 1827–1837.

72. Big data preprocessing: methods and prospects / [S. Garcia, S. Ramirez-Gallego, J. Luengo et al.] // Big Data Analytics. – 2016. – Vol. 1. – P. 1–22.

73. Розробка еволюційних алгоритмів у задачах управління асортиментом мережових аптек / [Є.М. Федорченко, А.О. Олійник, О.О. Степаненко та ін.] // Штучний інтелект. – 2019. – № 3–4. – С. 58–71.

74. Cheng, H. Application of improved BP neural network based on LM algorithm in desulfurization system of thermal power plant / H. Cheng, L. Cui, J. Li // 2017 Chinese Automation Congress (CAC), Jinan, 20–22 Oct. 2017 : proceedings. – Jinan: IEEE, 2017. – P. 1–4.

75. Development of Genetic Methods of Network Pharmacy Financial Indicators Optimization / [A. Oliinyk, I. Fedorchenko, T. Zaiko et al.] // 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), Kyiv, 8-11 October 2019 : proceedings. – Kyiv: IEEE, 2019. – P. 607–612.

76. Субботін, С. О. Інтелектуальні системи : Навчальний посібник / С. О. Субботін, А. О. Олійник ; під заг. ред. проф. С. О. Субботіна. – Запоріжжя : ЗНТУ, 2014. – 218 с.

77. Research on Gas Concentration Prediction Models Based on LSTM Multidimensional Time Series / [T. Zhang, S. Song, S. Li et al.] // Energies. – 2019. – Vol. 12. – P. 161–176.

78. Development of an evolutionary optimization method for financial indicators of pharmacies / [A. Oliinyk, I. Fedorchenko, V. Donenko et al.] // International Journal of Computing – 2020. – Vol. 19. – P. 449–463.

79. Development of a Genetic Method for Image Recognition in the Form of Radiographs / [I. Fedorchenko, A. Oliinyk, A. Stepanenko et al.] // 3rd International Workshop on Informatics and Data-Driven Medicine (IDDM'2020), Växjö, Sweden, 19-21 November 2020 : proceedings. – Växjö: IDDM, 2020. – Vol. 2753. – P. 94 – 107.

80. Development of genetic methods for predicting the incidence of volumes of emissions of pollutants in air / [A. Oliinyk, I. Fedorchenko, A. Stepanenko et al.] // 2nd International Workshop on Informatics and Data-Driven Medicine : Second international conference, Lviv, 11-13 November 2019 : proceedings. – Lviv: IDDM, 2019. – P. 340–353.

81. Construction of a genetic method to forecast the population health indicators based on neural network models / [I. Fedorchenko, A. Oliinyk, A. Stepanenko et al.] // Eastern-European Journal of Enterprise Technologies. – 2020. – Vol. 1, Issue 4103. – P. 52–63.

82. Hui, C. Artificial neural networks / C. Hui. – Rijeka : InTech, 2011. – 600 p.

83. Yasnitsky, L. Algorithm for Searching and Analyzing Abnormal Observations of Statistical Information Based on The Arnold – Kolmogorov – Hecht-Nielsen Theorem / L. Yasnitsky // International Journal of Advanced Trends in Computer Science and Engineering. – 2020. – Vol. 9. – P. 1814–1819.

84. Schmidt-Hieber, J. Nonparametric regression using deep neural networks with ReLU activation function / J. Schmidt-Hieber // The Annals of Statistics. – 2020. – Vol. 48. – P. 1875–1897.

85. Jais, I. Adam Optimization Algorithm for Wide and Deep Neural Network / I. Jais, A. Ismail, S. Nisa // Knowledge Engineering and Data Science. – 2019. – Vol. 2. – P. 41–46.

86. Zhang, Z. Improved Adam Optimizer for Deep Neural Networks /

Z. Zhang // 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS), Banff, 4-6 June 2018 : proceedings. – Banff: IEEE, 2019. – P. 1–4.

87. Каширина, И. Л. Исследование и сравнительный анализ методов оптимизации, используемых при обучении нейронных сетей / И. Л. Каширина, М. В. Демченко // Системный анализ и информационные технологии. – 2018. – № 4. – С. 123–132.

88. Олійник, А. О. Еволюційні обчислення та програмування : навчальний посібник / А. О. Олійник, С. О. Субботін, О. О. Олійник. – Запоріжжя : ЗНТУ, 2010. – 324 с.

89. Genetic Algorithm for Solution of the Problem of Optimal Location of the Distributed Electrical Networks / [D. Goncharenko, A. Oliinyk, I. Fedorchenko et al.] // 2020 10th International Conference on Advanced Computer Information Technologies (ACIT), Deggendorf, 16-18 September 2020 : proceedings. – Deggendorf: IEEE, 2020. – P. 380–385.

90. The Development of a Genetic Method for Predicting the Incidence of Diabetes / [A. Kharchenko, I. Fedorchenko, T. Fedoronchak et al.] // 2021 11th International Conference on Advanced Computer Information Technologies (ACIT), Deggendorf, 15–17 Sept. 2021 : proceedings. – Deggendorf: IEEE, 2021. – P. 708–713.

91. Development of the genetic algorithm for technical diagnostics of aircraft / [I. Fedorchenko, A. Oliinyk, A. Stepanenko et al.] // Information Technology and Interactions (IT&I-2020), Kyiv, Ukraine, 02–03 December 2020 : proceedings. – Kyiv: CEUR Workshop Proceedings, 2020. – Vol. 2833. – P. 11 – 25.

92. Jang, H. Analysis the Compressive Strength of Flue Gas Desulfurization Gypsum Using Artificial Neural Network / H. Jang, X. Shuli, S. So // Journal of Nanoscience and Nanotechnology. – 2020. – Vol. 20. – P. 485–490.

93. Development and research of a genetic method for the analysis and determination of the location of power grid objects / [I. Fedorchenko, A. Oliinyk,

S. Korniienko et al.] // Artificial Intelligence Scientific Journal. – 2020. – Vol. 25, № 1. – P. 20–42.

94. Modified genetic algorithm to determine the location of the distribution power supply networks in the city / [I. Fedorchenko, A. Oliinyk, J. Alsayaydeh et al.] // ARPN Journal of Engineering and Applied Sciences. – 2020. – Vol. 15, № 23. – P. 2850–2867.

95. Development of a Genetic Method for Determining the Optimal Location of Power Supply Objects / [I. Fedorchenko, A. Oliinyk, A. Stepanenko et al.] // 2020 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), Kharkiv, 6–9 October 2019 : proceedings. – Kharkiv: IEEE, 2020. – P. 47–52.

96. Особливості побудови рішень генетичного алгоритму в задачі розпізнавання образів захворювання пневмонії / [Є. М. Федорченко, С. К. Корнієнко, А. С. Харченко та ін.] // Реєстрація, зберігання і обробка даних. – 2020. – № 22. – С. 76-95.

97. Selection of sand models and identification of parameters using an enhanced genetic algorithm / [Y. Jin, Z. Yin, S. Shen et al.] // International Journal for Numerical and Analytical Methods in Geomechanics. – 2015. – Vol. 40. – P. 1219–1240.

98. A Novel Selection Approach for Genetic Algorithms for Global Optimization of Multimodal Continuous Functions / [E. Haq, I. Ahmad, A. Hussain et al.] // Computational Intelligence and Neuroscience. – 2019. – Vol. 2019. – P. 1–14.

99. Tang, P. Adaptive directed mutation for real-coded genetic algorithms / P. Tang, M. Tseng // Applied Soft Computing. – 2013. – Vol. 13. – P. 600-614.

100. Basak, A. A Rank based Adaptive Mutation in Genetic Algorithm / P. A. Basak // International Journal of Computer Applications. – 2020. – Vol. 175. – P. 49–55.

101. The Development of a Genetic Method to Optimize the Flue Gas Desulfurization Process / [I. Fedorchenko, A. Oliinyk, T. Fedoronchak et al.] // 3rd

International Workshop on Modern Machine Learning Technologies and Data Science (MoMLLeT+DS 2021), Lviv-Shatsk, 5 June 2021 : proceedings. – Lviv : CEUR-WS, 2021. – Vol. 2917. – P. 161 – 173.

102. The development of a genetic method to optimize the flue gas desulfurization process / [I. Fedorchenko, A. Oliinyk, A. Stepanenko et al.] // Artificial Intelligence. – 2021. – Vol. 26, – P. 59–73.

ДОДАТОК А
Технічне завдання

Вступ

Розроблюваний програмний продукт використовується для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Він являє собою набір бібліотек за допомогою яких можна створювати та досліджувати власні моделі залежності ступеня десульфурації методу мокрої абсорбційної очистки димових газів від використовуваних реагентів. ПП «Flue Gas Treatment 1.0» представлений у вигляді окремого віконного додатку завдяки якому можна обчислити прогноз вмісту діоксиду сірки у димовому газі після процесу десульфурації від обраних параметрів процесу за допомогою методу найближчого сусіда, випадкового лісу та опорних векторів, а також можна здійснювати тренування моделі та прогноз на основі багатосарового перцептрону, завантажуючи і застосовуючи власні експериментальні дані

A.1 Мета розробки і її призначення

Об'єкт дослідження – процес прогнозування результатів очищення викидів димових газів в атмосферу.

Предмет дослідження – методи прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

Мета роботи – дослідження та програмна реалізація методів прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

Метод дослідження – традиційні моделі (метод найближчого сусіда, випадковий ліс, метод опорних векторів), нейронні мережі, комбіновані методи (нейронні мережі і генетичні алгоритми).

Актуальність роботи полягає в тому, що останнім часом збільшилась кількості викидів діоксиду сірки (SO_2) в атмосферу, це обумовлено зростаючими показниками енергоспоживання у світі. Найбільшими

джерелами викидів SO_2 є теплові електростанції, які працюють на твердому та рідкому паливі, а також металургійні підприємства. Більшість підприємств використовують систему мокрої десульфурації димових газів (СМДДГ) як основний метод видалення SO_2 . В даний час при роботі МДДГ відсутній метод точного моделювання для прогнозування концентрації SO_2 на виході після процесу десульфурації. В результаті робочі параметри та робочий стан СМДДГ коригуються на основі досвіду експертів, що призводить до нераціональних витрат матеріалів та надмірних викидів.

А.2 Стислий зміст основних теоретичних посилок до розробки на основі проведених пошуків

За даними Nature Geoscience супутники NASA виявили 500 нових джерел забруднення повітря, близько 40 з них небезпечні діоксидом сірки (SO_2). Ця речовина вважається один з найчастіше зустрічаючих газів, який забруднює повітря та несе велику кількість небезпек для здоров'я людини і довкілля. Діоксид сірки пов'язана з різними респіраторними захворюваннями і підвищенням рівня смертності. Вдихання діоксиду сірки може викликати підвищений опір дихальних шляхів через звуження проходів у легенях. Основними джерелами викидів двоокису сірки є теплові електростанції, що працюють на твердому і рідкому паливі, а також металургійні підприємства. Отже, контроль вмісту SO_2 в димових газах від спалювання вугілля є ефективним методом щодо скорочення викидів в атмосферу. Викиди SO_2 можна зменшити за рахунок встановлення обладнання для сіркоочистки на новозбудованих та існуючих вугільних блоках, а також відповідних вимог до ефективності знесірчення. Щоб підвищити ефективність знесірчення і більш ефективно знизити викиди SO_2 , необхідно додатково оптимізувати систему управління знесірчення відповідно до потреб галузі. Точне встановлення взаємозв'язку між параметрами процесу і фактичної ефективністю сіркоочистки є основою для оптимізації системи управління сіркоочистки.

A.3 Підстави для розробки

Підставою для виконання розробки є завдання на дипломний проєкт на тему “Дослідження та програмна реалізація методів прогнозування результатів процесу десульфурації димових газів від діоксиду сірки”, затверджене наказом по НУ «Запорізька політехніка» № 435 від 5 листопада 2021 року.

A.4 Основні вимоги до програми, що розробляється, початковим даним і результату

A.4.1 Вимоги до функціональних характеристик

Вимоги до функціональних характеристик:

- повинна бути реалізована операція попередньої обробки даних (повинно бути видалення порожніх рядків, усунення викидів, видалення рядків, що повторюються, заповнення пропущених значень та нормалізація даних);
- повинна бути реалізована можливість здійснення обчислення прогнозу з використанням методу найближчого сусіда;
- повинна бути реалізована можливість здійснення обчислення прогнозу з використанням методу випадковий ліс;
- повинна бути реалізована можливість здійснення обчислення прогнозу з використанням методу опорних векторів;
- повинна бути реалізована можливість здійснення обчислення прогнозу з використанням моделі на основі багатошарового перцептрону;
- повинна бути реалізована можливість завантаження власних даних для навчання та тренування моделі багатошарового перцептрону;
- повинна бути реалізована можливість тренування моделі на основі багатошарового перцептрону;

- повинна бути реалізована можливість виводу графіка зміни значення середньої абсолютної похибки під час тренування багатошарового перцептрону;
- повинна бути реалізована можливість виводу графіка зміни значення середньої квадратичної похибки під час тренування багатошарового перцептрону;
- повинна бути реалізована можливість обробки помилок під час виконання програми та виведення повідомлення про помилку;
- повинна бути реалізована можливість виводу інформації про програмний продукт у окремому вікні.

А.4.2 Вимоги до надійності

ПП повинна виконувати наступні вимоги до надійності:

- захищати ПП від спроби завантажити файли, тип яких не відповідає вимогам системи;
- захищати ПП від помилкових дій користувача (наприклад користувач ввів невалідні дані).

ПП повинен нормально функціонувати при безперервній роботі комп'ютера. У разі виникнення збоїв в роботі апаратури, відновлення нормальної роботи ПП повинно проводитися після перезавантаження системи.

А.4.3 Умови експлуатації

Умови експлуатації:

- програмне забезпечення не потребує обслуговуючого персоналу;
- для користування програмним забезпеченням не потрібна додаткова кваліфікація.

A.4.4 Вимоги до складу та параметрів технічних засобів

Вимоги до складу та параметрів технічних засобів:

- платформа x64-based;
 - операційна система Microsoft Windows 10/8/7 (рекомендується встановити всі останні оновлення);
 - процесор з мінімальною тактовою частотою 1,8 ГГц.
- Рекомендується використовувати як мінімум двоядерний процесор;
- оперативна пам'ять від 2 ГБ (рекомендується 4 ГБ та мінімум 2,5 ГБ при використанні на віртуальній машині);
 - роздільна здатність екрану 1024x768 пікселів з 16-бітовим кольором;
 - мінімум 2 ГБ вільного місця на жорсткому диску.
 - середовище Python 3.7;
 - бібліотека NumPy та Pandas для роботи з масивами даних і формування датасетів;
 - бібліотека Scikit-learn для реалізації алгоритмів машинного навчання;
 - бібліотека Matplotlib для побудови графіків та візуалізації даних Matplotlib;
 - бібліотека Tkinter для розробки інтерфейсу користувача.

A.4.5 Вимоги до маркування та упакування

Програмне забезпечення не потребує додаткових дій при маркуванні та упакуванні.

A.4.6 Вимоги до транспортування та збереження

Вимоги до транспортування й зберігання аналогічні вимогам, які пропонуються до носія, на якому записана програма.

A.5 Вимоги до програмної документації

Склад програмної документації повинен включати в себе:

- технічне завдання;
- керівництво програміста;
- керівництво оператора.

A.6 Основні обмеження на установку і використання програмного продукту

Розроблюваний програмний продукт потребує інсталяції. Програмний продукт повинен використовуватись на комп'ютерах які задовільняють вимогам та параметрам технічних засобів. Програмний продукт може використовуватись користувачем у будь-який момент часу, не конфліктуючи з іншими програмами.

ДОДАТОК Б**Опис програмного продукту**

Б.1 Загальні відомості

Назва програмного продукту – «Flue Gas Treatment 1.0».

Розроблюваний програмний продукт використовується для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Він являє собою набір бібліотек за допомогою яких можна створювати та досліджувати власні моделі залежності ступеня десульфурації методу мокрої абсорбційної очистки димових газів від використовуваних реагентів. Програмний продукт представлений у вигляді окремого віконного додатку завдяки якому можна обчислити прогноз вмісту діоксиду сірки у димовому газі після процесу десульфурації від обраних параметрів процесу за допомогою методу найближчого сусіду, випадкового лісу та опорних векторів, а також можна здійснювати тренування моделі та прогноз на основі багатопараметричного перцептрону, завантажуючи і застосовуючи власні експериментальні дані.

Для коректної роботи ПП необхідні такі мінімальні системні вимоги:

- платформа x64-based;
- операційна система Microsoft Windows 10/8/7 (рекомендується встановити всі останні оновлення);
- процесор з мінімальною тактовою частотою 1,8 ГГц.

Рекомендується використовувати як мінімум двоядерний процесор;

- оперативна пам'ять від 2 ГБ (рекомендується 4 ГБ та мінімум 2,5 ГБ при використанні на віртуальній машині);
- роздільна здатність екрану 1024x768 пікселів з 16-бітовим кольором;
- мінімум 2 ГБ вільного місця на жорсткому диску.

У своїй роботі ПП використовує такі необхідні компоненти:

- середовище Python 3.7;
- бібліотека NumPy та Pandas для роботи з масивами даних і формування датасетів;

- бібліотека Scikit-learn для реалізації алгоритмів машинного навчання;
- бібліотека Matplotlib для побудови графіків та візуалізації даних Matplotlib;
- бібліотека Tkinter для розробки інтерфейсу користувача

Б.2 Функціональне призначення

Вимоги до функціональних характеристик:

- повинна бути реалізована операція попередньої обробки даних (повинно бути видалення порожніх рядків, усунення викидів, видалення рядків, що повторюються, заповнення пропущених значень та нормалізація даних);
- повинна бути реалізована можливість здійснення обчислення прогнозу з використанням методу найближчого сусіда;
- повинна бути реалізована можливість здійснення обчислення прогнозу з використанням методу випадковий ліс;
- повинна бути реалізована можливість здійснення обчислення прогнозу з використанням методу опорних векторів;
- повинна бути реалізована можливість здійснення обчислення прогнозу з використанням моделі на основі багатос шарового перцептрону;
- повинна бути реалізована можливість завантаження власних даних для навчання та тренування моделі багатос шарового перцептрону;
- повинна бути реалізована можливість тренування моделі на основі багатос шарового перцептрону;
- повинна бути реалізована можливість виводу графіка зміни значення середньої абсолютної похибки під час тренування багатос шарового перцептрону;

- повинна бути реалізована можливість виводу графіка зміни значення середньої квадратичної похибки під час тренування багатошарового перцептрону;
- повинна бути реалізована можливість обробки помилок під час виконання програми та виведення повідомлення про помилку;
- повинна бути реалізована можливість виводу інформації про програмний продукт у окремому вікні.

Б.3 Опис логічної структури

На рис. Б.1 представлена діаграма класів програмного продукту.

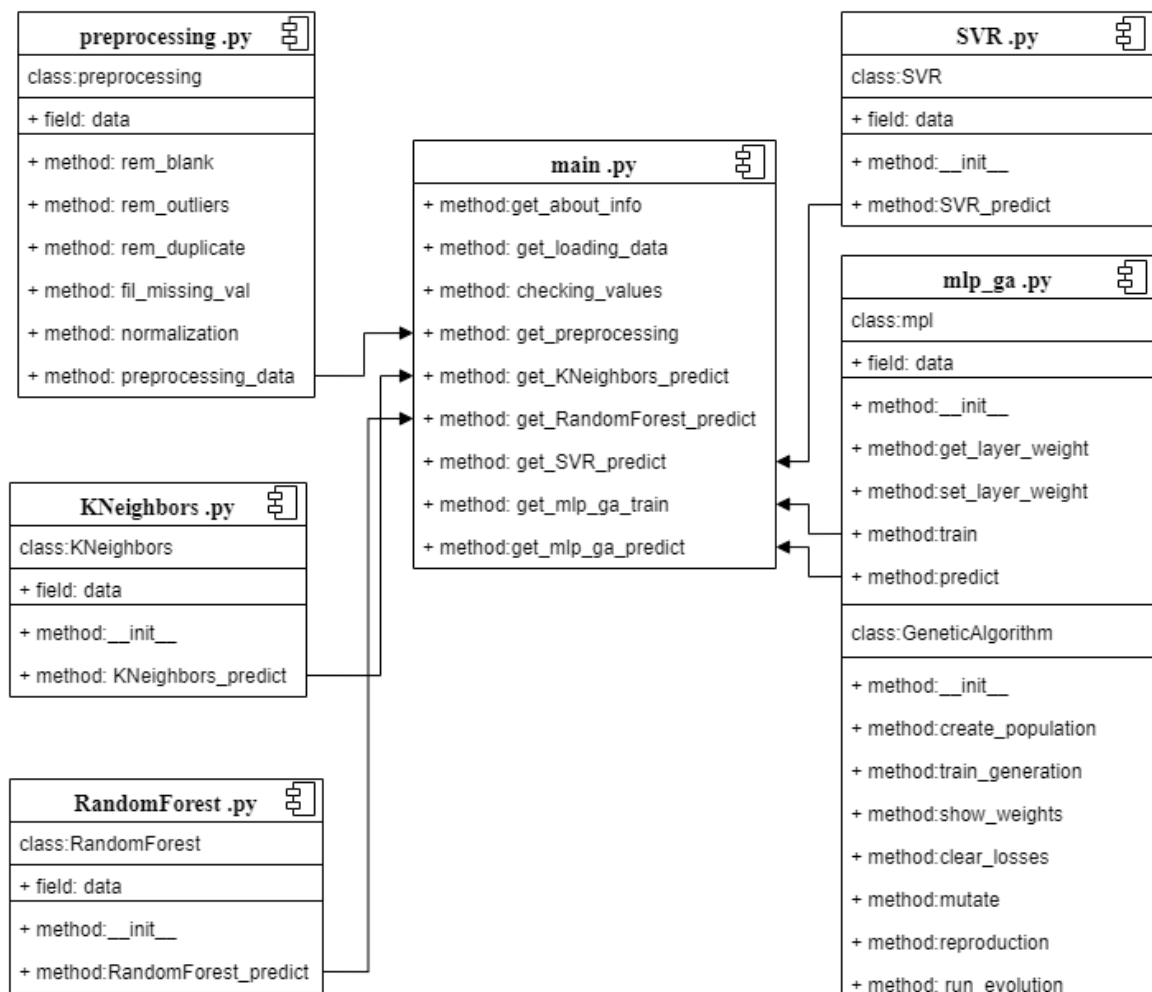


Рисунок Б.1 – Діаграма класів програмного продукту

Модуль `main.py` – головний модуль проєкту, що реалізує ініціалізацію ПП та елементів інтерфейсу користувача, він містить такі методи:

- `get_about_info` – здійснює вивід інформації про програмний продукт у окремому вікні;
- `checking values` – здійснює перевірку правильності значень введеним користувачем;
- `get_loading_data` – здійснює завантаження даних для навчання та тренування моделі;
- `get_preprocessing` – створює екземпляр класу `preprocessing` (модуля `preprocessing.py`) для здійснення попередньої обробки даних;
- `get_KNeighbors_predict` – створює екземпляр класу `KNeighbors` (модуля `KNeighbors.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу найближчого сусіда;
- `get_RandomForest_predict` – створює екземпляр класу `RandomForest` (модуля `RandomForest.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу випадковий ліс;
- `get_SVR_predict` – створює екземпляр класу `SVR` (модуля `SVR.py`) для здійснення обчислення прогнозу з використанням моделі на основі методу опорних векторів;
- `get_mlp_ga_train` – створює екземпляр класу `mlp_ga` (модуля `mlp_ga.py`) для тренування моделі на основі багатошарового перцептрону.
- `get_mlp_ga_predict` – створює екземпляр класу `mlp_ga` (модуля `mlp_ga.py`) для здійснення обчислення прогнозу з використанням моделі на основі багатошарового перцептрону.

Модуль `preprocessing.py` – реалізує попередню обробку даних, він виконує видалення порожніх рядків, усунення викидів, видалення рядків, що повторюються, заповнення пропущених значень та нормалізацію даних.

Модуль `KNeighbors.py` – реалізує прогнозування за допомогою методу найближчого сусіду;

Модуль RandomForest.py – реалізує прогнозування за допомогою методу випадковий ліс.

Модуль SVR.py – реалізує прогнозування за допомогою методу опорних векторів.

Модуль mlp_ga.py – містить два класи mlp та GeneticAlgorithm, він реалізує прогнозування за допомогою моделі на основі багат шарового перцептрон з використанням генетичного алгоритму для настройки вагових коефіцієнтів.

Для навчання на тестування моделей залежності ступеня сіркоочистки димових газів від використовуваних реагентів програмний продукт використовує експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів. Використовувані дані представляють собою файл у форматі csv, де кожний рядок файлу – це один проведений експеримент процесу десульфурації димових газів від діоксиду сірки, значення параметрів процесу розділені між собою комами.

У наборі даних є сім параметрів, останній з яких є цільовим:

- вміст SO₂ на вході (мг/м³);
- first_stage – об'єм газу на 1-у сходинку (м³/год);
- second_stage – об'єм газу на 2-у сходинку (м³/год);
- water – витрати води у абсорбері (м³/год);
- lime – витрати вапна (т/год);
- second_reagen – витрати вторинного реагенту (т/год);
- вміст SO₂ на виході (мг/м³).

Б.4 Вибір інструментарію розробки програмного продукту

Для розробки програмного продукту було обрано мову програмування R, бо вона володіє тими перевагами, що вона легко адаптується і може ефективно працювати у будь-якому середовищі. Крім того, з незначними змінами вона може працювати в будь-якій операційній системі і може бути

інтегрована з іншими мовами програмування. Python має багатий набір бібліотек та інструментів, що спрощує виконання завдань з аналізу даних, а також має широкий спектр інструментів із відкритим вихідним кодом для візуалізації та машинного навчання. Крім того, у Python є величезна база спільноти, де інженери та фахівці за даними можуть запитувати свої запити та відповідати на запитання інших.

Для розробки ПП було обрано середовище Spyder, бо воно має такі переваги як простота в освоєнні, кросплатформне з відкритим вихідним кодом, яке було спеціально створене для аналізу даних. Spyder об'єднує основні бібліотеки для науки про дані, такі як NumPy, SciPy, Matplotlib та IPython, крім того, воно може бути розширене за допомогою плагінів. Spyder містить такі функції, як текстовий редактор з підсвічуванням синтаксису, автозавершення коду та вивчення змінних, значення яких можна редагувати за допомогою графічного інтерфейсу користувача.

Б.5 Виклик і завантаження

Для запуску ПП необхідно запустити на робочому столі файл FlueGasTreatment1.0.exe, після чого відкриється головне вікно, яке представлено на рис. Б.2, також у консолі виводу інформації буде відображено повідомлення «Вітаємо! Програмний продукт «Flue Gas Treatment 1.0» завантажений і готовий до роботи».

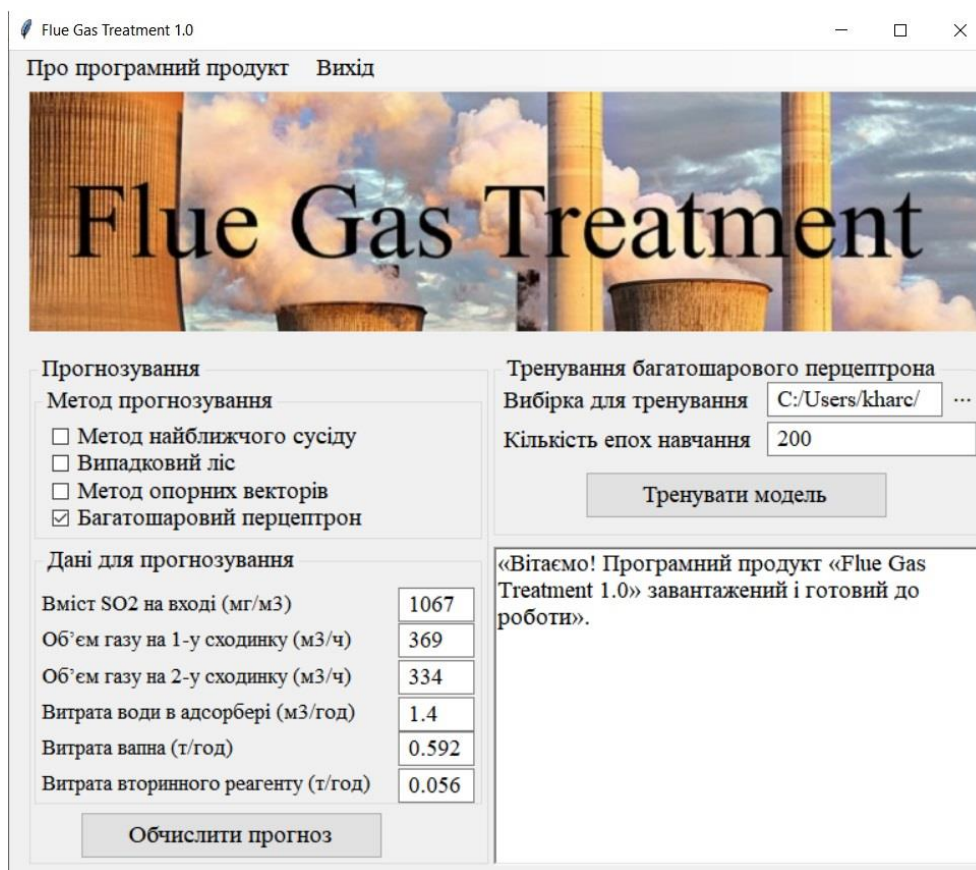


Рисунок Б.2 – Головне вікно ПП

Б.6 Приклади реалізації

Для того, щоб здійснити прогнозування результатів процесу десульфурзації димових газів від діоксиду сірки, спочатку необхідно обрати один або декілька методів із запропонованих (метод найближчого сусіду, випадковий лісу, метод опорних векторів та багатошаровий перцептрон). Потім необхідно ввести значення параметрів процесу, а саме:

- вміст SO₂ на вході (мг/ м³);
- об'єм газу на 1-у сходинку (м³/год);
- об'єм газу на 2-у сходинку (м³/год);
- витрати води у абсорбері (м³/год);
- витрати вапна (т/год);
- витрати вторинного реагенту (т/год).

Результатами прогнозування є значення вмісту діоксиду сірки (мг/м^3 , ціле число) на виході абсорбційної вежі у димовому газі після процесу десульфурації та час (секунди, число (крапка є роздільником десяткової частини)) виконання розрахунків. Результати прогнозування виводяться у консоль виводу інформації. Приклад виведення результатів прогнозування ПП приведено на рис. Б.3.

The screenshot shows the 'Flue Gas Treatment 1.0' application window. It has a title bar with the text 'Flue Gas Treatment 1.0' and standard window controls. Below the title bar is a menu bar with 'Про програмний продукт' and 'Вихід'. The main area features a large banner image of industrial smokestacks with the text 'Flue Gas Treatment' overlaid. Below the banner, there are two main sections: 'Прогнозування' (Prediction) and 'Тренування багатошарового перцептрона' (Training of a multilayer perceptron). The 'Прогнозування' section includes a 'Метод прогнозування' (Prediction method) dropdown menu with options: 'Метод найближчого сусіду' (unchecked), 'Випадковий ліс' (checked), 'Метод опорних векторів' (checked), and 'Багатошаровий перцептрон' (unchecked). Below this is a 'Дані для прогнозування' (Prediction data) table with input fields for various parameters. The 'Тренування' section includes a 'Вибірка для тренування' (Training data) dropdown menu with the path 'C:/Users/kharc/' and a 'Кількість епох навчання' (Number of training epochs) input field set to '100'. A 'Тренувати модель' (Train model) button is located below these fields. At the bottom of the 'Тренування' section, there is a summary of the used method and results: 'Використовуваний метод: Випадковий ліс' (Used method: Random forest), 'Вміст SO2 на виході: 920 мг/м3' (SO2 content at output: 920 mg/m3), 'Використовуваний метод: Метод опорних векторів' (Used method: Support vector method), 'Вміст SO2 на виході: 935 мг/м3' (SO2 content at output: 935 mg/m3), and 'Час виконання: 0.6593 сек.' (Execution time: 0.6593 sec.).

Дані для прогнозування	
Вміст SO2 на вході (мг/м^3)	1663
Об'єм газу на 1-у сходинку ($\text{м}^3/\text{ч}$)	344
Об'єм газу на 2-у сходинку ($\text{м}^3/\text{ч}$)	321
Витрата води в адсорбері ($\text{м}^3/\text{год}$)	1
Витрата вапна (т/год)	0.618
Витрата вторинного реагенту (т/год)	0.134

Обчислити прогноз

Тренування багатошарового перцептрона
Вибірка для тренування: C:/Users/kharc/ ...
Кількість епох навчання: 100
Тренувати модель

Використовуваний метод: Випадковий ліс
Вміст SO2 на виході: 920 мг/м^3
Використовуваний метод: Метод опорних векторів
Вміст SO2 на виході: 935 мг/м^3
Час виконання: 0.6593 сек.

Рисунок Б.3 – Вікно виводу результатів прогнозування

Для того, щоб здійснити навчання на тестування моделі залежності ступеня сіркоочистки димових газів від використовуваних реагентів на основі багатошарового перцептрону необхідно завантажити експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів у форматі csv та вказати кількість епох навчання в межах від одного до 1000.

Результатами навчання на тестування моделі багатошарового перцептрона є значення середньої абсолютної і середньої квадратичної похибки моделі на навчальній і тестовій вибірці відповідно та час виконання, а також графік зміни значення цих похибок під час тренування цієї моделі. Результати навчання на тестування моделі багатошарового перцептрона виводяться у консоль виводу інформації. Приклад виведення результатів прогнозування ПП приведено на рис. Б.4–Б.6.

The screenshot shows the 'Flue Gas Treatment 1.0' application window. It features a title bar, a menu bar with 'Про програмний продукт' and 'Вихід', and a main content area with a background image of industrial smokestacks. The interface is divided into several sections:

- Прогнозування (Forecasting):** Includes a 'Метод прогнозування' (Forecasting method) section with four options: 'Метод найближчого сусіду' (Nearest neighbor method), 'Випадковий ліс' (Random forest), 'Метод опорних векторів' (Support vector method), and 'Багатошаровий перцептрон' (Multi-layer perceptron), which is selected with a checkmark.
- Дані для прогнозування (Data for forecasting):** A table with input parameters and their values:

Вміст SO ₂ на вході (мг/м ³)	794
Об'єм газу на 1-у сходинку (м ³ /ч)	360
Об'єм газу на 2-у сходинку (м ³ /ч)	307
Витрата води в адсорбері (м ³ /год)	1.3
Витрата вапна (т/год)	0.09
Витрата вторинного реагенту (т/год)	0.08
- Тренування багатошарового перцептрона (Multi-layer perceptron training):** Includes a 'Вибірка для тренування' (Training sample) field with the path 'C:/Users/kharc/' and a 'Кількість епох навчання' (Number of training epochs) field with the value '50'. A 'Тренувати модель' (Train model) button is present.
- Тренування моделі багатошаровий перцептрон (Multi-layer perceptron model training):** Displays the results of the training:
 - Результати: (Results)
 - MAE на навчальній вибірці: 0.0550797
 - MAE на тестовій вибірці: 0.0789698
 - MSE на навчальній вибірці: 0.0052979
 - MSE на тестовій вибірці: 0.0133135
 - Час виконання: 14.4692 сек. (Execution time)

Buttons for 'Обчислити прогноз' (Calculate forecast) and 'Тренувати модель' (Train model) are located at the bottom of their respective sections.

Рисунок Б.4 – Вікно виводу результатів навчання на тестування моделі багатошарового перцептрона

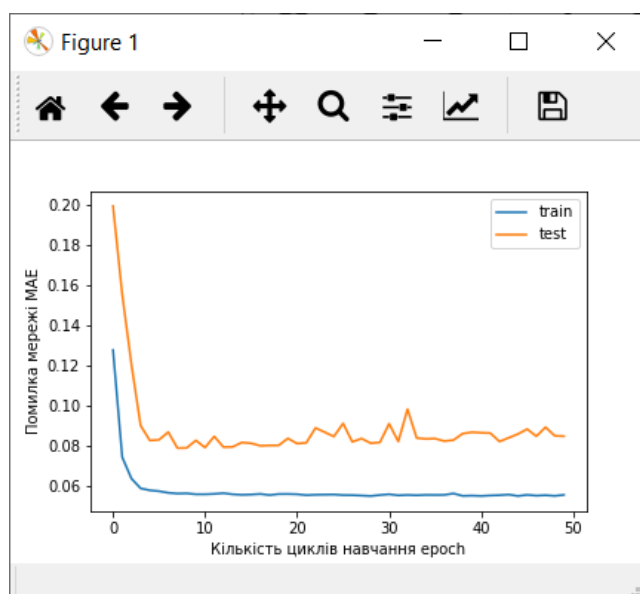


Рисунок Б.5 – Вікно виводу графіка зміни значення метрики MAE на навчальній і тестовій вибірці під час навчання

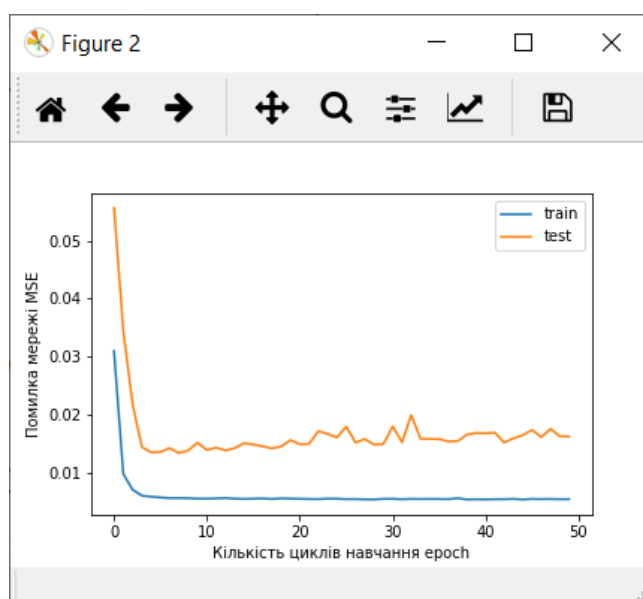


Рисунок Б.6 – Вікно виводу графіка зміни значення метрики MSE на навчальній і тестовій вибірці під час навчання

Для того щоб отримати керівництво по роботі з ПП, користувачу необхідно у верхньому лівому кутку у меню обрати «Про програмний продукт» і відкриється вікно з інформацією про ПП та довідка як з ним працювати.

Б.7 Вхідні та вихідні дані

Для навчання на тестування моделей залежності ступеня сіркоочистки димових газів від використовуваних реагентів ПП використовує експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів та значення кількості епох навчання, введеним користувачем. Експериментальні дані представляють собою файл у форматі csv, де кожний рядок файлу – це один проведений експеримент процесу десульфурації димових газів від діоксиду сірки, значення параметрів процесу розділені між собою комами.

У наборі даних є сім параметрів, останній з яких є цільовим:

- cont_entr – вміст SO_2 на вході (мг/м^3);
- first_stage – об'єм газу на 1-у сходинку ($\text{м}^3/\text{год}$);
- second_stage – об'єм газу на 2-у сходинку ($\text{м}^3/\text{год}$);
- water – витрати води у абсорбері ($\text{м}^3/\text{год}$);
- lime – витрати вапна (т/год);
- second_reagen – витрати вторинного реагенту (т/год);
- cont_out – вміст SO_2 на виході (мг/м^3).

На рис. Б.7 представлено приклад вхідних даних для навчання та тестування моделі на основі багатошарового перцептрон.

	A	B	C	D	E	F	G
1	cont_entr;first_stage;second_stage;water;lime;second_reagen;cont_out						
2	1537;331;311;1.5;0.756;0;1169						
3	1501;367;329;1.5;0.651;0;881						
4	1571;365;328;1.5;0.48;;994						
5	1591;375;322;0.7;0;0.195;1180						
6	1554;380;322;1.6;0.457;0.212;980						
7	1445;373;319;1.5;0.766;0.254;816						
8	1472;359;311;1.6;0.776;0.361;728						
9	1534;385;320;2;0.765;0.134;806						
10	1391;372;314;1.7;0.42;0;812						
11	1553;376;321;1.7;0.486;0;946						
12							

Рисунок Б.7 – Експериментальні дані роботи абсорбційної вежі мокрої десульфурації димових газів

Вихідними даними є значення середньої абсолютної і середньої квадратичної похибки моделі на навчальній і тестовій вибірці відповідно та час виконання, а також графік зміни значення цих похибок під час тренування цієї моделі.

Для здійснення прогнозу вмісту діоксиду сірки у димовому газі після процесу десульфурації ПП використовує в якості вхідних даних значення параметрів процесу десульфурації введеним користувачем (табл. Б.1):

Таблиця Б.1 – Вхідні дані для здійснення прогнозу вмісту діоксиду сірки у димовому газі після процесу десульфурації

Параметр	Тип даних
Вміст SO ₂ на вході (мг/м ³)	Ціле число
Об'єм газу на 1-у сходинку (м ³ /год)	Ціле число
Об'єм газу на 2-у сходинку (м ³ /год)	Ціле число
Витрати води у абсорбері (м ³ /год)	Число (крапка є роздільником десяткової частини)
Витрати вапна (т/год)	Число (крапка є роздільником десяткової частини)
Витрати вторинного реагенту (т/год)	Число (крапка є роздільником десяткової частини)

Вихідними даними прогнозу є значення вмісту діоксиду сірки (мг/м³, ціле число) на виході абсорбційної вежі у димовому газі після процесу десульфурації та час (секунди, число (крапка є роздільником десяткової частини)) виконання розрахунків.

ДОДАТОК В
Лістинги файлів роботи

B.1 Файл preprocessing.py

```

class Preprocessing:
    dataframe=0
    def __init__(self, df):
        self.df = df

    def rem_blank(self):
        df[0].fillna(method='ffill', inplace=True)
        df.dropna(how='all',subset=[2,3,8,9,11,12,13], inplace=True)
        df.dropna(how='all',subset=[2,3], inplace=True)
        df.dropna(how='all',subset=[11,12,13], inplace=True)
        df.dropna(how='all',subset=[3,11], inplace=True)
        df.dropna(how='all',subset=[11], inplace=True)
        df[13].fillna(0,inplace=True)
        for i in df.index:
            if type(df.loc[i,12])==str:
                print (df.loc[i].name)
                df.loc[2053,12]=0.174

    def fit_missing_val(self):
        Preprocessing.dataframe=df.iloc[:,[2,3,8,9,11,12,13]].values
        imp_median = SimpleImputer( missing_values=np.nan,strategy='median')
        imp_median.fit(Preprocessing.dataframe)
        Preprocessing.dataframe=imp_median.transform(Preprocessing.dataframe)

    def rem_duplicate(self):

        i=0
        while i<len(Preprocessing.dataframe):
            if Preprocessing.dataframe[i,2]==Preprocessing.dataframe[i,3]==Preprocessing.dataframe[i,4]==0:
                Preprocessing.dataframe=np.delete(Preprocessing.dataframe,i,0)
                i-=1
            i+=1
        #
        =====
        #Delete
        Preprocessing.dataframe=np.unique(Preprocessing.dataframe, axis=0)
        #
        =====

    def rem_outliers(self):
        i=0
        while i<=4:

            # IQR
            Q1 = np.percentile(Preprocessing.dataframe[:,i], 25, interpolation = 'midpoint')
            Q3 = np.percentile(Preprocessing.dataframe[:,i], 75, interpolation = 'midpoint')
            IQR = Q3 - Q1
            # Upper bound
            upper = np.where(Preprocessing.dataframe[:,i] >= (Q3+1.5*IQR))
            # Lower bound
            Preprocessing.dataframe=np.delete(Preprocessing.dataframe,upper,0)
            lower = np.where(Preprocessing.dataframe[:,i] <= (Q1-1.5*IQR))
            Preprocessing.dataframe=np.delete(Preprocessing.dataframe,lower,0)
            i+=1

    def normalization(self):
        scaler = preprocessing.MinMaxScaler()
        Preprocessing.dataframe=scaler.fit_transform(Preprocessing.dataframe)

```

В.2 Файл KNeighbors.py

```
class KNeighbors:
    dataframe=0
    def __init__(self, dataframe):
        self.df = dataframe

    def KNeighbors_predict(self, Xnew):
        X=np.column_stack(((dataframe[:,0],dataframe[:,2:5])))
        y=dataframe[:,1]
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
        calc_start = time.time()

        slr= KNeighborsRegressor(n_neighbors=5)
        slr.fit(X_train, y_train)
        y_train_pred = slr.predict(X_train)
        y_test_pred = slr.predict(X_test)
        print('MAE:', metrics.mean_absolute_error(y_train, y_train_pred))
        print('MSE:', metrics.mean_squared_error(y_train, y_train_pred))
        print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_pred)))
        print('R^2:', metrics.r2_score(y_train, y_train_pred))
        print('\n')
        print('MAE:', metrics.mean_absolute_error(y_test, y_test_pred))
        print('MSE:', metrics.mean_squared_error(y_test, y_test_pred))
        print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_pred)))
        print('R^2:', metrics.r2_score(y_test, y_test_pred))

        scaler = preprocessing.MinMaxScaler()
        Xnew=scaler.fit_transform(Xnew)
        ynew= slr.predict(Xnew)

        #ynew = preprocessing.MinMaxScaler().inverse_transform(ynew)
        #Xnew = preprocessing.MinMaxScaler().inverse_transform(Xnew)
        calc_end = time.time()
        print("X=%s, Predicted=%s" % (Xnew[0], ynew[0]))
        print(' Час виконання: ' + str(calc_end - calc_start) + 'сек.')
```

В.3 Файл RandomForest.py

```
class RandomForest:
    dataframe=0
    def __init__(self, dataframe):
        self.df = dataframe

    def RandomForest_predict(self, Xnew):
        X=np.column_stack(((dataframe[:,0],dataframe[:,2:5])))
        y=dataframe[:,1]
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
        calc_start = time.time()

        from sklearn.ensemble import RandomForestRegressor
        slr = RandomForestRegressor(max_depth=3, random_state=0).fit(X_train, y_train)
        y_train_pred = slr.predict(X_train)
        y_test_pred = slr.predict(X_test)

        print('MAE:', metrics.mean_absolute_error(y_train, y_train_pred))
        print('MSE:', metrics.mean_squared_error(y_train, y_train_pred))
        print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_pred)))
        print('R^2:', metrics.r2_score(y_train, y_train_pred))
```

```

print('\n')
print('MAE:', metrics.mean_absolute_error(y_test, y_test_pred))
print('MSE:', metrics.mean_squared_error(y_test, y_test_pred))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_pred)))
print('R^2:', metrics.r2_score(y_test, y_test_pred))

scaler = preprocessing.MinMaxScaler()
Xnew=scaler.fit_transform(Xnew)
ynew= slr.predict(Xnew)

#ynew = preprocessing.MinMaxScaler().inverse_transform(ynew)
#Xnew = preprocessing.MinMaxScaler().inverse_transform(Xnew)
calc_end = time.time()
print("X=%s, Predicted=%s" % (Xnew[0], ynew[0]))
print( ' Час виконання: ' + str(calc_end - calc_start) + 'сек.')

```

B.4 Файл SVR.py

```

class SVR:
    dataframe=0
    def __init__(self, dataframe):
        self.df = dataframe
    def SVR(self, Xnew):
        X=np.column_stack(((dataframe[:,0],dataframe[:,2:5])))
        y=dataframe[:,1]
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
        calc_start = time.time()
        from sklearn.ensemble import RandomForestRegressor
        slr = svm.SVR().fit(X_train, y_train)
        y_train_pred = slr.predict(X_train)
        y_test_pred = slr.predict(X_test)
        print('MAE:', metrics.mean_absolute_error(y_train, y_train_pred))
        print('MSE:', metrics.mean_squared_error(y_train, y_train_pred))
        print('RMSE:', np.sqrt(metrics.mean_squared_error(y_train, y_train_pred)))
        print('R^2:', metrics.r2_score(y_train, y_train_pred))
        print('\n')
        print('MAE:', metrics.mean_absolute_error(y_test, y_test_pred))
        print('MSE:', metrics.mean_squared_error(y_test, y_test_pred))
        print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_test_pred)))
        print('R^2:', metrics.r2_score(y_test, y_test_pred))
        scaler = preprocessing.MinMaxScaler()
        Xnew=scaler.fit_transform(Xnew)
        ynew= slr.predict(Xnew)
        #ynew = preprocessing.MinMaxScaler().inverse_transform(ynew)
        #Xnew = preprocessing.MinMaxScaler().inverse_transform(Xnew)
        calc_end = time.time()
        print("X=%s, Predicted=%s" % (Xnew[0], ynew[0]))
        print( ' Час виконання: ' + str(calc_end - calc_start) + 'сек.')

```

B.5 mpl_ga.py

```

class mpl:
    def __init__(self):
        model = Sequential()
        model.add(Dense(7, input_dim=4, kernel_initializer='normal', activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(6, activation='relu'))
        model.add(Dense(1, activation='relu'))

```



```

        member.set_layer_weight(i, new_weight)
def reproduction(self):
    popul_idx = [i for i in range(len(self.popul))]
    for i in range(len(self.population)):
        #selects two parents probabilistic accroding to the fitness
        if sum(self.norm_acc) != 0:
            par1 = np.random.choice(popul_idx, p = self.norm_acc)
            par2 = np.random.choice(popul_idx, p = self.norm_acc)
        else:
            # if there are no "best" parents choose randomly
            par1 = np.random.choice(popul_idx)
            par2 = np.random.choice(popul_idx)
        # picking random midpoint for crossing over name/DNA
        par1_weights = self.population[par1].give_weights()
        par2_weights = self.population[par2].give_weights()
        mid_point = np.random.choice([i for i in range(len(par1_weights))])
        # adding DNA-Sequences of the parents to final DNA
        self.children_population_weights.append(par1_weights[:mid_point] + par2_weights[mid_point:])
    # old population gets the new and proper weights
    for i in range(len(self.population)):
        for j in range(len(self.children_population_weights)):
            self.population[i].load_layer_weights(self.children_population_weights[j])
def run_evolution(self):
    for epis in range(self.gener):
        self.clear_losses()
        self.train_generation()
        self.predict()
        if epis != self.gener -1:
            self.normalize()
            self.reproduction()
            self.mutate()
        else:
            pass
    for a in range(self.gener):
        for mem in self.population:
            plt.plot(mem.acc_history)

```

ДОДАТОК Д
Слайди презентації

Національний університет «Запорізька політехніка»
Кафедра програмних засобів

Дослідження та програмна реалізація методів прогнозування результатів процесу десульфурації димових газів від діоксиду сірки

Виконала:
студентка групи КНТ-120м

А.С. Харченко

Керівник:
доцент, д.т.н, професор

А.О. Олійник

Рисунок Д.1 – Слайд № 1

2

Мета, об'єкт, предмет дослідження

Об'єкт дослідження – процес прогнозування результатів очищення викидів димових газів в атмосферу.

Предмет дослідження – методи прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

Мета роботи – дослідження та програмна реалізація методів прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

Актуальність та проблематика роботи полягає в тому, що останнім часом збільшилась кількість викидів діоксиду сірки (SO_2) в атмосферу, це обумовлено зростаючими показниками енергоспоживання у світі. В даний час при роботі з системою мокрої десульфурації відсутній метод точного моделювання для прогнозування концентрації SO_2 на виході після процесу. В результаті робочі параметри та стан системи десульфурації коригуються на основі досвіду експертів, що призводить до нераціональних витрат матеріалів та надмірних викидів.

Рисунок Д.2 – Слайд № 2

Постановка завдання до роботи

Проаналізувавши принцип роботи методу мокрої абсорбційної очистки димових газів, було встановлено, що узагальнену модель залежності ступеня десульфурації від використовуваних реагентів з деяким припущенням, можна привести у вигляді формули:

$$K_{des} = f(x_{con}, x_{first_stage}, x_{second_stage}, x_{water}, x_{lime}, x_{second_reagent})$$

де x_{con} – вміст SO_2 на вході (мг/м³);
 x_{first_stage} – об'єм газу на 1-у сходинку;
 x_{second_stage} – об'єм газу на 2-у сходинку;
 x_{water} – витрати води у абсорбері;
 x_{lime} – витрати вапна;
 $x_{second_reagent}$ – витрати вторинного реагенту.

Рисунок Д.3 – Слайд № 3

Огляд існуючих методів

Можливості	Метод найближчого сусіду	Випадковий ліс	Метод опорних векторів	Генетичний алгоритм	Штучна нейронна мережа
Простота реалізації	Так	Ні	Ні	Так	Ні
Можливість розпаралелювання	Ні	Так	Так	Так	Так
Схильність до перенавчання	Ні	Ні	Ні	Ні	Так
Стійкість до шуму та викидів	Ні	Так	Ні	Ні	Так
Легкість інтерпретації одержаних результатів	Так	Ні	Ні	Так	Ні
Адаптація до змін у вирішуваному завданні	Ні	Ні	Ні	Так	Так
Автоматично обробляє пропущені значення	Ні	Так	Ні	Ні	Ні
Обчислювально витратний зі збільшенням кількості атрибутів	Так	Так	Так	Ні	Ні

Рисунок Д.4 – Слайд № 4

Модифікований генетичний метод

- Ініціалізація початкової популяції хромосомами, що містять інформацію про значення вагових коефіцієнтів мережі.
- Оцінювання хромосом поточної популяції за формулою:

$$F = \frac{1}{2C_i} \sum_{i=1}^{C_i} (O_i - R_i)^2$$

де C_i – потужність множини навчальних пар;

O_i – отримане за допомогою мережі значення вихідного нейрона при i -му наборі навчання;

R_i – необхідне значення вихідного нейрона при i -му наборі навчання

- Відбір хромосом-батьків для схрещування за лінійним ранжуванням.
- SBX – схрещування.

Рисунок Д.5 – Слайд №5

Модифікований генетичний метод

- Ймовірність мутації хромосоми розраховується за формулою:

$$\beta_{mut} = \begin{cases} 0.5 \frac{F_{max}(e) - F(P(e))}{F_{max}(e) - F(e)}, & \text{if } F(P(e)) \geq \bar{F}(e) \\ 0.5, & \text{if } F(P(e)) < \bar{F}(e) \end{cases}$$

де $F_{max}(e)$ – найбільше значення фітнес-функції поточної популяції;

$F(e)$ – середнє значення фітнес-функції поточної популяції;

$F(P(e))$ – значення фітнес-функції хромосоми.

Якщо ген p_j піддається мутації, то його нове змінене значення p_j^{new} визначається за формулою:

$$p_j^{new} = p_j + b(o_j - p_j) \left(1 - \frac{e}{e_{max}} \right)^{\xi}$$

де b – випадкове згенероване число з інтервалу $[0, 1]$;

o_j – випадковим чином генерується з множини $\{p_{min}, p_{max}\}$;

e – номер поточного покоління;

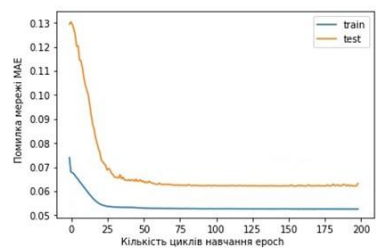
ξ – параметр уточнення;

e_{max} – максимальне число поколінь.

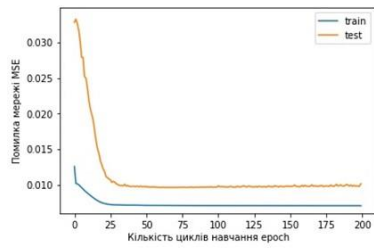
- Селекція, яка проріджує популяцію і відсіює особин з високою ймовірністю мутації.

Рисунок Д.6 – Слайд № 6

Приклад реалізації



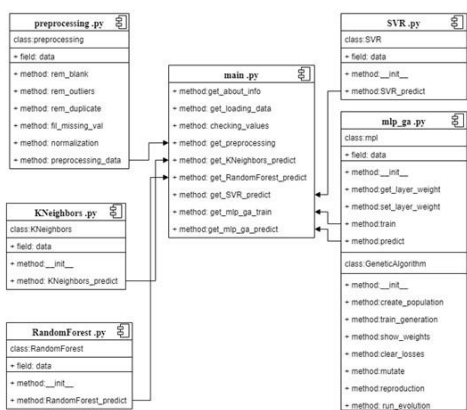
Графік зміни значення метрики MAE для багаторовового перцептрона з двома прихованими шарами та генетичним методом з адаптивною мутацією



Графік зміни значення метрики MSE для багаторовового перцептрона з двома прихованими шарами та генетичним методом з адаптивною мутацією

Рисунок Д.7 – Слайд № 7

Схема структури ПП



Діаграма класів програмного продукту

Рисунок Д.8 – Слайд № 8

Вибір мови програмування

Критерії порівняння	R	C++	Python	Java	Julia
Ліцензія	GNU GPL	—	Python Software Foundation License	GNU GPL	MIT
Простота навчання та написання коду	—	+	+	+	+
Кросплатформеність	+	+	+	+	+
Можливість обробляти операції з великим обсягом даних	+	+	+	+	+
Паралельне програмування	+	+	+	+	+
Наявність бібліотек для машинного навчання	+	±	+	±	+
Інтеграція з іншими мовами	+	+	+	+	±
Наявність зрозумілої та повної документації	+	+	+	+	+
Висока візуалізація даних	+	—	+	—	+
Підвищена продуктивність	±	+	+	+	+
Великий список предметно-орієнтованих пакетів з відкритим вихідним кодом	+	+	+	+	—
Сильна підтримка спільноти	+	+	+	+	—

Рисунок Д.9 – Слайд № 9

Вибір середовища розробки

Критерії порівняння	PyCharm	Spyder	JupyterLab	Visual Studio Code
Ліцензія	Apache License 2.0	MIT	—	MIT
Кросплатформеність	+	+	+	—
Автозавершення коду	+	+	+	+
Підтримка репозиторіїв	+	+	—	+
Простота встановлення	—	+	+	—
Перевірка та аналіз коду	+	+	+	+
Швидкий та безпечний рефакторинг коду	+	+	—	+
Наявність інтерактивної консолі	+	+	+	+
Наявність зручного користувацького інтерфейсу	+	+	—	+
Швидкий та зручний пошук	+	+	+	+
Підсвічування синтаксису	+	+	+	+
Підтримка наукових пакетів	+	+	+	+
Наявність функції статичного аналізу коду	+	+	+	+
Потребує малої кількості пам'яті для зберігання	—	+	+	+

Рисунок Д.10 – Слайд № 10

Інтерфейсні вікна програмного продукту

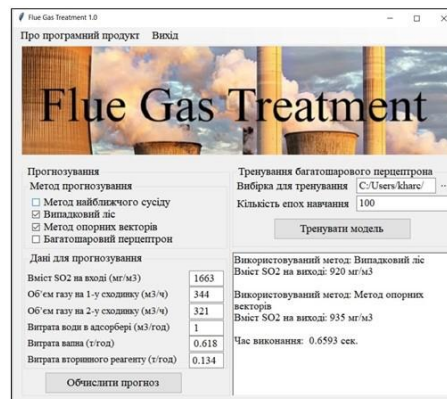


Рисунок Д.11 – Слайд № 11

Інтерфейсні вікна програмного продукту

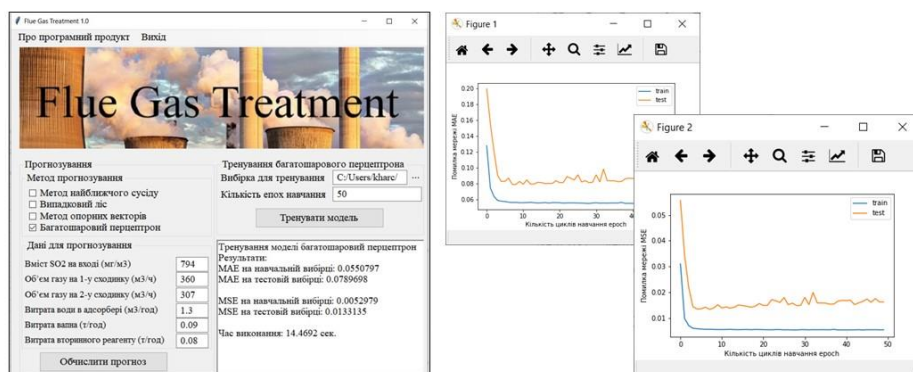


Рисунок Д.12 – Слайд № 12

Результати

Таблиця 1 – Порівняльний аналіз результатів розроблених моделей прогнозування результатів процесу десульфурації димових газів від діоксиду сірки

Назва методу	MSE	MAE	Час виконання, с
Метод найближчого сусіду	0.0153	0.0897	362
Випадковий ліс	0.0152	0.0954	315
Метод опорних векторів	0.0131	0.0842	374
Багатошаровий перцептрон з одним прихованим шаром	0.0149	0.0940	764
Багатошаровий перцептрон з двома прихованими шарами	0.0129	0.0794	867
Багатошаровий перцептрон з двома прихованими шарами та розробленим ГА	0.0101	0.0677	489

Рисунок Д.13 – Слайд № 13

Висновок

Під час виконання роботи було виконано дослідження та програмна реалізація методу прогнозування результатів процесу десульфурації димових газів від діоксиду сірки.

Запропоновано модифікований генетичний метод для настройки вагових коефіцієнтів штучної нейронної мережі для прогнозування результатів процесу десульфурації димових газів від діоксиду сірки. Принципова відмінність запропонованого генетичного методу від існуючих аналогів полягає у використанні адаптивної мутації, яка використовує в процесі своєї роботи рівень розвитку популяції.

Розроблено ПП для прогнозування результатів процесу десульфурації димових газів від SO_2 на основі створеного методу.

Було проведено порівняльний аналіз розробленого генетичного методу та відомих, в результаті чого було встановлено, що він дає найменшу помилку MAE та MSE.

Отримані результати дозволяють запропонувати ефективний метод для підвищення точності прогнозування та оптимізації експлуатаційних витрат процесу десульфурації димових газів від діоксиду сірки, внаслідок чого знизити викиди SO_2 в атмосферу від вугільних електростанцій.

Рисунок Д.14 – Слайд № 14

Наукові публікації та досягнення у творчих конкурсах

За результатами досліджень було опубліковано 15 наукових праць, з яких 3 статті і 8 доповідей на конференціях, що індексується в міжнародній наукометричній базі Scopus, та 4 статті у наукових фахових журналах України.

Посіла II місце у вузівському конкурсі на кращу студентську наукову роботу 2018/2019 н. р. (секція «Комп'ютерні науки, програмні засоби, системний аналіз та інформаційні технології», підсекція «Комп'ютерні науки»).

Посіла I місце у вузівському конкурсі на кращу студентську наукову роботу 2019/2020 н. р. (секція «Комп'ютерні науки, програмні засоби, системний аналіз та інформаційні технології», підсекція «Комп'ютерні науки»).

Посіла I місце у вузівському конкурсі на кращу студентську наукову роботу 2020/2021 н. р. (секція «Комп'ютерні науки, програмні засоби, системний аналіз та інформаційні технології», підсекція «Комп'ютерні науки»).

Отримала диплом III ступеня переможниці Всеукраїнського конкурсу студентських наукових робіт зі спеціальності «Комп'ютерні науки» в 2020/2021 н. р.



Рисунок Д.15 – Слайд № 15