Вінницький національний технічний університет

Факультет комп'ютерних систем і автоматики

Кафедра системного аналізу та інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року »

Виконав: студент 2 курсу, групи ІСТ-20м
спеціальності 126 – «Інформаційні системи та технології»
_____ Поліщук Д.А.

Керівник: к.т.н., доц. каф. САІТ
_____Ящолт А.Р.
«___» _____ 2021 р.

Опонент: д.т.н., проф. каф. АІІТ
_____ Бісікало О.В.
«___» _____ 2021 р.

Допущено до захисту

Завідувач кафедри САІТ

_____д.т.н., проф. Мокін В. Б.

«___» _____ 2021 р.

Вінниця ВНТУ – 2021 рік

Вінницький національний технічний університет
Факультет комп'ютерних систем і автоматики
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – ІІ-й (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 126 Інформаційні системи та технології
Освітньо-професійна програма – Інформаційні технології аналізу даних та зображень

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ

_____ д.т.н., проф. Мокін В. Б.

«___» _____ 2021 р.

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Поліщуку Дмитру Анатолійовичу

1. Тема роботи: «Інформаційна технологія аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року», керівник роботи: Ящолт А.Р., к.т.н., доц. каф. САІТ,
затверджені наказом закладу вищої освіти від «___»__ _____2021 року №__
2. Строк подання студентом роботи «___»__ _____2021 року
3. Вихідні дані до роботи:
Набір даних, який містить інформацію про моніторинг змін завислих речовин у річковій воді у річці Південний Буг
4. Зміст текстової частини:
- обґрунтування проблеми створення інформаційної технології аналізу та прогнозування завислих речовин у річковій воді;
- формалізація моделі інформаційної технології аналізу та прогнозування завислих речовин у річці;
- розробка інформаційної технології аналізу та прогнозування завислих речовин у річці;
- економічна частина.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень):
- загальна структура інформаційної системи аналізу та прогнозування завислих речовин у річці Південний Буг;
- діаграма важливості ознак;
- прогнозування навчальних даних;
- прогнозування тестових даних;
- прогнозування валідаційних даних;
- найкраща модель прогнозування.

6. Консультанти розділів МКР

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Ратушняк О.Г., к.т.н., доц. каф. ЕПВМ		
2-3	Зав. каф. САІТ Мокін В.Б., д.т.н., проф.		

7. Дата видачі завдання «___»_____2021 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів МКР	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області	09.2021	
2	Обґрунтування проблеми створення інформаційної технології аналізу та прогнозування завислих речовин у річковій воді	09.2021	
3	Розробка інформаційної технології	09.2021	
4	Економічна частина	10.2021	
5	Реалізація інформаційної технології	10.2021	
6	Аналіз результатів прогнозування	11.2021	
7	Оформлення матеріалів до захисту МКР	12.2021	

Студент _____ Поліщук Д.А.

Керівник роботи _____ Яцолт А.Р.

АНОТАЦІЯ

УДК 004.09

Поліщук Д.А. Інформаційна технологія аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 рок. Магістерська кваліфікаційна робота зі спеціальності 126 – інформаційні системи та технології, освітньо-професійна програма – інформаційні технології аналізу даних та зображень. Вінниця: ВНТУ, 2021. 114 с.

На укр. мові. Бібліогр.: 21 назв; рис. 77; табл.: 10.

В магістерській кваліфікаційній роботі звернено увагу на проблему моніторингу якості води у річках України. Запропоновані технології, які допоможуть аналізувати та прогнозувати концентрації завислих речовин у річковій воді за заданий період часу. Здійснено реалізацію інформаційної технології прогнозування концентрації завислих речовин у річковій воді. Об’єкт досліджень є процес прогнозування вмісту завислих речовин у водах річки, дані для аналізу отримано із датасету з сервісу Kaggle. Галузь застосування – екологічні установи та організації, які займаються аналізом якості водних ресурсів.

Ілюстративна частина складається з 6 плакатів із результатами моделювання.

У розділі економічної частини розглянуто питання про доцільність розробки та впровадження інформаційної технології аналізу та прогнозування концентрації завислих речовин у річковій воді.

Ключові слова: інформаційна система, моніторинг, якість водних ресурсів.

ABSTRACT

Polishchuk D.A. Information technology of analysis and forecasting of changes in the concentration of suspended solids in the Southern Bug River in the 4th quarter of 2021. Master's qualification work in the specialty 126 – information systems and technologies, educational and professional program – information technologies of data and image analysis. Vinnytsia: VNTU, 2021. – 114 p.

In Ukrainian language. Bibliographer: 21 titles; fig. 77; table: 10.

In the master's qualification work attention is paid to the problem of water quality monitoring in the rivers of Ukraine. Technologies that will help to analyze and predict the concentration of suspended solids in river water for a given period of time are proposed. The implementation of information technology for forecasting the concentration of suspended solids in river water has been implemented. The object of research is the process of predicting the content of suspended solids in river waters, the data for analysis were obtained from a dataset from the Kaggle service. Scope - environmental institutions and organizations that analyze the quality of water resources.

The illustrative part consists of 6 posters with simulation results.

In the section of the economic part the question of expediency of development and introduction of information technology of the analysis and forecasting of concentration of the suspended substances in river water is considered.

Key words: information system, monitoring, quality of water resources.

ЗМІСТ

ВСТУП 4

1 ОБҐРУНТУВАННЯ ПРОБЛЕМІ СТВОРЕННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЗАВИСЛИХ РЕЧОВИН У РІЧКОВІЙ ВОДІ 6

1.1 Аналіз предметної області 6

1.2 Огляд і опис методів та підходів щодо прогнозування завислих речовин у водах річки 12

1.3 Обґрунтування вибору бібліотеки Scikit Learn 20

1.4 Висновки 29

2 ФОРМАЛІЗАЦІЯ МОДЕЛІ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЗАВИСЛИХ РЕЧОВИН У РІЧЦІ 30

2.1 Ідея інформаційної системи аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року 30

2.2 Методологія інформаційної технології 41

2.3 Формалізація інформаційної моделі аналізу та прогнозування змін концентрації завислих речовин у річці Південний 60

2.4 Огляд та вибір технологій для інформаційної системи аналізу та прогнозування завислих речовин у річковій воді 66

2.5 Висновки 71

3 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЗАВИСЛИХ РЕЧОВИН У РІЧЦІ 72

3.1 Модель об'єкта 72

3.2 Висновки 80

4 ЕКОНОМІЧНА ЧАСТИНА 81

4.1 Оцінювання комерційного потенціалу розробки 81

4.2 Прогнозування витрат на виконання науково-дослідної роботи 87

4.3 Розрахунок економічної ефективності науково-технічної розробки 92

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності . 93

4.5 Висновки 96

ВИСНОВКИ 97
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 99
Додаток А - Технічне завдання 100
Додаток Б - Протокол перевірки кваліфікаційної роботи 102
Додаток Г - Фрагмент лістингу програми 103
Додаток Д - Ілюстративна частина 114

ВСТУП

Актуальність теми. Антропогенний вплив на природу невпинно призводить до її змін та погіршення, якості води, в тому числі і різкого збільшення завислих речовин у річках. При наявній державній системі моніторингу якості цих вод є можливість отримати доступ до накопичених моніторингових даних та спробувати з використанням сучасних інформаційних технологій проаналізувати та спрогнозувати стан завислих речовин у річках України. Усі ці процеси призводять до підтримки стану водних ресурсів у нормі.

Мета і завдання роботи. Метою даної роботи є збільшення точності прогнозування вмісту завислих речовин у водних ресурсах за допомогою інструментів інформаційних технологій.

У роботі планується вирішити наступні завдання:

- дослідити дані моніторингу якості вод у річках України та провести аналіз (розвідувальний аналіз)цих даних для завислих речовин у річці;
- виконати аналіз даних та прогнозування завислих речовин у річковій воді;
- розробити інформаційну технологію для прогнозування завислих речовин у річковій воді.

Об’єктом дослідження магістерської кваліфікаційної роботи є процес прогнозування вмісту завислих речовин у водах річки, дані для аналізу отримано із датасету з сервісу Kaggle.

Предметом дослідження магістерської кваліфікаційної роботи є методи обробки і прогнозування, які ґрунтуються на сучасних інформаційних технологіях щодо прогнозування та аналізу стану вод у річках по завислим речовинам.

Новизна одержаних результатів. Дістала подальший розвиток спеціалізована модель аналізу та прогнозування завислих речовин у річковій воді розрахунку, за рахунок обробки існуючих даних моніторингу якості води

у річках дозволить робити точні прогнози по даним із створів вище по течії про стан та рівень завислих речовин у створах нище по течії річки.

Практичне значення роботи полягає у тому, що за даними створів що вище по течії можна отримати точні пронози по завислим речовина у створах що ниже по течії в певний часовий період.

Апробація результатів магістерської кваліфікаційної роботи. Результати кваліфікаційної роботи доповідались на Всеукраїнській науково-практичній інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2021-2022 рр.).

Публікації результатів магістерської кваліфікаційної роботи. Під час виконання магістерської кваліфікаційної роботи опубліковано тези у збірнику матеріалів конференції «Молодь в науці: дослідження, проблеми, перспективи» (Вінниця, 2021-2022 рр.)" [1].

1 ОБҐРУНТУВАННЯ ПРОБЛЕМИ СТВОРЕННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЗАВИСЛИХ РЕЧОВИН У РІЧКОВІЙ ВОДІ

1.1 Аналіз предметної області

Оцінка водних об'єктів здійснюється на основі використання гідрологічних та гідрохімічних характеристик, які в сукупності визначають його еколого-гідрохімічний стан.

При аналізі формування водного та гідрохімічного режиму визначаються основні фактори, що суттєво впливають на зміну того чи іншого інгредієнта водойми. До таких відносяться:

- зміна річкового стоку;
- вплив підземного та дренажного стоку;
- надходження дренажних вод із гідромеліоративних систем;
- надходження вод із забрудненими речовинами з площі водозабору у вигляді промислових та
- побутових стоків;
- седиментація зважених у воді частинок;
- збовтування донних відкладень;
- безповоротне водоспоживання;
- трансформація речовин унаслідок фізико-хімічних, біологічних перетворень;
- випаровування води з поверхневих водойм [2].

Математичне моделювання біологічних процесів є однією із складових екологічного прогнозу У цій сфері широко використовується імітаційне моделювання

Імітаційне моделювання – це тип моделювання, який включає групу

методів, що дозволяють створювати моделі та описувати процеси, що відбуваються в екосистемах. Сама екосистема замінюється імітатором. З ним проводяться експерименти для отримання інформації про системі.

За допомогою класифікації Е. А. Зілова, можна визначили найбільш популярні моделі цього типу з високою прогностичною ефективністю. До таких відносяться:

1. Комплексні динамічні моделі;

2. Прості статистичні розрахункові моделі.

Вище зазначені моделі базуються на диференціальних рівняннях. Складність їх вирішення у тому, що немає систематичних правил виведення даних рівнянь. Їх складають на основі напівемпіричних закономірностей, міркувань та аналогій [3]. До того ж, при роботі з системами з десятків диференціальних рівнянь виключити помилки дуже складно.

Зараз математичне моделювання та експерименти з модельними екосистемами не є єдиним методом прогнозування антропогенного на водні об'єкти. Однак, у практиці цей метод успішно застосовується і може бути додатковим та надійним джерелом інформації про водоймища і дозволить з точністю спрогнозувати їх стан.

Суспензії – окремий випадок суспензії. Вони є гетерогенними системами (грубодисперсними системами з твердою дисперсною фазою та рідкою фазою – дисперсійним середовищем), на відміну від розчинених речовин, що утворюють гомогенні системи – справжні розчини. Суспензії не відносять до колоїдних систем, оскільки у разі суспензій тверда фаза піддається осіданню під дією сили тяжіння – седиментації – що, поряд з розміром частинки (розмір частинок у суспензіях більше 1 мкм, а в колоїдних системах – від 0,01 мкм) , і відрізняє суспензії від колоїдних розчинів.

Зважені речовини поділяють на органічні та неорганічні. Поширені різновиди завислих частинок - нерозчинний діоксид кремнію (річковий пісок), мул, планктон.

Завислі речовини розрізняють типологію за складом:

- Мінеральні частки;

Річковий пісок разом з алюмосилікатами та іншими нерозчинними неорганічними оксидами та солями становить більшу частину завислих мінеральних частинок. У воду частинки такого роду потрапляють у процесі водної чи вітрової ерозії берегів та прибережних ґрунтів, а також шляхом абразії берегів водойми.

- Органічні частки;

Органічні зважені частки – це домішки, які у водойми внаслідок промислової діяльності підприємств м'ясної, рибної, молочної продукції; заводів із виробництва різних органічних речовин, наприклад, полімерів.

Органічні частинки потрапляють у водоймища зі стічними водами побутових відходів. У таких стоках багато мікропластику - найдрібніших частинок, що утворюються при розпаді пластмасових виробів. Такі суспензії можуть становити небезпеку для природи. Справа в тому, що пластикова крихта не схильна до біодеградації, плаває на поверхні води і в її приповерхневому шарі. Через це вона потрапляє до харчового циклу водних жителів, стаючи причиною зниження рівня їх популяції, погіршення екологічної ситуації [2].

- Змішані.

Змішаними називаються суспензії, які містять органічні та неорганічні частинки. Майже всі суспензії, що спостерігаються в природних водоймах, є змішаними.

Антропогенна діяльність також призводить до утворення змішаного типу стоків, наприклад, на підприємствах целюлозно-паперової, нафтовидобувної, фармацевтичної промисловості. Сам факт ставлення

суспензії до змішаного типу не є негативним, оскільки на небезпеку вказує не класифікація суспензії, а інформація про її хімічний склад.

Охарактеризуємо органолептичні властивості води та ВР. Органолептика – аналіз матеріалу з урахуванням сприйняття його органами почуттів: нюхом, дотиком, зором, смаком, слухом. Воді притаманні органолептичні властивості, які визначаються цим методом.

Ступінь каламутності води залежить від наявності в ній завислих дрібнодисперсних домішок. Мутність – функція насиченості води цими частинками. Можливі випадки, коли менш насичені суспензії будуть більш каламутними через тип зважених частинок.

Визначається цей показник на польових випробуваннях фотометрично та/або візуально (ГОСТ 1030-81 "Вода господарсько-питного призначення. Польові методи аналізу"). При візуальному визначенні ступеня мутності води використовується скляна пробірка, лист темного паперу і, при необхідності, джерело світла. Результатом такого випробування є визначення градації каламутності води:

- каламутність непомітна (відсутня);
- слабка опалесценція;
- опалесценція;
- слабка каламутність;
- каламутність;
- сильна каламутність.

Зважені речовини у твердому стані, що знаходяться у воді, що має природну природу, включають наступні компоненти: глиняні частинки, пісок, мул, суспендовані частинки, що мають органічне та неорганічне походження, планктон та інші мікроорганізми.

На те, наскільки сконцентровані у воді ці складові, впливають фактори зміни сезонів і режим надходження стічних вод з поверхні. Також важливу роль відіграє процес танення снігу та складу порід, які формують русло/донну частину водойми. Крім того, не варто недооцінювати і вплив факторів, що

мають антропогенне походження – проведення сільськогосподарських робіт, гірничодобувні роботи.

Зазначені частки впливають на те, яку прозорість матиме вода, на її світлопроникність, на температурні значення. Крім іншого впливає на процес абсорбції токсичних домішок, на компоненти і розподіл відкладень, швидкість утворення опадів. Воду, що у великій кількості містить зазначені речовини, не допускають для застосування з метою рекреації з міркувань естетики [3].

Якісний склад води об'єктів, розташованих поблизу місць господарсько-питного та культурно-побутового значення, має відповідати деяким вимогам. У процесі спуску стічних вод рівень концентрації зазначених частинок не повинен зрости більш ніж на 0.25 мг/дм3 та 0.75 мг/дм3.

Водні об'єкти, що містять понад 30 мг/дм3 мінеральних речовин природного походження, допустимо зростання концентрації завислих речовин у водних масах у коридорі до 5%.

Нерозчинні у воді частинки поділяються на суспензії та колоїди.

Вкрай важливо проводити контроль рівня скупчення зважених частинок під час перевірки біологічної та фізико-хімічної обробки стічних вод та у процесі оцінки якості природної води.

Грубодисперсні речовини можна визначити за допомогою гравіметричного способу.

Він полягає у відділенні цих речовин за допомогою фільтра "синя стрічка" (застосовується для проб, які мають прозорість менше 10 см).

Сучасні водоочисні споруди використовують основи фізики та високі технології для очищення найбруднішої води, щоб вона могла повернутися у навколишнє середовище як повноправний учасник водного круговороту. У давнину люди брали воду з річок, озер для приготування їжі і на госп-побутові потреби. Але ця вода не була чистою, звідси випливали хвороби, епідемії та висока смертність. У древній Індії зменшення вмісту завислих речовин у стоках використовувалися гравій, пісок і кип'ятіння [3].

Світові комерційні та промислові виробничі підприємства вимагають найкращих доступних технологій очищення стоків для досягнення відповідності вимогам скидання зважених твердих частинок. Підприємства, що скидають стічні води в міську каналізацію або водні шляхи, повинні дотримуватися суворих норм щодо рівнів загального вмісту завислих речовин.

Розширення виробництва по всьому світу збільшує загальну кількість завислих твердих частинок. Стічні води, що потрапляють на очисні споруди, включають такі предмети, як дерево, каміння, очищення, завислі речовини та багато іншого. Якщо їх не видалити, можуть викликати проблеми: забити труби, викликати поломку насосів та іншого обладнання в процесі очищення.

Екологічні служби контролюють та обмежують рівні забруднення промислових стоків, які підприємство може скидати у каналізацію та водні шляхи. Щоб забезпечити дотримання граничних значень, підприємства повинні впроваджувати програму попереднього очищення стічних вод. Деякі компанії обирають вивіз стоків. Інші вважають за краще встановлювати на своєму підприємстві технологію очищення та фільтрації [4].

Ця система забезпечує комплексне очищення стічних вод на основі механічних, біологічних та фізико-хімічних методів.

Налаштування очищення індивідуально підбираються для кожного Замовника.

Механічна фільтрація фізично видаляє обложені тверді частинки з мулу.

Хімічна обробка розділяє тверді та рідкі частинки на частини шляхом автоматичного додавання хімікатів для попередньої обробки, таких як регулятори pH, полімери (коагулянти та флокулянти) або глина.

Після видалення твердих частинок рідкі стічні води фільтруються через речовину, зазвичай пісок, під дією сили тяжіння. Цей метод позбавляє певної кількості бактерій, зменшує каламутність і колір, видаляє запахи, зменшує кількість заліза та видаляє більшість інших твердих частинок, що залишилися у воді. Іноді фільтрують воду через вугільні частинки, які видаляють органічні частинки. Нарешті, стічні води потрапляють у резервуар для «контакту з

хлором», куди додається хімічний хлор для знищення бактерій, які можуть становити небезпеку здоров'ю, як це робиться в плавальних басейнах. Хлор в основному видаляється при знищенні бактерій, але іноді його необхідно нейтралізувати, додаючи інші хімічні речовини. Це захищає рибу та інші морські організми, яким може зашкодити щонайменше кількість хлору. Хлор також призводить до численних захворювань людей. Очищена вода потім скидається в місцеву річку чи океан [4].

1.2 Огляд і опис методів та підходів щодо прогнозування завислих речовин у водах річки

Методи машинного навчання можуть використовуватись для класифікації та прогнозування проблем часових рядів.

Перш ніж досліджувати методи машинного навчання для часових рядів, рекомендується переконатися, що ви вичерпали класичні методи прогнозування лінійних часових рядів. Класичні методи прогнозування часових рядів можуть бути орієнтовані на лінійні відносини, проте вони є складними і добре справляються з широким спектром проблем, за умови, що ваші дані підготовлені належним чином і метод добре налаштований [5].

Продемонструємо 11 різних класичних методів прогнозування часових рядів. Популярними методами є:

- авторегресія (AR);
- ковзна середня (мА);
- авторегресійна ковзна середня (ARMA);
- авторегресійне інтегроване ковзне середнє (ARIMA);
- сезонні авторегресійні інтегровані ковзні середні (SARIMA);
- сезонні авторегресійні інтегровані ковзні середні з екзогенними регресорами (SARIMAX);
- Векторна авторегресія (VAR);
- змінна середня векторна авторегресія (VARMA);

– ковзне середнє векторне авторегресії з екзогенними регресорами (VARMAX);

– просте експонентне згладжування (SES);

– експонентне згладжування Холта Вінтера (HWES);

– авторегресія (AR).

Метод авторегресії (AR) моделює наступний крок у послідовності як лінійну функцію спостережень на попередніх тимчасових кроках.

Позначення моделі включають вказівку порядку моделі p як параметр для функції AR, наприклад, AP (p). Наприклад, AR (1) є моделлю авторегресії першого порядку.

Метод підходить для одномірних часових рядів без трендових та сезонних складових.

Метод Ковзного середнього (мА). Метод ковзного середнього (MA) моделює наступний крок у послідовності як лінійну функцію від залишкових помилок із середнього процесу на попередніх тимчасових кроках.

Модель ковзного середнього відрізняється від обчислення ковзного середнього часового ряду.

Позначення для моделі включає вказівку порядку моделі q як параметр функції MA, наприклад, M. A. (д). Наприклад, MA (1) є моделлю ковзного середнього першого порядку.

Метод підходить для одномірних часових рядів без трендових та сезонних складових.

Авторегресійна ковзна середня (ARMA). Метод авторегресійного ковзного середнього (ARMA) моделює наступний крок у послідовності як лінійну функцію спостережень та випадкових помилок на попередніх тимчасових кроках. Він поєднує моделі авторегресії (AR) і ковзної середньої (MA).

Позначення моделі включає вказівку порядку для моделей AR (p) і MA (q) як параметри функції ARMA, наприклад, ARMA (p, q). Модель ARIMA можна використовувати для розробки моделей AR або MA.

Метод підходить для одномірних часових рядів без трендових та сезонних складових.

Метод Авторегресійне інтегроване ковзне середнє (ARIMA). Даний метод авторегресійного інтегрованого ковзного середнього (ARIMA) моделює наступний крок у послідовності як лінійну функцію різницевих спостережень та залишкових помилок на попередніх тимчасових кроках.

Він поєднує моделі авторегресії (AR) та ковзного середнього (MA), а також етап попередньої обробки різницевої послідовності, щоб зробити послідовність стаціонарною, званою інтеграцією (I).

Позначення для моделі включають вказівку порядку моделей AR (p), I (d) і MA (q) як параметри функції ARIMA, наприклад, APIMA (p, d, q). Модель ARIMA також може бути використана для розробки моделей AR, MA та ARMA.

Метод підходить для одномірних часових рядів із трендом і без сезонних компонентів [5].

Метод сезонні авторегресійні інтегровані ковзні середні (SARIMA). Даний метод сезонного авторегресійного інтегрованого ковзного середнього (SARIMA) моделює наступний крок у послідовності як лінійну функцію різницевих спостережень, помилок, сезонних спостережень і сезонних помилок на попередніх тимчасових кроках. Він поєднує в собі модель ARIMA з можливістю виконання тієї ж моделі авторегресії, диференціювання та ковзного середнього на сезонному рівні.

Позначення для моделі включають вказівку порядку моделей AR (p), I (d) і MA (q) як параметри функції ARIMA і AR (P), I (D), MA (Q) і m. параметри на сезонному рівні, наприклад SARIMA (p, d, q) (P, D, Q) m, де "m" - кількість тимчасових кроків у кожному сезоні (сезонний період). Модель SARIMA може використовуватись для розробки моделей AR, MA, ARMA та ARIMA [5].

Метод підходить для одновимірних часових рядів із трендовими та/або сезонними компонентами.

Метод сезонні авторегресійні інтегровані ковзні середні з екзогенними регресорами (SARIMAX). Сезонна авторегресійна інтегрована ковзна середня з екзогенними регресорами (SARIMAX) є розширенням моделі SARIMA, яка також включає моделювання екзогенних змінних [6].

Екзогенні змінні також називаються коваріатами, і їх можна розглядати як паралельні вхідні послідовності, спостереження яких виконуються з тими самими тимчасовими кроками, що й вихідний ряд. Первинні серії можуть бути названі ендогенними даними, щоб відрізняти їх від екзогенної послідовності (послідовності). Спостереження за екзогенними змінними входять у модель безпосередньо кожному тимчасовому кроці і моделюються як і, як первинна эндогенная послідовність (наприклад, процес AR, MA тощо. буд.).

Метод SARIMAX також можна використовувати для моделювання підгрупованих моделей із зовнішніми змінними, такими як ARX, MAX, ARMAX та ARIMAX.

Метод підходить для одновимірних часових рядів з трендовими та/або сезонними компонентами та екзогенними змінними.

Метод векторна авторегресія (VAR). Даний метод векторної агресії (VAR) моделює наступний крок у кожному часовому ряду, використовуючи модель AR. Це узагальнення AR на кілька паралельних часових рядів, наприклад, багатовимірний часовий ряд.

Позначення моделі включає вказівку порядку для моделі AR (p) як параметри функції VAR, наприклад, ВДП (p).

Метод підходить для багатовимірних часових рядів без трендових та сезонних складових [7].

Метод змінна середня векторна авторегресія (VARMA). Даний метод ковзної середньої векторної агресії (VARMA) моделює наступний крок у кожному часовому ряду, використовуючи модель ARMA. Це узагальнення ARMA для кількох паралельних часових рядів, наприклад, багатовимірний часовий ряд.

Позначення моделі включає вказівку порядку для моделей AR (p) і MA (q) як параметри функції VARMA, наприклад, Варма (p, д). Модель VARMA може бути використана для розробки моделей VAR або VMA. Метод підходить для багатовимірних часових рядів без трендових та сезонних складових.

Метод Ковзне середнє векторне авторегресії з екзогенними регресорами (VARMAX). Даний метод ковзне середнє векторне авторегресії з екзогенними регресорами (VARMAX) є розширенням моделі VARMA, яка також включає моделювання екзогенних змінних. Це багатоваріантна версія методу ARMAX.

Екзогенні змінні також називаються коваріатами, і їх можна розглядати як паралельні вхідні послідовності, спостереження яких виконуються з тими самими тимчасовими кроками, що й вихідний ряд. Первинний ряд називають ендогенними даними, щоб відрізняти їх від екзогенної послідовності (послідовностей). Спостереження за екзогенними змінними входять у модель безпосередньо кожному тимчасовому кроці і моделюються як і, як первинна эндогенная послідовність (наприклад, процес AR, MA тощо. буд.).

Метод VARMAX також можна використовувати для моделювання підгрупованих моделей із зовнішніми змінними, такими як VARX та VMAX.

Метод підходить для багатовимірних часових рядів без трендових та сезонних компонентів та екзогенних змінних.

Розглянемо метод просте експонентне згладжування (SES). Даний метод простого експонентного згладжування (SES) моделює наступний тимчасовий крок як експоненційно зважену лінійну функцію спостережень на попередніх тимчасових кроках. Метод підходить для одномірних часових рядів без трендових та сезонних складових [7].

Розглянемо метод експонентне згладжування Холта Вінтера (HWES). Метод експонентне згладжування Холта Вінтера (HWES) також називається методом потрійного експонентного згладжування, моделює наступний тимчасовий крок як експоненційно зважену лінійну функцію спостережень на попередніх тимчасових кроках з урахуванням тенденцій та сезонності.

Метод підходить для одновимірних часових рядів із трендовими та/або сезонними компонентами. У лабораторії моделювання природних систем Національного центру когнітивних розробок Університету ІТМО ми активно досліджуємо питання щодо застосування автоматичного машинного навчання для різних завдань. У цій статті ми хочемо розповісти про застосування AutoML для ефективного прогнозування часових рядів, а також про те, як це реалізовано у рамках open-source фреймворку FEDOT. Це друга стаття із серії публікацій, присвяченій даній розробці (з першою з них можна ознайомитись за посиланням).

Сучасний метод автоматичне машинне навчання (AutoML). Сучасна Data Science стала дуже популярною та потрібною частиною IT сфери. Фахівці збирають дані, займаються їх очищенням, пробують різні моделі, роблять валідацію, вибирають найкращі з них. І все це для того, щоб надати бізнесу рішення, яке принесе найбільшу користь. При цьому деякі етапи отримання таких рішень з кожним роком все більше автоматизуються. Як правило, це стосується найрутинніших частин. Таким чином звільняється час експертів для найважливіших завдань [7].

Отже, уявімо, що перед фахівцем стоїть завдання побудувати модель машинного навчання і "обернути" її в web-сервіс, щоб ця модель виконувала корисну роботу - передбачала що-небудь. Але перш ніж дійти до етапу навчання моделі, потрібно пройти кілька кроків, зокрема:

- зібрати дані з багатьох джерел, очистити їх;
- здійснити передобробку, нормалізацію, закодувати деякі з ознак;
- відібрати найбільш корисні з них або синтезувати на їх основі нові;
- видалити можливі викиди у даних.

Такі багатоступінчасті послідовності операцій, що включають етапи від первинної обробки даних до навчання моделі та складання прогнозів, називають пайплайнами. Працювати з пайплайнами вже дещо складніше, ніж із одиночними моделями машинного навчання, оскільки чим більше складових блоків, тим більше гіперпараметрів, які потрібно оптимізувати.

Також вища ймовірність того, що на якомусь етапі виникне помилка, та й загалом таку громіздку систему важче налаштовувати та контролювати її поведінку. Для вирішення цієї проблеми реалізовані спеціальні інструменти – MLFlow, Apache AirFlow тощо – щось на зразок workflow management system (WMS) у світі машинного навчання. Вони мають спростити контроль за станом пайплайнів обробки даних.

Чому потреба таких інструментів виникла, адже раніше обходилися без них?

Відповідь у цьому, галузь “дорослішає”, з'являються добре оптимізовані рішення загальних завдань. Поступово індустрія уникає самописних "сервісів" і переходить до використання стандартних підходів і технологій до обробки даних для ML завдань.

Найбільш амбітним завданням у цій галузі машинного навчання є автоматична генерація цих пайплайнів. Існує кілька фреймворків, які надають подібні функції серед open-source, наприклад це TPOT, AutoGluon, MLJAR або H2O. Такі AutoML фреймворки вирішують завдання оптимізації виду "побудувати такий пайплайн, який дає кінцевий прогноз із найменшою (серед усіх розглянутих рішень) помилкою". В основному структура пайплайна зафіксована і підбираються тільки гіперпараметри, але деякі фреймворки здатні отримувати рішення моделі довільної структури. Дана оптимізаційна задача (знаходження пайплайну довільної структури) вирішується, як правило, за допомогою еволюційних алгоритмів, приклади: фреймворки TPOT і FEDOT [7-8].

Існують також і пропрієтарні SaaS-рішення, такі як DataRobot, GoogleAutoTables, Amazon SageMaker, які допомагають не лише автоматизувати ML експерименти, але й надають можливості AutoML.

Як правило, AutoML бібліотеки та сервіси успішно вирішують дві найпопулярніші завдання у машинному навчанні: класифікація та регресія на табличних даних. Рідше підтримуються завдання, пов'язані з обробкою зображень, тексту та прогнозування часових рядів. У рамках цієї статті ми не

розглядатимемо плюси та мінуси відомих рішень, а зупинимося на можливостях автоматичного машинного навчання в задачі прогнозування часових рядів.

Незважаючи на те, що завдання прогнозування користується попитом в науці та бізнесі, більшість open-source бібліотек автоматичного машинного навчання не надають можливостей для формування пайплайнів для завдання прогнозування часових рядів. Причин тому може бути кілька, одна з яких — складність адаптації поточної функціональності бібліотеки для прогнозування рядів без переробки інструментарію для інших завдань (класифікації та регресії).

Обробка часових рядів відрізняється від звичного набору дій при вирішенні задачі регресії. Відмінності починаються вже з розбиття вихідної вибірки: наприклад, перемішувати дані у випадковому порядку для валідації моделі часового ряду наполегливо немає сенсу. Інакше для часових рядів формуються і ознаки: на вихідному ряді, як правило, моделі машинного навчання не навчають його потрібно перевести в інше уявлення. Впроваджувати такі конструкції у вже існуючий AutoML проект зі своїм legacy буває проблематично - швидше за все, саме тому розробники часто відмовляються від тимчасових рядів (щоб "не гнатися за двома зайцями") і концентруються на обробці конкретних типів даних: тільки табличних або тільки тексту [9]..

Деякі команди, які все-таки наважуються підтримувати прогнозування тимчасових рядів, обмежуються лише цим типом даних. Хороший open-source приклад – фреймворк AutoTS. У подібних бібліотеках зазвичай використовуються "класичні" статистичні моделі для прогнозування, наприклад AR або ARIMA. "Всередині" фреймворку проводиться налаштування цих моделей, а потім вибирається найкраща (за метрикою помилки на валідаційній вибірці), але нових моделей тут не генеруються. На такій логіці, наприклад, заснована бібліотека pmdarima.

Інший спосіб – адаптувати функціональність готового AutoML-інструменту для завдання прогнозування. Цю роль добре підходять регресійні моделі. Так, наприклад, було зроблено в H2O, де у своїй комерційній версії продукту розробники надали таку можливість. Однак, судячи з деяких прикладів використання open-source версії, користувачеві доведеться взяти на себе завдання первинної обробки вихідних рядів, наприклад, вилучення ознак. Для повноцінної роботи такого урізаного інструментарію може не вистачати [8].

А які функції хотілося б мати у AutoML-арсеналі?

Хоча завдання прогнозування поведінки одновимірного масиву виглядає тривіальною, існує безліч інструментів, якими інженер хотів би мати при роботі з тимчасовими рядами. Наприклад:

Можливість будувати ансамблі, що інтерпретуються, з моделей (наприклад, щоб одна модель відтворювала високочастотну складову тимчасового ряду, друга — низькочастотну, а третя — об'єднувала їх прогнози);

Мати можливість здійснювати налаштування гіперпараметрів у пайплайнах для часових рядів;

Використовувати екзогенні (допоміжні) часові лави;

Застосовувати специфічні методи попередньої обробки (від згладжування ковзним середнім до перетворення Бокса-Кокса);

Мати можливість застосовувати in-sample та out-of-sample прогнозування;

А якщо тимчасовий ряд із перепустками — як їх усунути?

Врахувати всі перераховані можливості в одному фреймворку, і, при цьому, не обмежуватися лише тимчасовими рядами досить складне завдання.

1.3 Обгрунтування вибору бібліотеки Scikit Learn

Scikit-learn – один з найбільш широко використовуваних пакетів Python для Data Science та Machine Learning. Він дозволяє виконувати безліч операцій та надає безліч алгоритмів. Scikit-learn також пропонує чудову документацію про свої класи, методи та функції, а також опис використовуваних алгоритмів [5].

Scikit-Learn підтримує:

- попередню обробку даних;
- зменшення розмірності;
- вибір моделі;
- регресії;
- класифікації;
- кластерний аналіз.

Він також надає кілька наборів даних, які можна використовувати для тестування ваших моделей.

Scikit-learn не реалізує все, що пов'язане із машинним навчанням. Наприклад, він не має комплексної підтримки для:

- нейронних мереж;
- самоорганізуються карт (мереж Кохонена);
- навчання асоціативних правил;
- навчання з підкріпленням (reinforcement learning).

Scikit-learn заснований на NumPy та SciPy, тому необхідно зрозуміти хоча б ази цих двох бібліотек, щоб ефективно застосовувати Scikit-learn.

Scikit-learn – це пакет з відкритим вихідним кодом. Як і більшість матеріалів з екосистеми Python, він є безкоштовним навіть для комерційного використання. Його ліцензовано під ліцензією BSD.

Попередня обробка даних

Ви можете використовувати scikit-learn для підготовки ваших даних до алгоритмів машинного навчання: стандартизації або нормалізації даних, кодування категоріальних змінних та багато іншого.

Давайте спочатку визначимо масив NumPy, з яким працюватимемо (рис. 1.1).

```
>>> import numpy as np
>>> x = np.array([[0.1, 1.0, 22.8],
...               [0.5, 5.0, 41.2],
...               [1.2, 12.0, 2.8],
...               [0.8, 8.0, 14.0]])
>>> x
array([[ 0.1,  1. , 22.8],
       [ 0.5,  5. , 41.2],
       [ 1.2, 12. ,  2.8],
       [ 0.8,  8. , 14. ]])
```

Рисунок 1.1 – Масив NumPy

Часто потрібно перетворювати дані таким чином, щоб середнє значення кожного стовпця (елемента) дорівнювало нулю, а стандартне відхилення - одиниці. У цьому випадку можна використовувати sklearn.preprocessing.StandardScaler (рис. 1.2).

```
>>> from sklearn.preprocessing import StandardScaler
>>> scaler = StandardScaler()
>>> scaled_x = scaler.fit_transform(x)
>>> scaler.scale_
array([ 0.40311289,  4.03112887, 14.04421589])
>>> scaler.mean_
array([ 0.65,  6.5 , 20.2 ])
>>> scaler.var_
array([1.6250e-01, 1.6250e+01, 1.9724e+02])
>>> scaled_x
array([[-1.36438208, -1.36438208,  0.18512959],
       [-0.3721042 , -0.3721042 ,  1.4952775 ],
       [ 1.36438208,  1.36438208, -1.23894421],
       [ 0.3721042 ,  0.3721042 , -0.44146288]])
>>> scaled_x.mean().round(decimals=4)
0.0
>>> scaled_x.mean(axis=0)
array([ 1.66533454e-16, -1.38777878e-17,  1.52655666e-16])
>>> scaled_x.std(axis=0)
array([1., 1., 1.])
>>> scaler.inverse_transform(scaled_x)
array([[ 0.1,  1. , 22.8],
       [ 0.5,  5. , 41.2],
       [ 1.2, 12. ,  2.8],
       [ 0.8,  8. , 14. ]])
```

Рисунок 1.2 – Використання klearn.preprocessing.StandardScaler

Якщо є деякі категоріальні дані, і потрібно перетворити їх у числа. Один із способів зробити це – використовувати клас sklearn.preprocessing.OneHotEncoder.

Розглянемо наступний приклад з масивами ролей у компанії (рис. 1.3):

```
>>> from sklearn.preprocessing import OneHotEncoder
>>> roles = np.array([('Tom', 'manager'),
...                   ('Mary', 'developer'),
...                   ('Ann', 'recruiter'),
...                   ('Jim', 'developer')])
>>> roles
array([['Tom', 'manager'],
       ['Mary', 'developer'],
       ['Ann', 'recruiter'],
       ['Jim', 'developer']], dtype='<u9')>>> encoder = OneHotEnco
der()
>>> encoded_roles = encoder.fit_transform(roles[:, [1]])
>>> encoded_roles.toarray()
array([[0., 1., 0.],
       [1., 0., 0.],
       [0., 0., 1.],
       [1., 0., 0.]])</u9')>
```

Рисунок 1.3 – Використання класу sklearn.preprocessing.OneHotEncoder

У наведеному вище прикладі (рис. 1.3) перший стовпець об'єкта encoded_roles вказує, чи кожен співробітник є розробником. Другий та четвертий співробітник (Мері та Джим) - розробники. Другий стовпець пов'язаний із посадою менеджера. Лише перший працівник (Том) має цю посаду. Нарешті, третій стовпець відповідає посаді рекрутера, і це третій співробітник (Энн).

Розглянемо приклад використання зменшення розмірності. Зменшення розмірності включає вибір або вилучення найбільш важливих компонентів (ознак) багатовимірного набору даних. Scikit-learn пропонує кілька підходів до зменшення розмірності. Одним із них є аналіз основних компонентів (PCA) [9].

Розглянемо підходи до вибору моделі. Для навчання та тестування моделей машинного навчання, вам необхідно випадково розбивати дані на

підмножини. Це включає як входи, і їх відповідні виходи. Функція sklearn.model_selection.train_test_split () корисна у таких випадках (рис. 1.4).

```
>>> import numpy as np
>>> from sklearn.model_selection import train_test_split
>>> x, y = np.arange(1, 21).reshape(-1, 2), np.arange(3, 40, 4)
>>> x
array([[ 1,  2],
       [ 3,  4],
       [ 5,  6],
       [ 7,  8],
       [ 9, 10],
       [11, 12],
       [13, 14],
       [15, 16],
       [17, 18],
       [19, 20]])
>>> y
array([ 3,  7, 11, 15, 19, 23, 27, 31, 35, 39])
>>> x_train, x_test, y_train, y_test =\
...     train_test_split(x, y, test_size=0.4, random_state=0)
>>> x_train
array([[ 3,  4],
       [13, 14],
       [15, 16],
       [ 7,  8],
       [ 1,  2],
       [11, 12]])
>>> y_train
array([ 7, 27, 31, 15,  3, 23])
>>> x_test
array([[ 5,  6],
       [17, 18],
       [ 9, 10],
       [19, 20]])
>>> y_test
array([11, 35, 19, 39])
```

Рисунок 1.4 – Використання функції sklearn.model_selection.train_test_split

На додаток до звичайного поділу наборів даних, scikit-learn надає засоби для здійснення перехресної перевірки, налаштування гіперпараметрів ваших моделей за допомогою пошуку по сітці, обчислення багатьох величин, які показують продуктивність моделі (наприклад, коефіцієнт детермінації, середньоквадратична помилка, показник відхилення з поясненням, матриця помилок, звіт про класифікацію, f-показники та багато іншого).

Scikit-learn надає кілька наборів даних, які підходять для вивчення та тестування ваших моделей. В основному це відомі набори даних. Вони є достатнім обсягом даних для тестування моделей і в той же час не дуже великий, що забезпечує прийнятну тривалість навчання [10].

Наприклад, функція sklearn.datasets.load_boston (рис. 1.5) відображає дані про ціни на будинки в районі Бостона (ціни не оновлюються!). Є 506 спостережень, а вхідна матриця має 13 стовпців (ознаки):

```
>>> from sklearn.datasets import load_boston
>>> x, y = load_boston(return_X_y=True)
>>> x.shape, y.shape
((506, 13), (506,))
```

Рисунок 1.5 – Використання функції sklearn.datasets.load_boston

Цей набір даних підходить для багатоваріантної регресії.

Інший приклад – набір даних, пов'язаний з вином. Його можна отримати за допомогою функції sklearn.datasets.load_wine(рис. 1.6):

```
>>> from sklearn.datasets import load_wine
>>> x, y = load_wine(return_X_y=True)
>>> x.shape, y.shape
((178, 13), (178,))
>>> np.unique(y)
array([0, 1, 2])
```

Рисунок 1.6 – Використання функції klearn.datasets.load_wine

Набір даних підходить для класифікації. Він містить 13 функцій, пов'язаних з трьома різними виноробними компаніями з Італії, та 178 спостережень.

Scikit-learn підтримує різні методи регресії починаючи з лінійної регресії, методу k-найближчих сусідів за допомогою поліноміальної регресії,

регресії опорних векторів, дерев прийняття рішень і т.д., до ансамблю методів, таких як random forest та градієнтний бустинг. Він також підтримує нейронні мережі, але не настільки, як спеціалізовані бібліотеки, наприклад, як TensorFlow [11].

Наведемо приклад реалізації регресії random forest. Розпочинаємо з імпорту необхідних нам пакетів, класів та функцій (рис. 1.7):

```
>>> import numpy as np
>>> from sklearn.datasets import load_boston
>>> from sklearn.ensemble import RandomForestRegressor
>>> from sklearn.model_selection import train_test_split
```

Рисунок 1.7 - Імпорт основних класів та функцій

Наступним кроком є отримання даних для роботи та поділ цих даних на навчальну та тестову підмножини. Ми будемо використовувати набір даних Бостона (рис. 1.8):

```
>>> x, y = load_boston(return_X_y=True)
>>> x_train, x_test, y_train, y_test =\
...     train_test_split(x, y, test_size=0.33, random_state=0)
```

Рисунок 1.8 - Завантаження даних Бостона

Деякі методи вимагають масштабування (стандартизації) даних, тоді як інших методів це необов'язково. На цей раз ми продовжимо без масштабування.

Тепер нам потрібно створити наш регресор і навчити його за допомогою підмножини даних, вибраної для навчання (рис. 1.9):

```
>>> regressor = RandomForestRegressor(n_estimators=10, random_stat
e=0)
>>> regressor.fit(x_train, y_train)
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=N
one,
                      max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_spli
t=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=1
0,
                      n_jobs=None, oob_score=False, random_state=
0, verbose=0,
                      warm_start=False)
```

Рисунок 1.9 - Навчання моделі

Після того, як модель навчена, ми перевіряємо коефіцієнт детермінації на підмножині даних, що навчає, і, що більш важливо, на тестовому підмножині даних, який не використовувався при навчанні моделей (рис. 1.10).

```
>>> regressor.score(x_train, y_train)
0.9680930547240916
>>> regressor.score(x_test, y_test)
0.8219576562705848
```

Рисунок 1.10 - Навчання моделі на тестових даних

Модель, навчена досить добре, може використовуватися для прогнозування вихідних даних за інших нових вхідних даних x_new. У цьому випадку використовується команда .predict(): regressor.predict(x_new).

Scikit-learn виконує класифікацію майже так само, як регресію. Він підтримує різні методи класифікації, такі як логістична регресія і k-найближчі сусіди, метод опорних векторів, наївний класифікатор байесу, дерево

прийняття рішень, а також ансамбль методів, такі як random forest, AdaBoost і градієнтний бустинг [12].

Досить поширений підхід використовувати метод random forest для класифікації (рис. 1.11). Цей підхід дуже аналогічний підходу, що застосовується у разі регресії. Але тепер ми використовуватимемо набір даних, пов'язаних з вином, визначатимемо класифікатор і оцінюємо його з точністю класифікації замість коефіцієнта детермінації.

```
>>> import numpy as np
>>> from sklearn.datasets import load_wine
>>> from sklearn.ensemble import RandomForestClassifier
>>> from sklearn.model_selection import train_test_split
>>> x, y = load_wine(return_X_y=True)
>>> x_train, x_test, y_train, y_test =\
...     train_test_split(x, y, test_size=0.33, random_state=0)
>>> classifier = RandomForestClassifier(n_estimators=10, random_st
ate=0)
>>> classifier.fit(x_train, y_train)
RandomForestClassifier(bootstrap=True, class_weight=None, criterio
n='gini',
                       max_depth=None, max_features='auto', max_le
af_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_spl
it=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=
10,
                       n_jobs=None, oob_score=False, random_state=
0, verbose=0,
                       warm_start=False)
>>> classifier.score(x_train, y_train)
1.0
>>> classifier.score(x_test, y_test)
1.0
```

Рисунок 1.11 - Використання методу random forest для класифікації

Модель, навчена досить добре, може використовуватися для прогнозування вихідних даних за інших нових вхідних даних. У цьому випадку використовується команда .predict(): regressor.predict(x_new) [13].

Ще один метод, який користується популярністю - Кластерний аналіз. Кластеризація - це галузь неконтрольованого навчання, що широко підтримується в scikit-learn. На додаток до методу k-середніх, є можливість застосовувати метод поширення близькості, спектральну кластеризу.

1.4 Висновки

В розділі було проведено аналіз предметної області, зроблено огляд і опис методів та підходів щодо прогнозування завислих речовин у водах річок, проведенно обгрунтування вибору бібліотеки Scikit Learn.

2 ФОРМАЛІЗАЦІЯ МОДЕЛІ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЗАВИСЛИХ РЕЧОВИН У РІЧЦІ

2.1 Ідея інформаційної системи аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року

Як було зазначено вище для початку варто розпочати з розвідувального аналізу. Exploratory data analysis займається наближеним експрес-аналізом даних.

Алгоритмом наближеного аналізу даних включає наступні етапи: обчислення характеристик; перевірка стохастичності вибірки; перевірка гіпотез; видалення аномальних спостережень [1].

Аби здійснити аналіз та передбачення завислих речовин у річковій воді дані взято з роботи «WQ SB river: EDA and Forecasting» на сайті Kaggle, власником даного ноутбука був завідувач кафедри Мокін В. Б. [11]. Дані взято з датасету платформі Kaggle з даними моніторингу якості води [12].

Для того щоб почати, необхідно завантажити пакети маніпуляцій даними, моделі навчання та інженерні особливості. Процес завантаження зображений на рисунку 2.1.

```
# Work with Data - the main Python libraries
import numpy as np
import pandas as pd
import pandas_profiling as pp

# Visualization
import matplotlib.pyplot as plt

# Preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, KFold, ShuffleSplit, GridSearchCV

# Modeling
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
import xgboost as xgb
from xgboost.sklearn import XGBRegressor

# Metrics
from sklearn.metrics import r2_score

import warnings
warnings.simplefilter('ignore')
```

Рисунок 2.1 – Завантаження пакетів даних

Підключаємо дані по вмісту речовин в р. Південний Буг (рис. 2.2).

```
data = pd.read_csv('../input/wq-southern-bug-river-01052021/PB_All_2000_2021.csv', sep=';', hea
der=0)
data
```

Рисунок 2.2 – Підключення даних

Таблиця даних спостереження представоено на рисунку 2.3. Де «id» це номер станції спостереження, «date» дата спостереження і 8 показників речовин. Нас найбільше цікавить O2 через те, що він є об’єктом дослідження даної роботи.

	id	date	NH4	BSK5	Suspended	O2	NO3	NO2	SO4	PO4	CL
0	1	17.02.2000	0.330	2.77	12.0	12.30	9.50	0.057	154.00	0.454	289.50
1	1	11.05.2000	0.044	3.00	51.6	14.61	17.75	0.034	352.00	0.090	1792.00
2	1	11.09.2000	0.032	2.10	24.5	9.87	13.80	0.173	416.00	0.200	2509.00
3	1	13.12.2000	0.170	2.23	35.6	12.40	17.13	0.099	275.20	0.377	1264.00
4	1	02.03.2001	0.000	3.03	48.8	14.69	10.00	0.065	281.60	0.134	1462.00
...	...	...	...	...	...	...	...	...	...	...	...
2856	22	06.10.2020	0.046	2.69	3.6	8.28	3.80	0.038	160.00	0.726	77.85
2857	22	27.10.2020	0.000	1.52	0.5	11.26	0.56	0.031	147.20	0.634	71.95
2858	22	03.12.2020	0.034	0.29	0.8	11.09	2.58	0.042	209.92	0.484	61.17
2859	22	12.01.2021	0.000	2.10	0.0	14.31	3.94	0.034	121.60	0.424	63.49
2860	22	10.02.2021	0.000	1.78	0.0	14.30	6.30	0.033	134.40	0.582	66.31

Рисунок 2.3 – Таблиця даних

Формування таблиці показників з додаванням кілометражу та впорядкування їх за зменшенням від витоку до гирла (рис. 2.4).

```
data_about = pd.read_csv('../input/wq-southern-bug-river-01052021/PB_stations.csv', sep=';', he
ader=0, encoding='cp1251')
data_about.sort_values(by=['length'], ascending=False)
```

Рисунок 2.4 – Формування таблиці

Список всіх показників які знаходяться на р. Південний Буг приведено на рисунку 2.5.

	id	length	name_station
20	21	773.0	р. Південний Буг, 773 км, смт. Чорний Острів, Мар'янівське вдсх.
19	20	755.0	р. Південний Буг, 755 км, м. Хмельницький , Хмельницьке вдсх.
18	19	744.0	р. Південний Буг, 744 км, с. Копистин, нижче м.Хмельницький
17	18	711.0	р. Південний Буг, 711 км, смт. Меджибіж, Меджибіжське вдсх.
16	17	692.0	р. Південний Буг, 692 км, с. Щедрове, Щедрівське вдсх.
15	16	652.0	р. Південний Буг, 652 км, м. Хмільник, питний в/з, вище міста
14	15	607.0	р. Південний Буг, 607 км, с. Гущинці, нижче села , питний водозабір м.Калинівка
13	14	582.0	р. Південний Буг, 582 км, м. Вінниця, Сабарівське вдсх, питний в/з міста, вище міста
12	13	569.5	р. Південний Буг, 569,5 км, 500 м нижче скиду ВОКВП ВКГ "Вінницяводоканал" (1,5 км нижче греблі Сабарівського вдсх.)
11	12	537.0	р. Південний Буг, 537 км, смт. Сутиски, Сутиське вдсх., н/б'єф
9	10	413.0	р. Південний Буг, 413 км, с. Маньківка, вище села, питний в/з м.Ладижин
8	9	400.0	р. Південний Буг, 400 км, м. Ладижин, Ладижинське вдсх.
7	8	372.0	р. Південний Буг, 372 км, с. Глибочок, Глибочекське вдсх.
6	7	327.0	р. Південний Буг, 327 км, с. Ставки, кордон Вінницької та Кіровоградської обл.
5	6	316.0	р. Південний Буг, 316 км, м.Гайворон, Гайворонське вдсх.
4	5	237.0	р. Південний Буг, 237 км, питний водозабір смт Побузьке
3	4	206.0	р. Південний Буг, 206 км, м. Первомайськ, Первомайське вдсх.
2	3	153.0	р. Південний Буг, 153 км, с. Олексіївка, питний в/з м. Південно-Українськ
1	2	136.0	р. Південний Буг, 136 км, с. Олександрівка, Олександрівське вдсх.
21	22	97.0	р. Південний Буг, 97 км, м. Вознесенськ, пит.в/з м. Вознесенськ, 2 км до в'їзду у м. Вознесенськ по трасі з м. Миколаїв
10	11	50.0	р. Південний Буг, 50 км, с. Ковалівка, Південно-Бузька ЗС
0	1	0.5	р. Південний Буг, 0,5 км, м. Миколаїв, Бузький лиман, тех. в/з Миколаївської ТЕЦ (ліва частина морського порту)

Рисунок 2.5 − Показники на р. Південний Буг

Дивимося на самі данні, бачимо що порядок постів даних багато міряли часто дані доступні за багато років, але по деяких постам даних не вистачає (рис. 2.6).

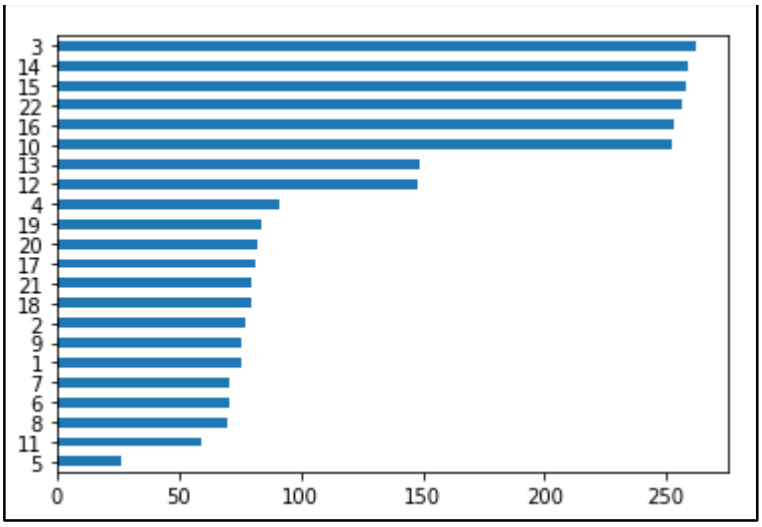


Рисунок 2.6 – Кількість даних по постах

Перетворимо дані в формат «date» та створимо ознаку «year» (рис. 2.7).

```
# Determination the year of observations
data['ds'] = pd.to_datetime(data['date'])
data['year'] = data['ds'].dt.year
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2861 entries, 0 to 2860
Data columns (total 13 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   id         2861 non-null   int64
 1   date       2861 non-null   object
 2   NH4        2858 non-null   float64
 3   BSK5       2860 non-null   float64
 4   Suspended  2845 non-null   float64
 5   O2         2858 non-null   float64
 6   NO3        2860 non-null   float64
 7   NO2        2858 non-null   float64
 8   SO4        2812 non-null   float64
 9   PO4        2833 non-null   float64
 10  CL         2812 non-null   float64
 11  ds         2861 non-null   datetime64[ns]
 12  year       2861 non-null   int64
dtypes: datetime64[ns](1), float64(9), int64(2), object(1)
memory usage: 290.7+ KB
```

Рисунок 2.7 – Перетворення даних

Далі витягуємо номер поста і рік, групуємо за номером поста та впорядковуємо за зменшенням (рис. 2.8).

```
data[['id', 'year']].groupby(by=['id']).min().sort_values(by=['year'], ascending=False)
```

Рисунок 2.8 – Групування за номером поста та впорядкування за зменшенням

Тобто ми бачимо що на 5 посту дані починають лише з 2019 року на 13 посту дані починаються з 2006 року, а решта з 2000 (рис. 2.9).

	year
id	
5	2019
13	2006
1	2000
21	2000
20	2000
19	2000
18	2000
17	2000
16	2000
15	2000
14	2000
12	2000
2	2000
11	2000
10	2000
9	2000
8	2000
7	2000
6	2000
4	2000
3	2000
22	2000

Рисунок 2.9 – Кількість даних по роках

Тепер переглянемо кінцеві дати даних все так само як і з минулим тільки з «max» рисунок 2.10.

```
data[['id', 'year']].groupby(by=['id']).max().sort_values(by=['year'], ascending=False)
```

Рисунок 2.10 – Групування за номером поста та впорядкування за збільшення

Можна побачити що пости 3,5,10,14,15,16 і 22 мають свіжі дані по 2021 рік, а решта лише по 2018 рік рисунок 2.11.

id	year
22	2021
14	2021
3	2021
5	2021
10	2021
16	2021
15	2021
21	2018
20	2018
19	2018
18	2018
17	2018
1	2018
13	2018
2	2018
11	2018
9	2018
8	2018
7	2018
6	2018
4	2018
12	2018

Рисунок 2.11 – Кількість даних по роках

Отже, зі списку доцільним є відібрати 3 послідовних пости 14,15,16 подивимося на них в таблиці «data_about» (рис. 2.12).

```
stations_good = [14, 15, 16]
data_about[data_about['id'].isin(stations_good)]
```

	id	length	name_station
13	14	582.0	р. Південний Буг, 582 км, м. Вінниця, Сабарівське вдсх, питний в/з міста, вище міста
14	15	607.0	р. Південний Буг, 607 км, с. Гущинці, нижче села , питний водозабір м.Калинівка
15	16	652.0	р. Південний Буг, 652 км, м. Хмільник, питний в/з, вище міста

Рисунок 2.12 – Відібрані послідовно розташовані пости

Це послідовно розташовані пости біля питних водозаборів зокрема Вінниця, Гущинці, Хмільник. Отже, доцільно зробити Вінницький пост цільовим тобто тим який було б цікаво прогнозувати за даними попередніх постів. Але як видно з рисунка 2.12. пости спостеження моли б бути відсортовані у зворотньому порядку. Далії в програмі цю проблему було знято через тагргетування даних з 14 створу. Особливості взаємодії речовин у воді показують, що більш доцільно моделювати не один показник по його ж значенню вище по течії, а також інші показники що можуть на це вплинути. Тобто для того щоб моделювати завислі речовини буде доцільним взяти до нього ще NO_2 і NO_3 вони суттєво впливають на рівень завислих речовин в воді відповідно і впливають на процеси забруднення (рис. 2.13).

```
# Set target indicator
target_data_name = 'Suspended'
#feature_target_all = ['NH4', 'BSK5', 'NO3', 'NO2', 'SO4', 'PO4', 'CL']
feature_target_all = ['NO3', 'NO2']
feature_data_all = feature_target_all + [target_data_name]
feature_data_all
```

```
['NO3', 'NO2', 'Suspended']
```

Рисунок 2.13 – Вибір речовин прогнозування

Тепер витягуємо ці три показники. Тут є така особливість, що є сенс видалити ті рядки де дані не повні щоб залишися лише повні рядки тому як так легше моделювати результат (рис. 2.14).

```
# Data sampling only for good stations
df_indicator = data[['id', 'ds'] + feature_data_all]
df_indicator = df_indicator[df_indicator['id'].isin(stations_good)].dropna().reset_index(dro
p=True)
df_indicator
```

	id	ds	NO3	NO2	Suspended
0	14	2000-10-01	6.30	0.300	9.0
1	14	2000-01-02	8.80	0.270	8.0
2	14	2000-01-03	8.80	0.090	11.0
3	14	2000-04-04	4.60	0.090	13.0
4	14	2000-05-16	3.00	0.050	10.0
...	...	...	...	...	...
758	16	2020-06-10	1.90	0.100	12.0
759	16	2020-03-11	2.06	0.044	9.0
760	16	2020-08-12	3.38	0.060	13.0
761	16	2021-03-16	7.70	6.790	9.0
762	16	2021-06-04	6.38	0.164	9.0

Рисунок 2.14 – Показники по трьох даним

Ось ми бачимо 762 рядків, але це по як ви можете бачити по різних постах, а нам потрібно що у нас окремо виділений «таргет» яким є 14 пост по Suspended. Тому таблиці потрібно перекомпілювати (рис. 2.15).

```
cols = []
for station in stations_good:
    for feature in feature_data_all:
        cols.append(str(station) + "_" + feature)
cols
```

```
['14_NO3',
 '14_NO2',
 '14_Suspended',
 '15_NO3',
 '15_NO2',
 '15_Suspended',
 '16_NO3',
 '16_NO2',
 '16_Suspended']
```

Рисунок 2.15 – Синтезування даних прогнозування

Для подальшого компілювання таблиці потрібно використати команду «pivot_table» з бібліотеки Pandas. Індекс обробимо дату, номер колонок це будуть номери постів, а значення будуть ті самі значення що і в таблиці (рис. 2.15). При цьому для того щбо підписи стовпців були більш зручними й не багаторівневими спеціальним циклом ми синтезуємо їх назви роблячи парами номер стовпця через знак «_» і показників. Маємо ось таку табличку (рис 2.16).

```
df = pd.pivot_table(df_indicator, index=["ds"], columns=["id"], values=feature_data_all).dro
pna()
df.columns = cols
df
```

	14_NO3	14_NO2	14_Suspended	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspended
ds									
2000-01-02	0.270	0.130	0.130	8.80	8.40	7.70	8.0	12.0	9.0
2000-01-03	0.090	0.170	0.270	8.80	9.10	8.80	11.0	17.0	9.0
2000-01-08	0.240	0.150	0.160	1.50	7.00	0.90	18.0	20.0	22.0
2000-04-04	0.090	0.140	0.090	4.60	4.90	3.50	13.0	12.0	18.0
2000-04-07	0.170	0.320	0.360	2.30	2.10	1.70	15.0	18.0	17.0
...	...	...	...	...	...	...	...	...	...
2020-09-15	0.064	0.051	0.072	1.03	0.57	0.84	15.0	13.0	10.0
2020-10-06	0.037	0.038	0.052	0.24	0.44	0.96	6.0	9.0	7.0
2020-12-08	0.061	0.036	0.046	0.68	1.03	0.71	18.0	10.0	12.0
2021-03-16	0.122	0.134	6.790	10.90	8.56	7.70	8.0	10.0	9.0
2021-06-04	0.129	0.179	0.164	6.57	8.07	6.38	8.0	8.0	9.0

Рисунок 2.16 – Таблиця прогнозування

Відповідно третій стовпчик і є той самий «таргет» який ми хочемо прогнозувати. Це вміст Завислих речовин на 14 пості за даними інших двох

створів, які вище потечії.

На графіку видно що в цілому показники однорідні, але можна побачити що у 2006-2012 роках є деякі аномальні дані які дуже різко відрізняються від всіх інших, що демонструється на рисунку 2.17.

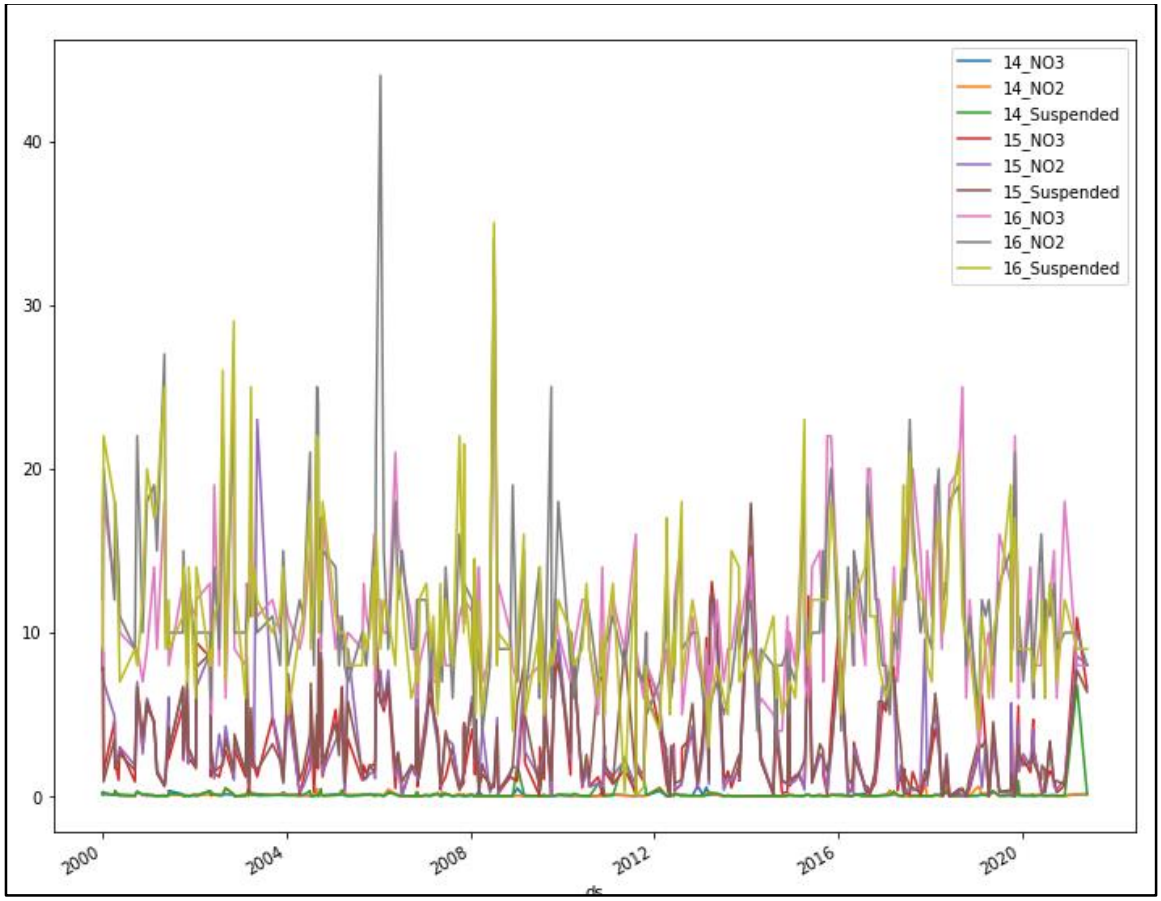


Рисунок 2.17 – Графік показників за роками

2.2 Методологія інформаційної технології

Технологія складатиметься з наступних основних етапів [13]:

- Import libraries
- Download data
- EDA & FE & Preprocessing data Statistics & FE

Data standartization

Training data splitting

- Cross-validation of training data

- Modeling

Linear Regression

Random Forest Regressor

LGBoost Regressor

- Test prediction
- Results visualization
- Select the best model

Коротко дамо характеристику кожному з наведених етапів (рис. 2.18).

1. Import libraries

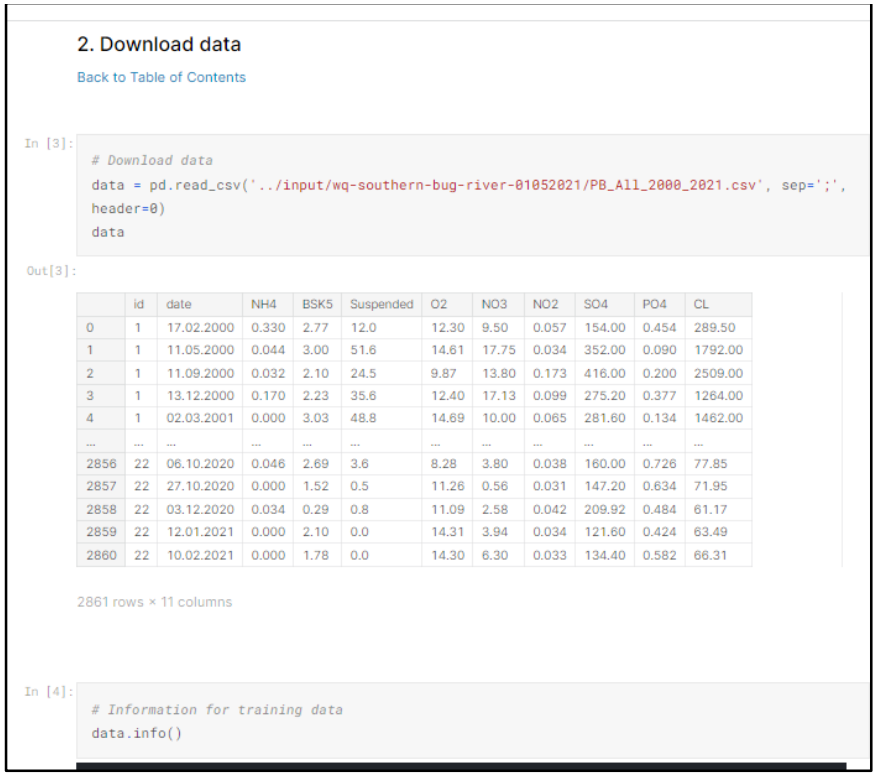
Back to Table of Contents

```
# Work with Data - the main Python libraries
import numpy as np
import pandas as pd
import pandas_profiling as pp

# Visualization
import matplotlib.pyplot as plt

# Preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, KFold, ShuffleSplit, GridSearchCV

# Modeling
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
import lightgbm as lgb
from lightgbm import LGBMRegressor

# Metrics
from sklearn.metrics import r2_score

import warnings
warnings.simplefilter('ignore')
```

```
pd.set_option('max_colwidth', 200)
```

Рисунок 2.18 – Фрагмент коду блока Import libraries

Наступним етапом після підключення необхідних бібліотек є завантаження даних для роботи (рис. 2.19-2.20).



2. Download data

Back to Table of Contents

In [3]:

```
# Download data
data = pd.read_csv('../input/wq-southern-bug-river-01052021/PB_All_2000_2021.csv', sep=';',
header=0)
data
```

Out[3]:

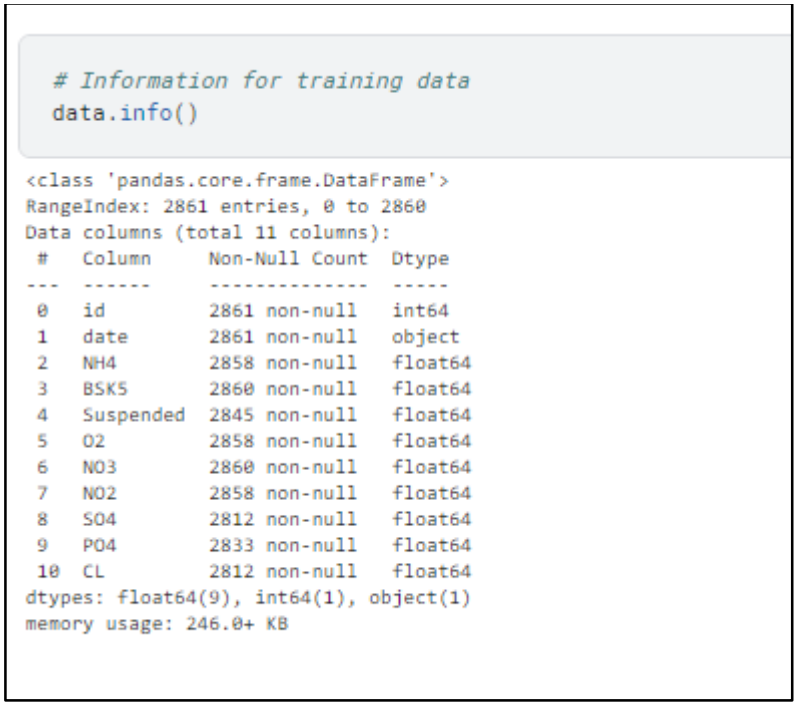
	id	date	NH4	BSK5	Suspended	O2	NO3	NO2	SO4	PO4	CL
0	1	17.02.2000	0.330	2.77	12.0	12.30	9.50	0.057	154.00	0.454	289.50
1	1	11.05.2000	0.044	3.00	51.6	14.61	17.75	0.034	352.00	0.090	1792.00
2	1	11.09.2000	0.032	2.10	24.5	9.87	13.80	0.173	416.00	0.200	2509.00
3	1	13.12.2000	0.170	2.23	35.6	12.40	17.13	0.099	275.20	0.377	1264.00
4	1	02.03.2001	0.000	3.03	48.8	14.69	10.00	0.065	281.60	0.134	1462.00
...	...	...	...	...	...	...	...	...	...	...	...
2856	22	06.10.2020	0.046	2.69	3.6	8.28	3.80	0.038	160.00	0.726	77.85
2857	22	27.10.2020	0.000	1.52	0.5	11.26	0.56	0.031	147.20	0.634	71.95
2858	22	03.12.2020	0.034	0.29	0.8	11.09	2.58	0.042	209.92	0.484	61.17
2859	22	12.01.2021	0.000	2.10	0.0	14.31	3.94	0.034	121.60	0.424	63.49
2860	22	10.02.2021	0.000	1.78	0.0	14.30	6.30	0.033	134.40	0.582	66.31

2861 rows × 11 columns

In [4]:

```
# Information for training data
data.info()
```

Рисунок 2.19 – Фрагмент коду блока Download data



```
# Information for training data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2861 entries, 0 to 2860
Data columns (total 11 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   id          2861 non-null   int64
 1   date        2861 non-null   object
 2   NH4         2858 non-null   float64
 3   BSK5        2860 non-null   float64
 4   Suspended   2845 non-null   float64
 5   O2          2858 non-null   float64
 6   NO3         2860 non-null   float64
 7   NO2         2858 non-null   float64
 8   SO4         2812 non-null   float64
 9   PO4         2833 non-null   float64
 10  CL          2812 non-null   float64
dtypes: float64(9), int64(1), object(1)
memory usage: 246.0+ KB
```

Рисунок 2.20 – Фрагмент коду блока Download data

```
# Download data about monitoring stations
data_about = pd.read_csv('../input/wq-southern-bug-river-01052021/PB_stations.csv', sep=';', header=0, encoding='cp1251')
data_about.sort_values(by=['length'], ascending=False)
```

	id	length	name_station
20	21	773.0	р. Південний Буг, 773 км, смт. Чорний Острів, Мар'янівське вдсх.
19	20	755.0	р. Південний Буг, 755 км, м. Хмельницький , Хмельницьке вдсх.
18	19	744.0	р. Південний Буг, 744 км, с. Копистин, нижче м.Хмельницький
17	18	711.0	р. Південний Буг, 711 км, смт. Меджибіж, Меджибіжське вдсх.
16	17	692.0	р. Південний Буг, 692 км, с. Щедрове, Щедрівське вдсх.
15	16	652.0	р. Південний Буг, 652 км, м. Хмільник, питний в/з, вище міста
14	15	607.0	р. Південний Буг, 607 км, с. Гущинці, нижче села , питний водозабір м.Калинівка
13	14	582.0	р. Південний Буг, 582 км, м. Вінниця, Сабарівське вдсх, питний в/з міста, вище міста
12	13	569.5	р. Південний Буг, 569,5 км, 500 м нижче скиду ВОКВП ВКГ "Вінницяводоканал" (1.5 км нижче греблі Сабарівського вдсх.)
11	12	537.0	р. Південний Буг, 537 км, смт. Сутиски, Сутиське вдсх., н/б'єф
9	10	413.0	р. Південний Буг, 413 км, с. Маньківка, вище села, питний в/з м.Ладижин
8	9	400.0	р. Південний Буг, 400 км, м. Ладижин, Ладижинське вдсх.
7	8	372.0	р. Південний Буг, 372 км, с. Глибочок, Глибочекське вдсх.
6	7	327.0	р. Південний Буг, 327 км, с. Ставки, кордон Вінницької та Кіровоградської обл.
5	6	316.0	р. Південний Буг, 316 км, м.Гайворон, Гайворонське вдсх.
4	5	237.0	р. Південний Буг, 237 км, питний водозабір смт Побузьке
3	4	206.0	р. Південний Буг, 206 км, м. Первомайськ, Первомайське вдсх.
2	3	153.0	р. Південний Буг, 153 км, с. Олексіївка, питний в/з м. Південно-Українськ
1	2	136.0	р. Південний Буг, 136 км, с. Олександрівка, Олександрівське вдсх.
21	22	97.0	р. Південний Буг, 97 км, м. Вознесенськ, пит.в/з м. Вознесенськ, 2 км до в'їзду у м. Вознесенськ по трасі з м. Миколаїв
10	11	50.0	р. Південний Буг, 50 км, с. Ковалівка, Південно-Бузька ЗС

Рисунок 2.21 – Фрагмент коду блока Download data

Наступний досить важливий крок це Statistics & FE (рисунок 2.22).

Після завантаженння та підключення до даних потрібно проаналізувати отримані дані, зробити розвідку EDA & FE & Preprocessing data (рис. 2.23).

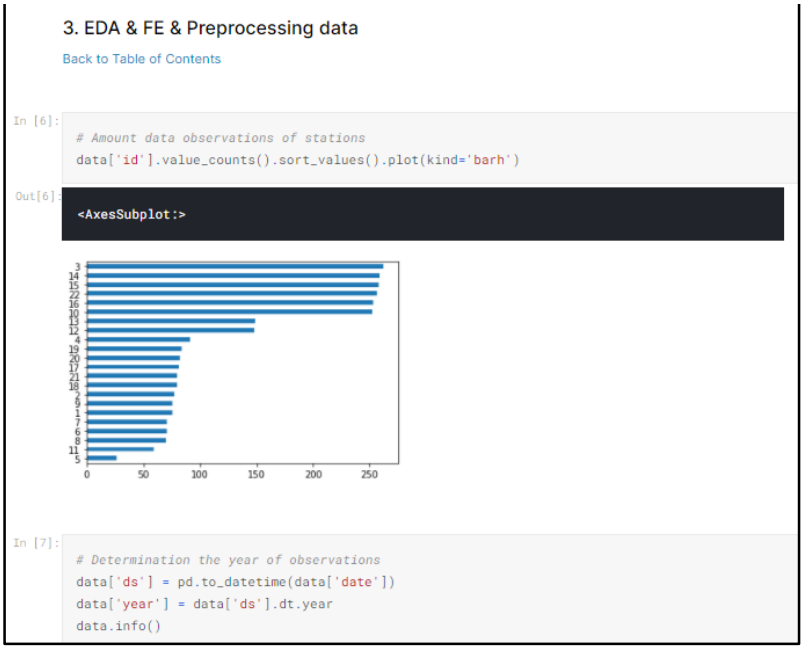


Рисунок 2.23 – Фрагмент коду блока EDA & FE & Preprocessing data

Після EDA & FE & Preprocessing data важливим кроком є Statistics & FE (рис. 2.24 - 2.27).

3.1. Statistics & FE

Back to Table of Contents

The analysis showed that many values are only available in stations 1 and 2, while others have much less data. I propose select only these two stations.

In [24]:
```
# Display the statistics for training data
train.describe([.05, .5, .96])
```

Out[24]:

	14_NO3	14_NO2	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspend
count	130.000000	130.000000	130.000000	130.000000	130.000000	130.000000	130.000000	130.000000
mean	0.059285	0.064815	1.901231	2.138846	2.227769	10.317692	10.507692	10.442308
std	0.040138	0.052150	1.746691	1.857869	1.711474	3.566717	3.280188	3.398378
min	0.000000	0.000000	0.000000	0.000000	0.000000	3.500000	3.000000	3.000000
5%	0.013450	0.015900	0.131500	0.173500	0.242500	5.000000	6.000000	5.450000
50%	0.050000	0.050000	1.330000	1.600000	1.800000	10.000000	10.000000	10.000000
96%	0.148400	0.188240	5.584000	6.436000	6.367200	17.840000	16.680000	17.000000
max	0.180000	0.300000	7.800000	8.070000	7.000000	20.000000	19.000000	19.000000

In [25]:
```
# Display the statistics for test data
test.describe()
```

Out[25]:

	14_NO3	14_NO2	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspended
count	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000
mean	0.071152	0.074333	2.393030	2.513030	2.576970	10.060606	10.363636	9.575758
std	0.053075	0.060727	1.801051	1.973774	1.859994	3.287661	3.806394	3.409823
min	0.009000	0.010000	0.020000	0.100000	0.090000	5.000000	4.000000	5.000000

Рисунок 2.24 – Фрагмент коду Statistics & FE

```
# Determination the start year of observations for all stations
data[['id', 'year']].groupby(by=['id']).min().sort_values(by=['year'], ascending=False)
```

id	year
5	2019
13	2006
1	2000
21	2000
20	2000
19	2000
18	2000
17	2000
16	2000
15	2000
14	2000
12	2000
2	2000
11	2000
10	2000
9	2000
8	2000
7	2000
6	2000
4	2000
3	2000
22	2000

Рисунок 2.25 – Фрагмент коду Statistics & FE

```
# Data sampling only for good stations
df_indicator = data[['id', 'ds'] + feature_data_all]
df_indicator = df_indicator[df_indicator['id'].isin(stations_good)].dropna().reset_index(drop=True
df_indicator
```

	id	ds	NO3	NO2	Suspended
0	14	2000-10-01	6.30	0.300	9.0
1	14	2000-01-02	8.80	0.270	8.0
2	14	2000-01-03	8.80	0.090	11.0
3	14	2000-04-04	4.60	0.090	13.0
4	14	2000-05-16	3.00	0.050	10.0
...	...	...	...	...	...
758	16	2020-06-10	1.90	0.100	12.0
759	16	2020-03-11	2.06	0.044	9.0
760	16	2020-08-12	3.38	0.060	13.0
761	16	2021-03-16	7.70	6.790	9.0
762	16	2021-06-04	6.38	0.164	9.0

763 rows × 5 columns

Рисунок 2.26 – Фрагмент коду Statistics & FE

```
df = pd.pivot_table(df_indicator, index=["ds"], columns=["id"], values=feature_data_all).dropna()
df.columns = cols
df
```

ds	14_NO3	14_NO2	14_Suspended	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspended
2000-01-02	0.270	0.130	0.130	8.80	8.40	7.70	8.0	12.0	9.0
2000-01-03	0.090	0.170	0.270	8.80	9.10	8.80	11.0	17.0	9.0
2000-01-08	0.240	0.150	0.160	1.50	7.00	0.90	18.0	20.0	22.0
2000-04-04	0.090	0.140	0.090	4.60	4.90	3.50	13.0	12.0	18.0
2000-04-07	0.170	0.320	0.360	2.30	2.10	1.70	15.0	18.0	17.0
...	...	...	...	...	...	...	...	...	...
2020-09-15	0.064	0.051	0.072	1.03	0.57	0.84	15.0	13.0	10.0
2020-10-06	0.037	0.038	0.052	0.24	0.44	0.96	6.0	9.0	7.0
2020-12-08	0.061	0.036	0.046	0.68	1.03	0.71	18.0	10.0	12.0
2021-03-16	0.122	0.134	6.790	10.90	8.56	7.70	8.0	10.0	9.0
2021-06-04	0.129	0.179	0.164	6.57	8.07	6.38	8.0	8.0	9.0

238 rows × 9 columns

Рисунок 2.27 – Фрагмент коду Statistics & FE

На рисунку 2.28 та 2.29 приведено результати Statistics & FE для 2021 року.

```
# Display the statistics for test data
test.describe()
```

	14_NO3	14_NO2	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspended
count	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000
mean	0.071152	0.074333	2.393030	2.513030	2.576970	10.060606	10.363636	9.575758
std	0.053075	0.060727	1.801051	1.973774	1.859994	3.287661	3.806394	3.409823
min	0.009000	0.010000	0.020000	0.100000	0.090000	5.000000	4.000000	5.000000
25%	0.040000	0.029000	1.000000	0.800000	1.020000	7.000000	8.000000	7.000000
50%	0.058000	0.060000	2.200000	2.600000	1.990000	10.000000	9.000000	9.000000
75%	0.080000	0.107000	3.300000	3.900000	4.000000	12.000000	12.000000	12.000000
max	0.250000	0.280000	6.820000	7.060000	6.890000	20.000000	20.000000	19.000000

Рисунок 2.28 – Фрагмент коду Statistics & FE

На рисунку 2.29 наведено приклад візуалізації Statistics & FE.

```
# Data visualization
df.plot(figsize=(12,10))
```

```
<AxesSubplot:xlabel='ds'>
```

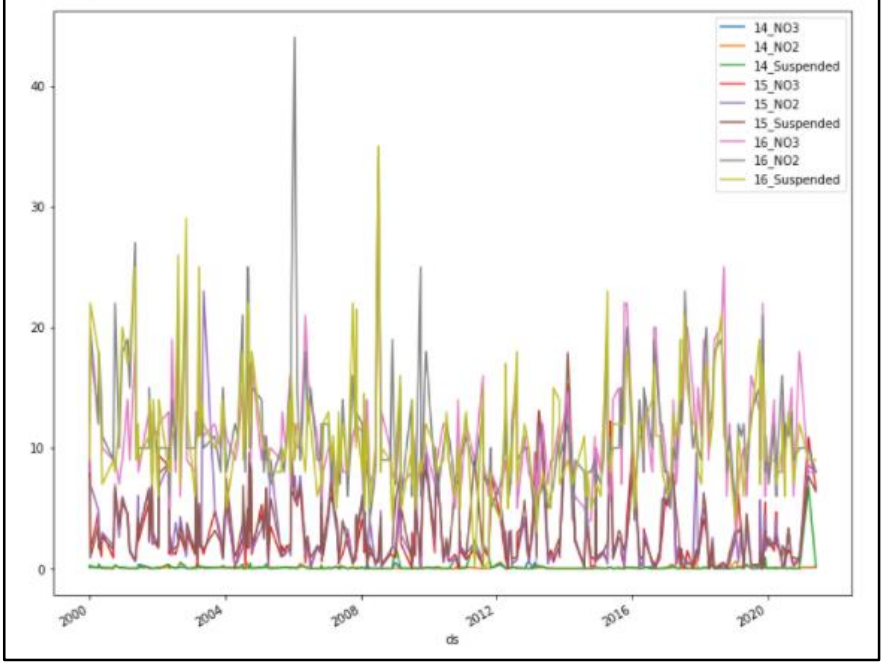


Рисунок 2.29 – Фрагмент коду Statistics & FE

Далі йде етап Data standartization (рис. 2.30).

3.2. Data standartization

Back to Table of Contents

In [26]:

```
# Standartization data
scaler = StandardScaler()
train = pd.DataFrame(scaler.fit_transform(train), columns = train.columns)

# Display training data
train
```

Out[26]:

	14_NO3	14_NO2	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspended
0	0.768205	0.292298	1.665998	1.491944	0.570264	0.473493	0.150666	-0.130656
1	-1.107577	-0.862679	0.631491	0.573377	-0.133599	-0.652325	-1.379534	-0.721449
2	-0.232212	-0.670183	-0.403016	-1.047623	-0.186388	-0.652325	-0.767454	-0.130656
3	-1.232630	-0.862679	0.574019	0.627410	0.922195	-0.370870	-0.155374	0.755533
4	-1.132588	-0.708682	-0.989236	-1.053027	-1.077947	0.754948	-0.461414	-0.130656
...	...	...	...	...	...	...	...	...
125	2.018727	0.484794	2.068306	2.356477	2.623196	0.754948	-0.155374	0.164740
126	-1.232630	-0.285191	-0.994983	-1.107060	-1.236316	-1.496688	-0.767454	0.164740
127	-0.407285	-0.670183	-0.403016	-0.345190	-0.544185	-0.089416	-0.461414	-0.426052
128	-1.182609	-0.901179	-0.788082	-0.901733	-0.866789	0.192039	-0.155374	0.460137
129	0.267996	-0.670183	-0.747851	-0.615357	-0.426875	0.192039	-0.920474	-0.721449

130 rows × 8 columns

In [27]:

```
# Standartization data
test = pd.DataFrame(scaler.transform(test), columns = test.columns)
```

Рисунок 2.30 – Фрагмент коду Data standartization

```
# Determination the final year of observations for all stations
data[['id', 'year']].groupby(by=['id']).max().sort_values(by=['year'], ascending=False)
```

id	year
22	2021
14	2021
3	2021
5	2021
10	2021
16	2021
15	2021
21	2018
20	2018
19	2018
18	2018
17	2018
1	2018
13	2018
2	2018
11	2018
9	2018
8	2018
7	2018
6	2018
4	2018
12	2018

Рисунок 2.31 – Фрагмент коду Data standartization

Приклад реалізації Data standartization наведено на рисунку 2.32.

Although, if you limit yourself to 2018, then you can take all the stations.

Consider only stations 14, 15, 16.

```
# Information about stations 14, 15, 16
stations_good = [14, 15, 16]
data_about[data_about['id'].isin(stations_good)]
```

	id	length	name_station
13	14	582.0	р. Південний Буг, 582 км, м. Вінниця, Сабарівське вдсх, питний в/з міста, вище міста
14	15	607.0	р. Південний Буг, 607 км, с. Гущинці, нижче села , питний водозабір м.Калинівка
15	16	652.0	р. Південний Буг, 652 км, м. Хмільник, питний в/з, вище міста

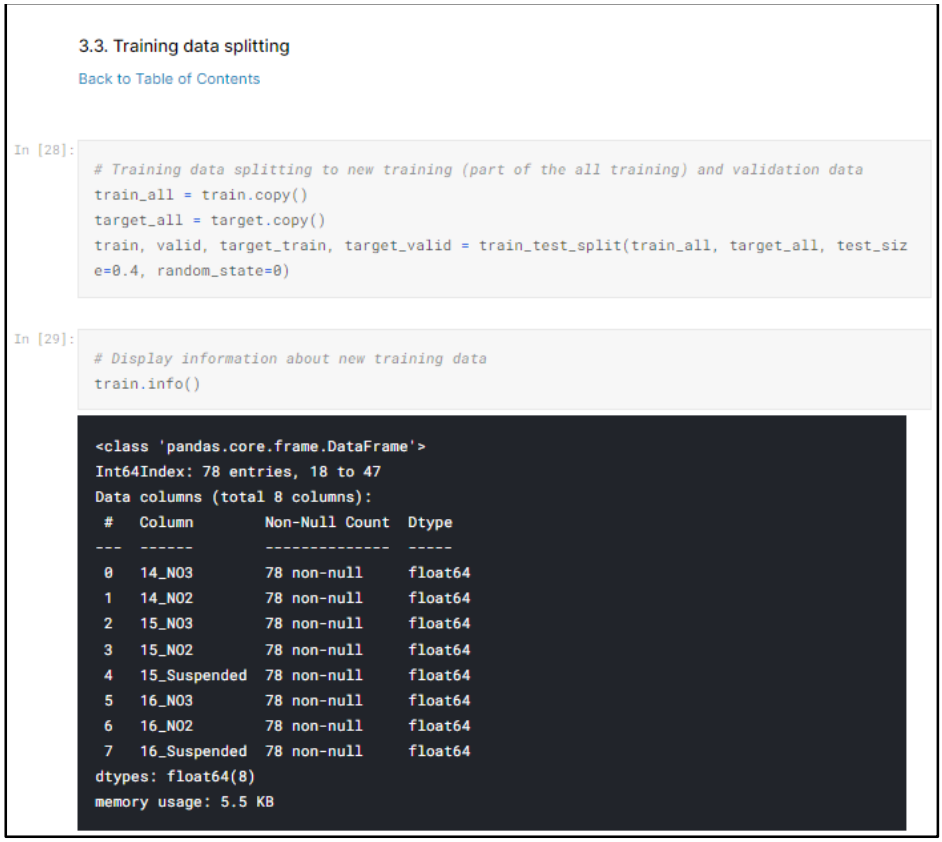
```
# Set target indicator
target_data_name = 'Suspended'
#feature_target_all = ['NH4', 'BSK5', 'NO3', 'NO2', 'SO4', 'PO4', 'CL']
feature_target_all = ['NO3', 'NO2']
feature_data_all = feature_target_all + [target_data_name]
feature_data_all
```

```
['NO3', 'NO2', 'Suspended']
```

```
# Data sampling only for good stations
df_indicator = data[['id', 'ds'] + feature_data_all]
df_indicator = df_indicator[df_indicator['id'].isin(stations_good)].dropna().reset_index(drop=True)
df_indicator
```

Рисунок 2.32 – Фрагмент коду Data standartization

Далі йде крок Training data splitting (рис.2.33 - 2.34).



3.3. Training data splitting

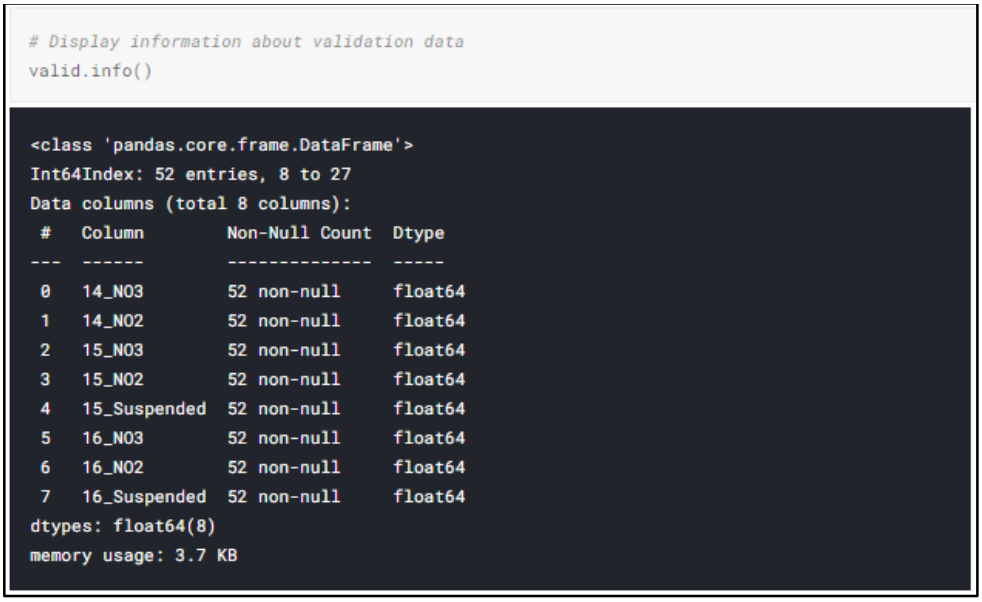
Back to Table of Contents

In [28]:
```
# Training data splitting to new training (part of the all training) and validation data
train_all = train.copy()
target_all = target.copy()
train, valid, target_train, target_valid = train_test_split(train_all, target_all, test_siz
e=0.4, random_state=0)
```

In [29]:
```
# Display information about new training data
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 78 entries, 18 to 47
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   14_NO3        78 non-null     float64
 1   14_NO2        78 non-null     float64
 2   15_NO3        78 non-null     float64
 3   15_NO2        78 non-null     float64
 4   15_Suspended  78 non-null     float64
 5   16_NO3        78 non-null     float64
 6   16_NO2        78 non-null     float64
 7   16_Suspended  78 non-null     float64
dtypes: float64(8)
memory usage: 5.5 KB
```

Рисунок 2.33 – Фрагмент коду Data standartization



```
# Display information about validation data
valid.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 52 entries, 8 to 27
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   14_NO3        52 non-null     float64
 1   14_NO2        52 non-null     float64
 2   15_NO3        52 non-null     float64
 3   15_NO2        52 non-null     float64
 4   15_Suspended  52 non-null     float64
 5   16_NO3        52 non-null     float64
 6   16_NO2        52 non-null     float64
 7   16_Suspended  52 non-null     float64
dtypes: float64(8)
memory usage: 3.7 KB
```

Рисунок 2.34– Фрагмент коду Data standartization

Логічно що після етапу Training data splitting йтиме Cross-validation of training data (рисунок 2.35).

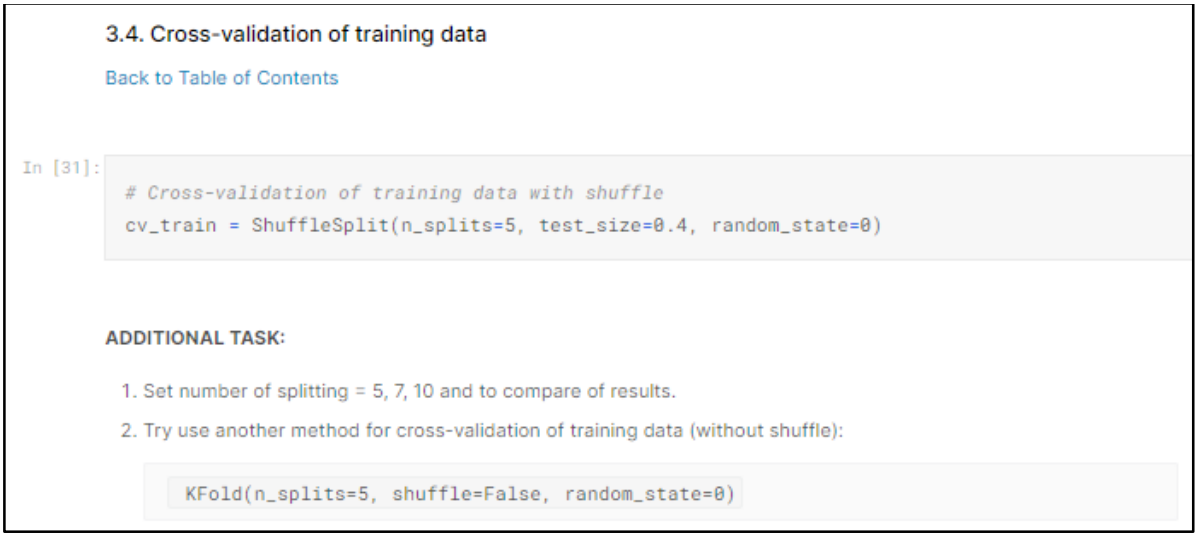


Рисунок 2.35 – Фрагмент коду Cross-validation of training data

Після етапу Cross-validation of training data йтиме крок про Modeling (рисунок 2.36).

Рисунок 2.36 – Фрагмент коду Modeling

Оскільки моделювання проходитиме по декільком моделям то матимемо окремі етами для кожної з них. Парад моделей розпочинає Linear Regression (рис.2.37).

Рисунок 2.37 – Фрагмент коду Linear Regression

Продовжує після Linear Regression модель Random Forest Regressor (рис. 2.38)

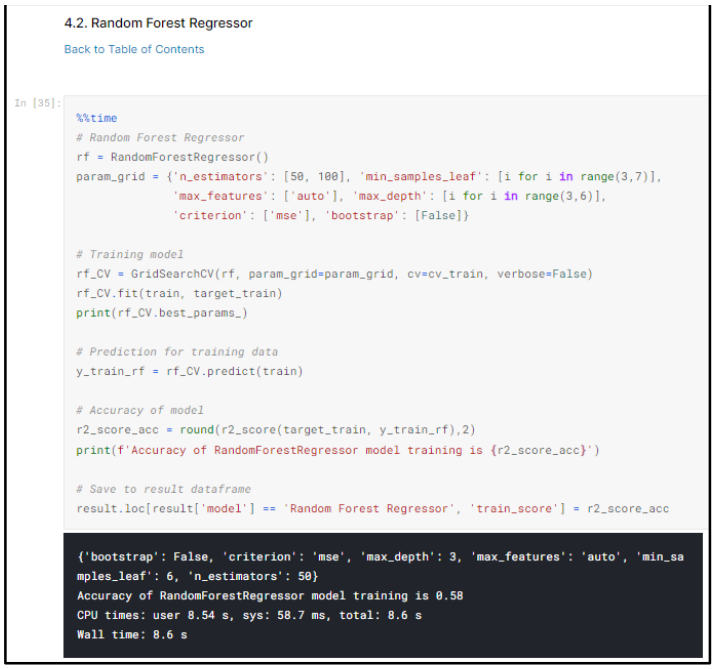


Рисунок 2.38 – Фрагмент коду Random Forest Regressor

Наступна фінальна модель LGBoost Regressor (рис. 2.39).

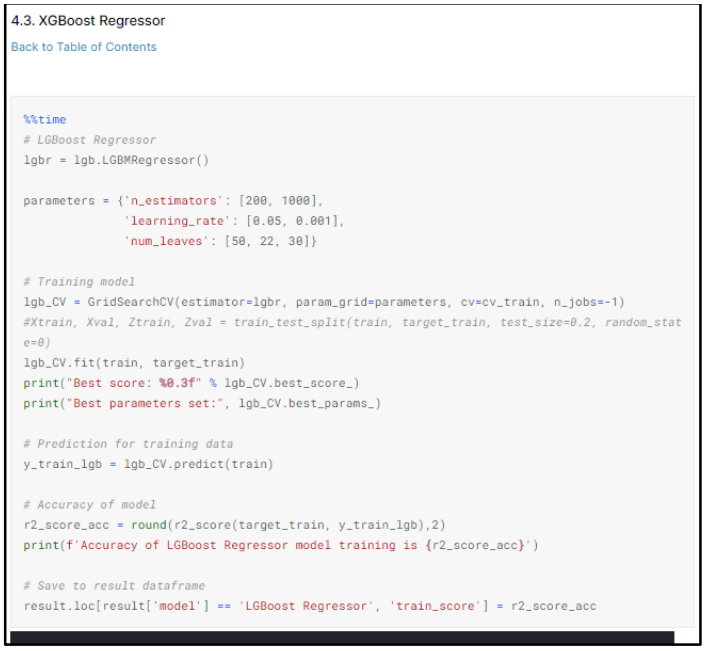


Рисунок 2.39 – Фрагмент коду Random Forest Regressor

Слідуючим кроком буде Test prediction (рис. 2.40).

5. Test prediction

Back to Table of Contents

```
# Prediction of target for test data for all models
y_test_lr = lr.predict(test)
y_test_rf = rf_CV.predict(test)
y_test_lgb = lgb_CV.predict(test)
```

Рисунок 2.40 – Фрагмент коду Test prediction

Передостаннім етапом є Results visualization (рисунок 2.41-2.42).

6. Visualization

Back to Table of Contents

```
# Building plot for prediction for the training data
x = np.arange(len(train))
plt.figure(figsize=(16,10))
plt.scatter(x, target_train, label = "Target training data", color = 'g')
plt.scatter(x, y_train_lr, label = "Linear Regression prediction", color = 'b')
plt.scatter(x, y_train_rf, label = "Random Forest prediction", color = 'y')
plt.scatter(x, y_train_lgb, label = "LGBoost Regressor prediction", color = 'brown')
plt.plot(x, np.full(len(train), 0.5), label = "Maximum allowable value", color = 'r')
plt.title('Prediction for the training data')
plt.legend(loc='best')
plt.grid(True)
```

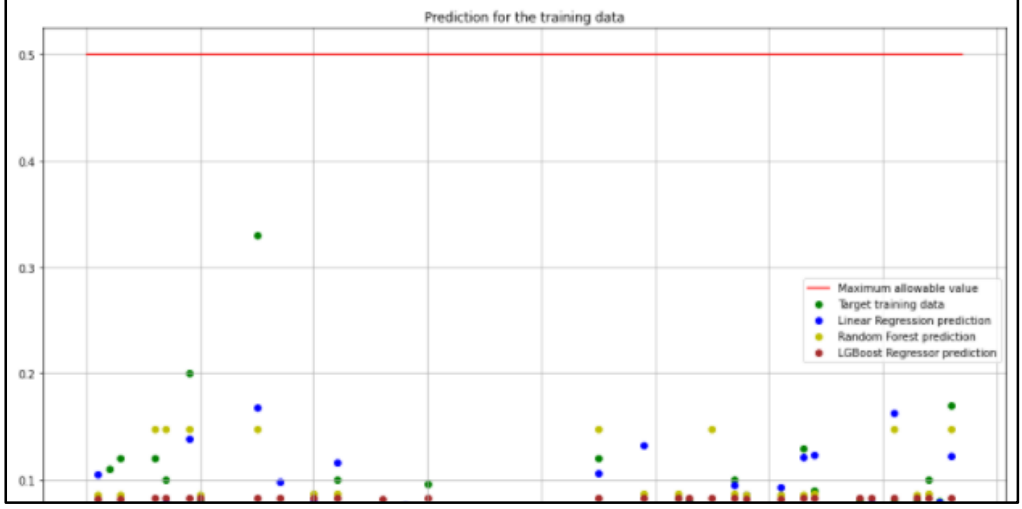


Рисунок 2.41 – Фрагмент коду Results visualization

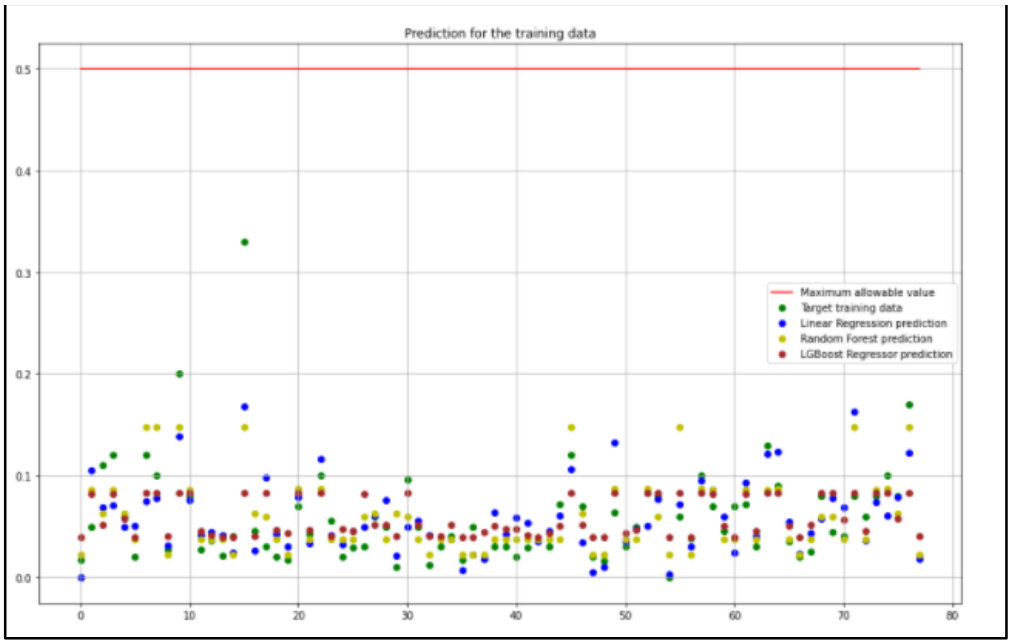


Рисунок 2.42 – Фрагмент коду Results visualization

```
# Building plot for prediction for the valid data
x = np.arange(len(valid))
plt.figure(figsize=(16,10))
plt.scatter(x, target_valid, label = "Target valid data", color = 'g')
plt.scatter(x, y_val_lr, label = "Linear Regression prediction", color = 'b')
plt.scatter(x, y_val_rf, label = "Random Forest prediction", color = 'y')
plt.scatter(x, y_val_lgb, label = "LGBoost Regressor prediction", color = 'brown')
plt.plot(x, np.full(len(valid), 0.5), label = "Maximum allowable value", color = 'r')
plt.title('Prediction for the valid data')
plt.legend(loc='best')
plt.grid(True)
```

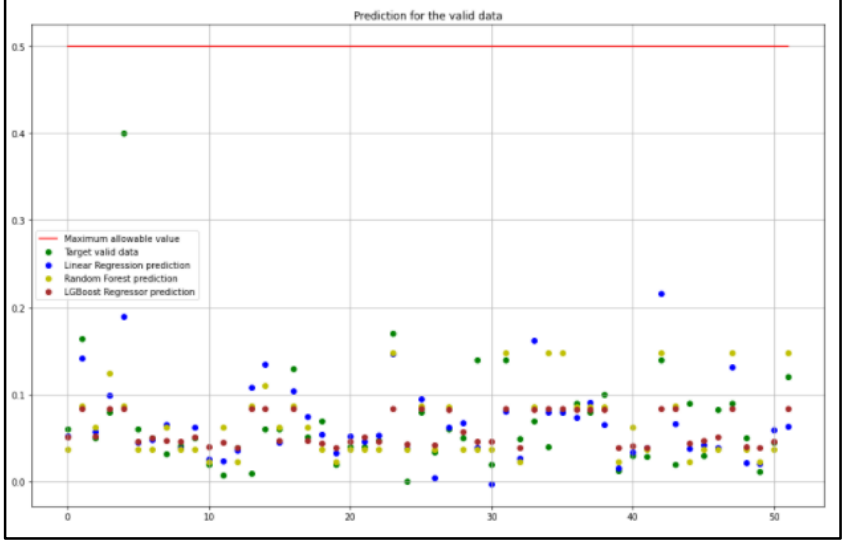


Рисунок 2.43 – Фрагмент коду Results visualization

```
# Building plot for prediction for the test data
x = np.arange(len(test))
plt.figure(figsize=(16,10))
plt.scatter(x, target_test, label = "Target test data", color = 'g')
plt.scatter(x, y_test_lr, label = "Linear Regression prediction", color = 'b')
plt.scatter(x, y_test_rf, label = "Random Forest prediction", color = 'y')
plt.scatter(x, y_test_lgb, label = "LGBoost Regressor prediction", color = 'brown')
plt.plot(x, np.full(len(test), 0.5), label = "Maximum allowable value", color = 'r')
plt.title('Prediction for the test data')
plt.legend(loc='best')
plt.grid(True)
```

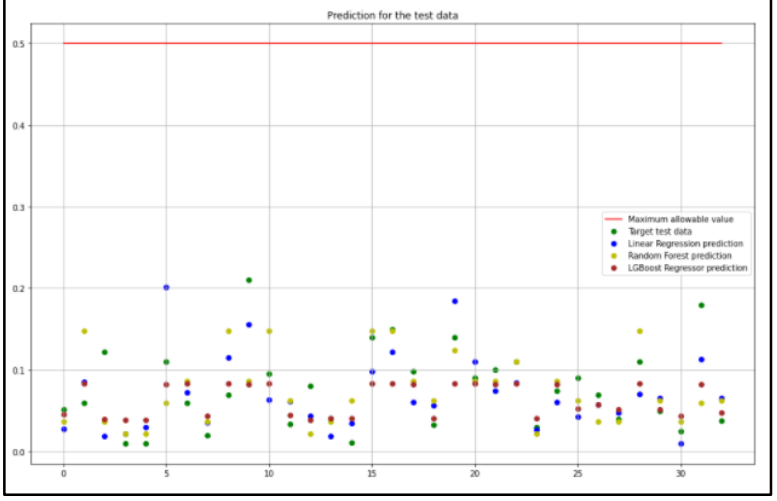


Рисунок 2.44 – Фрагмент коду Results visualization

Заключним етапом методології даної технології буде кроком буде Select the best model (рис. 2.45).

7. Select the best model

Back to Table of Contents

```
# Display results of modeling
result.sort_values(by=['valid_score', 'train_score'], ascending=False)
```

	model	train_score	valid_score
0	Linear Regression	0.54	0.43
2	LGBoost Regressor	0.35	0.20
1	Random Forest Regressor	0.58	0.13

```
# Select models with good training results
result_best = result[(result['train_score'] > result['valid_score'])]
```

```
# Select models with minimal overfitting
result_best = result_best[(result_best['train_score'] - result_best['valid_score']).abs() < 0.15]
result_best.sort_values(by=['valid_score', 'train_score'], ascending=False)
```

Рисунок 2.45– Фрагмент коду Select the best model

2.3 Формалізація інформаційної моделі аналізу та прогнозування змін концентрації завислих речовин у річці Південний

Для того щоб можна було будувати далі інформаційну систему потрібно провести розвідувальний аналіз.

Є потреба перетворити дані або як варіант представити їх в зручній формі, коректній формі. Результатами розвідувального аналізу зазвичай зображаються як схеми, графіки, діаграми, таблиці тощо. Методами за допомогою яких відбувається аналіз називають:

- Кластерний аналіз;
- Аналіз часових рядів;
- Аналіз відповідностей;
- Дерева класифікацій;
- Факторний аналіз;
- Покрокова лінійна та нелінійна регресія;
- Багатовимірне шкалювання;
- Аналіз дискримінантних функцій;
- Логлінійний аналіз;
- Канонічні кореляції.

Зазвичай результатами розвідувального аналізу не користуються за для вироблення управлінських рішень. Їхнє застосування — допомога в розробці найкращої стратегії для поглибленого вивчення, аналізу, висування гіпотез, чи інших математичних методів [8].

Виходячи з графіку на рисунку 2.43 ми вже знаємо що у нас є аномальні дані які погано впливають на кінцевий результат прогнозування, тому потрібно відтворити новий фільтр даних щоб зменшити розкид між ними (рис. 2.46).

```
print(len(df))
for col in df.columns.tolist():
    df = df[df[col] <= float(df.quantile([.96])[col])]
df = df.reset_index(drop=True)
print(len(df))
df.describe()
```

```
238
163
```

Рисунок 2.46 – Фільтр даних

Тепер на побудованій таблиці після фільтрації даних видно що розкид між даними зменшився і вони стали схожі на більш однорідні (рис. 2.47).

	14_NO3	14_NO2	14_Suspended	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2
count	163.000000	163.000000	163.000000	163.000000	163.000000	163.000000	163.000000	163.0000
mean	0.061687	0.066742	0.065417	2.000798	2.214601	2.298466	10.265644	10.47852
std	0.043153	0.053933	0.053746	1.763373	1.881712	1.742312	3.503691	3.381298
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	3.500000	3.000000
25%	0.030000	0.030000	0.030000	0.665000	0.705000	1.040000	8.000000	8.000000
50%	0.050000	0.050000	0.050000	1.400000	1.600000	1.900000	10.000000	10.00000
75%	0.080000	0.081500	0.083000	2.940000	3.155000	3.000000	12.000000	13.00000
max	0.250000	0.300000	0.400000	7.800000	8.070000	7.000000	20.000000	20.00000

Рисунок 2.47 – Таблиці відфільтрованих даних

Для того щоб до кінця переконатися побудуємо графік (рис.2.48).

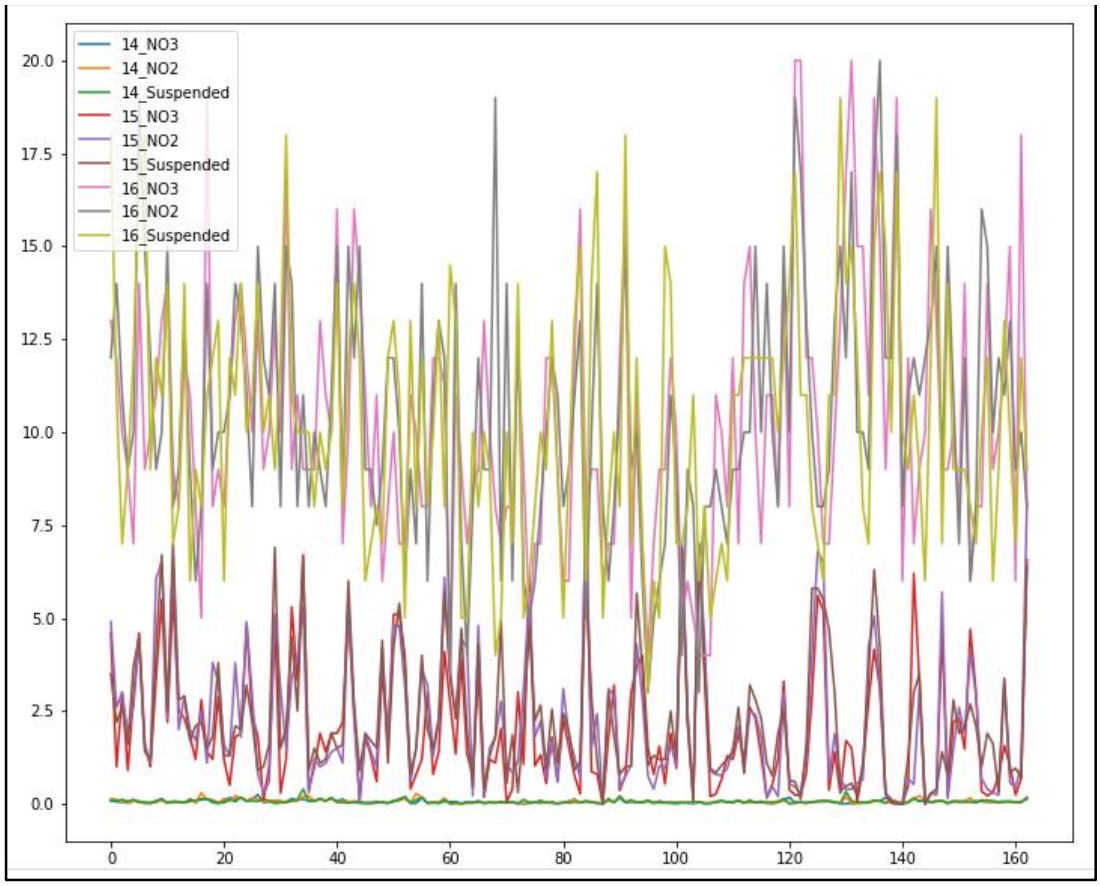


Рисунок 2.48 – Графік показників

Видно що дані однорідні, тепер маємо змогу розпочати подальшу розробку прогнозування «таргету» для цього ми виділяємо його щоб він був окремо рисунок 2.49.

```
# Set target data
target_name = '14_' + target_data_name
target_data = df.pop(target_name)
target_data
```

```
0      0.090
1      0.080
2      0.120
3      0.070
4      0.120
       ...
158    0.060
159    0.072
160    0.052
161    0.046
162    0.164
Name: 14_Suspended, Length: 163, dtype: float64
```

Рисунок 2.49 – Виділений «таргет»

Зараз розіб'ємо наші дані на тренувальні й тестові за допомогою команди «train_test_split», тестовий обсяг буде 40% рисунок 2.50.

```
# Dividing data into training and test
train, test, target, target_test = train_test_split(df, target_data, test_size=0.2, random_s
tate=0)
print(train.shape, test.shape)
```

```
(130, 8) (33, 8)
```

Рисунок 2.50 – Розподіл даних

Далі можна побачити статистику і кількість даних які пішли до тренувальної та тестової дати на рисунках 2.51-2.52.

```
# Display the statistics for training data
train.describe([.05, .5, .96])
```

	14_NO3	14_NO2	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspend
count	130.000000	130.000000	130.000000	130.000000	130.000000	130.000000	130.000000	130.000000
mean	0.059285	0.064815	1.901231	2.138846	2.227769	10.317692	10.507692	10.442308
std	0.040138	0.052150	1.746691	1.857869	1.711474	3.566717	3.280188	3.398378
min	0.000000	0.000000	0.000000	0.000000	0.000000	3.500000	3.000000	3.000000
5%	0.013450	0.015900	0.131500	0.173500	0.242500	5.000000	6.000000	5.450000
50%	0.050000	0.050000	1.330000	1.600000	1.800000	10.000000	10.000000	10.000000
96%	0.148400	0.188240	5.584000	6.436000	6.367200	17.840000	16.680000	17.000000
max	0.180000	0.300000	7.800000	8.070000	7.000000	20.000000	19.000000	19.000000

Рисунок 2.51 – Статистика тренувальних даних

```
# Display the statistics for test data
test.describe()
```

	14_NO3	14_NO2	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspended
count	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000	33.000000
mean	0.071152	0.074333	2.393030	2.513030	2.576970	10.060606	10.363636	9.575758
std	0.053075	0.060727	1.801051	1.973774	1.859994	3.287661	3.806394	3.409823
min	0.009000	0.010000	0.020000	0.100000	0.090000	5.000000	4.000000	5.000000
25%	0.040000	0.029000	1.000000	0.800000	1.020000	7.000000	8.000000	7.000000
50%	0.058000	0.060000	2.200000	2.600000	1.990000	10.000000	9.000000	9.000000
75%	0.080000	0.107000	3.300000	3.900000	4.000000	12.000000	12.000000	12.000000
max	0.250000	0.280000	6.820000	7.060000	6.890000	20.000000	20.000000	19.000000

Рисунок 2.52 - Статистика тестових даних

Для поліпшення роботи з даними потрібно їх стандартизувати у двох вибірках train і test використовуючи команду «StandardScaler» рисунок 2.53.

```
# Standartization data
scaler = StandardScaler()
train = pd.DataFrame(scaler.fit_transform(train), columns = train.columns)

# Display training data
train
```

Рисунок 2.53 – Стандартизація test і train

У свою чергу тренувальні дані ми ще раз розділимо виділимо в них валідаційні, це та частина «датасету» які необхідна для вибору найкращої моделі щоб її потому застосувати для тестових даних. Тут валідаційні дані вибираються 20% рисунок 2.54.

```
# Training data splitting to new training (part of the all training) and validation data
train_all = train.copy()
target_all = target.copy()
train, valid, target_train, target_valid = train_test_split(train_all, target_all, test_size
=0.4, random_state=0)
```

Рисунок 2.54 – Вибірка валідаційних даних

Виконуємо трьох разову кросс-валідацію для тренувальної дати що відображено на рисунку 2.55.

```
# Cross-validation of training data with shuffle
cv_train = ShuffleSplit(n_splits=5, test_size=0.4, random_state=0)
```

Рисунок 2.55– Валідація train

Згідно з вище вказаних підготовлених даних можна сказати, що у нас є все для побудови подальшої лінійної моделі. В котрій вже можливо буде підбивати підсумки прогнозування завислих речовин в річці Південний Буг.

2.4 Огляд та вибір технологій для інформаційної системи аналізу та прогнозування завислих речовин у річковій воді

Оскільки це буде інформаційна система, яка використовує дані моніторингу, то варто скористатись наступними інструментами машиного :

- Linear Regression;
- Random Forest Regressor;
- LGBoost Regressor.

Дамо невелику характеристику кожного ыз запропонованих ынструментів [14].

Linear Regression. Лінійна регресія – один із найпростіших інструментів статистичного моделювання, але саме у його простоті і полягає його ефективність. Спочатку необхідно зрозуміти значення терміна. У спрощеному вигляді регресійний аналіз можна визначити як спосіб опису взаємовідносин між залежною змінною та набором незалежних змінних. Наприклад, за допомогою цього методу за наявності набору даних про зростання, вагу, поле та інші фізичні параметри дитини, ми можемо описати вплив цих змінних на цільову змінну – зростання цієї людини у дорослому віці.

Що означає сам термін "регресія?" В основному це просто данина традиції: він виник кілька століть тому і міцно закріпився в побуті. Напевно, вам знайомий термін "регресія до середнього значення". За ним ховається думка про те, що дані можуть відхилятися від очікуваного "середнього" значення, але якщо спостережень багато, вони повертаються або "регресують" до прогнозованого значення. Наприклад, у високої людини майже, напевно, будуть високі діти. Вони можуть “регресувати” до середнього значення зростання, але навряд чи виявляться вищими за свого батька.

Зверніть увагу, що регресія моделює числові відносини, тобто. вихідні дані регресійного аналізу – завжди число. Цим він відрізняється від категоризації, результатом якої є клас об'єктів або позначка для вихідних

даних. Друга частина терміна "лінійна регресія" наголошує на лінійному характері залежності між змінними.

Звернемося до відомого прикладу з даними про різні сорти квіток ірису. Розглянемо залежність між довжиною чашолистки та довжиною пелюстки квітів сорту Virginica. Лінійна регресія визначить залежність цих двох змінних за допомогою лінії найкращої відповідності [15].

Ви виділили, як визначаються умови лінійної моделі шляхом мінімізації рівняння залишку регресії (рівняння нев'язки), але сама модель поки що не завершена. На відміну від моделей машинного навчання, які можуть містити мільйони прихованих параметрів, кожен параметр лінійної регресії можна аналізувати. Модель регресії створює кілька метрик, які можуть бути використані надалі. Насамперед, це R2 або коефіцієнт детермінації. R2 дозволяє виміряти, наскільки модель може пояснити дисперсію даних. Якщо R-квадрат дорівнює 1 це означає, що модель описує всі дані. Якщо R-квадрат дорівнює 0,5, модель пояснює лише 50% дисперсії даних. Відхилення, що залишилися, не мають пояснення. Що ближче R2 до одиниці, то краще.

Крім того, для кожного коефіцієнта моделі ми можемо розрахувати p-величину для того, щоб визначити статистичну значущість оцінки. Якщо p-величини великі, варто подумати про те, чи варто включати такі змінні в модель, оскільки вони можуть бути марними. Замість того, щоб видаляти створену модель, рекомендується додавати по одній змінній один раз, а потім перевіряти, як змінилася статистична значущість моделі після додавання змінної. Такий підхід називають дисперсійним аналізом AnoVa (англ. Analysis of Variance). Ще один спосіб перевірки моделі полягає в аналізі розподілу залишків регресії. У ідеалі вони повинні мати нормальне розподілення і бути гомоскедастичними, тобто. вони повинні зберігати більш менш постійну дисперсію для підібраних значень і вихідних даних моделі. Важливо, щоб залишок регресії був однаковим за всіма даними, оскільки це гарантує чітке відпрацювання моделі.

Головний недолік лінійної регресії у тому, що може моделювати лише прямі лінійні залежності, тоді як часто виникає необхідність створення моделі інших типів відносин між даними. На щастя, існують прості способи відображення даних, між якими немає лінійної залежності, за допомогою лінійної регресії. Перший спосіб - перетворення вихідних даних. Замість використання змінних для створення моделі в їх вихідному вигляді, на практиці часто доводиться використовувати різні перетворення, такі як визначення логарифму значень або зведення в ступінь. Навіть якщо між значеннями немає прямої лінійної залежності, вона може існувати між логарифмами цих значень. У такому випадку модель покаже, що зі збільшенням залежної змінної на 1% цільова змінна збільшиться на X%.

Інший спосіб передбачає включення до моделі квадратних членів. Для цього дані зводяться в квадрат і додаються до рівняння як додаткова змінна. Ще одна технологія перетворення даних передбачає врахування взаємодії предикторів, коли вихідні змінні розглядаються у комбінаціях. Наприклад, якщо в модель додати такі змінні як Пол і Зростання, модель можна включити об'єднаний член Пол:Рост. Фактично при цьому створюються дві нові змінні - Чоловік: Зростання (вона додаватиме в модель тільки зростання чоловіків) і Жінка: Зростання (зростання жінок). Така взаємодія предикторів дозволить створити модель залежності зростання від статі окремо для жінок та чоловіків [16].

Моделі лінійної регресії дуже популярні в різних сферах досліджень завдяки швидкості їх створення та простоті інтерпретації. Завдяки можливості перетворення даних вони можуть бути використані для моделювання широкого спектра залежностей. Завдяки простоті форми, в порівнянні з нейронними мережами, їх статистичні параметри легко піддаються аналізу та порівнянню, що дозволяє вилучати з них цінну інформацію. Лінійна регресія використовується у прогностичних цілях; вона також показала свою ефективність у описі систем. Якщо ви хочете змоделювати значення числової змінної, у вас є відносно невеликий список незалежних змінних, і ви

розраховуєте на те, що модель буде вам зрозуміла, ви, швидше за все, оберете саме лінійну регресію як інструмент моделювання. Система PolyAnalyst пропонує користувачам можливість порівняти різні інструменти моделювання та вибрати ту технологію, яка оптимально підходить для використання із конкретними вихідними даними.

Random Forest Regressor. Випадковий ліс — один із найбільш приголомшливих алгоритмів машинного навчання, придумані Лео Брейманом та Адель Катлер ще у минулому столітті. Він дійшов до нас у «первозданном вигляді» (ніякі евристики не змогли його суттєво покращити) і є одним із небагатьох універсальних алгоритмів. Універсальність полягає, по-перше, у тому, що він добрий у багатьох завданнях (за моїми оцінками, 70% з тих, що зустрічаються на практиці, якщо не враховувати задачі із зображеннями), по-друге, у тому, що є випадкові ліси для вирішення завдань класифікації, регресії, кластеризації, пошуку аномалій, селекції ознак тощо.
RF (random forest) – це безліч вирішальних дерев. У задачі регресії їх відповіді усереднюються, у задачі класифікації приймається рішення голосуванням у більшості [17].

LGBoost Regressor - Бібліотека xgboost написана на C++ і може використовуватися автономно (за допомогою інтерфейсу командою рядка), так і за допомогою бібліотек-інтерфейсів для R, Python, Julia і Scala. У цьому повідомленні розглядається R-версія, робота з іншими мовами виглядає аналогічно.

Встановлювати можна з CRAN, але краще використовувати найсвіжішу версію з GitHub (актуально для всіх ОС).

Ідея градієнтного бустингу полягає у побудові ансамблю послідовно уточнюючих один одного елементарних моделей. n-на елементарна модель навчається на "помилках" ансамблю з n-1 моделей, відповіді моделей виважено підсумовуються. "Помилки" тут у лапках, оскільки насправді кожна наступна модель наближає антиградієнт функції втрат, який не обов'язково дорівнює різниці фактичних та передбачених значень (тобто помилки в

буквальному значенні). Різні функції втрат мають різні похідні, але для середньоквадратичної функції втрат, заданої як 12[yi−f(yi)]2, антиградієнт (похідна зі зворотним знаком) є саме різницею між фактичними та передбаченими значеннями: yi−f(yi). Це найбільш інтуїтивний варіант, далі як складніший приклад буде розглянута логістична функція втрат.

Елементарна модель тут названа елементарною зовсім не тому, що повинна бути дуже простою, такою як неглибоке вирішальне дерево. Під елементарністю мається на увазі можливість присутності моделі як складової частини моделі вищого порядку, у цьому випадку - моделі градієнтного бустингу. Бустити можна практично будь-які моделі - загальні лінійні, узагальнені лінійні, дерева рішень, K-найближчих сусідів та інші, наприклад, на StackOverflow в одній з відповідей згадувався бустинг нейромереж з 5 прихованими шарами. Тобто як такого обмеження на складність моделі немає, а є лише загальна для машинного навчання дилема зміщення-дисперсії (bias-variance tradeoff): модель має бути досить гнучкою, щоб відновлювати потрібну залежність, але при цьому по можливості не повинна перевчитися. Елементарна модель у градієнтному бустингу відома також як weak learner (неперекладний та досить абстрактний термін) [18].

До особливостей реалізації бустингового алгоритму в xgboost можна віднести використання крім першої ще й другої похідної від функції втрат, наявність вбудованої регуляризації, а також можливість задавати функції втрат і метрики якості. Перша суто технічна особливість підвищує ефективність алгоритму. Вбудована регуляризація допомагає боротися з перенавчанням: на черговій ітерації вирішальне дерево не будуватиметься до максимальної глибини, якщо це дуже незначно покращує якість моделі ціною її значного ускладнення. Нарешті, як функція втрат користувач може задати будь-яку функцію, яка має безперервну першу і другу похідну. Зазначимо, що у ранньому пакеті gbm представлений широкий асортимент функцій втрат залежно від розподілу цільової змінної.

2.5 Висновки

У даному розділі було здійснено наступні кроки інформаційної системи як підготовку даних і виконання розвідувального аналізу факторів. Проведено підготовку даних для подальшої побудови моделей лінійних регресій.

Проведено огляд технологій за допомогою яких буде проводитися прогнозування завислих речовин у річковій воді, наведено їх переваги та недоліки та обрано технології за допомогою яких буде реалізована система.

3 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЗАВИСЛИХ РЕЧОВИН У РІЧЦІ

3.1 Модель об'єкта

Отже, у попередньому розділі було підготовлено все для подальшого прогнозування завислих речовин, приступимо безпосередньо до виконання цієї задачі методами LGboost , Linear Regression, Random Forest Regressor (рис. 3.1). А по завершенню виберемо найкращу модель із цих трьох можливих [19].

```
# Creation the dataframe with the resulting score of all models
result = pd.DataFrame({'model' : ['Linear Regression', 'Random Forest Regressor', 'LGBoost R
egressor'],
                       'train_score': 0, 'valid_score': 0})
result
```

	model	train_score	valid_score
0	Linear Regression	0	0
1	Random Forest Regressor	0	0
2	LGBoost Regressor	0	0

Рисунок 3.1 – Фрагмент коду з оцінкою усіх досліджуваних моделец

Отже, ідентифікуємо лінійну регресію для налаштування рахуємо валідаційні значення, похибки. Можна побачити що на тренувальному лінійна регресія дає на тренувальному 0.54 (рис. 3.2).

```
# Linear Regression
lr = LinearRegression()
lr.fit(train, target_train)

# Prediction for training data
y_train_lr = lr.predict(train)

# Accuracy of model
r2_score_acc = round(r2_score(target_train, y_train_lr), 2)
print(f'Accuracy of Linear Regression model training is {r2_score_acc}')

# Save to result dataframe
result.loc[result['model'] == 'Linear Regression', 'train_score'] = r2_score_acc
```

```
Accuracy of Linear Regression model training is 0.54
```

Рисунок 3.2 – Точність навчальної моделі лінійної регресії

А тепер проведемо таку саму маніпуляцію для тестових даних рисунок 3.3.

```
# Print rounded r2_lr = lr.predict(valid)
y_val_lr = lr.predict(valid)
r2_score_acc_valid = round(r2_score(target_valid, y_val_lr),2)
result.loc[result['model'] == 'Linear Regression', 'valid_score'] = r2_score_acc_valid
print(f'Accuracy of Linear Regression model prediction for valid dataset is {r2_score_acc_va
lid}')
```

```
Accuracy of Linear Regression model prediction for valid dataset is 0.43
```

Рисунок 3.3 – Точність прогнозування дійсної моделі лінійної регресії

На тестових даних точність валідаційної моделі лінійної регресії склала 0.43 Коли дані зачумлені лінійна регресія показує доволі таки непоганий результат, звичайно було би краще якби результат був ближче до 0.9, але

важливим ще є щоб похибка на тренувальних не сильно відрізнялась від похибки на валідаційних [19].

Далі переглянемо результати тренувальних даних які були прогнозовані за допомогою «Random Forest Regressor» (рис. 3.4).

```
%%time
# Random Forest Regressor
rf = RandomForestRegressor()
param_grid = {'n_estimators': [50, 100], 'min_samples_leaf': [i for i in range(3,7)],
              'max_features': ['auto'], 'max_depth': [i for i in range(3,6)],
              'criterion': ['mse'], 'bootstrap': [False]}

# Training model
rf_CV = GridSearchCV(rf, param_grid=param_grid, cv=cv_train, verbose=False)
rf_CV.fit(train, target_train)
print(rf_CV.best_params_)

# Prediction for training data
y_train_rf = rf_CV.predict(train)

# Accuracy of model
r2_score_acc = round(r2_score(target_train, y_train_rf),2)
print(f'Accuracy of RandomForestRegressor model training is {r2_score_acc}')

# Save to result dataframe
result.loc[result['model'] == 'Random Forest Regressor', 'train_score'] = r2_score_acc
```

```
{'bootstrap': False, 'criterion': 'mse', 'max_depth': 3, 'max_features': 'auto', 'min_sa
mples_leaf': 6, 'n_estimators': 50}
Accuracy of RandomForestRegressor model training is 0.58
CPU times: user 8.54 s, sys: 58.7 ms, total: 8.6 s
Wall time: 8.6 s
```

Рисунок 3.4 – Точність прогнозування навчальної моделі «Random Forest Regressor»

В даному випадку ми бачимо що на тренувальних даних 0.58, а на валідаційних 0.13 (рис. 3.5).

```
# Print rounded r2_score_acc to 2 decimal values after the text
y_val_rf = rf_CV.predict(valid)
r2_score_acc_valid = round(r2_score(target_valid, y_val_rf),2)
result.loc[result['model'] == 'Random Forest Regressor', 'valid_score'] = r2_score_acc_valid
print(f'Accuracy of RandomForestRegressor model prediction for valid dataset is {r2_score_ac
c_valid}')
```

```
Accuracy of RandomForestRegressor model prediction for valid dataset is 0.13
```

Рисунок 3.5 – Точність прогнозування дійсної моделі «Random Forest Regressor»

І 3 остання модель LGBoost Regressor її прогнозування тренувальних даних показано на рисунку 3.6.

```
%%time
# LGBoost Regressor
lgbr = lgb.LGBMRegressor()

parameters = {'n_estimators': [200, 1000],
              'learning_rate': [0.05, 0.001],
              'num_leaves': [50, 22, 30]}

# Training model
lgb_CV = GridSearchCV(estimator=lgbr, param_grid=parameters, cv=cv_train, n_jobs=-1)
#Xtrain, Xval, Ztrain, Zval = train_test_split(train, target_train, test_size=0.2, random_sta
e=0)
lgb_CV.fit(train, target_train)
print("Best score: %0.3f" % lgb_CV.best_score_)
print("Best parameters set:", lgb_CV.best_params_)

# Prediction for training data
y_train_lgb = lgb_CV.predict(train)

# Accuracy of model
r2_score_acc = round(r2_score(target_train, y_train_lgb),2)
print(f'Accuracy of LGBoost Regressor model training is {r2_score_acc}')

# Save to result dataframe
result.loc[result['model'] == 'LGBoost Regressor', 'train_score'] = r2_score_acc
```

```
Best score: 0.254
Best parameters set: {'learning_rate': 0.001, 'n_estimators': 1000, 'num_leaves': 50}
Accuracy of LGBoost Regressor model training is 0.35
CPU times: user 752 ms, sys: 94.8 ms, total: 847 ms
```

Рисунок 3.6 – Точність прогнозування навчальної моделі «LGBoost Regressor»

Тут ми бачимо на тренувальних даних прогнозування показало дивовижні 0.35, але на тестових є всього 0.2 (рис. 3.7).

```
# Print rounded r2_score_acc to 2 decimal values after the text
y_val_lgb = lgb_CV.predict(valid)
r2_score_acc_valid = round(r2_score(target_valid, y_val_lgb),2)
result.loc[result['model'] == 'LGBoost Regressor', 'valid_score'] = r2_score_acc_valid
print(f'Accuracy of LGBoost Regressor model prediction for valid dataset is {r2_score_acc_v
alid}')
```

```
Accuracy of LGBoost Regressor model prediction for valid dataset is 0.2
```

Рисунок 3.7 – Точність прогнозування дійсної моделі «LGBoost Regressor»

Далі побудуємо діаграму важливості ознак. Код, для побудови діаграми наведено на рисунку 3.8.

```
xgbr = xgb.XGBRegressor(**xgb_CV.best_params_)
xgbr.fit(train, target_train)
fig =  plt.figure(figsize = (10,8))
axes = fig.add_subplot(111)
xgb.plot_importance(xgbr,ax = axes,height = 0.5)
plt.show();
plt.close()
```

Рисунок 3.8 – Код для побудови діаграми важливості ознак

Далі на діаграмі важливості ознак відображено який із показників найбільше впливає на прогнозований «таргет». Тобто на Suspended в 14 створі найбільше впливають показник того ж самого 14 створу NO_2 показано у додатку Б на рисунку Б.1.

Візуалізуємо прогнозування навчальних валідаційних і тестових даних. Код, для побудови на рисунках 3.9-3.11.

```
# Building plot for prediction for the training data
x = np.arange(len(train))
plt.figure(figsize=(16,10))
plt.scatter(x, target_train, label = "Target training data", color = 'g')
plt.scatter(x, y_train_lr, label = "Linear Regression prediction", color = 'b')
plt.scatter(x, y_train_rf, label = "Random Forest prediction", color = 'y')
plt.scatter(x, y_train_lgb, label = "LGBoost Regressor prediction", color = 'brown')
plt.plot(x, np.full(len(train), 0.5), label = "Maximum allowable value", color = 'r')
plt.title('Prediction for the training data')
plt.legend(loc='best')
plt.grid(True)
```

Рисунок 3.9 – Код для побудови графіку прогнозування навчальних даних

```
# Building plot for prediction for the valid data
x = np.arange(len(valid))
plt.figure(figsize=(16,10))
plt.scatter(x, target_valid, label = "Target valid data", color = 'g')
plt.scatter(x, y_val_lr, label = "Linear Regression prediction", color = 'b')
plt.scatter(x, y_val_rf, label = "Random Forest prediction", color = 'y')
plt.scatter(x, y_val_lgb, label = "LGBoost Regressor prediction", color = 'brown')
plt.plot(x, np.full(len(valid), 0.5), label = "Maximum allowable value", color = 'r')
plt.title('Prediction for the valid data')
plt.legend(loc='best')
plt.grid(True)
```

Рисунок 3.10 – Код для побудови прогнозування валідаційних даних

```
# Building plot for prediction for the test data
x = np.arange(len(test))
plt.figure(figsize=(16,10))
plt.scatter(x, target_test, label = "Target test data", color = 'g')
plt.scatter(x, y_test_lr, label = "Linear Regression prediction", color = 'b')
plt.scatter(x, y_test_rf, label = "Random Forest prediction", color = 'y')
plt.scatter(x, y_test_lgb, label = "LGBoost Regressor prediction", color = 'brown')
plt.plot(x, np.full(len(test), 0.5), label = "Maximum allowable value", color = 'r')
plt.title('Prediction for the test data')
plt.legend(loc='best')
plt.grid(True)
```

Рисунок 3.11 – Код для побудови прогнозування тестових даних

Самі графіки прогнозування навчальних, валідаційних і тестових даних зображені у додатку Б на рисунках Б.2-4 відповідно.

Тепер коли всі підрахунки виконані можна приступати до вибору найкращої моделі прогнозування. Далі відображено таблицю результатів прогнозування рисунок 3.12.

	model	train_score	valid_score
0	Linear Regression	0.56	0.62
2	LGBoost Regressor	0.35	0.20
1	Random Forest Regressor	0.58	0.13

Рисунок 3.12 – Таблиця результатів прогнозування

Тут важливо щоб різниця між показниками була не дуже великою. Передусім вибирається найкраща позиція де різниця між тренувальними і валідаційними даними не більше ніж 0.1, а далі між тими вибирається та у якої найбільший «valid_score» рисунок 3.13.

```
# Select models with minimal overfitting
result_best = result_best[(result_best['train_score'] - result_best['valid_score']).abs() <
0.15]
result_best.sort_values(by=['valid_score', 'train_score'], ascending=False)
```

	model	train_score	valid_score
0	Linear Regression	0.56	0.62
2	LGBoost Regressor	0.35	0.20

Рисунок 3.13 – Вибір моделі з мінімальним відхиленням

Останнім кроком являється показ найкращого результату прогнозування. Така дія показана на рисунку 3.14.

In [47]:
```
# Select the best model
result_best.nlargest(1, 'valid_score')
```

Out[47]:

	model	train_score	valid_score
0	Linear Regression	0.84	0.76

Рисунок 3.14 – Найкраща модель прогнозування

3.2 Висновки

Отже, в даному розділі було розкрито основну ідею інформаційної технології аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року.

Проведено прогнозування завислих речовин у річковій воді та обрано найкращу модель із найвдалішим прогнозом на валідаційній вибірці даних.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки

Метою проведення комерційного та технологічного аудиту є оцінювання комерційного потенціалу впровадження інформаційної технології аналізу та прогнозування концентрації завислих речовин у річковій воді.

Для проведення технологічного аудиту було залучено 3-х незалежних експертів Вінницького національного технічного університету кафедри системного аналізу та інформаційних технологій – зав. каф. САІТ, Мокін Віталій Борисович – проф. каф., Козачко Олексій Миколайович – доц. каф та доц. каф., Жуков Сергій Олександрович. Для проведення технологічного аудиту було використано таблицю 4.1 в якій за п’ятибальною шкалою використовуючи 12 критеріїв здійснено оцінку комерційного потенціалу [20, 21].

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів

Продовження таблиці 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування

Продовження табл. 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
10	Необхід на розробка нових матеріал ів	Потрібні матеріали, що використовуютьс я у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовують ся у виробництві
11	Термін реалізаці ї ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхід на розробка регламен тних докумен тів та отриман ня великої кількості дозвільн их докумен тів на виробни цтво та реалізаці ю продукт у	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки пові-домлення відповідним органам про виробництв о та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

В таблиці 4.2 наведено рівні комерційного потенціалу розробки.

Таблиця 4.2 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

В таблиці 4.3 наведено результати оцінювання експертами комерційного потенціалу розробки.

Таблиця 4.3 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	1. Мокін В.Б.	2. Козачко О.М.	3. Жуков С.О.
	Бали, виставлені експертами:		
1	4	4	3
2	4	3	3
3	3	3	4
4	3	4	3
5	2	1	3
6	2	3	2
7	4	4	3
8	2	2	3
9	3	4	2
10	4	4	3
11	4	4	4
12	4	3	4
Сума балів	$СБ_1$=39	$СБ_2$=39	$СБ_3$=37
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{39+39+37}{3} = 38$		

Середньоарифметична сума балів, розрахована на основі висновків експертів склала 38 бали, що згідно таблиці 4.2 вважається, що рівень комерційного потенціалу проведених досліджень є вище середнього.

Нову розробку можна реалізовувати через веб – портал Держводагенства України. В якості аналога для розробки було обрано інструменти пакету програм MS Excel 2007

До недоліків аналогу можна віднести: відсутність алгоритму аналізу та прогнозування; необхідність спеціалізований знань для побудови прогнозів; немає адаптації до індивідуальних особливостей користувача.

Основна мета розробки інформаційної технології – вирішити задачі, які будуть давати переваги перед конкурентами, а саме реалізувати: персоналізацію аналізу та прогнозу концентрації завислих речовин у річці при динамічному наборі вхідних даних; контроль процесу прогнозування та можливість підняти точність прогнозування; сучасний, простий, інтуїтивний інтерфейс користувача; українська локалізація.

Проведемо оцінку якості і конкурентоспроможності нової розробки порівняно з аналогом. В таблиці 4.4 наведені основні техніко-економічні показники аналога і нової розробки.

Таблиця 4.4 – Основні параметри нової розробки та товару-конкурента

Показник	Варіанти		Відносний показник якості	Коефіцієнт вагомості параметра
	Базовий (товар-конкурент)	Новий (інноваційне рішення)		
1	2	3	4	5
Об'єм пам'яті, мб	32	48	1,5	30%
Кільсть вправ у базі, шт	>300	+/- 80	3,75	10
Швидкість первинного запуску, с	2	5	2,5	15
Витрати трафіку, мб/міс	100-150	600-800	6	15
Підтримка версій	8-12	7.2-12	1	30%

Проведемо оцінку якості продукції, яка є найефективнішим засобом забезпечення вимог споживачів та порівняємо її з аналогом.

Визначимо відносні одиничні показники якості по кожному параметру за формулами (4.1) та (4.2) і занесемо їх у відповідну колонку таблиці 4.5.

$$q_i = \frac{P_{Hi}}{P_{Бi}} \tag{4.1}$$

або

$$q_i = \frac{P_{Бi}}{P_{Hi}}, \tag{4.2}$$

де P_{Hi} $P_{Бi}$, – числові значення *i*-го параметру відповідно нового і базового виробів.

$$q_1 = \frac{48}{32} = 1{,}5;$$

$$q_2 = \frac{300}{80} = 3{,}75;$$

$$q_3 = \frac{5}{2} = 2{,}5;$$

$$q_4 = \frac{600}{100} = 6;$$

$$q_5 = \frac{12}{12} = 1.$$

Відносний рівень якості нової розробки визначаємо за формулою:

$$К_{я.в.} = \sum_{i=1}^{n} q_i \cdot \alpha_i, \tag{4.3}$$

$$К_{я.в.} = 1{,}5 \cdot 0{,}3 + 3{,}75 \cdot 0{,}1 + 2{,}5 \cdot 0{,}15 + 6 \cdot 0{,}15 + 1 \cdot 0{,}3 = 2{,}4.$$

Відносний коефіцієнт показника якості нової розробки більший одиниці, отже нова розробка якісніший базового товару-конкурента.

Наступним кроком є визначення конкурентоспроможності товару. Конкурентоспроможність товару є головною умовою конкурентоспроможності підприємства на ринку і важливою основою прибутковості його діяльності.

Однією із умов вибору товару споживачем є збіг основних ринкових характеристик виробу з умовними характеристиками конкретної потреби

покупця. Такими характеристиками найчастіше вважають нормативні та технічні параметри, а також ціну придбання та вартість споживання товару.

В таблиці 4.5 наведено технічні та економічні показники для розрахунку конкурентоспроможності нової розробки відносно товару-аналога, технічні дані взяті з попередніх розрахунків.

Таблиця 4.5 – Нормативні, технічні та економічні параметри нової розробки і товару-виробника

Показники	Варіанти	
	Базовий (товар-конкурент)	Новий (інноваційне рішення)
1	2	3
1. Нормативно-технічні показники		
Об'єм пам'яті, мб	32	48
Кільсть вправ у базі, шт	>300	+/- 80
Швидкість первинного запуску, с	2	5
Витрати трафіку, мб/міс	100-150	600-800
Підтримка версій	8-12	7.2-12
2. Економічні показники		
Ціна придбання, грн.	350	280

Загальний показник конкурентоспроможності інноваційного рішення (К) з урахуванням вищезазначених груп показників можна визначити за формулою:

$$K = \frac{I_{m.n.}}{I_{e.n.}}, \tag{4.4}$$

де $I_{m.n.}$ – індекс технічних параметрів; $I_{e.n.}$ – індекс економічних параметрів.

Індекс технічних параметрів є відносним рівнем якості інноваційного рішення. Індекс економічних параметрів визначається за формулою (4.5).

$$I_{e.n.} = \frac{\sum_{i=1}^{n} P_{Нei}}{\sum_{i=1}^{n} P_{Бei}}, \tag{4.5}$$

де P_{Hei}, $P_{Бei}$ – економічні параметри (ціна придбання та споживання товару) відповідно нового та базового товарів.

$$I_{е.п.} = \frac{280}{350} = 0{,}8;$$

$$К = \frac{2{,}4}{0{,}8} = 3.$$

Зважаючи на розрахунки, можна зробити висновок, що нова розробка буде конкурентоспроможніше, ніж конкурентний товар.

4.2 Прогнозування витрат на виконання науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи групуються за такими статтями: витрати на оплату праці, витрати на соціальні заходи, матеріали, паливо та енергія для науково-виробничих цілей, витрати на службові відрядження, програмне забезпечення для наукових робіт, інші витрати, накладні витрати.

1. Основна заробітна плата кожного із дослідників $З_О$, якщо вони працюють в наукових установах бюджетної сфери визначається за формулою:

$$З_О = \frac{М}{Т_р} * t(грн), \quad (4.6)$$

де M – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.;

$T_Р$ – число робочих днів в місяці; приблизно $T_Р \approx 21…23$ дні;

t – число робочих днів роботи дослідника.

Для розробки програмних засобів, які б автоматично зчитували дані моніторингу про стан водних ресурсів у річках та робився автоматично прогноз концентрації завислих речовин необхідно залучити програміста з посадовим окладом 10000 грн. Кількість робочих днів у місяці складає 22, а кількість робочих днів програміста складає 22. Зведемо сумарні розрахунки до таблиця 4.6.

Таблиця 4.6 – Заробітна плата дослідника в науковій установі бюджетної сфери

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник	20000	909,1	5	4545
Програміст	10000	454,5	22	10000
Всього				14545

2. Розрахунок додаткової заробітної плати робітників

Додаткова заробітна плата $З_д$ всіх розробників та робітників, які приймали устають в розробці нового технічного рішення розраховується як 10 - 12 % від основної заробітної плати робітників.

На даному підприємстві додаткова заробітна плата начисляється в розмірі 10% від основної заробітної плати.

$$З_д = \frac{(З_о + З_р) * Н_{доп}}{100\%} \quad (4.7)$$

$$З_д = 0{,}11 * 14545 = 1600(грн).$$

3. Нарахування на заробітну плату $Н_{ЗП}$ дослідників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою (4.3).

$$Н_{ЗП} = \frac{(З_о + З_д) * \beta}{100} (грн), \quad (4.8)$$

де $З_о$ – основна заробітна плата розробників, грн;

$З_д$ – додаткова заробітна плата всіх розробників та робітників, грн;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % .

Дана діяльність відноситься до бюджетної сфери, тому ставка єдиного внеску на загальнообов'язкове державне соціальне страхування буде складати 22%, тоді**:**

$$Н_{ЗП} = \frac{(14545+1600)*22}{100} = 3552(грн).$$

4. Витрати на матеріали М та комлектуючі К, що були використані під час виконання даного етапу роботи, розраховуються по кожному виду матеріалів за формулою:

$$М = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i - \sum_{1}^{n} В_i \cdot Ц_в \quad грн., \qquad (4.9)$$

де $Н_i$ – витрати матеріалу *і*-го найменування, кг;

$Ц_i$ – вартість матеріалу *і*-го найменування, грн/кг;

$К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15);

$В_i$ – маса відходів матеріалу *і*-го найменування, кг;

$Ц_в$ – ціна відходів матеріалу *і*-го найменування, грн/кг;

n – кількість видів матеріалів.

Матеріали, що використані на розробку приведено в табл. 4.7.

Таблиця 4.7 – Матеріали, що використані на розробку

Найменування матеріалу	Ціна за одиницю, грн.	Витрачено	Вартість витраченого матеріалу, грн.
Папір	130	1	130
Ручка	12	1	12
CD-диск	12	1	12
Флешка	135	1	135
Всього			289
З врахуванням коефіцієнта транспортування			317,9

5. Програмне забезпечення для наукової роботи включає витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення необхідного для проведення дослідження.

Оскільки для нової розробки Android Studio, яке є безкоштовним ми дані витрати враховувати не будемо.

6. Амортизація обладнання, комп'ютерів та приміщень, які використовувались під час виконання даного етапу роботи

Дані відрахування розраховують по кожному виду обладнання, приміщенням тощо.

$$A = \frac{Ц \cdot T}{Ткор \cdot 12} \quad [грн], \tag{4.10}$$

де Ц – балансова вартість даного виду обладнання (приміщень), грн.;

$Т_{кор}$ – час користування;

Т – термін використання обладнання (приміщень), цілі місяці.

Згідно пункта 137.3.3 Податкового кодекса амортизація нараховується на основні засоби вартістю понад 2500 грн. В нашому випадку для написання магістерської роботи використовувався персональний комп'ютер вартістю 35000 грн.

$$А = \frac{35000 \cdot 1}{2 \cdot 12} = 1458{,}33.$$

7. До статті «Паливо та енергія для науково-виробничих цілей» відносяться витрати на всі види палива й енергії, що безпосередньо використовуються з технологічною метою на проведення досліджень.

$$В_е = \sum_{i=1}^{n} \frac{W_{yt} \cdot t_i \cdot Ц_е \cdot К_{впі}}{\eta_і}, \tag{4.11}$$

де W_{yt} – встановлена потужність обладнання на певному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_е$ – вартість 1 кВт-години електроенергії, грн;

$К_{впі}$ – коефіцієнт, що враховує використання потужності, $К_{впі} < 1$;

$\eta_і$ – коефіцієнт корисної дії обладнання, $\eta_і < 1$.

Для написання магістерської роботи використовується персональний комп'ютер для якого розрахуємо витрати на електроенергію.

$$В_е = \frac{0{,}3 \cdot 180 \cdot 4{,}1 \cdot 0{,}5}{0{,}8} = 138{,}38.$$

Витрати на службові відрядження, витрати на роботи, які виконують сторонні підприємства, установи, організації та інші витрати в нашому досліджені не враховуються оскільки їх не було.

Накладні (загальновиробничі) витрати Внзв охоплюють: витрати на управління організацією, оплата службових відряджень, витрати на утримання, ремонт та експлуатацію основних засобів, витрати на опалення, освітлення, водопостачання, охорону праці тощо. Накладні (загальновиробничі) витрати Внзв можна прийняти як (100…150)% від суми основної заробітної плати розробників та робітників, які виконували дану МКНР, тобто:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.12)$$

де $Н_{нзв}$ – норма нарахування за статтею «Інші витрати».

$$В_{нзв} = 14545 \cdot \frac{100}{100\%} = 14545 грн$$

Сума всіх попередніх статей витрат дає витрати, які безпосередньо стосуються даного розділу МКР

$$В = 14545+1600+3552+317,9+458,33+138,38+14545=36157,5$$

Прогнозування загальних втрат ЗВ на виконання та впровадження результатів виконаної МКР здійснюється за формулою:

$$ЗВ = \frac{В}{\eta}, \quad (4.13)$$

де η – коефіцієнт, який характеризує стадію виконання даної НДР.

Оскільки, робота знаходиться на стадії науково-дослідних робіт, то коефіцієнт β = 0,5.

Звідси:

$$ЗВ = \frac{36157,5}{0,5} = 72315,03 \text{ грн.}$$

4.3 Розрахунок економічної ефективності науково-технічної розробки

У даному підрозділі кількісно спрогнозуємо, яку вигоду, зиск можна отримати у майбутньому від впровадження результатів виконаної наукової роботи. Розрахуємо збільшення чистого прибутку підприємства $\Delta П_i$, для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, за формулою

$$\Delta П_i = \sum_{1}^{n} \left(\Delta Ц_o \cdot N + Ц_o \cdot \Delta N \right)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\upsilon}{100} \right) \qquad (4.14)$$

де $\Delta Ц_0$ – покращення основного оціночного показника від впровадження результатів розробки у даному році.

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки:

$Ц_o$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки:

л – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт $л = 0{,}8333$.

р – коефіцієнт, який враховує рентабельність продукту. р = 0,25;

х – ставка податку на прибуток. У 2021 році – 18%.

Припустимо, що при впроваджені результатів наукової розробки покращується якість прогнозу концентрації завислих речовин у річковій воді. Припустимо, що ціна від зросте на 50 грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року на 3000 шт., протягом другого року – на 4000 шт., протягом третього року на 5000 шт. Реалізація

продукції до впровадження розробки складала 1 шт., а її ціна до складає 280 грн. Розрахуємо прибуток, яке отримає підприємство протягом трьох років.

$$\Delta П_1 = [50 \cdot 1 + (280 + 50) \cdot 3000] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 169126{,}78\text{грн.}$$

$$\Delta П_2 = [50 \cdot 1 + (280 + 50) \cdot (3000 + 4000)] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 394659{,}22\text{грн.}$$

$$\Delta П_3 = [50 \cdot 1 + (280 + 50) \cdot (3000 + 4000 + 5000)] \cdot 0{,}833 \cdot 0{,}25 \cdot \left(1 + \frac{18}{100}\right) = 676522{,}94\text{грн.}$$

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Розрахуємо основні показники, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності.

Розрахуємо величину початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки.

$$PV = k_{\text{інв}} \cdot \text{ЗВ}, \tag{4.15}$$

де $k_{\text{інв}}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо ($k_{\text{інв}} = 2…5$).

$$PV = 2 \cdot 72315{,}03 = 144630{,}06$$

Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$ згідно наступної формули:

$$E_{абс} = (ПП - PV) \tag{4.16}$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство від реалізації результатів наукової розробки, грн.;

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t},$$

(4.17)

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДМКР, грн.;

Т – період часу, протягом якою виявляються результати впровадженої НДМКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,2;

t – період часу (в роках).

$$\text{ПП} = \frac{169126{,}78}{(1+0{,}2)^1} + \frac{394659{,}22}{(1+0{,}2)^2} + \frac{676522{,}94}{(1+0{,}2)^3} = 808335{,}17\text{грн.}$$

$$\text{Е}_{\text{абс}} = (808335{,}17 - 144630{,}06) = 663705{,}1\text{грн.}$$

Оскільки $E_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього користуються формулою:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1,$$

(4.18)

$T_ж$ – життєвий цикл наукової розробки, роки.

$$\text{Е}_\text{в} = \sqrt[3]{1 + \frac{663705{,}1}{144630{,}06}} - 1 = 1{,}17 = 117\%$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f,$$

(4.19)

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2021 році в Україні d= (0,14…0,2);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,1).

$$\tau_{\min} = 0,18 + 0,05 = 0,23.$$

Так як $E_в > \tau_{\min}$ то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в}$$

(4.20)

$$\text{Т}_{\text{ок}} = \frac{1}{1,17} = 0,9 \text{ роки}$$

Так як $T_{ок} \leq$ 3...5-ти років, то фінансування даної наукової розробки в принципі є доцільним.

4.5 Висновки

Було проведено оцінку комерційного потенціалу програмного засобу для аналізу та прогнозування концентрацій завислих речовин у річковій воді, який є на вище середньому рівні. При порівнянні нової розробки з аналогом виявлено, що вона є якіснішою і конкурентоспроможнішою відносно аналога, а також краще по технічним і економічним показникам.

Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 36157,5 грн. Загальна ж величина витрат на

виконання та впровадження результатів даної НДР буде складати 72315,03 грн.

Вкладені інвестиції в даний проект окупляться через 9 місяців при прогнозованому прибутку 808335,17 грн. за три роки.

ВИСНОВКИ

В магістерській кваліфікаційній роботі розроблено інформаційну технологію аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року. В роботі приведено огляд проблем розробки інформаційної технології аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг. Запропоновані оптимальні технології та формати для реалізації даної системи з точки зору можливості її використання на персональних комп'ютерах із платним ліцензійним забезпеченням, так і з вільним для розповсюдження програмним забезпеченням.

В першому розділі проведено обґрунтування проблеми створення інформаційної системи аналізу та прогнозування завислих речовин у річковій воді та проведено аналіз предметної області, зроблено огляд і опис методів та підходів щодо прогнозування завислих речовин у водах річки. В результаті цього було обґрунтовано вибір бібліотеки Scikit Learn.

В другому розділі формалізовано модель інформаційної технології аналізу та прогнозування завислих речовин у річці, запропоновано ідею інформаційної технології аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року, сформовано методологію інформаційної технології, проведено огляд та вибір технологій для інформаційної технології аналізу та прогнозування завислих речовин у річковій воді.

В третьому розділі сформовано передбачення за допомогою обраних інструментів у попередніх розділах та обрано найкращу модель, яка дозволить будувати прогнози високої точності. Визначено що оптимальним методом бустингу є бібліотеки це RandomForestRegressor, LGBRegressor, LinearRegression за використанням мови Python.

Розроблено та наведено код, який виконує поставлене з завдання цими способами. Побудовано діаграму важливості ознак та графіки прогнозування

даних. Аналіз показав що найбільшим чинником який вливає на завислі речовини є NO_2. Прогнозування виявило що найкращою моделлю прогнозування завислих речовин у воді є «Linear Regression» із точністю прогнозу 72%. Вміст завислих речовин у р. Південний Буг за проаналізовані роки та періоди являється в межах ГДК.

В четвертому розділі було виконано економічну частину кваліфікаційної роботи під час розрахунків було доведено:

– Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 36157,5 грн. Загальна ж величина витрат на виконання та впровадження результатів даної НДР буде складати 72315,03 грн.

– Вкладені інвестиції в даний проект окупляться через 9 місяців при прогнозованому прибутку 808335,17 грн за три роки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Поліщук Д.А, Мокін В.Б., Ящолт А.Р. Інформаційна технологія аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року. Матеріали КОНФЕРЕНЦІЇ ВНТУ електронні наукові видання, Молодь в науці: дослідження, проблеми, перспективи (МН-2022). Вінниця, 2021. № 2. [Електронний ресурс]. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2022/paper/view/14217/12042.
2. Панкратова Н.Д. Системний аналіз: теорія та застосування : підручник. – Видво “Наукова думка” НАН України, 2019. – 352 с.
3. Кузьмін О. Є. Системний аналіз і прийняття інноваційних рішень : навчальний посібник / О. Є. Кузьмін, О. О. Жовтанецька, Н. О. Заяць ; МОН України. – Львів : Новий Світ-2000, 2018. – 227 с.
4. Функціональний аналіз, адаптований до прикладних задач в галузі інформаційних технологій : навчальний посібник / Б. І. Мокін, В. Б. Мокін, О. Б. Мокін. – Вінниця : ВНТУ, 2020. – 192 с.
5. Практикум для самостійної роботи студентів з навчальної дисципліни «Методологія та організація наукових досліджень». Частина 1: від постановки задачі до синтезу та ідентифікації математичної моделі [Електронне видання] / Б. І. Мокін, В. Б. Мокін, О. Б. Мокін. – Вінниця : ВНТУ, 2018. – 179 с.
6. Попередній аналіз даних. Описова статистика [Електронний ресурс] URL: https://stud.com.ua/93299/statistika/poperedniy_analiz_danih_opisova_statistika#73
7. Хмарні технології. Переваги та недоліки [Електронний ресурс]. URL: https://valtek.com.ua/ua/system-integration/it98 infrastructure/clouds/cloud-technologies (дата звернення 12.09.2021). – Назва з екрану.
8. Основи статистики та аналізу даних [Електронний ресурс]. URL: https://socialdata.org.ua/manual4/
9. Бібліотека Scikit-learn [Електронний ресурс] URL: https://en.wikipedia.org/wiki/Scikit-learn

10. Бібліотека NumPy [Електронний ресурс] URL: https://en.wikipedia.org/wiki/NumPy
11. Бібліотека Pandas [Електронний ресурс] URL: https://en.wikipedia.org/wiki/Pandas
12. Бібліотека Matplotlib [Електронний ресурс] URL: https://en.wikipedia.org/wiki/Matplotlib
13. Avimanyu Bandyopadhyay Hands-On GPU Computing with Python / Chapter 4[Електронний ресурс]. The Anaconda Python distribution for package managementP.118-120
14. Метод ExtraTrees [Електронний ресурс] URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html
15. Метод AdaBoost[Електронний ресурс] URL: https://ru.wikipedia.org/wiki/AdaBoost
16. Ammonium prediction in river water[Електронний ресурс] URL: https://www.kaggle.com/vbmokin/ammonium-prediction-in-river-water
17. Prediction BOD inriverwater[Електронний ресурс] URL: https://www.kaggle.com/vbmokin/bod-prediction-in-river-15-regression-models
18. Data set - Prediction BOD in river water. [Електронний ресурс] URL: https://www.kaggle.com/vbmokin/prediction-bod-in-river-water
19. Методичні рекомендації з комерціалізації розробок, створених в результаті науково-технічної діяльності – К. : Наказ Державного комітету України з питань науки, інновацій та інформатики (Лист № 1/06-4-97 від 13.09.2010 р.).
20. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт. / Укладачі В.О. Козловський, О.Й. Лесько, В.В.Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

Додаток А
(обов'язковий)

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет комп'ютерних систем і автоматики

ЗАТВЕРДЖУЮ

Завідувач кафедри САІТ

_____ д.т.н., проф. Мокін В. Б.

«___» _____ 2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на магістерську кваліфікаційну роботу

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЗМІН КОНЦЕНТРАЦІЇ ЗАВИСЛИХ РЕЧОВИН У РІЧЦІ ПІВДЕННИЙ БУГ У 4 КВАРТАЛІ 2021 РОКУ»

08-53.МКР.005.02.000.ТЗ

Керівник: к.т.н., доцент

_____ Ящолт А.Р.

«___» _____ 2021 р.

Розробив: студент гр. 2ІСТ-20м

_____ Поліщук Д.А.

«___» _____ 2021 р.

Вінниця 2021

1. Підстава для проведення робіт

Підставою для виконання роботи є наказ № __ по ВНТУ від «__» _____ 2021 р., та індивідуальне завдання на МКР, затверджене протоколом № __ засідання кафедри САІТ від «__» _____ 2021 р.

2. Джерела розробки:

- Датасети з Конкурса на платформі Kaggle - WQ SB river : EDA and Forecasting. NEW [Електронний ресурс]. URL : https://www.kaggle.com/nikaapril/wq-sb-river-eda-and-forecasting-new
- Дані про моніторинг якості річкової води у р. П. Буг із сайта [Електронний ресурс]. URL : http://monitoring.davr.gov.ua/EcoWaterMon/GDKMap/Index

3. Мета і призначення роботи:

Розробити інформаційну технологію для аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року, яка б давала можливість робити прогнози змін концентрації завислим речовинам враховуючи дані вимірювань із створів, що розташовані вище по течії.

1. Вихідні дані для проведення робіт:

- Дані конкурсу « River Water Quality EDA and Forecasting» платформи Kaggle.
- Електронна карта Вінницької області.
- Дані моніторингу Держводагентства України.

2. Методи дослідження:

Машинне навчання, лінійна регресія, random forest regressor, lgboost regressor. регресивний аналіз, метод лінійної регресії.

6. Етапи роботи і терміни їх виконання:

1. Аналіз предметної області ..... 09.2021 – 09.2021;
2. Розробка інформаційної технології..... 09.2021 – 09.2021;
3. Реалізація інформаційної технології..... 10.2021 – 11.2021;
4. Оформлення пояснювальної записки. ..... 11.2021 – 12.2021.

7. Очікувані результати та порядок реалізації:

Розробка та апробація інформаційної технології аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року.

8. Вимоги до розробленої документації

Пояснювальна записка оформлена у відповідності до вимог «Методичних вказівок до виконання та оформлення магістерських кваліфікаційних робіт для студентів спеціальності 126 – «Інформаційні системи та технології» денної форми навчання».

9. Порядок приймання роботи

Публічний захист ..... «___» _____.2021 р.
Початок розробки..... «___» _____ 2021 р.
Граничні терміни виконання МКР ..... «___» _____ 2021 р.

Розробив студент групи 2ІСТ-20м _____ Поліщук Д.А.

Додаток Б

(обов'язковий)

Протокол перевірки кваліфікаційної роботи

Назва роботи: «Інформаційна технологія аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг у 4 кварталі 2021 року
Тип роботи: магістерська кваліфікаційна робота
Підрозділ: кафедра САІТ
Науковий керівник: Ящолт А.Р. к.т.н., доцент

Показники звіту подібності

Unicheck	
Оригінальність	81,3 %
Схожість	18,7 %

Аналіз звіту подібності (відмітити потрібне)

- ☐ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.
- ☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і самостійності її автора. Роботу направити на доопрацювання.
- ☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Заявляю, що ознайомлений з повним звітом подібності, який був згенерований системою щодо роботи

Автор _____ Поліщук Д.А.
(підпис)

Опис прийнятого рішення

Робота допускається до захисту

Особа, відповідальна за перевірку _____ Жуков С. О.
(підпис)

Додаток Г

(довідниковий)

Фрагмент Лістингу програмного коду.

Dataset River Water Quality EDA and Forecasting
Dataset contains data on river water quality for 8 indicators for 22 monitoring stations.

Data for 2000-2021 for the Southern Bug (or Pivdennyi Booh) river.

Possible Tasks:
Analysis of data dependences, including EDA.

Prediction of the data in the certain station by data from upstream stations with the highest accuracy.

Map of the stations:
http://monitoring.davr.gov.ua/EcoWaterMon/GDKMap/Index

image.png

The water quality state monitoring stations of the Southern Bug (or Pivdennyi Booh) river.

Acknowledgements
Data Science for tabular data: Advanced Techniques
EDA for tabular data: Advanced Techniques
Datasets for river water quality prediction
AI-ML-DS Training. L1T : Titanic - Decision Tree
AI-ML-DS Training. L1T : NH4 - linear regression
Heart Disease - Automatic AdvEDA & FE & 20 models
BOD prediction in river - 15 regression models
The system "MONITORING AND ENVIRONMENTAL ASSESSMENT OF WATER RESOURCES OF UKRAINE", State Agency of Water Resources of Ukraine
WQ SB river : EDA and Forecasting

Table of Contents

Import libraries

Download data

EDA & FE & Preprocessing data

Statistics & FE

Data standartization

Training data splitting

Cross-validation of training data

Modeling

Linear Regression

Random Forest Regressor

LGBoost Regressor

Test prediction

Results visualization

Select the best model

1. Import libraries

Back to Table of Contents

```
# Work with Data - the main Python libraries
import numpy as np
import pandas as pd
import pandas_profiling as pp

# Visualization
import matplotlib.pyplot as plt

# Preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, KFold, ShuffleSplit, GridSearchCV

# Modeling
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
import lightgbm as lgb
from lightgbm import LGBMRegressor
```

```
# Metrics
from sklearn.metrics import r2_score

import warnings
warnings.simplefilter('ignore')
pd.set_option('max_colwidth', 200)
```

2. Download data

Back to Table of Contents

```
# Download data
data = pd.read_csv('../input/wq-southern-bug-river-01052021/PB_All_2000_2021.csv', sep=';', header=0)
data
```

id	date	NH4	BSK5	Suspended	O2	NO3	NO2	SO4	PO4	CL	
0	1	17.02.2000	0.330	2.77	12.0	12.30	9.50	0.057	154.00	0.454	289.50
1	1	11.05.2000	0.044	3.00	51.6	14.61	17.75	0.034	352.00	0.090	1792.00
2	1	11.09.2000	0.032	2.10	24.5	9.87	13.80	0.173	416.00	0.200	2509.00
3	1	13.12.2000	0.170	2.23	35.6	12.40	17.13	0.099	275.20	0.377	1264.00
4	1	02.03.2001	0.000	3.03	48.8	14.69	10.00	0.065	281.60	0.134	1462.00
...	...	...	...	...	...	...	...	...	...	...	...
2856	22	06.10.2020	0.046	2.69	3.6	8.28	3.80	0.038	160.00	0.726	77.85
2857	22	27.10.2020	0.000	1.52	0.5	11.26	0.56	0.031	147.20	0.634	71.95
2858	22	03.12.2020	0.034	0.29	0.8	11.09	2.58	0.042	209.92	0.484	61.17
2859	22	12.01.2021	0.000	2.10	0.0	14.31	3.94	0.034	121.60	0.424	63.49
2860	22	10.02.2021	0.000	1.78	0.0	14.30	6.30	0.033	134.40	0.582	66.31

2861 rows × 11 columns

```
# Information for training data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2861 entries, 0 to 2860
```

Data columns (total 11 columns):

#	Column	Non-Null Count	Dtype
0	id	2861 non-null	int64
1	date	2861 non-null	object
2	NH4	2858 non-null	float64
3	BSK5	2860 non-null	float64
4	Suspended	2845 non-null	float64
5	O2	2858 non-null	float64
6	NO3	2860 non-null	float64
7	NO2	2858 non-null	float64
8	SO4	2812 non-null	float64
9	PO4	2833 non-null	float64
10	CL	2812 non-null	float64

dtypes: float64(9), int64(1), object(1)

memory usage: 246.0+ KB

```
# Download data about monitoring stations
data_about = pd.read_csv('../input/wq-southern-bug-river-01052021/PB_stations.csv', sep=';', header=0, encoding='cp1251')
data_about.sort_values(by=['length'], ascending=False)
```

id	length	name_station
20	21	773.0 р. Південний Буг, 773 км, смт. Чорний Острів, Мар'янівське вдсх.
19	20	755.0 р. Південний Буг, 755 км, м. Хмельницький , Хмельницьке вдсх.
18	19	744.0 р. Південний Буг, 744 км, с. Копистин, нижче м.Хмельницький
17	18	711.0 р. Південний Буг, 711 км, смт. Меджибіж, Меджибіжське вдсх.
16	17	692.0 р. Південний Буг, 692 км, с. Щедрове, Щедрівське вдсх.
15	16	652.0 р. Південний Буг, 652 км, м. Хмільник, питний в/з, вище міста
14	15	607.0 р. Південний Буг, 607 км, с. Гущинці, нижче села , питний водозабір м.Калинівка
13	14	582.0 р. Південний Буг, 582 км, м. Вінниця, Сабарівське вдсх, питний в/з міста, вище міста
12	13	569.5 р. Південний Буг, 569,5 км, 500 м нижче скиду ВОКВП ВКГ "Вінницяводоканал" (1,5 км нижче греблі Сабарівського вдсх.)
11	12	537.0 р. Південний Буг, 537 км, смт. Сутиски, Сутиське вдсх., н/б'єф

9 10 413.0 р. Південний Буг, 413 км, с. Маньківка, вище села, питний в/з м.Ладижин

8 9 400.0 р. Південний Буг, 400 км, м. Ладижин, Ладижинське вдсх.

7 8 372.0 р. Південний Буг, 372 км, с. Глибочок, Глибочекське вдсх.

6 7 327.0 р. Південний Буг, 327 км, с. Ставки, кордон Вінницької та Кіровоградської обл.

5 6 316.0 р. Південний Буг, 316 км, м.Гайворон, Гайворонське вдсх.

4 5 237.0 р. Південний Буг, 237 км, питний водозабір смт Побузьке

3 4 206.0 р. Південний Буг, 206 км, м. Первомайськ, Первомайське вдсх.

2 3 153.0 р. Південний Буг, 153 км, с. Олексіївка, питний в/з м. Південно-Українськ

1 2 136.0 р. Південний Буг, 136 км, с. Олександрівка, Олександрівське вдсх.

21 22 97.0 р. Південний Буг, 97 км, м. Вознесенськ, пит.в/з м. Вознесенськ, 2 км до в'їзду у м. Вознесенськ по трасі з м. Миколаїв

10 11 50.0 р. Південний Буг, 50 км, с. Ковалівка, Південно-Бузька ЗС

0 1 0.5 р. Південний Буг, 0,5 км, м. Миколаїв, Бузький лиман, тех. в/з Миколаївської ТЕЦ (ліва частина морського порту)

3. EDA & FE & Preprocessing data

Back to Table of Contents

```
# Amount data observations of stations
data['id'].value_counts().sort_values().plot(kind='barh')
```

<AxesSubplot:>

```
# Determination the year of observations
data['ds'] = pd.to_datetime(data['date'])
data['year'] = data['ds'].dt.year
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2861 entries, 0 to 2860
Data columns (total 13 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   id         2861 non-null   int64
 1   date       2861 non-null   object
```

```
 2   NH4         2858 non-null   float64
 3   BSK5        2860 non-null   float64
 4   Suspended   2845 non-null   float64
 5   O2          2858 non-null   float64
 6   NO3         2860 non-null   float64
 7   NO2         2858 non-null   float64
 8   SO4         2812 non-null   float64
 9   PO4         2833 non-null   float64
 10  CL          2812 non-null   float64
 11  ds          2861 non-null   datetime64[ns]
 12  year        2861 non-null   int64
dtypes: datetime64[ns](1), float64(9), int64(2), object(1)
memory usage: 290.7+ KB
# Determination the start year of observations for all stations
data[['id', 'year']].groupby(by=['id']).min().sort_values(by=['year'], ascending=False)
year
id
5         2019
13        2006
1         2000
21        2000
20        2000
19        2000
18        2000
17        2000
16        2000
15        2000
14        2000
12        2000
2         2000
11        2000
10        2000
9         2000
8         2000
7         2000
```

6 2000

4 2000

3 2000

22 2000

As you can see, the stations 5 & 13 have little data.

Determination the final year of observations for all stations

data[['id', 'year']].groupby(by=['id']).max().sort_values(by=['year'], ascending=False)

year

id

22 2021

14 2021

3 2021

5 2021

10 2021

16 2021

15 2021

21 2018

20 2018

19 2018

18 2018

17 2018

1 2018

13 2018

2 2018

11 2018

9 2018

8 2018

7 2018

6 2018

4 2018

12 2018

As you can see, only stations 3, 5, 10, 14, 15, 16 and 22 have modern data.

Although, if you limit yourself to 2018, then you can take all the stations.

Consider only stations 14, 15, 16.

```
# Information about stations 14, 15, 16
stations_good = [14, 15, 16]
data_about[data_about['id'].isin(stations_good)]
```

id	length	name_station
13	14	582.0 р. Південний Буг, 582 км, м. Вінниця, Сабарівське вдсх, питний в/з міста, вище міста
14	15	607.0 р. Південний Буг, 607 км, с. Гущинці, нижче села , питний водозабір м.Калинівка
15	16	652.0 р. Південний Буг, 652 км, м. Хмільник, питний в/з, вище міста

```
# Set target indicator
target_data_name = 'Suspended'
#feature_target_all = ['NH4', 'BSK5', 'NO3', 'NO2', 'SO4', 'PO4', 'CL']
feature_target_all = ['NO3', 'NO2']
feature_data_all = feature_target_all + [target_data_name]
feature_data_all
['NO3', 'NO2', 'Suspended']
# Data sampling only for good stations
df_indicator = data[['id', 'ds'] + feature_data_all]
df_indicator =
df_indicator[df_indicator['id'].isin(stations_good)].dropna().reset_index(drop=True)
df_indicator
```

id	ds	NO3	NO2	Suspended
0	14	2000-10-01	6.30	0.300 9.0
1	14	2000-01-02	8.80	0.270 8.0
2	14	2000-01-03	8.80	0.090 11.0
3	14	2000-04-04	4.60	0.090 13.0
4	14	2000-05-16	3.00	0.050 10.0
...	...	...	...	... ...
758	16	2020-06-10	1.90	0.100 12.0
759	16	2020-03-11	2.06	0.044 9.0
760	16	2020-08-12	3.38	0.060 13.0
761	16	2021-03-16	7.70	6.790 9.0

762 16 2021-06-04 6.38 0.164 9.0

763 rows × 5 columns

```
cols = []
for station in stations_good:
    for feature in feature_data_all:
        cols.append(str(station) + "_" + feature)
cols
['14_NO3',
 '14_NO2',
 '14_Suspended',
 '15_NO3',
 '15_NO2',
 '15_Suspended',
 '16_NO3',
 '16_NO2',
 '16_Suspended']
df = pd.pivot_table(df_indicator, index=["ds"], columns=["id"],
values=feature_data_all).dropna()
df.columns = cols
df
```

	14_NO3	14_NO2	14_Suspended	15_NO3	15_NO2	15_Suspended	16_NO3	16_NO2	16_Suspended
ds									
2000-01-02	0.270	0.130	0.130	8.80	8.40	7.70	8.0	12.0	9.0
2000-01-03	0.090	0.170	0.270	8.80	9.10	8.80	11.0	17.0	9.0
2000-01-08	0.240	0.150	0.160	1.50	7.00	0.90	18.0	20.0	22.0
2000-04-04	0.090	0.140	0.090	4.60	4.90	3.50	13.0	12.0	18.0
2000-04-07	0.170	0.320	0.360	2.30	2.10	1.70	15.0	18.0	17.0
...	...	...	...	...	...	...	...	...	...
2020-09-15	0.064	0.051	0.072	1.03	0.57	0.84	15.0	13.0	10.0
2020-10-06	0.037	0.038	0.052	0.24	0.44	0.96	6.0	9.0	7.0
2020-12-08	0.061	0.036	0.046	0.68	1.03	0.71	18.0	10.0	12.0
2021-03-16	0.122	0.134	6.790	10.90	8.56	7.70	8.0	10.0	9.0
2021-06-04	0.129	0.179	0.164	6.57	8.07	6.38	8.0	8.0	9.0

```
plt.scatter(x, y_test_rf, label = "Random Forest prediction", color = 'y')
plt.scatter(x, y_test_lgb, label = "LGBoost Regressor prediction", color = 'brown')
plt.plot(x, np.full(len(test), 0.5), label = "Maximum allowable value", color = 'r')
plt.title('Prediction for the test data')
plt.legend(loc='best')
plt.grid(True)
```

7. Select the best model

Back to Table of Contents

```
# Display results of modeling
result.sort_values(by=['valid_score', 'train_score'], ascending=False)
```

	model	train_score	valid_score
0	Linear Regression	0.54	0.43
2	LGBoost Regressor	0.35	0.20
1	Random Forest Regressor	0.58	0.13

```
# Select models with good training results
result_best = result[(result['train_score'] > result['valid_score'])]
# Select models with minimal overfitting
result_best = result_best[(result_best['train_score'] - result_best['valid_score']).abs() < 0.15]
result_best.sort_values(by=['valid_score', 'train_score'], ascending=False)
```

	model	train_score	valid_score
0	Linear Regression	0.54	0.43
2	LGBoost Regressor	0.35	0.20

```
# Select the best model
result_best.nlargest(1, 'valid_score')
```

	model	train_score	valid_score
0	Linear Regression	0.54	0.43

```
# Find a name of the best model (with maximal valid score)
best_model_name =
result_best.loc[result_best['valid_score'].idxmax(result_best['valid_score'].max()), 'model']
print(f'The best model is "{best_model_name}"')
```

The best model is "Linear Regression"

I hope you find this notebook useful and enjoyable.

Your comments and feedback are most welcome.

Go to Top

Додаток Д

(обов’язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ЗМІН КОНЦЕНТРАЦІЇ ЗАВИСЛИХ РЕЧОВИН У РІЧЦІ ПІВДЕННИЙ БУГ У 4 КВАРТАЛІ 2021 РОКУ

Виконав: студент гр. 2ІСТ-20м

_____ Поліщук Д.А.

«___» _____ 2021 р.

Керівник: к.т.н., доцент

_____ Ящолт А.Р.

«___» _____ 2021 р.

Нормоконтроль: к.т.н., доцент

_____ Жуков С. О.

«___» _____ 2021 р.

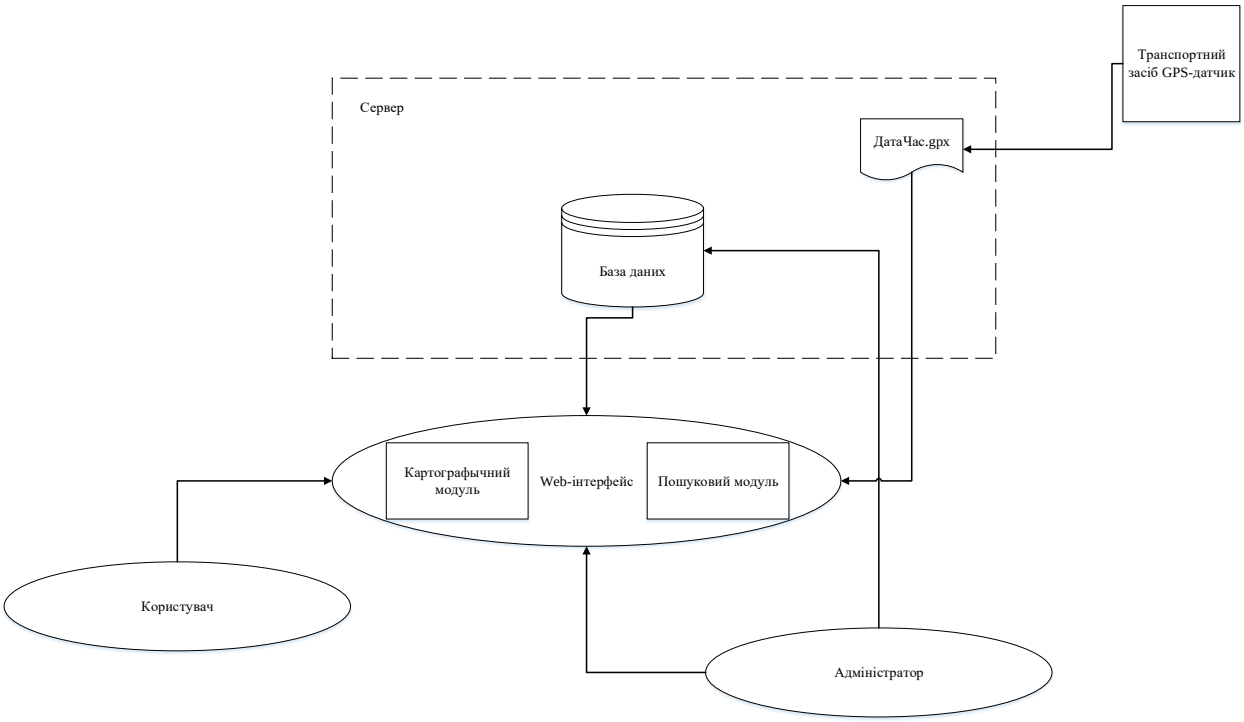


Рисунок Д.1 – Загальна структура інформаційної технології аналізу та прогнозування змін концентрації завислих речовин у річці Південний Буг

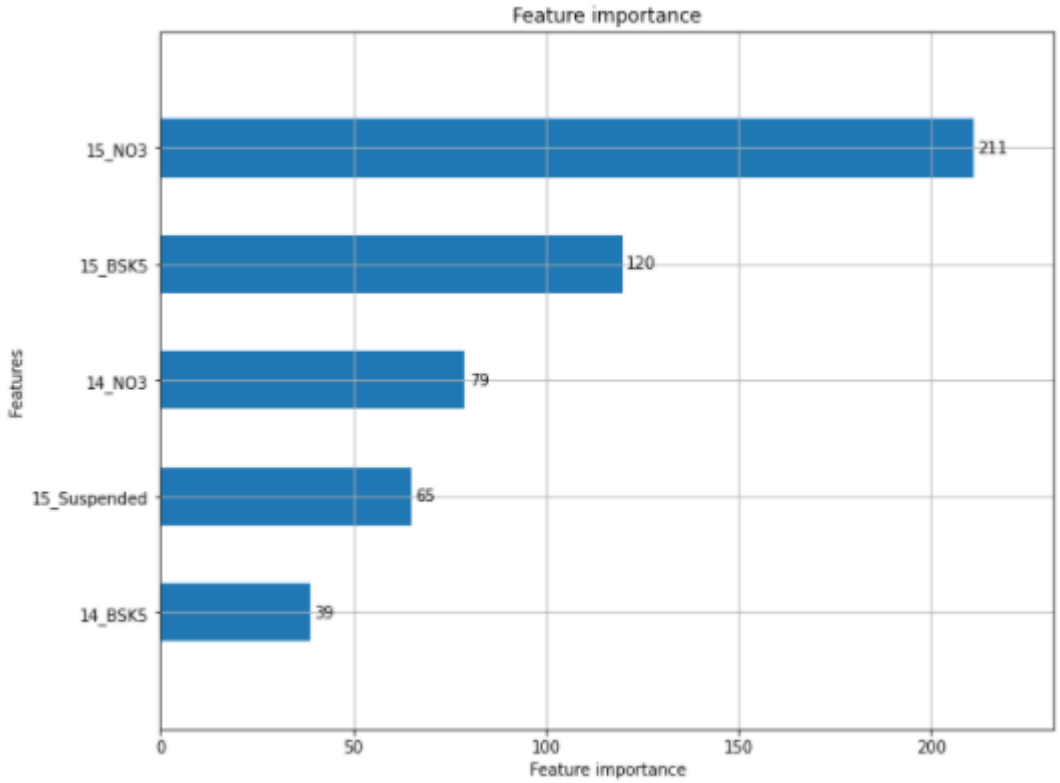


Рисунок Д.2 – Діаграма важливості ознак

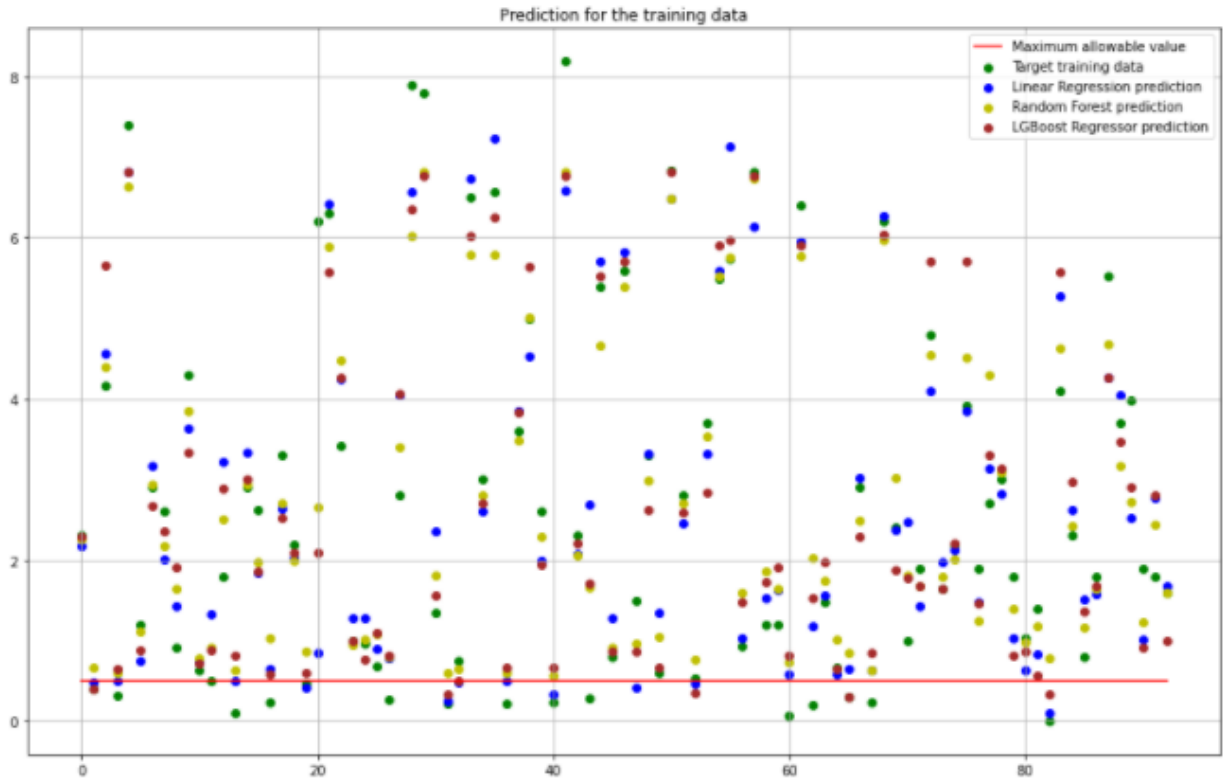


Рисунок Д.3 – Прогнозування навчальних даних

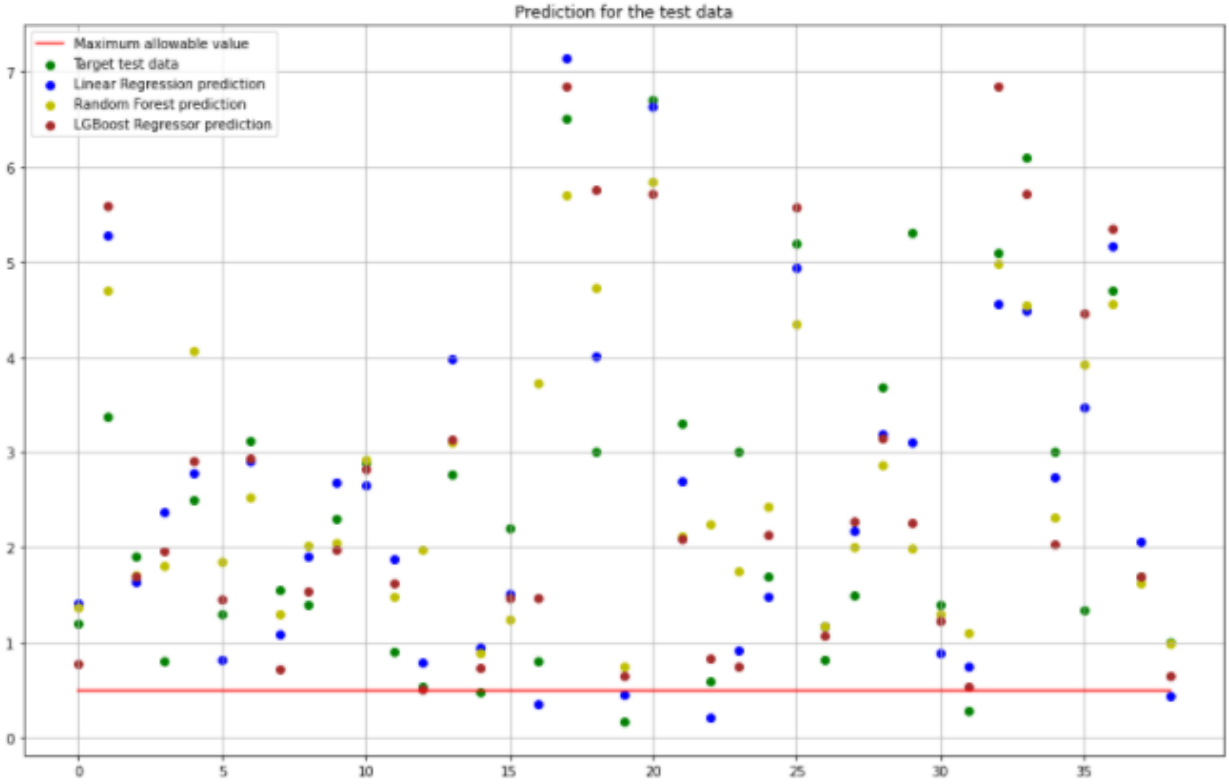


Рисунок Д.4 – Прогнозування тестових даних

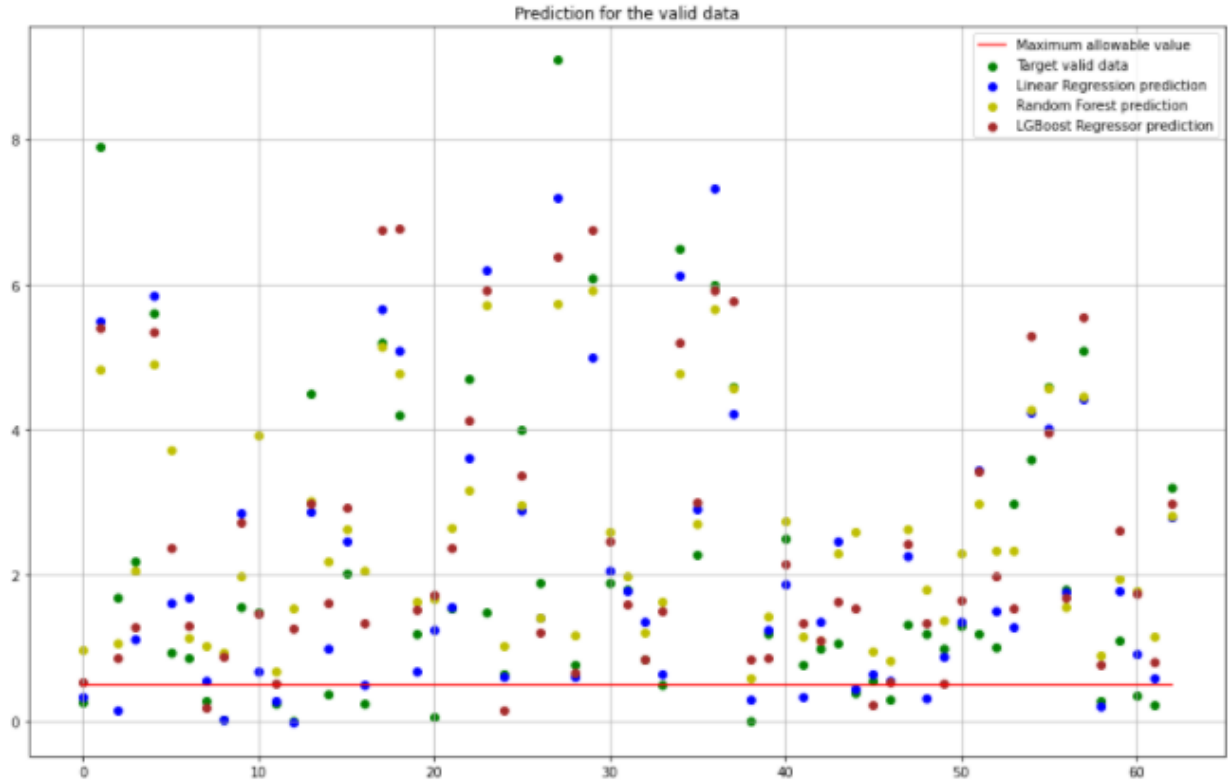


Рисунок Д.5 – Прогнозування валідаційних даних

In [47]:

```
# Select the best model
result_best.nlargest(1, 'valid_score')
```

Out[47]:

	model	train_score	valid_score
0	Linear Regression	0.84	0.76

Рисунок Д.6 – Найкраща модель прогнозування