Національний університет «Полтавська політехніка імені Юрія Кондратюка»
(повне найменування вищого навчального закладу)

Навчально-науковий інститут інформаційних технологій та робототехніки
(повна назва інституту)

Кафедра комп'ютерних та інформаційних технологій і систем
(повна назва кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

магістра
(рівень вищої освіти)

на тему:

«Використання методів машинного навчання для аналізу та прогнозування екологічних викликів мегаполісів»

Виконав: студент 6 курсу, групи 6д-ТН

спеціальності:
_____122 Комп'ютерні науки_____
(шифр і назва спеціальності)

_____Пищида В.В._____
(прізвище та ініціали)

Керівник: ___Скакаліна О.В._____

(прізвище та ініціали)

Полтава – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»
НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І РОБОТОТЕХНІКИ

КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА
спеціальність 122 «Комп'ютерні науки»
на тему

«Використання методів машинного навчання для аналізу та прогнозування екологічних викликів мегаполісів»

Студента групи 6д-ТН Пищиди Володимира Васильовича

Керівник роботи
кандидат технічних наук,
доцент Скакаліна О.В.

Завідувач кафедри
кандидат фізико-математичних наук,
Двірна О.А

Полтава – 2025

РЕФЕРАТ

Кваліфікаційна робота магістра: 76 с., 31 малюнків, 2 додатки, 20 джерел.

Об’єкт дослідження: рівень трафік, загальний рівень забрудненості повітря міст як залежність від метеорологічних чинників

Мета роботи: збір метеорологічних даних, даних трафіку та рівня забрудненості повітря низки міст з метою проведення аналізу, порівняльної характеристики та побудови моделей машинного навчання з використанням алгоритмів МГУА в поєднанні з генетичним алгоритмом

Методи: розробка власного сервісу мовою програмування JavaScript та запуск його на віддаленому сервері для збору даних щогодини у режимі реального часу. Розробка та використання Python скриптів для обробки даних, кластеризації, побудови моделей та візуалізації даних

Ключові слова: алгоритми МГУА, генетичний алгоритм, трафік, TomTom індекс, індекс забруднення повітря (AQI), лаги

ABSTRACT

Master's Thesis: 76 pages, 31 figures, 2 appendices, 20 sources.

Research object: Traffic level, overall air pollution level in cities as a function of meteorological factors.

Objective: To collect meteorological data, traffic data, and air pollution levels from various cities for analysis, comparative characterization, and to develop machine learning models using GMDH algorithms combined with a genetic algorithm.

Methods: Development of a custom service in JavaScript and its deployment on a remote server for real-time hourly data collection. Development and use of Python scripts for data processing, clustering, model building, and data visualization.

Keywords: GMDH algorithms, genetic algorithm, traffic, TomTom index, Air Quality Index (AQI), lag

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

МГУА (GMDH) – сімейство індуктивних алгоритмів для комп'ютерного моделювання наборів даних. Алгоритми автоматично будують та оптимізують моделі поліноміальних нейронних мереж на основі даних і дозволяють таким чином виявляти взаємозв'язки між вхідними та вихідними змінними.

TomTom індекс – показник, розроблений компанією TomTom, що відома своїми продуктами GPS-навігації. Цей індекс використовується для вимірювання рівня заторів на дорогах у різних містах і країнах по всьому світу. Він визначає, наскільки час поїздки у певні години пік вищий у порівнянні з часом поїздки за умов ідеального руху, тобто коли на дорогах відсутні затори.

Агрегований квадратичний TomTom індекс ($ATTI^2$) - інтегральний показник, що обраховує середній квадрат TomTom індексу протягом певного тривалого проміжку часу для певного міста

Індекс забруднення повітря (AQI) - це показник, який використовується для оцінки якості повітря та визначення ступеню його забруднення

ЗМІСТ

ВСТУП

У сучасному світі проблема транспортного трафіку та атмосферного забруднення стає дедалі актуальнішою через стрімке зростання урбанізації, інтенсивність використання автомобільного транспорту та недостатню екологічну політику в багатьох країнах. Велике навантаження на транспортні мережі міст не лише погіршує якість життя, а й значно впливає на стан навколишнього середовища, особливо через збільшення викидів парникових газів та шкідливих речовин та здоров'я місцевих мешканців, що при тривалій дії може мати істотний негативний вплив на здоров'я. Таким чином, аналіз транспортного трафіку та його зв'язок із рівнем атмосферного забруднення є важливим завданням для розробки рішень, спрямованих на покращення екологічної ситуації.

Важливим інструментом у вирішенні цієї проблеми є використання сучасних методів аналізу даних та машинного навчання, які дозволяють виявляти приховані закономірності та створювати прогностичні моделі. Одним із перспективних підходів у цій сфері є метод групового урахування аргументів (МГУА), який дозволяє будувати точні та інтерпретовані моделі для аналізу складних залежностей між рівнем транспортного трафіку та параметрами забруднення атмосфери.

Метою цієї роботи є дослідження возначення рівня заторів у містах та взаємозв'язку між транспортним трафіком та атмосферним забрудненням у близько 25 відомих містах світу з використанням методів машинного навчання, зокрема, моделі МГУА. У рамках дослідження передбачається зібрати дані про рівень трафіку, концентрації основних забруднювальних речовин у повітрі (CO_2, PM2.5, NO_2 та інші) та у режимі реального часу із використаням сервісу, що був спеціально написаний для даної роботи. Аналіз отриманих даних та побудова прогнозних моделей дозволять визначити ключові чинники, що впливають на

рівень забруднення, та запропонувати рекомендації для зменшення негативного впливу транспорту на довкілля.

Результати цього дослідження можуть бути використані для розробки ефективних транспортних стратегій та екологічних програм, спрямованих на поліпшення якості повітря в міських зонах. Впровадження таких рішень є важливим кроком у досягненні сталого розвитку міст і забезпеченні здорового середовища для їхніх мешканців.

РОЗДІЛ 1
ОСНОВНІ ТЕОРЕТИЧНІ ЗАСАДИ МЕТОДУ ГРУПОВОГО УРАХУВАННЯ АРГУМЕНТІВ

1.1 Виникнення МГУА та основні особливості

Метод групового урахування аргументів (англ. Group Method of Data Handling, GMDH) — алгоритм машинного навчання, розроблений у 1968 році радянським академіком Олексієм Григоровичем Івахненком [15]. Його основою є індуктивний підхід і принцип самоорганізації. Варто зазначити, що даний метод було розроблено ще до активної розробки та використання нейронних мереж.

МГУА базується на ідеї індуктивного підходу, що означає побудову моделі на основі емпіричних даних. Мета методу полягає в тому, щоб навчитися обирати оптимальну модель з множини можливих моделей, використовуючи дані для поступового поліпшення моделі. Однією з ключових концепцій методу є самоорганізація. Це означає, що модель, створена за допомогою МГУА, автоматично налаштовується на найкращу структуру, яка підходить для опису залежностей між вхідними змінними та вихідними результатами

МГУА, як ключовий інструмент теорії індуктивного моделювання, є одним із найсучасніших методів обчислювального інтелекту та м'яких обчислень. Цей метод відзначається оригінальністю і високою ефективністю при вирішенні широкого кола завдань у сфері штучного інтелекту, включаючи ідентифікацію та прогнозування, розпізнавання образів, кластеризацію, інтелектуальний аналіз даних і виявлення закономірностей. Ефективність методу багато разів підтверджувалася розв'язанням безлічі конкретних задач з областей екології, економіки та інших галузей. Розв'язувані задачі:

1. Апроксимація функцій.

2. Прогнозування часових рядів.
3. Класифікація образів.
4. Кластеризація вибірки даних.
5. Налаштування структури нейронних мереж.

Протягом останнього десятиліття інтерес до МГУА значно зріс у всьому світі. Це пов'язано не тільки з відомою ефективністю методу, але й з підвищенням популярності технологій штучних нейронних мереж. Справа в тому, що структуру МГУА можна розглядати як нейронну мережу, яка має унікальну здатність до самоорганізації як на рівні структури, так і параметрів [2]. Серед явних переваг МГУА виділяються автоматична побудова мережевої структури, простота і швидкість налаштування параметрів, а також можливість отримати явний математичний вираз для побудованої мережі.

Популярність МГУА підтверджується проведенням міжнародних форумів з індуктивного моделювання, а також створеними компаніями, котрі спеціалізуються на створенні програмного забезпечення, ключовим елементом якого є саме даних набір алгоритмів.

Алгоритми МГУА мають низки особливостей, що для певних типів задач може мати вирішальне значення та являти собою велику перевагу у порівнянні з іншими методами машинного навчання. Завдяки своїм властивостям МГУА вважається одним із першопрохідних методів машинного навчання і самоорганізації.

1.2 Фундаментальні принципи МГУА та відмінність від інших методів машинного навчання

Фундаментальні принципи, що лежать в основі МГУА та клас задач, у яких ці алгоритми навіть часто можуть показувати кращі результати, ніж нейронні мережі.

Принцип №1 Самоорганізація. Під час навчання алгоритм МГУА не лише підбирає числові параметри, але й самостійно визначає найкращу структуру моделі, використовуючи процес відбору серед моделей-кандидатів різної складності. Наприклад, для моделювання задачі, що описується поліномом, потрібно підібрати не лише коефіцієнти, але й саму форму полінома.

Принцип №2 Запобігання перенавчанню. МГУА автоматично надає перевагу моделям, що мають оптимальний баланс між точністю та складністю, що допомагає знизити ризик перенавчання. Як відомо, нейронні мережі, як і інші численні класичні методи машинного навчання, мують важу перенавчання, тобто ускладнивши сильно модель можна досягти чудового результату на тестови даних, але на реальних даних модель може показувати суттєво гірш результати через надмірну спеціалізацію, а не на встановленні загальних особливостей, що і має відбуватися під час навчання

На першій ітерації обираються й навчаються найпростіші моделі-кандидати із заданого класу моделей. Далі, на кожному наступному етапі, складність моделей поступово збільшується. Цей процес триває доти, доки точність результатів покращується. Як тільки на черговому етапі виявляється, що якість моделей перестала зростати, процес навчання відразу завершується.

Рисунок 1.1 – Ітерації машинного навчання

На Рис. 1.1 показаний етап процесу класичного машинного навчання. На першій ітерації обираються та навчаються найпростіші моделі-кандидати із заданого класу. З кожним наступним етапом складність моделей зростає, і процес

триває, поки точність покращується. Ці етапи в МГУА називаються рядами, кожен з яких відповідає моделям однакової складності. Коли якість моделей перестає зростати, навчання зупиняється. Цей підхід ґрунтується на тому, що прості моделі недостатньо точно відображають закономірності, а складні — чутливі до шуму, тому МГУА вибирає оптимальні моделі, запобігаючи перенавчанню [15].

Існують різні методи запобігання перенавчання у машинному навчанні. Значною ж перевагою МГУА є те, що складність моделі визначається в процесі роботи алгоритму, що унеможливлює створення моделі з надмірно складність, а відтак перенавчання моделі[15].

Принцип №3 Використання критеріїв. Алгоритми МГУА застосовують внутрішні та зовнішні критерії для навчання моделей-кандидатів, що дозволяє обирати одну модель з оптимальною складністю. Як МГУА навчає моделі-кандидати та обирає оптимальну? Для цього використовуються 2 види критеріїв: внутрішні та зовнішні критерії [17]. Внутрішній критерій відповідає за підбір найкращих параметрів моделі. Одним з таких методів є метод найменших квадратів (МНК), який мінімізує суму квадратів відхилень між функцією та цільовими даними. Іншими словами, за допомогою лінійної алгебри підбираються параметри, при яких середньоквадратична помилка (MSE) є мінімальною.

$$\text{MSE} = \frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2 \qquad (1.1)$$

де y_i - істинне значення і-ї змінної, $\hat{y}_i$ - передбачене моделлю значення і-ї змінної, n - кількість змінних.

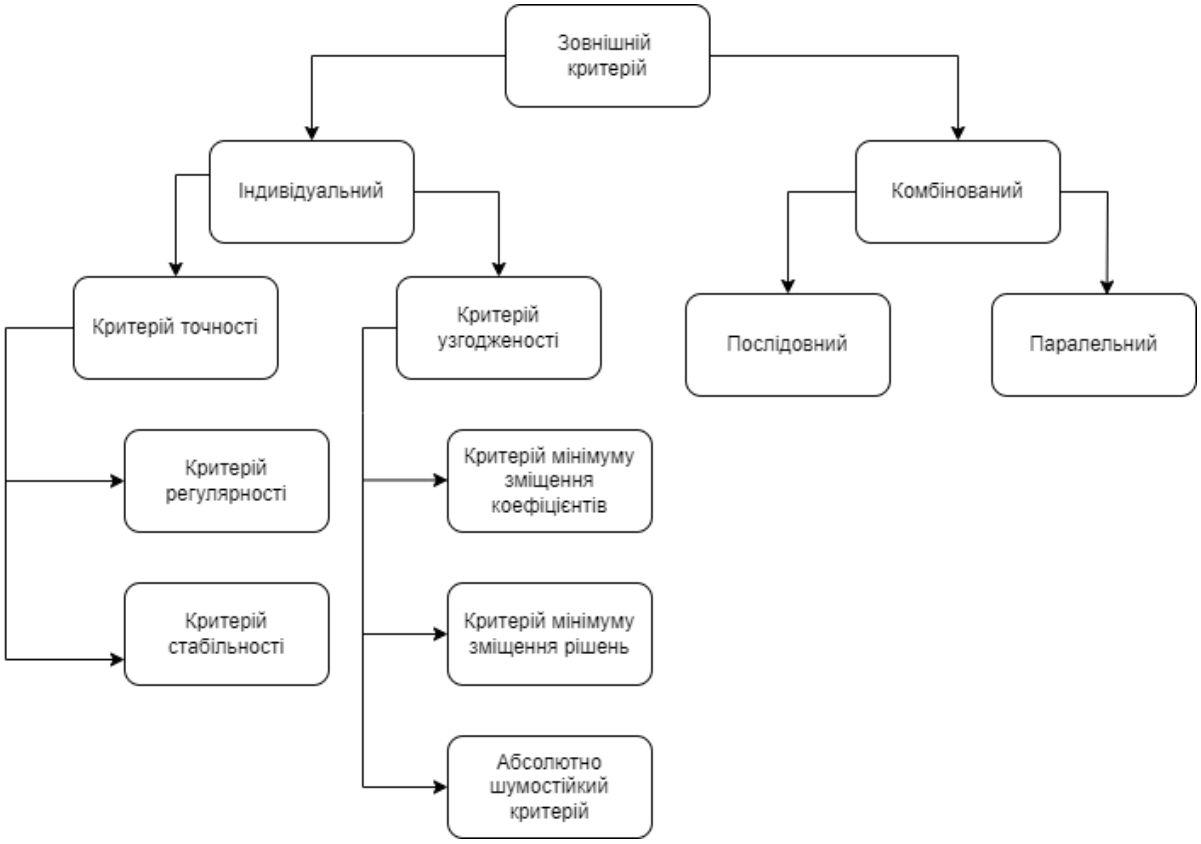

Рисунок 1.2 – Зовнішні критерії МГУА

Зовнішній критерій — це формула, яка використовує параметри навченої моделі та її прогнози на нових даних для визначення кількісної міри точності моделі. Чим менше значення цього критерію, тим краща модель. Зовнішній критерій є гіпер-параметром алгоритму і визначається перед початком навчання. На Рисунок 1.2 представлена класифікація та детальний опис різних типів зовнішніх критеріїв

Принцип №4 Поділ даних на три підвибірки. Поділ даних на три підвибірки — це важлива концепція в алгоритмі МГУА. На відміну від звичного розподілу на тренувальну та тестову вибірки, МГУА додає ще й перевірочну вибірку.

1. Навчальна вибірка складається з тренувальної частини — для підбору параметрів моделей за допомогою внутрішнього критерію та тестової частини — для вибору кращих моделей на основі зовнішнього критерію.

2. Перевірочна вибірка використовується після навчання для оцінки якості оптимальної моделі на нових даних/

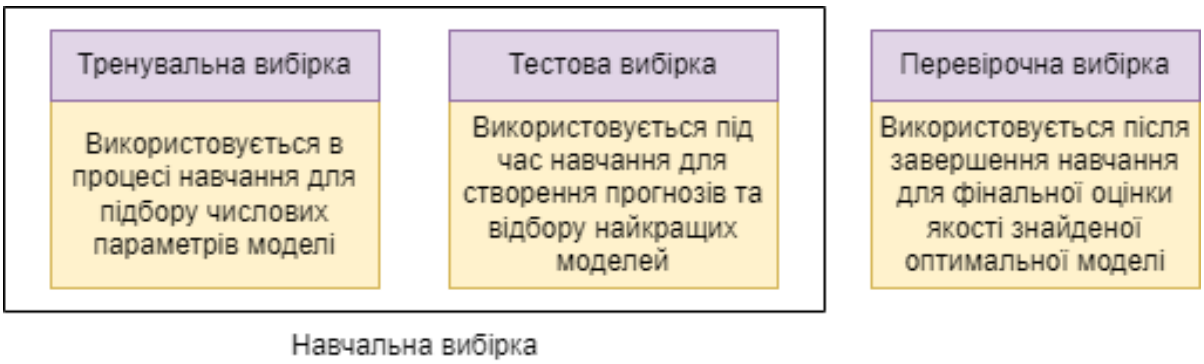

Рисунок 1.3 – Поділ даних для формування вибірок

Під час навчання одночасно використовуються тренувальна та тестова частини даних. На тренувальній частині застосовується внутрішній критерій для підбору параметрів моделей. Тестова частина служить для побудови прогнозів та відбору найкращих моделей на основі зовнішнього критерію. Після завершення навчання перевірочна вибірка використовується для об'єктивної оцінки якості знайденої оптимальної моделі.

1.3 Основні етапи роботи узагальненого алгоритму МГУА

До основних етапів роботи узагальненого алгоритму МГУА можна віднести:

1. Формування початкового набору аргументів: на першому етапі визначаються незалежні змінні (фактори), на основі яких буде побудовано

модель. Аргументи можуть бути простими змінними або базовими функціями від змінних (наприклад, поліномами).

2. Початкове навчання: алгоритм будує найпростішу модель на основі мінімальної кількості факторів (аргументів). Це може бути, наприклад, лінійна регресійна модель або модель, яка використовує базові поліноми.

3. Покрокове ускладнення моделі: алгоритм послідовно додає нові аргументи або модифікує вже наявні, формуючи нові, більш складні моделі. Цей процес відбувається шляхом перебору різних комбінацій аргументів, включаючи їх поліноміальні та нелінійні варіанти.

4. Оцінка якості моделей: для кожної побудованої моделі розраховується певний зовнішній критерій, який відображає її точність або стабільність. Це можуть бути різні критерії, такі як середньоквадратична помилка, критерій стабільності або регулярності. Моделі також можуть оцінюватися за критерієм узгодженості (наскільки стабільно модель поводиться на різних вибірках).

5. Відбір найкращих моделей: на кожному кроці алгоритм обирає найкращі моделі на основі їх продуктивності (точності, узгодженості, стабільності). Ці моделі використовуються для подальшого ускладнення.

6. Зупинка алгоритму: алгоритм продовжує будувати нові моделі доти, поки якість моделі покращується на нових ітераціях. Якщо покращення не відбувається або стає незначним, процес зупиняється.

7. Побудова фінальної моделі: фінальна модель може бути просто найкращою серед усіх побудованих або комбінацією кількох найкращих моделей. Це забезпечує кращу генералізацію і стабільність результатів.

1.4 Гіпер-параметри МГУА

Гіпер-параметри — це параметри алгоритму машинного навчання, налаштування яких дозволяє керувати процесом навчання. Гіпер-параметри задаються вручну до етапу навчання [2]. Загальними гіпер-параметрами алгоритмів МГУА є:

1. Зовнішній критерій (criterion).
2. Частка тестової частини в навчальній вибірці (test_size).
3. Кількість кращих моделей ряду, на основі яких буде розраховано якість усього ряду (p_average).
4. Мінімально необхідне покращення якості нового ряду для прийняття рішення про продовження навчання (limit). Крім того, деякі алгоритми мають додаткові гіпер-параметри.
5. Кількість кращих моделей, на основі яких будуть формуватися ускладнені моделі на наступному ряду (k_best).
6. Тип базових поліномів, з яких будуть формуватися моделі різного рівня складності (polynomial_type).

Гіпер-параметри та їх можливі значення докладно описані в розділах з оглядом конкретних алгоритмів МГУА.

1.6 МГУА та нейронні мережі

Нейронні мережі, як і МГУА (метод групового урахування аргументів), належать до алгоритмів глибокого навчання. Вони отримують знання з вхідних наборів даних і апроксимують складні закономірності, виявлені в цих даних. Однак підходи, що лежать в основі МГУА та нейронних мереж, значно відрізняються. Принципи роботи МГУА, описані вище, частково усувають деякі

поширені проблеми, з якими часто стикаються при роботі з нейронними мережами.

Переваги МГУА над нейронними мережами:

1. Самостійний підбір оптимальної структури МГУА автоматично підбирає оптимальну структуру моделі під час навчання. На відміну від нейронних мереж, де структуру (наприклад, кількість шарів і нейронів) необхідно вибирати вручну або використовувати пошукові алгоритми.

2. Результат не залежить від початкової ініціалізації параметрів У МГУА результат менш чутливий до початкових параметрів моделі, тоді як в нейронних мережах початкова ініціалізація ваг може суттєво впливати на кінцевий результат і вимагати ретельного підбору.

3. Автоматична боротьба з перенавчанням МГУА має вбудовані механізми для боротьби з перенавчанням (overfitting), що робить його більш стійким до ситуацій, коли модель може добре працювати на тренувальних даних, але не на тестових [15].

4. Чудово працює на дуже малих вибірках даних МГУА здатний показувати хороші результати навіть на малих наборах даних [2]. Нейронні мережі, навпаки, часто потребують великих обсягів даних для досягнення хороших результатів, оскільки вони мають більше параметрів для навчання.

1.7 Різновиди алгоритмів МГУА

Під МГУА на сьогодні мається на увазі вже не один конкретний метод, а ціле сімейство алгоритмів, заснованих на загальних принципах. Алгоритми відрізняються класом моделей-кандидатів, а також умовами формування та відбору змінних при переході від одного ряду до іншого. Розв'язання практичних задач і розробка теорії МГУА призвели до створення широкого спектру програмних алгоритмів.

Таблиця 1.1 – Алгоритми МГУА

Змінні	**Параметричні**	**Непараметричні**
Неперервні	Комбінаторний (COMBI)	Об'єктивне комп'ютерне кастерування (ОСС)
	Багатошаровий ітераційний (MIA)	Алгоритм кластеризації “Вказуючий палець” (PF)
	GN	Комплексування аналогів (АС)
	Об'єктивний системний аналіз (OSA)	
	Гармонійний	
	Дворівневий (ARIMAD)	
	Мультиплікативно-адаптивний (MAA)	
Дискретні або бінарні	Гармонічна повторна дискретизація	Алгоритм на основі теорії статистичних рішень (MTSD)

Алгоритми переважно відрізняються один від одного способом генерації набору моделей-кандидатів для заданої базової функції, методом ускладнення структури моделей і, нарешті, використаними зовнішніми критеріями. Вибір алгоритму залежить від специфіки задачі, рівня дисперсії шуму, достатності вибірки даних і того, чи містить вибірка лише неперервні дані.

Основні параметричні алгоритми МГУА, наведені в таблиці, були розроблені для роботи з неперервними змінними. Серед параметричних алгоритмів найбільш відомими є:

1. Основний комбінаторний алгоритм (COMBI): цей алгоритм базується на повному або частковому переборі поступово ускладнених моделей і їх оцінці за зовнішнім критерієм на окремій частині вибірки даних.

2. Алгоритм багатошарової ітерації (MIA): на кожному шарі процесу перебору використовується одна й та ж часткова описова модель (ітераційне правило). Він корисний, коли потрібно обробляти велику кількість змінних.

3. Алгоритм об'єктивного системного аналізу (OSA): основна особливість полягає в тому, що він розглядає не окремі рівняння, а системи алгебраїчних або різницевих рівнянь, отриманих за допомогою імпліцитних шаблонів (без цільової функції). Перевага алгоритму — поступове збільшення кількості регресорів, що дозволяє краще використовувати інформацію, вбудовану у вибірку даних.

4. Дворівневий алгоритм (ARIMAD): призначений для моделювання довгострокових циклічних процесів (наприклад, фондового ринку або погоди). Він використовує системи поліноміальних або різницевих рівнянь для ідентифікації моделей на двох часових шкалах, після чого обирає найкращу пару моделей за значенням зовнішнього критерію. Для цього можна використовувати будь-який із параметричних алгоритмів, описаних вище.

Менш відомі параметричні алгоритми, які застосовують повний перебір різницевих, гармонічних або гармонічно-експоненціальних функцій. Також є Мультиплікативно-адитивний алгоритм (MAA), у якому поліноміальні моделі отримуються шляхом логарифмування добутку вхідних змінних.

Параметричні алгоритми МГУА виявилися дуже ефективними для моделювання об'єктів із чіткими характеристиками, наприклад, інженерних об'єктів. Однак, коли йдеться про моделювання об'єктів із нечіткими характеристиками, ефективніше використовувати непараметричні алгоритми МГУА, у яких поліноміальні моделі замінюються вибірками, розділеними на

інтервали або кластери. Ці алгоритми повністю вирішують проблему усунення зміщення оцінок коефіцієнтів.

Приклади параметричних алгоритмів:

1. Алгоритм об'єктивного комп'ютерного кластерування (ОСС): працює з парами точок вибірки, розташованих близько одна до одної. Він знаходить фізичне кластерування, яке повинно бути схожим на двох підвибірках.

2. Алгоритм "Вказуючий палець" (PF): використовується для пошуку фізичного кластерування шляхом побудови двох ієрархічних дерев кластеризації та оцінки за балансним критерієм.

3. Алгоритм комплексування аналогів (АС): використовує набір аналогів замість моделей і кластеризацій. Рекомендований для моделювання об'єктів із високим рівнем нечіткості.

4. Ймовірнісний алгоритм, заснований на багатошаровій теорії статистичних рішень: рекомендований для розпізнавання і прогнозування бінарних об'єктів, а також для контролю достовірності вхідних даних з метою уникнення можливих помилок експертів.

1.8 Розгляд найпоширеніших алгоритмів

Розглянемо більш детально найпоширеніші алгоритми. До алгоритмів МГУА належать, що були створені найперше, належать:

- комбінаторний алгоритм COMBI;
- багаторядний ітеративний алгоритм MIA.

З часом обидва ці алгоритми отримали розвиток і їхні удосконалення перетворилися на нові самостійні алгоритми:

- комбінаторно-селекційний алгоритм MULTI;
- релаксаційний ітеративний алгоритм RIA.

Комбінаторний алгоритм (COMBI) є базовим алгоритмом МГУА. Він розглядає лише лінійні моделі, тобто виражає залежність цільової величини у як лінійну комбінацію змінних $x_1 \ldots x_n$. На першому етапі перебираються всі моделі-кандидати такого вигляду:

$$y(x_i) = w_0 + w_1 x_i, \ \text{i} \in [1, \text{n}] \tag{1.2}$$

Методом найменших квадратів (МНК) розраховуються параметри w_1 для кожної з моделей. Після цього обчислюється зовнішній критерій (criterion), на основі якого моделі сортуються в порядку зростання. Середнє значення зовнішнього критерію для першого ряду paveragep_{average}paverage моделі приймається за показник якості поточного ряду. Позначимо цю величину як E_1.

Далі алгоритм переходить до другого ряду, і моделі-кандидати ускладнюються до вигляду:

$$y(x_i, x_j) = w_0 + w_1 x_i + w_2 x_j, \ \text{i, j} \in [1, \text{n}], \ \text{i} \neq \text{j} \tag{1.3}$$

Знову за допомогою МНК розраховуються параметри w_1, w_2, після чого для кожної моделі другого ряду обчислюється зовнішній критерій. Після сортування за зовнішнім критерієм визначається показник якості другого ряду, і також обчислюється середнє значення критерію для кращих моделей. Далі якість другого ряду порівнюється з якістю попереднього ряду:

$$E_1 - E_2 \geq \text{limit}$$

Якщо якість другого ряду значно покращується, алгоритм продовжує навчання, переходячи до третього ряду, де перебираються ще більш складні моделі y(x_i, x_j, x_k).

Такий процес продовжується доти, поки не буде знайдено оптимальну модель із найменшою складністю, або поки не припиниться поліпшення показника.

Максимальна кількість рядів для даного алгоритму дорівнює nnn. Кількість моделей-кандидатів на кожному ряду визначається як M = C_n^k, де C_n^k — кількість комбінацій моделей на кожному етапі. Загальна кількість моделей на всіх рядах обчислюється за формулою [17]:

$$\text{M} = \sum_{k-1}^{n} C_n^k = 2^n - 1$$

Комбінаторно-селекційний алгоритм MULTI є вдосконаленим варіантом комбінаторного алгоритму COMBI. Він також розглядає лише лінійні моделі, тобто виражає залежність цільової величини у як лінійну комбінацію змінних $x_1 \ldots x_n$

Проте, цей алгоритм має додатковий механізм селекції, який дозволяє значно зменшити кількість моделей-кандидатів і тим самим прискорити процес навчання. Селекція на кожному етапі проводиться шляхом відбору найкращих моделей-кандидатів з попереднього ряду, на основі яких генеруються нові моделі для наступного ряду. Це допомагає уникнути перебору всіх можливих моделей-кандидатів, вибираючи лише найкращі комбінації для побудови складнішої моделі. На першому етапі перебираються всі моделі-кандидати такого вигляду:

$$y(x_i) = w_0 + w_1 x_i, \ \text{i} \in [1, \text{n}]$$

Методом найменших квадратів (МНК) розраховуються параметри w_1 для кожної з моделей. Після цього обчислюється зовнішній критерій (criterion), на основі якого моделі сортуються за зростанням. Далі обирається k_{best} найкращих

моделей. Середнє значення зовнішнього критерію для першого ряду приймається за показник якості поточного ряду. Позначимо цю величину як E_1.

Алгоритм переходить до другого ряду, де кожна з k_{best} найкращих моделей y(x_i) доповнюється новою змінною x_j, для якої i≠j. Це дозволяє побудувати нові моделі другого ряду:

$$y(x_i, x_j) = w_0 + w_1 x_i + w_2 x_j, \ \text{i, j} \in [1, \text{n}], \text{i} \neq \text{j} \qquad (1.4)$$

Знову ж таки, МНК використовується для розрахунку параметрів w_1 і w_2, і після цього обчислюється зовнішній критерій для кожної моделі другого ряду. Після сортування значень критерію обираються k_{best} найкращих моделей. Середнє значення зовнішнього критерію для другого ряду E_2 визначається як середній результат для найкращих моделей. Далі порівнюється якість поточного ряду з попереднім:

$$E_1 - E_2 \geq \text{limit}$$

Якщо якість другого ряду значно покращується, алгоритм продовжує навчання, переходячи до третього ряду і додаючи ще більш складні моделі y(x_i, x_j, x_k) .

Процес триває, поки покращення не стане незначним або зовсім зупиниться. У такому випадку найкраща модель оптимальної складності обирається на основі мінімального значення зовнішнього критерію.

Максимальна кількість рядів для даного алгоритму дорівнює n. Кількість моделей-кандидатів на кожному ряду обмежується до $M_j \leq k_{best}$. Загальна кількість моделей-кандидатів на всіх рядах дорівнює $M_j \leq k_{best} * n^2$.

Багаторядний ітеративний алгоритм (MIA) є історично першим алгоритмом МГУА [17]. Він дозволяє будувати складні нелінійні моделі за рахунок використання найкращих моделей одного ряду як нових змінних для наступного ряду. В основі методу генерації моделей-кандидатів лежить базовий поліном, який вибирається до початку навчання.

Спочатку на першому ряду розглядаються всі моделі, які відповідають обраному виду поліному. Починаючи з другого ряду, метод продовжує будувати нові моделі відповідно до заданого поліному, але при цьому як змінні використовуються вже не вихідна вибірка даних із nnn змінними, а поліноми k_{best} найкращих моделей попереднього ряду. Таким чином, підсумкова функція буде функцією від множини інших функцій, кожна з яких також може бути функцією від функцій і так далі.

Можливі види базових поліномів:

$$f(x_i, x_j) = w_0 + w_1 x_i + w_2 x_j$$
$$f(x_i, x_j) = w_0 + w_1 x_i + w_2 x_j + w_{12} x_i\, x_j$$
$$f(x_i, x_j) = w_0 + w_1 x_i + w_2 x_j + w_{11}\, {x_i}^2 + w_{22}\, {x_j}^2$$

Релаксаційний ітеративний алгоритм (RIA) є вдосконаленням багаторядного ітеративного алгоритму (MIA). Він дозволяє будувати складні нелінійні моделі, використовуючи найкращі моделі одного ряду як нові змінні для наступного ряду [2]. В основі генерації моделей-кандидатів лежить базовий поліном, який вибирається до початку навчання. Основна відмінність цього алгоритму від MIA полягає в модифікованому способі формування моделей-кандидатів на нових рівнях, що дозволяє прискорити процес навчання.

На першому етапі розглядаються всі моделі, що відповідають обраному виду поліному. Починаючи з другого ряду, алгоритм продовжує будувати нові

моделі відповідно до обраного поліному. Однак як змінні використовуються не лише початкові дані, а також найкращі моделі попереднього ряду. Моделі-кандидати формуються на основі двох змінних: одна змінна з початкового набору, а друга — це поліном моделі з попереднього ряду.

Можливі типи базових поліномів:

$$f(x_i, x_j) = w_0 + w_1 x_i + w_2 x_j$$
$$f(x_i, x_j) = w_0 + w_1 x_i + w_2 x_j + w_{12} x_i\, x_j$$
$$f(x_i, x_j) = w_0 + w_1 x_i + w_2 x_j + w_{11}\, {x_i}^2 + w_{22}\, {x_j}^2$$

1.9 Програмне забезпечення для застосування МГУА

Алгоритми МГУА (Метод групового урахування аргументів) мають давню історію і застосовуються в багатьох сферах, таких як прогнозування, моделювання, аналіз даних та машинне навчання. Ось кілька найпоширеніших бібліотек, програмного забезпечення та компаній, які використовують або популяризують алгоритми МГУА [17]:

1. GMDH Shell: це одне з найвідоміших комерційних програм для роботи з алгоритмами МГУА. Це програмне забезпечення спеціалізується на прогнозуванні, моделюванні та аналізі даних, використовуючи алгоритми МГУА для побудови багатошарових нейронних мереж і адаптивних моделей. GMDH Shell автоматизує процес вибору моделі та гіпер-параметрів, що робить його зручним для користувачів із різним рівнем технічної підготовки. Основні застосування включають фінансовий аналіз, енергетику, прогнозування ринку та інші сфери.

2. GMDH in Python (gmdh-py): це Python-бібліотека, яка реалізує базові концепції МГУА для машинного навчання та прогнозування. Вона дозволяє

користувачам легко будувати нелінійні моделі та нейронні мережі з використанням МГУА. Це зручний інструмент для дослідників і розробників, які працюють з науковими та інженерними задачами, де необхідна робота з часовими рядами та регресією.

3. MATLAB Toolbox for GMDH: відоме середовище для наукових обчислень і моделювання, яке має спеціалізовані інструменти для роботи з МГУА. MATLAB пропонує кілька реалізації алгоритмів МГУА для регресійного аналізу, прогнозування та моделювання складних процесів. Це особливо корисно для інженерів та дослідників, які працюють з великими обсягами даних і складними моделями, включаючи енергетику, економіку та природничі науки.

4. GMDH in Excel: набір інструментів для роботи з алгоритмами МГУА, вбудований у Microsoft Excel. Це рішення є зручним для бізнес-аналітиків та фінансових експертів, оскільки Excel залишається одним із найпопулярніших інструментів для обробки та аналізу даних.

5. Oracle: відомий постачальник баз даних та аналітичного програмного забезпечення, який також пропонує рішення з використанням алгоритмів МГУА. Oracle інтегрує МГУА в свої платформи аналітики даних, що дозволяє побудувати моделі прогнозування і автоматизувати процес вибору оптимальних параметрів.

РОЗДІЛ 2
АНАЛІЗ ТА ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА ВІДОМИХ МІСТ СВІТУ ЗА ЗАВАНТАЖЕНІСТЮ ДОРІГ

2.1 Огляд проблеми завантаженості доріг трафіком у сучасному світі та методи її обрахування

Завантаженість доріг трафіком — одна з ключових проблем сучасних урбаністичних центрів. Із зростанням населення міст, збільшенням кількості автомобілів та недостатнім розвитком громадського транспорту, затори стають щоденним викликом для мільйонів людей. Вони впливають не лише на комфорт пересування, але й на економіку, екологію та здоров’я населення.

Затори на дорогах призводять до значних втрат часу, збільшення витрат на пальне та зниження ефективності роботи. У середньому водії в мегаполісах витрачають десятки, а іноді й сотні годин на рік у заторах. Екологічний вплив також є значним — постійно працюючі автомобільні двигуни виділяють більше шкідливих викидів, що сприяє забрудненню повітря та зміні клімату.

Для оцінки та моніторингу завантаженості доріг використовуються спеціальні індекси та системи [1, 20]. Вони допомагають аналізувати ситуацію, прогнозувати розвиток трафіку та приймати ефективні рішення для покращення транспортної інфраструктури. Серед найпоширеніших індексів особливо можна виокремити:

1. Індекс завантаженості (Congestion index): цей показник визначає, наскільки більше часу займає дорога в умовах заторів у порівнянні з ідеальними умовами. Наприклад, індекс 50% означає, що час у дорозі збільшується наполовину через затори.

2. TomTom traffic index: використовується компанією TomTom для аналізу трафіку у різних містах світу. Цей індекс базується на реальних даних GPS-пристроїв і дозволяє порівнювати міста за рівнем завантаженості доріг.

3. Індекс затримки (Delay index): даний показник оцінює середній час затримки через затори, що дозволяє зрозуміти масштаб проблеми в конкретному регіоні.

4. INRIX Global traffic scorecard: глобальний індекс, який аналізує завантаженість доріг у багатьох країнах світу. Він враховує середню швидкість пересування, затримки та загальний вплив заторів на економіку.

5. Індекс мобільності (Mobility index): даний індекс оцінює ефективність транспортного руху через врахування середньої швидкості транспорту, кількість затримок та їх тривалості.

Ефективне використання цих індексів дозволяє містам краще планувати транспортну інфраструктуру, розвивати громадський транспорт та зменшувати кількість заторів. Для прикладу, у містах із високим індексом завантаженості активно впроваджуються системи інтелектуального управління трафіком, будуються нові дороги, додаткові смуги, велосипедні доріжки, вдосконалюється громадський транспорт тощо [18].

Розв’язання проблеми завантаженості доріг є складним завданням, яке потребує комплексного підходу, що враховує як технічні, так і соціальні аспекти. Впровадження сучасних технологій та раціональне планування можуть зробити міста комфортнішими для життя та роботи.

2.2 Індекс TomTom

У даній роботі проводиться аналіз трафіку у відібраних для експерименту містах із використанням індексу TomTom. Він є одним із найвідоміших інструментів для аналізу завантаженості доріг у містах по всьому світу [8]. Це

обумовлено відносною легкістю його обчислення так широким використанням для планування транспортної інфраструктури.

Він був розроблений компанією TomTom, яка спеціалізується на навігаційних системах і картах. Цей індекс допомагає оцінити, наскільки завантажені дороги в певному місті, порівнюючи час у дорозі в умовах заторів з ідеальними умовами. TomTom розпочала свою діяльність у 1991 році як компанія з розробки програмного забезпечення для електронних пристроїв. Згодом, на основі зростання попиту на GPS-навігацію, компанія почала випускати портативні навігатори. Завдяки збору великих обсягів даних із пристроїв, компанія отримала можливість аналізувати трафік у реальному часі [20].

У 2010-х роках TomTom почала використовувати зібрані дані для створення глобального індексу завантаженості доріг. Сьогодні TomTom Traffic Index є одним із найдетальніших і найточніших інструментів аналізу дорожнього трафіку.

Індекс базується на величезних обсягах даних, які збираються з GPS-пристроїв, підключених автомобілів, мобільних додатків і інших джерел. Процес включає кілька етапів:

1. Збір даних:

1.1. Дані збираються в реальному часі від мільйонів користувачів по всьому світу.

1.2. Інформація включає швидкість руху, місцезнаходження автомобілів і час подорожі.

2. Порівняння часу в дорозі:

2.1. Аналізується час, необхідний для подорожі певним маршрутом, у різних умовах:

2.1.1. Ідеальні умови: коли немає заторів.

2.1.2. Реальні умови: з урахуванням заторів.

3. Обчислення індексу:

3.1. Індекс виражається у відсотках і показує, на скільки більше часу займає дорога через затори. Наприклад, якщо індекс становить 30%, це означає, що поїздка займає на 30% більше часу, ніж в умовах без заторів.

4. Річний звіт:

4.1. Щороку компанія випускає звіт, у якому порівнюються міста за рівнем завантаженості. У звіті також наводиться інформація про найзавантаженіші дні, час пік і тривалість заторів.

2.3 Особливості індексу TomTom

Індекс використовується урядами, міськими планувальниками, транспортними компаніями та звичайними водіями. Він допомагає планувати транспортну інфраструктуру, виявляти найбільш завантажені райони для пріоритетного вирішення проблем та знижувати затори через оптимізацію маршрутів і введення систем інтелектуального управління трафіком.

Його ключовими особливостями є:

- глобальне охоплення: TomTom traffic index охоплює понад 400 міст у більш ніж 50 країнах [18];
- реальний час і історичні дані: крім звітів, система надає дані в реальному часі, які використовуються для прогнозування трафіку та покращення планування [16];
- деталізація: Аналіз проводиться для різних днів тижня, часу доби та навіть окремих районів міста [18];

TomTom traffic index є “ефективним інструментом для боротьби з дорожніми заторами та створення комфортних умов для високої мобільності в міста” [20].

Загальна формула для його розрахунку має вигляд:

$$\text{TomTom index} = \text{duration\_in\_traffic} / \text{duration} \quad (2.1)$$

2.4 Проектне рішення щодо підрахунку загального TomTom індекса

Для розрахунку трафік індекса компанія TomTom пропонує свій API - це кілька сервісів, які мають різні тарифні плани. Але для цієї роботи було вирішено використовувати Google API, а саме сервіс Google Maps API, який використовується для отримання маршрутів між точками за координатами він враховує поточний стан трафіку. Загалом отримання цих даних у Google є платним, але для використання даної роботи використовувався Free Trial. Його використання виявилося цілком достатнім для отримання даних щодо обраного міста за кілька тижнів.

Google API пропонує дані, які містять два найважливіших поля, що можна використати для обрахунку TomTom traffic index:

1) duration: показує скільки часу водій проведе у дорозі за нормальних умов

2) duration_in_traffic: показує, скільки фактично часу водій проводить часу в дорозі з урахуванням трафіку

Таким чином, використовуючи ці дані по певному маршруту для певного моменту часу можна отримувати в реальному часі значення TomTom індексу для заданого маршруту.

2.5 Агрегація маршрутів для більш точного розрахунку

Оскільки в цій роботі ми прагнемо отримати загальний TomTom index для певного міста, нам потрібно взяти кілька маршрутів і усереднити по них значення TomTom індексу в даний момент часу. Обчислення загального індексу для певного міста проводилося шляхом вибору 3-х довгих маршрутів по

кожному місту з урахуванням його специфіки, щоб охопити ключові догори та відобразити загальний стан трафіку. Якщо, наприклад, місто розділене річкою, то обов'язково враховувалися маршрути як на одному березі, так і на другому. Коли певне місто виявляло якісь інші специфічні географічні особливості, то вони теж враховувалися: мости, узвишшя тощо. Обов'язково бралися до уваги маршрути в центрі міста та на периферії, щоб враховувати рух транспорту як у самому місті, так і на його околицях. Як приклад вибору маршрутів, які використовувалися для обчислення загального індексу TomTom по певному місту, може бути наступний приклад (рисунок 2.1). Як видно із наведеного зображення, обрані маршрути враховують особливості міста Київ (правий та лівий берег) і охоплюють тривалі маршрути для більш об'єктивної картини щодо завантаженості транспортом доріг міста у обраний момент часу.

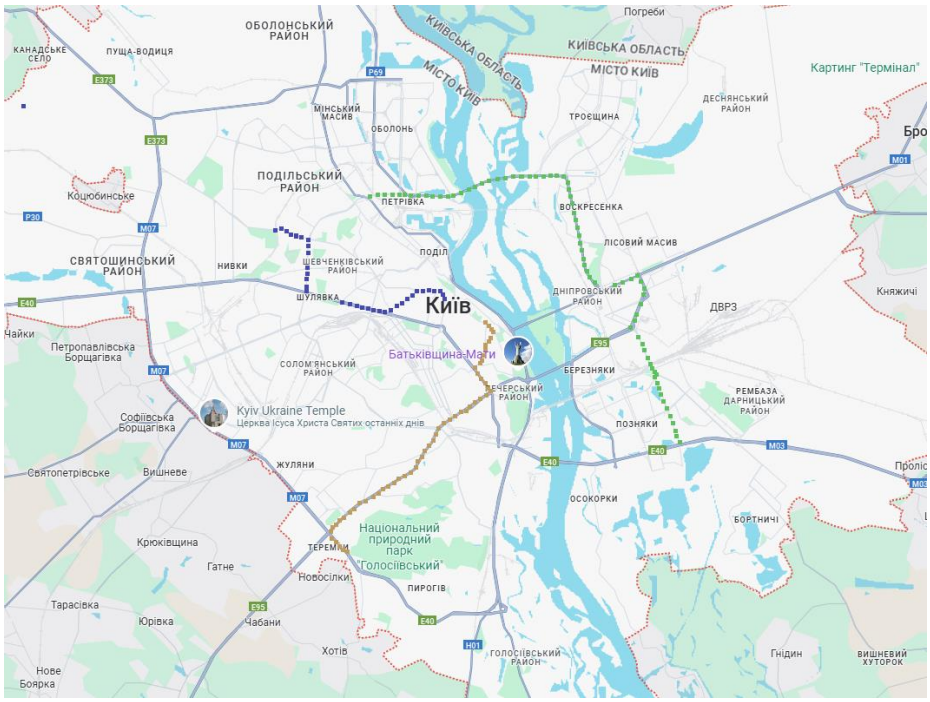

Рисунок 2.1 – Обрані 3 маршрути у місті Київ для обрахування загального TomTom індекса

Для отримання агрегованого значення TomTom index на певний момент часу по кожному місту була використана агрегація по всіх 3 обраним маршрутам:

$$\text{TomTom index} = \text{duration\_in\_traffic} / \text{duration}$$

$$\text{Aggregated TomTom index} = \sum_{i}^{n} \frac{\text{duration in traffic}_\text{i}}{\text{duration}}$$

де і змінюється від 1 n, де n - кількість маршрутів

(2.2)

Зрозуміло, що TomTom index зазвичай близький до 1, але у години пік його значення може істосно бути вищим, сягаючи кількох одиниць. Іншою його особливістю є те, що у певні момент він може бути меншим одиниці. На перший погляд це видається дивним, але пов'язане з ним, що інколи водії знаходять коротші маршрути і реальна тривалість поїздки за низького рівня завантаженості доріг виявляється мершою.

2.6 Вибір міст для аналізу

Всього для даної роботи було відібрано близько 30 міст із різних частин частин світу, різних кліматичних зон, розмірів. Головна ідея полягала в тому, щоб, зібравши достатньо даних по кожному з них, провести порівняльний аналіз та знайти особливості у рівні трафіку та забрудненості, що корелюють із зазначеними особливостями міст. Наприклад, вплив клімату - більш сухий чи більш вологий, наближеність до морського узбережжя, розмір міста тощо. Наявність широкого набору із даних міст, на мою думку, дозволяє провести більш широкий аналіз та виявити особливості, які б при більш вузькому відборі цілком імовірно залишилися б непоміченими. У відборі міст також враховувалась наявність достатньої кількості працюючих станцій для

вимірювання забруднення повітря, оскільки, як з'ясувалося, для значної кількості міст ці дані не оновлюються вчасно на сервісах, API яких використовувалося для отримання даних, що унеможливлює якісний аналіз.

До відібраних міст потрапили 2 міста України - Київ та Львів, великі столиці більшості європейських країн - Варшава, Берлін, Париж, Лондон, Афіни, Амстердам, Осло разом з меншими містами - Брно, Цюрих, Познань, а також великі міста на інших континентах такі як Лос-Анджелес, Ванкувер, Токіо, Пекін, Сідней та деякі інші.

2.7 Опис сервісу для збору статистичних даних

Як уже було сказано в попередньому розділі, у даній роботі використовується Google Maps API для розрахунку TomTom індексу. Цей сервіс розроблений з трьох основних елементів, кожен з яких відповідає інтеграції з необхідним сервісом. У даному розділі ми не розглядатимемо інтеграцію пов'язану з отриманням погодних даних та даних про забрудненість повітря оскільки це буде зроблено у наступному розділі й там є свої особливості, пов'язані зі структурою індекса забрудненості, вибором метеорологічних станцій для збору даних тощо.

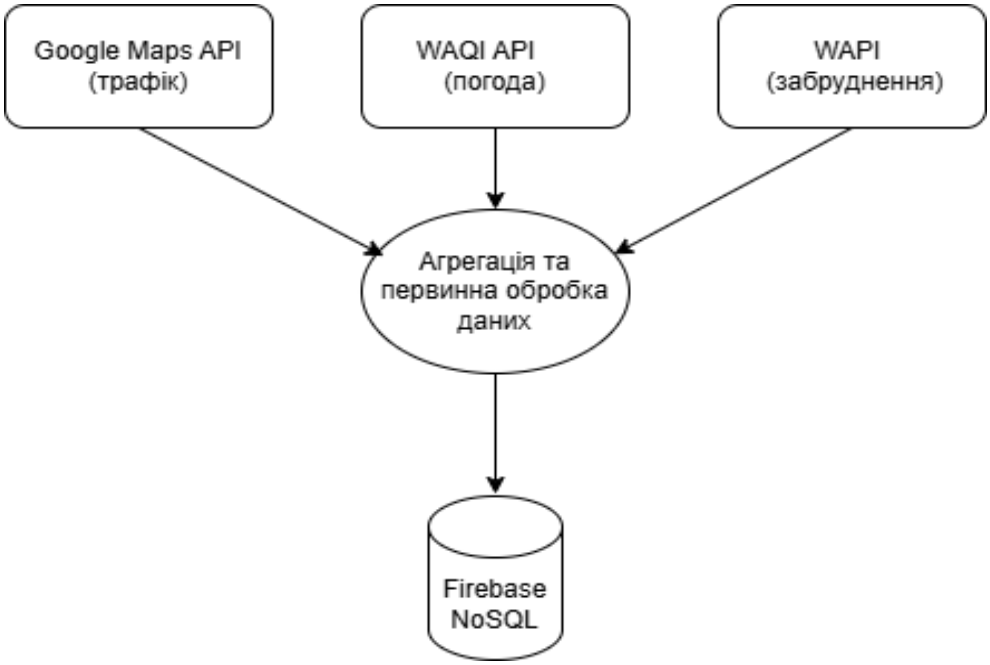

Рисунок 2.2 – Схема розробленого сервісу для збору статистики

На рисунку 2.2 зображена схема створеного сервісу, що був розроблений для збору усіх необхідних статистичних даних та запущений на віртуальному сервері. Робота сервісу полягала у зборі даних з 3-х сервісів через відкритий API: Google Maps - для трафіку, WAQI API - для збору забруднення та WAPI - для збору поточних погодних даних. Тут ми поки що обмежимося розглядом лише однієї частини, яка відповідальна за отримання даних щодо трафіку від Google API.

Сервіс було розроблено на Javascript із застосуванням системи контейнеризация Docker для швидкого розгортання а запуску на віртуальному сервері. Також для обробки та візуалізації даних використовувався Python. Посилання на створений репозиторій із вихідним кодом можна знайти у додатках:

2.8 Загальні висновки

Дані збиралися сервісом у вигляді часового ряду. Щогодини сервіс (рисунок 2.2) збирав дані з описаних сервісів та зберігав їх у NoSQL Firebase хмарній базі даних. Як можна бачити на рисунку. 2.3, для кожної мітки часу, для якої збиралися дані, робився запис у вигляді нового документа. Таким чином, можна легко знайти дані по кожному місту на потрібний момент часу. У підсумку був проведений збір даних за приблизно 20 днів, що дозволило охопити значну варіабельність як руху транспорту у робочі, вихідні дні, денні та нічні години, а також погодних умов - зміни температури, опадів, швидкості вітру.

Продовження збору даних на більш тривалий час могло б спричинити перевищення відведених для free-trial підписки обмежень для користування сервісом Google Maps.

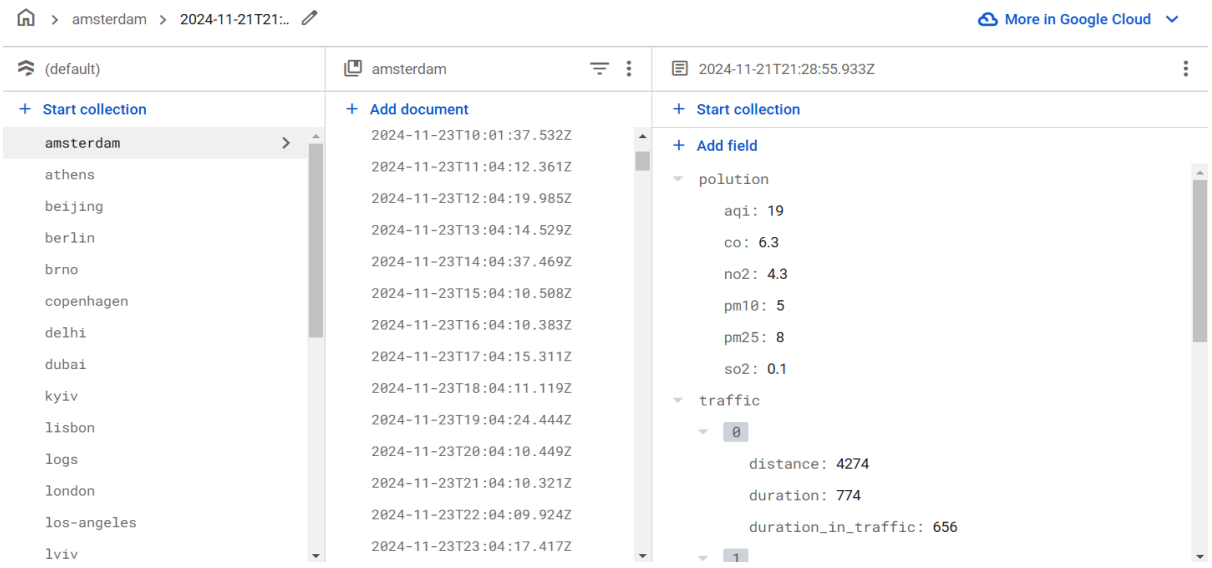

Рисунок 2.3 – Зібрані дані щодо різних міст у хмарній NoSQL базі даних Firebase

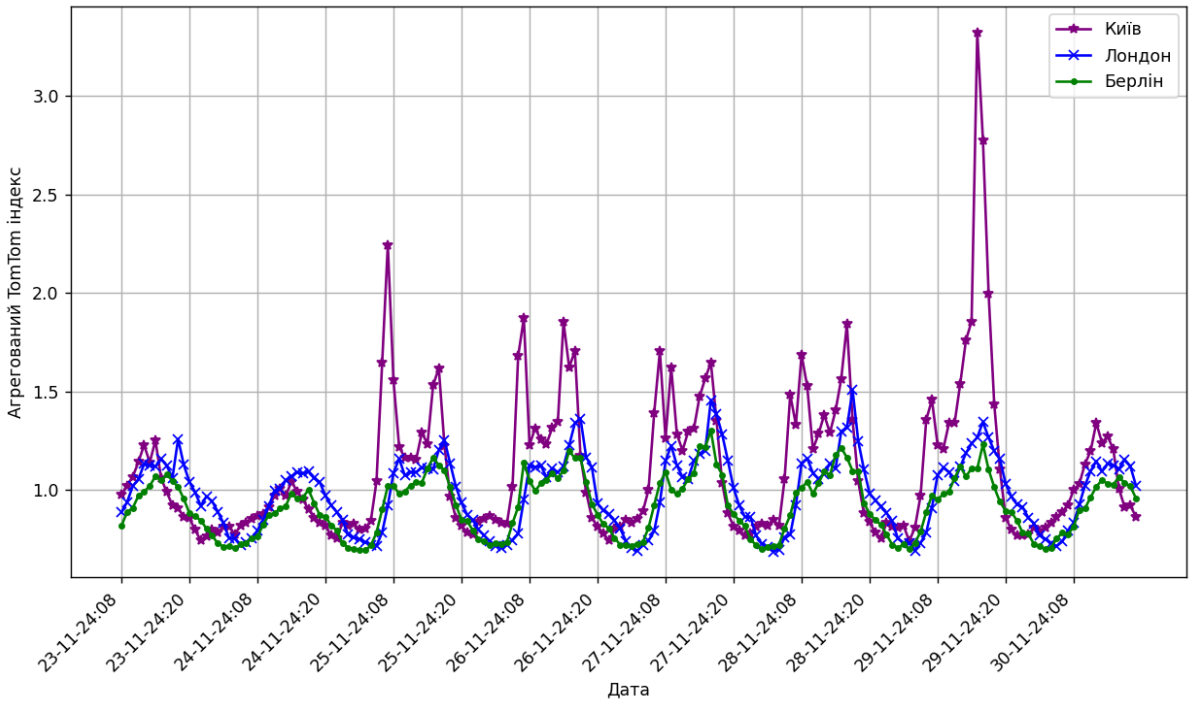

Рисунок 2.4 – Агрегований TomTom індекс у містах Київ, Лондон, Берлін з 23.11.24 до 01.12.24

Дані по 3-м містам (Київ, Берлін, Лондон) за 7 днів наведено для порівняння на рисунку 2.4. Із наведених графіків можна зробити наступні висновки:

1. Індекс TomTom для усіх 3-х міст добре описує як денну, так і тижневу специфіку руху. Чітко видно денні коливання рівня трафіку - характерні максимуми вранці та ввечері.

2. В той час як усі наведені міста виявляють відносно плавне збільшення трафіку вранці та зниження ввечері, особливо значні денні коливання помітні для Києва - чіткі піки вранці та ввечері.

3. Найменший рівень коливань спостерігається у Берліні - максимум 1.3, у Лондоні більше - 1.5. У Києві на початку та кінці робочого тижня TomTom індекс перевищує позначку 2 і сягає 3.5 у п'ятницю. Така особливість очевидно пов'язана із значним поверненням мешканців та гостей міста до Києва на початку робочого тижня та виїздом за межі міста у п'ятницю ввечері.

4. У вихідні дні індекс в усіх містах відрізняється незначно і є істотно нижчим за той, який характерний для робочих днів, проте все ж зберігається та послідовність, що і для робочих днів: для Берліна - найменший, для Лондона - більший та найвищий для Києва.

5. Для деяких міст, на даному графіку Києва, є характерним наявність чітко виражених піків. Вони можуть бути тривалими - близько 3 годин, досягати значень у 2 та більше.

Виходячи із вищезазначених міркувань 1-4, можна зробити висновки, що даний метод збору даних та підрахунку TomTom індекса відмінно пояснює особливості транспортного руху кожного міста і може слугувати надійною метрикою для визначення рівня трафіку, щоб у подальшому бути використаним для моделювання забрудненості міст. Це особливо важливо, позаяк загальновідомо, що викиди, які формуються через транспорт, є одним із головних забруднюючих чинників великих міст.

Висновок 5 вказує на те, що деякі міста можуть виявляти характерні сплески, котрі хоча й можуть бути відносно тривалими, суттєво погіршують транспортну ситуацію в місті, адже спричиняють значні затори і як наслідок уповільнення мобільності населення.

2.9 Обрахування загального рівня заторів

Як випливає із висновку 5, даному у попередньому розділі, хоча TomTom індекс дає дуже реалістичні оцінки поточної транспортної ситуації у місті, він виявляє недолік при спробі агрегувати дані за певний проміжок часу для розрахунку агрегованого рівня заторів у певному місті. І ось чому. Для обрахунку середнього рівня заторів по кожному місту, можна було б скористатися кривими (рисунок 2.4), проінтегрувавши їх на поділивши значення інтегралу на часовий інтервал, протягом якого було проведено вимірювання

$$\text{агрегований TomTom індекс (ATTI)} = \int_a^b \text{TomTom index} \; / \, (b - a) \tag{2.3}$$

Але, як було сказано у пункті 5 висновків попереднього розділу, ця формула досить слабко враховує нетривалі піки, що можуть мати значні величини. Якщо користуватися вище наведеною формулою для обрахунку, то міста із значною варіативністю трафіку, що однозначно є негативним явищем і одним із ключових показників проблем із транспортним рухом, але у яких водночас у не пікові години TomTom індекс відносно невисокий, будуть слабко виділятися на тлі міст, де відсутні подібні піки, але де, скажімо, більш активний рух у нічні години. Це може бути пов'язане із особливостями життя у тій чи іншій країні, заборонами - комендантські години, свята тощо. Щоб усунути цей очевидний недолік використання чистого агрегованого TTI, у даній роботі пропонується використовувати загальну техніку для більшого “штрафування” показників за значні відхилення - використання квадрату вихідної величини. Таким чином, навіть короткі піки у 1-2 години, але які при цьому мають значні значення, що перевищують 2, 3, будуть значно додавати до загальної величини інтервалу.

З урахуванням зазначених міркувань, фінальна формула для розрахунку агрегованого рівня заторів (квадратичного) для певного міста набуває вигляду:

$$\text{агрегований квадратичний TomTom індекс } (\text{ATTI}^2) = \int_a^b \text{TomTom index}^2 \; / \, (b - a) \tag{2.4}$$

Для обрахунку інтегралів TomTom індексу по різним містам був використаний більш точний чисельний метод Сімпсона, оскільки він забезпечує

кращу точність. Даний метод має більшу обчислювальну складність, але виходячи із наявних дата-сетів, це не викликало ніяких труднощів.

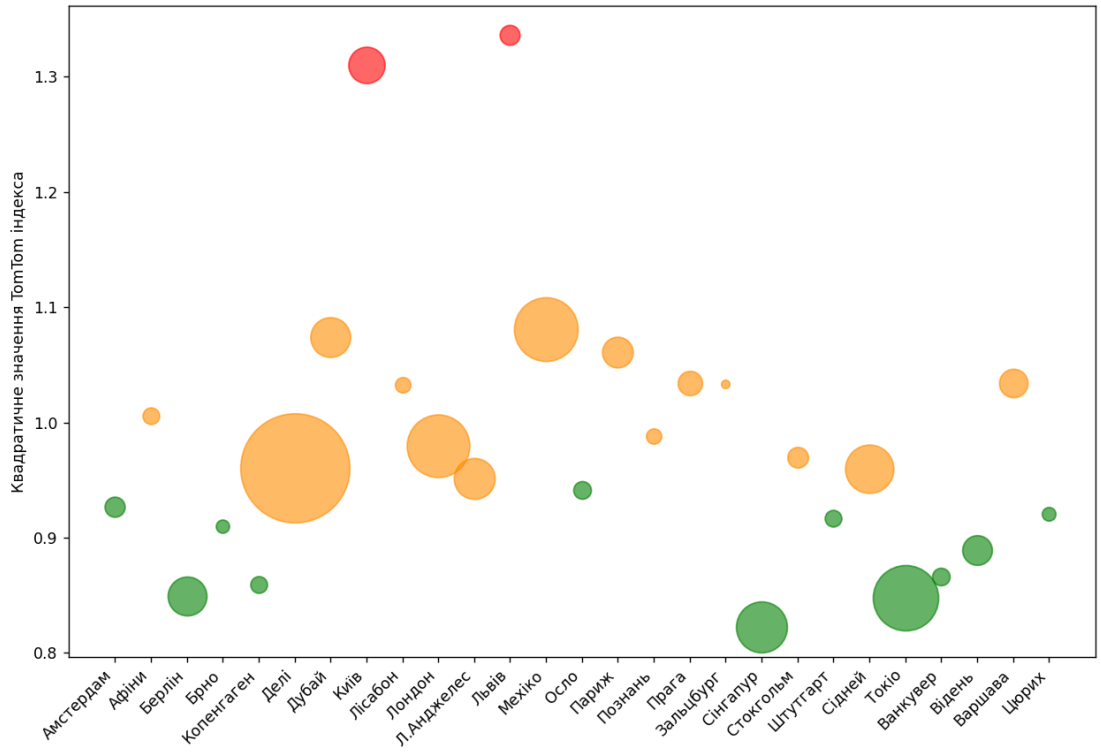

Рисунок 2.5 – Агрегований квадратичний TomTom індекс, K-means кластеризація

Результати знаходження агрегованого квадратичного TomTom індексу ($ATTI^2$) всіх міст, для яких збиралися дані із власного сервісу, у кластеризованому вигляді представлені на рисунку 2.5.

Кластеризація на рисунку 2.5 проведена за 3-а кластерами. Діаметр міста на діаграмі пропорційний кількості населення. До міст із найнижчим рівнем заторів очікувано потрапили такі міста як Копенгаген, Амстердам, Відень, Цюрих, Сингапур, Ванкувер. Певною неочікуваністю тут виявилося місто Токіо, адже відомо, що це місто налічує близько 9 млн. мешканців.

Токіо, один із найбільших мегаполісів світу, тривалий час зіштовхувався зі значною проблемою дорожніх заторів. Зі зростанням населення та економічним

розвитком Японії в другій половині XX століття ситуація на дорогах столиці ставала дедалі складнішою, а подекуди й просто критичною. Звичні методи розширення доріг і оптимізації руху вже не могли впоратися зі зростаючим рівнем дорожного завантаження. У таких умовах японські інженери та містобудівники запропонували інноваційне вирішення проблеми — систему багаторівневих міських доріг, відому як Шуто-експресвей або Шуто-дороги. Саме цьому інованіному рішенню, яке добре інтегровано у міську інфраструктуру, поєднане з іншими видами транспорту, слугує для зв'язку міста з передмістями Токіо переважною мірою завдячує свою позицію у “зеленому” кластері міст попри те, що це - одне з найбільших густонаселених міст світу.

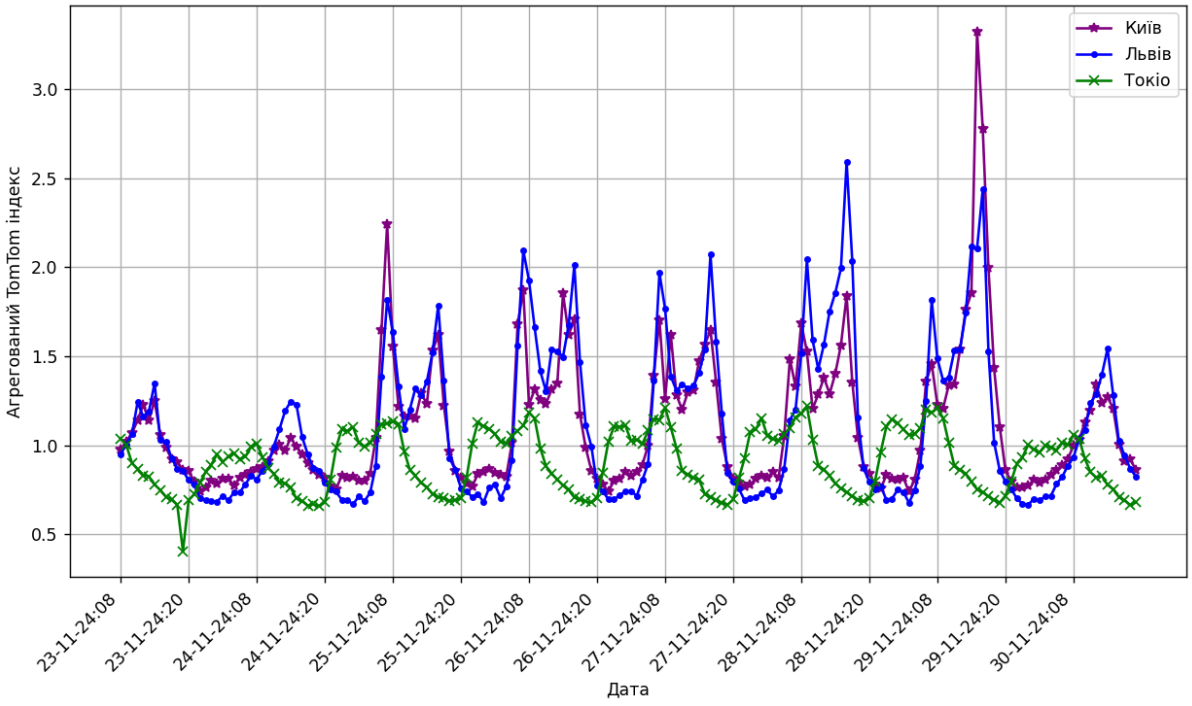

Рисунок 2.6 – TomTom індекс для Києва, Львова та Токіо

Багаторівнева система дала змогу суттєво збільшити загальну протяжність доріг без необхідності розширення їхньої площі на рівні землі. Шуто-дороги забезпечили відокремлення транзитного трафіку від місцевого руху, що значно

зменшило навантаження на звичайні міські вулиці. Окрім того, відсутність світлофорів і перехресть на Шуто-дорогах дозволяє підтримувати високу середню швидкість руху транспорту.

З іншого боку, помітно кластер міст із найскладнішою ситуацією із дорожніми заторами. До цих міст якраз належить Київ, який вже близько десятиліття відомий своїми регулярними заторами й посідає перші сходинки світового рейтингу за цим показником. Дещо неочікувано тут також розташувався Львів. Скоріше за все місто Лева перемістилося до кластеру міст зі значним рівнем заторів відносно нещодавно через істотне збільшення мешканців міста та підвищену роль у якості транспортного вузла останніми роками. Рисунок 2.6 дає більш детальне порівняння транспорту ситуацію у Києві, Львові та Токіо. Добре видно, що у Львові рівень заторів наближається до київських.

2.10 Висновки щодо використання квадратичного агрегованого TomTom індексу у якості індикатора рівня заторів

Із наведених графіків кластеризації (рисунок 2.5) та порівняння поточного агрегованого TomTom індексу, можна висновувати, що агрегований квадратичний TomTom індекс ($ATTI^2$) можна взяти за основу для визначення загального стану заторів у певномі місті. Він добре відображає не лише рівномірну завантаженість доріг у місті, але також має високу чутливість до піків, які є виразною ознакою проблем у транспортній інфраструктурі. TomTom індекс, що лежить у основі $ATTI^2$, як переконливо свідчать графіки на рисунку 2.4 та 2.6, добре відображає поточний стан трафіку, денні та вечірні коливання, зменшення та відповідне збільшення його рівня у вихідні та робочі дні, особливості понеділків та п'ятниць, коли пасажиропотік зазвичай вищий. З огляду на це, даний індекс та методика його обрахунку, на мою думку, може слугувати якісним індикатором для визначення як поточного, так і агрегованого

стану завантаженості доріг міст та може бути швидко обчислене з високою точністю й мінімальними витратами. Тому ця методика може бути рекомендована службам, що проводять моніторинг трафіку та муніципальним органам із метою постійного контролю ситуації з трафіком у місті та кількісної оцінки його змін.

РОЗДІЛ 3
МОДЕЛЮВАННЯ ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ РІВНЯ ЗАБРУДНЕНОСТІ

3.1 Постановка задачі моделювання рівня забрудненості повітря

Індекс якості повітря (AQI) є важливим показником, що використовується для оцінки стану атмосферного повітря та його впливу на здоров'я людей. Його значення обчислюється на основі концентрацій ключових забруднювачів, таких як PM2.5, PM10, CO, NO_2, SO_2 та O_3 [3]. Моделювання AQI ускладнюється через нелінійні взаємозв’язки між забруднювачами, погодними умовами та іншими факторами, які має враховувати модель. У цьому контексті метод групового урахування аргументів (МГУА) є одним із перспективних підходів для побудови прогнозних моделей.

Метод групового урахування аргументів (МГУА) — це індуктивний підхід до побудови математичних моделей, який базується на автоматичному відборі функцій та їх комбінацій для опису взаємозв’язків між змінними. Головною перевагою МГУА є здатність до пошуку адекватної моделі без потреби у попередньому визначенні структури залежностей. Алгоритм формує набір моделей, аналізує їхню ефективність і обирає найкращу за критеріями якості та простоти. Етапи моделювання AQI за допомогою МГУА передбачають:

1. Збір та підготовка даних:

   1.1) Збір даних про концентрації забруднювачів повітря.

   1.2) Отримання даних про погодні умови (температура, вологість, швидкість вітру, атмосферний тиск тощо).

   1.3) Формування тимчасових лагів для врахування затриманих ефектів впливу забруднювачів.

2. Формування навчальної вибірки: дані розділяються на навчальну та тестову вибірки. Підготовлені змінні включають як концентрації забруднювачів, так і погодні параметри, рівень трафіку.

3. Побудова моделі: алгоритм МГУА ітеративно формує базові функції та аналізує їхній внесок у підвищення точності моделі. Ці функції можуть включати нелінійні залежності, комбінації змінних і лагові змінні.

4. Оцінка моделі: для оцінки точності моделі використовуються метрики, такі як середня абсолютна помилка (MAE) та коефіцієнт детермінації (R^2). Модель перевіряється на тестовій вибірці для визначення її узагальнювальної здатності.

5. Інтерпретація результатів: МГУА забезпечує високу інтерпретованість моделі, що дозволяє оцінити вплив кожного аргументу на AQI. Це допомагає зрозуміти, які саме фактори найбільше впливають на якість повітря та дає змогу оцінити вплив кожного з них.

Алгоритми МГУА мають низку переваг, що є ключовими для виконання даної роботи, а саме:

1. Адаптивність: МГУА автоматично формує модель, що відповідає особливостям даних.

2. Інтерпретованість: побудовану модель легко зрозуміти, оскільки наявний її аналітичний вигляд, що дозволяє пояснювати вплив змінних на AQI.

3. Ефективність для малих вибірок: метод добре працює навіть у випадках обмеженої кількості даних, що є поширеною проблемою у моніторингу якості повітря.

Таким чином, МГУА є потужним інструментом для моделювання індексу якості повітря завдяки його здатності враховувати нелінійні залежності, інтерпретованісті моделей та ефективності при роботі з малими вибірками.

Комбінування цього підходу разом із звстосуванням генетичних алгоритмів для підбору параметрів моделі може ще покращити точність моделей.

Використання цих методів може значно підвищити точність прогнозів AQI та надати нові дані щодо впливу різних факторів на забруднення повітря, що є важливим для розробки екологічних політик і заходів щодо покращення якості повітря [14].

3.2 Особливості збору даних для моделювання та подальшого аналізу

Перш ніж перейти до створення моделей та їхнього аналізу на основі зібраних даних, потрібно сказати про особливості використовуваних даних та чому вони збиралися з допомого власного сервісу, а не було використано наявні

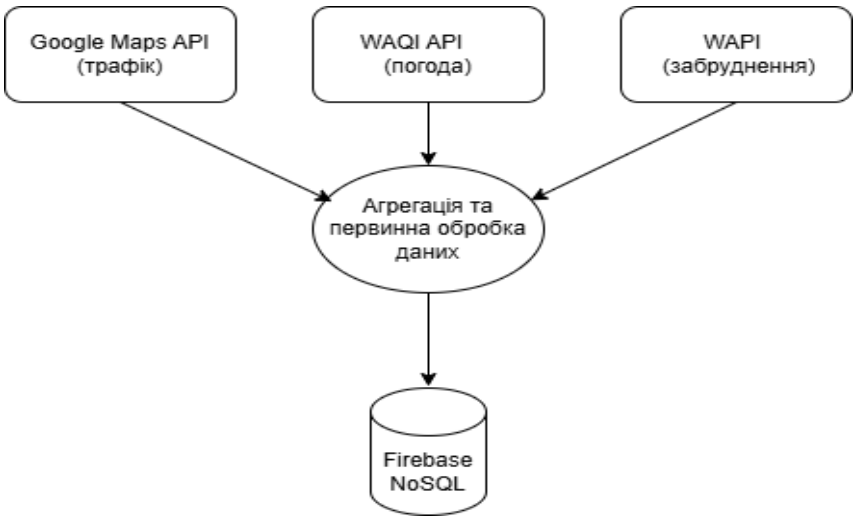

Рисунок 3.1 – Схема розробленого сервісу для збору даних

дата-сети за обраний проміжок часу. Погляньмо ще раз на схему сервісу, який використовувався для збору необхідних для моделювання даних (рисунок 3.1)

У попередньому розділі, в якому проводиться аналіз трафіку, було в деталях розказано про особливості використання Google Maps API (використання free trial, агрегація даних по 3-м маршрутам у кожно місті). Для збору ж даних щодо забруднення повітря та погодних даних, використовувалися

API інших сервісів, а саме WAQI, WAPI. Хоча WAPI передбачає отримання історичних даних через API, інший сервіс - WAQI, що використовувався для збору даних забруднення повітря, такої можливості не надає. Не надає такої можливості й сервіс, який використовувався для збору даних завантаженості доріг - Google Maps. У зв'язку із цим для максимальної узгодженості даних та переконання, що вони синхронізовані, було прийнято рішення для виконання даної роботи збирати дані у режимі реального часу із всіх 3 сервісів (рисунок 3.1).

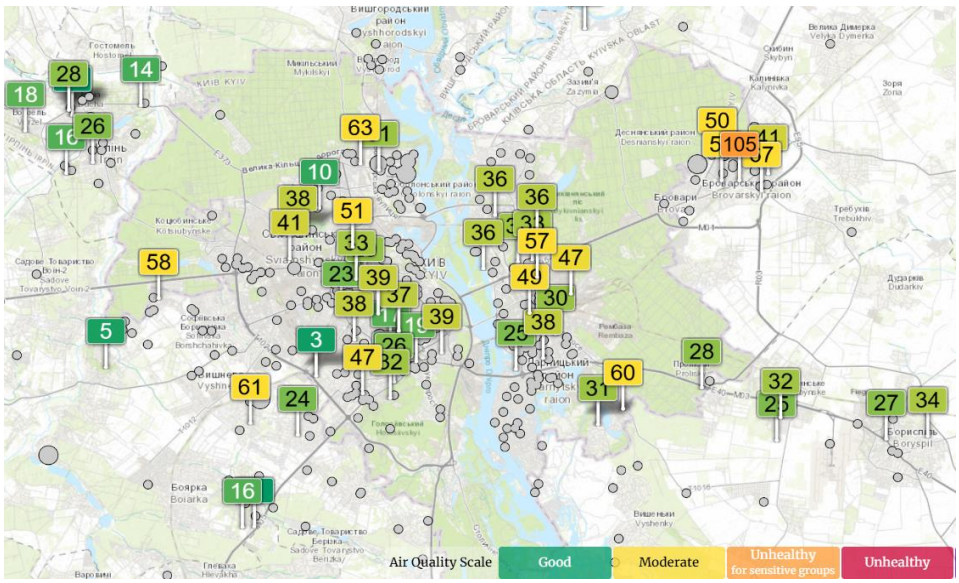

Рисунок 3.2 – Наявні станції із визначення забрудненості повітря у місті Києві, що підключені до сервісу WAQI

Важливо також наголосити на тому, як саме збиралися дані у режимі реального часу для визначення поточного рівня забрудненості повітря із сервісу WAQI. Даний ресурс пропонує у режимі реального часу дані з великої кількості станцій, підключених до системи. У одному місті може бути встановлена значна кількість станцій - від 2-3 до десятків. Рисунок 3.2 відображає це.

Як видно на рисунку 3.2, у Києві, для прикладу, є близько 20 таких станцій. Дані з них істотно відрізняються залежно від розташування до найбільш завантажених транспортом, промисловістю та густонаселених районах міста. Важливо зазначити, як саме відбувався відбір станцій для проведення дано роботи, з яких бралися дані. Це актуально з огляду на значну кількість наявних станцій та велику варіативність їхніх показів в залежності від географічного розташування. Для даної роботи з кожного міста, для якого збиралися дані, обиратися лише 1 станція за 3 основними критеріями:

1. Вона мала б бути у межах міста приблизно між центром та околицями.
2. Не бути розташованою безпосередньо біля значних трас, що могло б спотворити дані.
3. Вона мало б містити усі компоненти індекса забрудненості повітря (AQI), про структуру якого буде сказано детальніше у наступному розділі.

Таким чином, до потенційних із значного розмаїття станцій потрапляло декілька із дуже схожими показниками і вже з них обиралася 1. Це також було зроблено з міркувань спрощеного підходу зважаючи на обмежені наявні ресурси для проведення даної роботи. Адже введення більшої кількості станцій означало б пропорційне збільшення кількості даних, які потрібно було б зберігати у хмарній базі даних Firebase, яка є досить обмеженою для бюджетного тарифного плану.

3.3 Структура індексу забрудненості повітря AQI

Слід детальніше зупинитися й розібрати структуру індексу забрудненості повітря та спосіб, у який він обраховується AQI (air quality index) — це показник, який використовується для оцінки якості повітря та впливу його забруднення на

здоров'я людини. Він стандартизує дані про концентрацію різних забруднювачів, перетворюючи їх у шкалу, зрозумілу для широкої аудиторії.

AQI розраховується на основі концентрації ключових забруднювачів у повітрі, кожен з яких може мати різний вплив на здоров'я [1]:

1. PM2.5: дрібнодисперсні частки, діаметром до 2,5 мікрометрів. Джерелами є автомобільні викиди, спалювання палива, пил. Особливість щодо впливу на здоров'я - можуть проникати глибоко в легені та навіть у кров.

2. PM10: частки діаметром до 10 мікрометрів. Основними джерелами виступає пил, пилок, зола. Мають менший вплив, ніж PM2.5, але все ще шкідливі.

3. O_3: озон, вимірюється ні рівні грунту. Його джерелами є хімічні реакції між викидами транспортних засобів та сонячним світлом. Високі рівні озону є шкідливими для дихальної системи.

4. CO_2: Діоксид сірки. Основними джерелами є спалювання вугілля, нафтопродуктів та промисловість. Негативно впливає на органи дихання, а також може викликати кислотні дощі, що несуть екологічну загрозу.

5. NO_2: Оксиди азоту. Його джерелами є автомобілі, електростанції, промисловість. Вплив на здоров'я виявляється ц й подразненні легень. Крім того, він сприяє утворенню озону.

6. CO: чадний газ. Основним джерелом є неповне спалювання палива. Головна шкода для здоров'я полягає у його здатності заміщувати кисень у крові.

Як бачимо із наведеного переліку, всього налічують 6 компонентів у AQI. Загальний рівень AQI визначається на основі концентрації цих 6 компонентів за наступною схемою:

1. Дані про концентрацію забруднювачів збираються за допомогою спеціальних сенсорів, розташованих на станціях моніторингу.

2. Для кожного забруднювача розраховується субіндекс, який відповідає рівню його концентрації. Для його обрахунку використовуються

формули або таблиці, що встановлюють співвідношення між концентрацією та шкалою AQI.

3. Далі загальний AQI дорівнює максимальному значенню субіндексу серед усіх забруднювачів. Тобто, найшкідливіший компонент визначає кінцевий показник AQI.

Таким чином, формула для обрахунку AQI має вигляд [3]:

$$\text{AQI} = \max(\text{index PM2.5, index PM10, index } O_3, \text{ index } CO_2, \text{ index } NO_2, \text{ index CO}) \quad (3.1)$$

Важливо зазначити ту обставину, що таблиці концентрації для обрахунку індексу якості повітря (AQI) можуть відрізнятися в різних країнах і це зумовлено відмінностями у національних екологічних стандартах, підходах до оцінки впливу забруднювачів на здоров'я та місцевих умовах. Для прикладу, США використовують стандарти, затверджені EPA (Environmental Protection Agency). Європейський Союз керується Директивою про якість повітря. Китай має свої таблиці, які допускають вищі концентрації забруднювачів, ніж США чи ЄС.

Різні країни можуть акцентувати увагу на різних забруднювачах. Наприклад, у США більша увага приділяється PM2.5, тоді як у країнах із менш розвиненою індустрією може більше фокусуватися на SO_2 або CO[3].

Тому для того, щоб мати єдині покази індексу AQI, дані збиралися лише з одного ресурсу - WAQI, що використовує власну уніфіковану систему таблиць та підхід до розрахунку, щоб стандартизувати оцінку якості повітря по всьому світу. Це дозволяє користувачам отримувати порівняльну інформацію про забруднення повітря в різних країнах, незалежно від локальних стандартів [11].

Таблиця 3.1 – Приклад порівняння PM2.5 (мкг/м³)

Рівень AQI	**США (EPA)**	**Індія (NAQI)**	**Китай**	**ЄС**
Добре	0-12	0-30	0-35	0-10
Задовільно	12.1-35.4	31-60	36-75	10-20
Помірно	35.5-55.4	61-90	76-115	20-25
Шкідливо	55.5-150.4	91-250	116-250	25-50
Дуже шкідливо	150.5-250.4	251-350	251-350	50-75
Небезпечно	>250.4	>350	>350	>75

Таблиця 3.1 якраз показує розбіжності у існуючих стандартах, що прийняті за основу для обрахунку ступеня забруднення повітря. Тут лише наведені дані для частинок PM2.5. Але вже навіть з наведених даних помітна тенденція більш лояльного ставлення до рівня забруднення у країнах Азії, що активно розвиваються (Китай, Індія), менше у США і найбільш вимогливі норми у ЄС [11].

Таблиця 3.2 – Шкала AQI

Рівень	**Значення AQI**	**Опис**	**Вплив на здоров'я**
1	2	3	4
Добре	0-50	Чисте повітря	Безпечне для всіх
Задовільно	51–100	Прийнятна якість повітря	Незначний вплив на чутливі групи

1	2	3	4
Помірно	101–150	Ризик для чутливих груп	Людям із захворюваннями дихальної системи варто бути обережними
Шкідливо	151–200	Небезпечне для частини населення	Можливий вплив на здоров’я навіть у здорових людей
Дуже шкідливо	201–300	Серйозна загроза здоров’ю	Впливає на всіх
Небезпечно	301+	Критичний рівень	Загроза життю, слід уникати будь-якої активності на відкритому повітрі

Таблиця 3.2 описує загальну шкалу AQI із впливом на здоров'я кожного рівня забруднення. Виходячи з того, що для деяких міст цей показник регулярно перевищує 100, й нерідко навіть 200 пунктів, стає зрозуміло, настільки важливим для не лише комфортного, але й просто безпечного тривалого проживання людей є якість повітря.

3.4 Візуальний та кореляційний попередні аналізи отриманих даних

Перед власне побудовою моделей, корисно поглянути на зібрані дані та провести первинний кореляційних аналіз між основними показниками, щоб виявити певні закономірності та взяти це до уваги при моделюванні. Нижче наведено отримані дані за 8 дні для 3 міст: Берлін (рисунок 3.3), Афіни (рисунок 3.4) та Пекін (рисунок 3.5).

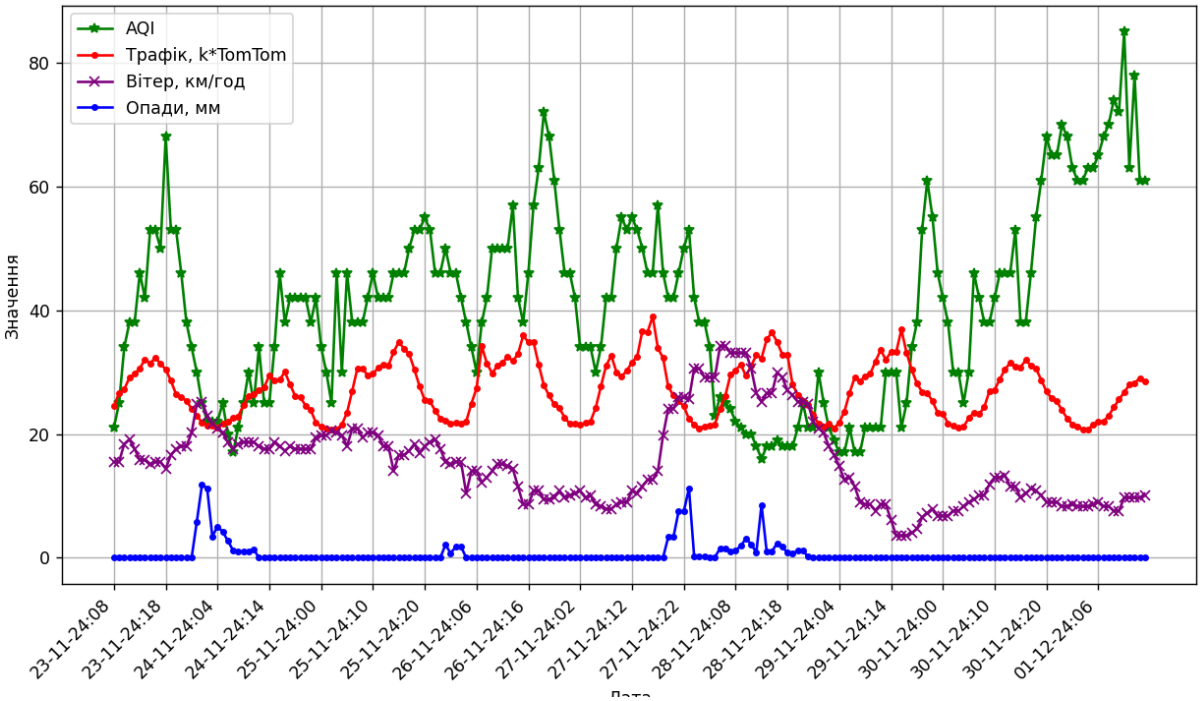

Рисунок 3.3 – Зміни AQI, трафіку та деяких погодних чинників у місті Берлін за 8 днів

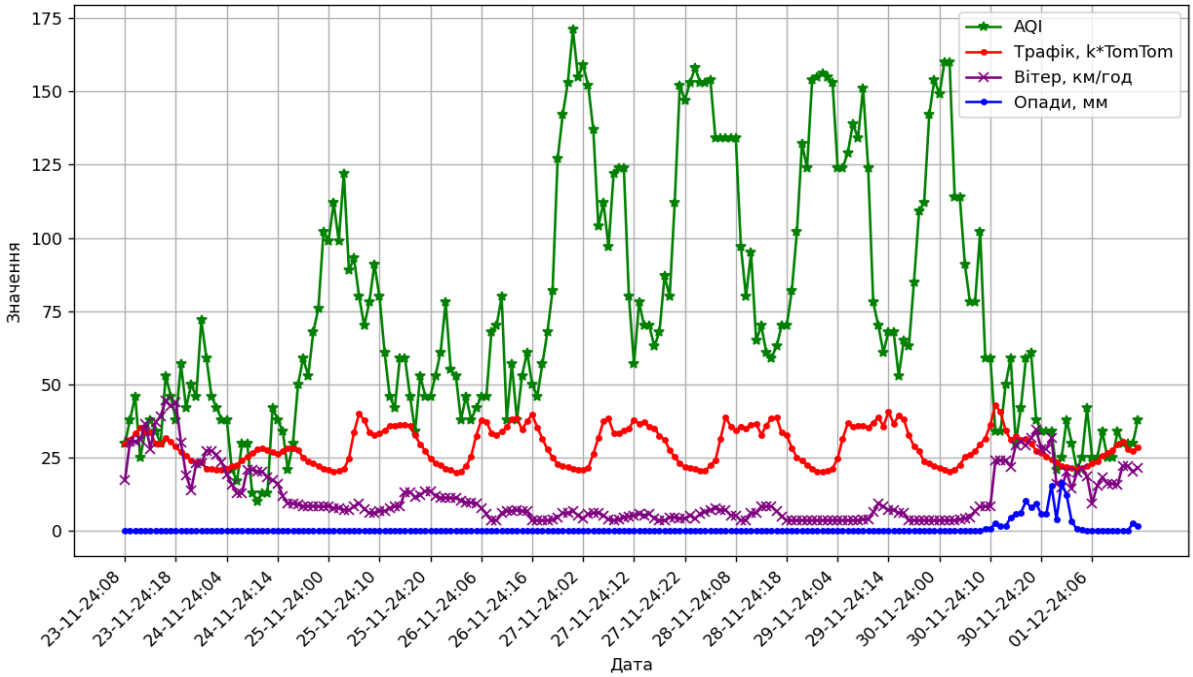

Рисунок 3.4 – Зміни AQI, трафіку та деяких погодних чинників у місті Афіни за 8 днів

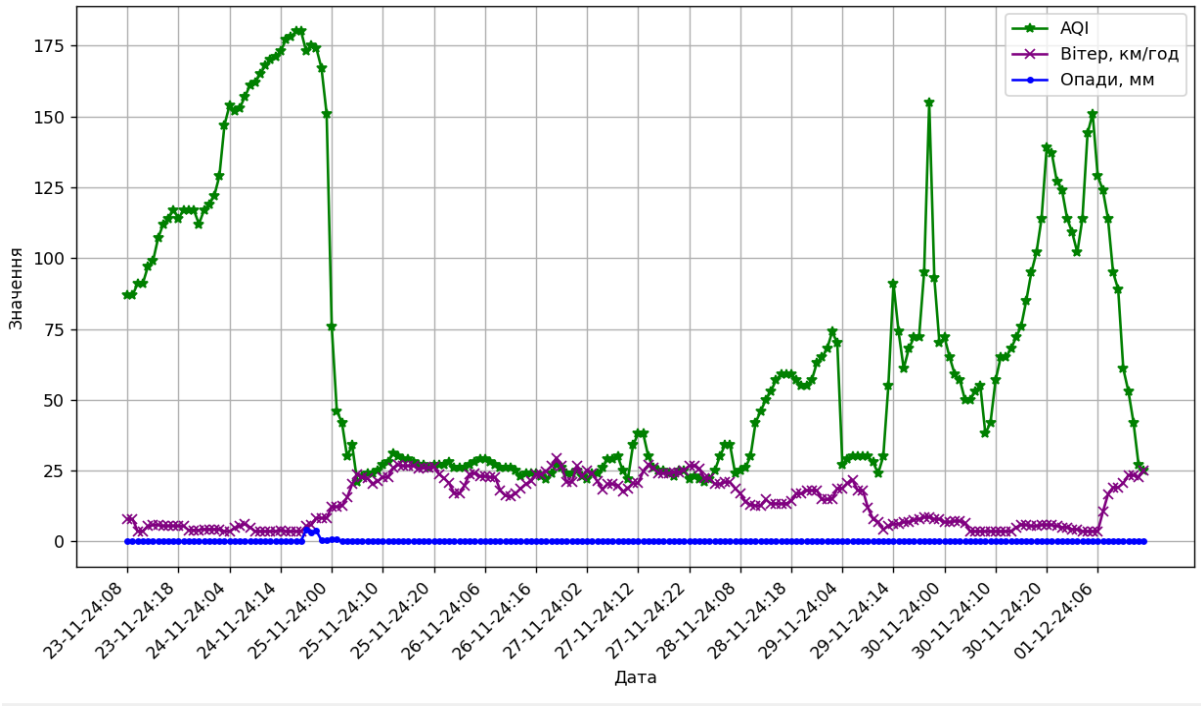

Рисунок 3.5 – Зміни AQI та деяких погодних чинників у місті Пекін за 8 днів

Висновки, які можна зробити з візуального аналізу наведених вище графіків:

1. Індекс забруднення повітря AQI виразно корелює із швидкістю вітру. Навіть за інтенсивного рівня трафіку та сильного вітру AQI досягає малих значень

2. Помітна чітка кореляція між AQI та рівнем трафіку. На зображених графіках рівень трафіку є TomTom індексом помноженим на деякий сталий коефіцієнт для приведення розмірності величин до співмірних значень. Вищий рівень трафіку істотно підвищує AQI, але із значним запізненням (лагом). Для Берліну (Рис. 3.4.4) значення лагу становить близько 3-4 годин - це часова різниця максимумів рівня трафіку та AQI. Для Афін ці значення досягають 10 годин і максимуми AQI припадають на мінімуми трафіку.

3. Висока швидкість вітру зазвичай означає негоду та опади. Відтак, для мінімумів AQI при сильному вітрові помітний також відчутно вищий рівень опадів. Тобто цілком вірогідним є припущення, що опади також корелюють із AQI, але з протилежним знаком.

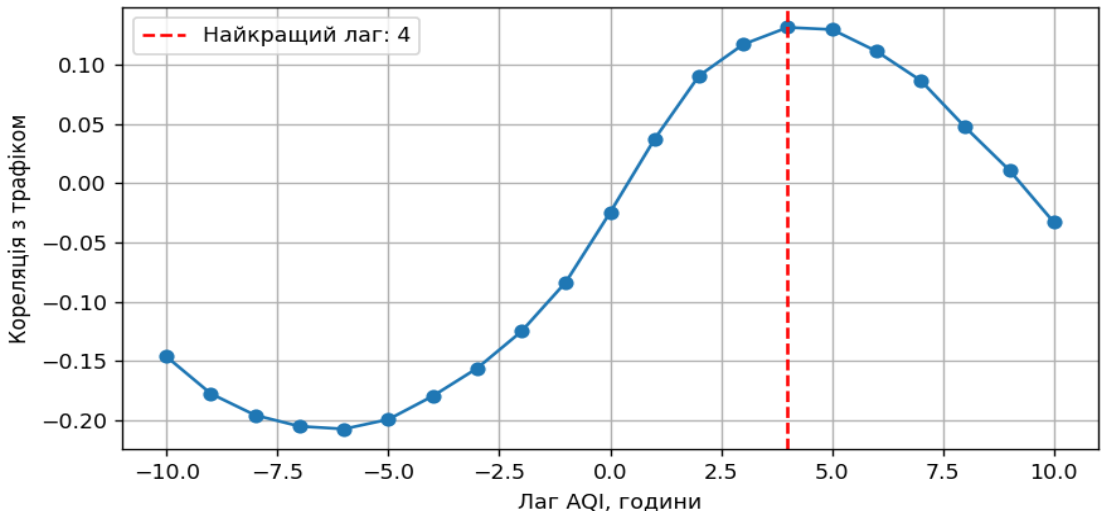

Рисунок 3.6 – Кореляція AQI з трафіком на усих зібраних даних за 3 тижні, Берлін

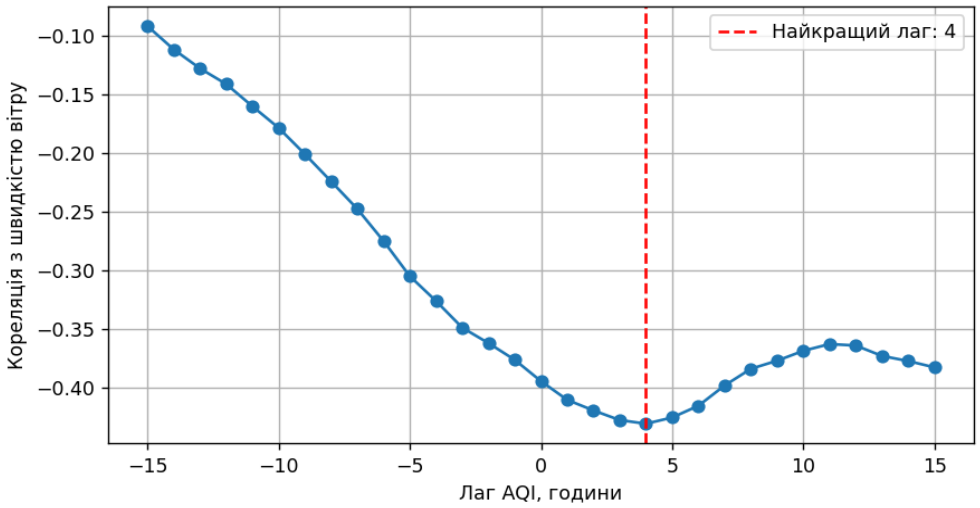

Рисунок 3.7 - Кореляція AQI з швидкістю вітру на зібраних даних за 3 тижні, Берлін

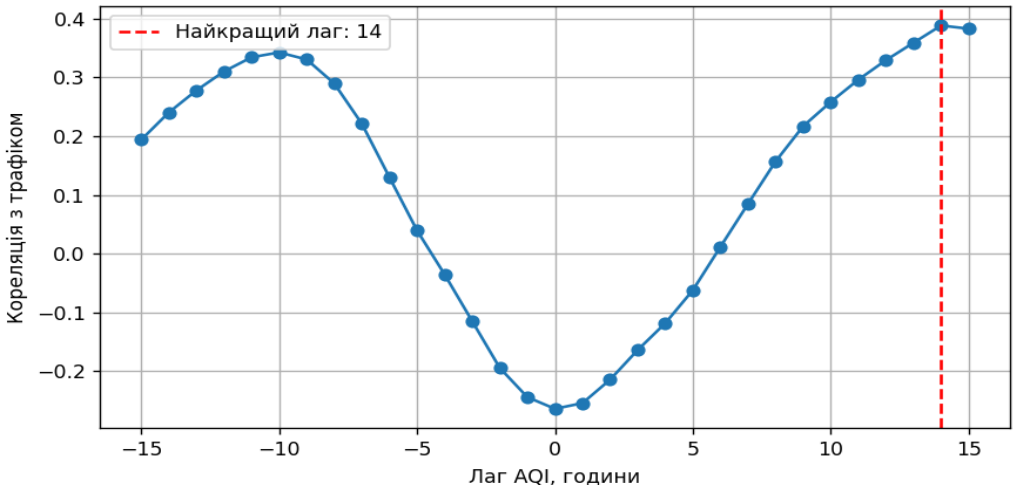

Рисунок 3.8 – Кореляція AQI з трафіком на зібраних даних за 3 тижні, Афіни

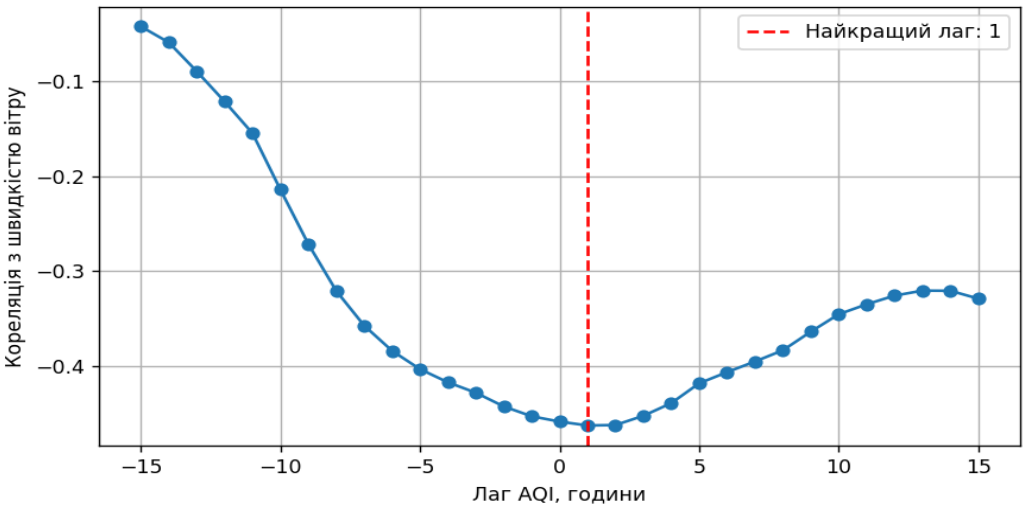

Рисунок 3.9 – Кореляція AQI з швидкістю вітру на зібраних даних за 3 тижні, Афіни

Наведені графіки кореляції між AQI та трафіком, швидкістю вітру для Берліну (рисунок 3.3, 3.7) та Афін (рисунок 3.4, 3.9) підтверджують висновки, зроблені на основі візуального аналізу (рисунок 3.3, 3.4, 3.5) і свідчать про тісний зв'язок між AQI та рівнем трафіку й швидкістю вітру. Важливо звернути увагу на те, що максимальний коефіцієнт кореляції досягається при лагах у декілька годин (Рис. 3.4.6 - 14 годин).

Таким чином, для якісного моделювання потрібно ввести в модель лаги принаймні для трафіку та швидкості вітру, які можуть сягати 10 та більше годин. Це, як буде показано в наступному розділі, призводить до збільшення кількості параметрів моделі до декількох десятків або навіть сотень параметрів.

3.5 Підготовка моделювання AQI.

Основною задачею, що має на меті дана основна частина роботи, є створення моделей для обрахунку AQI та порівняння отриманих моделей для різних міст із виявленням загальних спільних рис та особливостей. Як вже було обґрунтовано у розділі 3.1, за основу для моделювання було обрано алгоритми МГУА завдяки їхній здатності знаходити точний аналітичний вигляд моделі, відсіювати неактуальні параметри моделі та успішно оперувати з обмеженим набором даних.

Як вхідні параметри моделі використовувалися погодні дані та дані трафіку, отримані з допомогою власного сервісу, який щогодини проводив опитування через відкритий API 3 сервісів для збору необхідних даних із наступним збереженням їх до хмарної NoSQL бази даних Firebase: Google Maps, WAPI, WAQI. Вихідним параметром моделі є загальний AQI, отриманий з допомогою сервісу WAQI. Ось список зазначених параметрів моделі.

Вхідні:

1. TomTom індекс трафіку.
2. Швидкість вітру.
3. Рівень опадів.
4. Вологість.
5. Атмосферний тиск.
6. Температура.
7. Точка роси.

8. Хмарність.

Вихідний:

1. загальний AQI

Вхідні параметри мають різну розмірність (вологість - від 0 до 100), опади - міліметри та подібне, тому була проведена нормалізація типу min-max scale. Крім того, як помітно на рисунках 3.3, 3.4, AQI інколи зазнає істотних стрибків, що очевидно пов'язано із похибками вимірювання чи з іншими нерегулярностями. Для мінімізації їх впливу була проведена агрегація вхідних параметрів з інтервалом у 3 години. Як показало подальше моделювання, даний показник забезпечив найкраще значення точностей моделей при збережені інформативності.

Увесь дата-сет розбивався на навчальну та тестову вибірки у пропорції 80-20% із випадковим зміщенням. Моделювання проводилося із кількома випадковими зміщеннями і за результатами обиралася модель, яка забезпечувала максимальну точність на навчальних та тестувальних даних.

Як було показано раніше у попередньому розділі 3.4, модель повинна враховувати не лише ці вхідні 8 аргументів, але також і лаги, принаймні для трафіку та швидкості вітру. Тобто, потрібно для кожного параметру додати ще відповідні лаги, кількість яких в рази перевищує кількість основних параметрів. Так, якщо потрібно включити до моделі лаги з діапазоном від 1 до 15 годин для швидкості вітру, це означає додавання ще 15 параметрів - 1 лаг відповідає 1 годині. Як показав подальший аналіз, точності моделей >0.5 вдалося досягти шляхом введення лагів, що сягають 20-30 чи навіть інколи 72 годин. Це обумовлено істотними затримками процесів взаємодії метеорологічних чинників із складовими AQI, так званим відкладеним ефектом. Було зроблене припущення, що більшої точності моделі можна досягти, додавши лаги й до решти вхідних параметрів (температури, опадів, вологості…). Таким чином, вхідна кількість параметрів моделі подекуди сягала 200.

Алгоритми МГУА добре працюють з великою кількістю вхідних аргументів, відсіюючи неактуальні, які не впливають на вихідне значення. Експерименти із введенням різної кількості лагів показали, що просто їх збільшення далеко не завжди призводить до кращої точності моделі (рисунок 3.10). Висока вимірність даних ускладнює пошук адекватних залежностей між змінними, особливо якщо обсяг вибірки невеликий. МГУА може побудувати забагато базисних функцій, що збільшує ризик включення нерелевантних або навіть шкідливих функцій у модель. Для ефективного підбору вхідних параметрів для побудови оптимальної моделі було застосовано генетичний алгоритм.

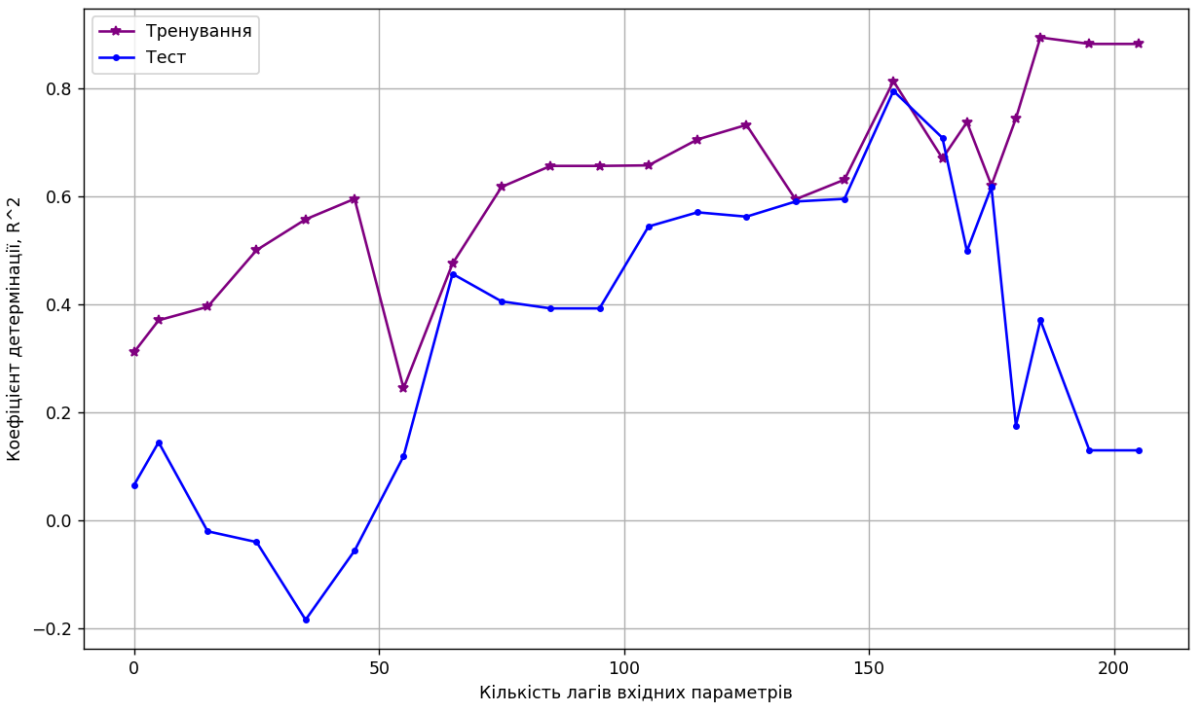

Рисунок 3.10 – Залежність коефіцієнту детермінації тренувального та тестового наборів від кількості вхідних параметрів (лагів) моделі

3.6 Введення генетичного алгоритму

Проблема врахування великої кількості вхідних параметрів включно з лагами призводила до погіршення якості моделі через введення значної кількості нерелевантних даних, що вносило додатковий шум та у підсудку призводило до гіршого сходження алгоритму, була вирішена шляхом додавання генетичного алгоритму [9, 16, 19]. Хоча це істотно подовжило час для пошуку оптимальної моделі через необхідність навчання великої кількості проміжних моделей популяції, але з іншого боку, це дозволило знайти оптимальні моделі, придатні для виконання подальшого якісного аналізу.

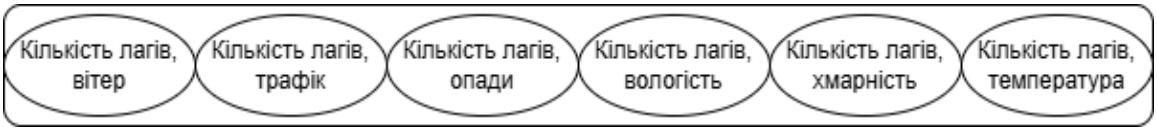

Рисунок 3.11 - Структура хромосоми для генетичного алгоритму для оптимального пошуку кількості лагів

Усі вхідні 8 параметрів обов'язково включаються до моделі. Задача генетичного алгоритму полягала у пошуку оптимальної кількості лагів для кожного з вхідних параметрів, що додавалися до початкових 8. Лаги не формувалися лише до таких параметрів як точка роси та атмосферний тиск через відсутність кореляції між цими показниками та AQI. Фітнес функція повертала значення R^2 моделі на тестовому наборі й задача алгоритма полягала у знаходженні хромосоми з таким набором лагових параметрів, щоб R^2 достигала абсолютного максимуму на допустимому проміжку значень для лагів. Обмеження на лаги встановлювалися у межах 100 годин для трафіку й швидкості вітру та по 20 годин для усіх інших менш релевантних параметрів для звуження множини потенційних рішень та пришвидшення роботи генетичного алгоритму.

Параметри генетичного алгоритму:

- розмір популяції – 30;
- фітнес-функція - R^2;
- хромосома - цілі додатні числа у заданомі діапазоні, що позначають кількість лагів, що потрібно додати до моделі для відповідних параметрів. Довжина хромосоми – 6;
- відбір турнірний, розмір 3;
- ймовірність мутації - 10%.

3.7 Результати моделювання

Всього було створено 5 моделей для 5 різних міст: Берлін, Лондон, Париж, Амстердам та Афіни. Вибір цих міст був зумовлений 2-ма ключовими обставинами:

1) Станції вимірювання AQI для цих міст містили усі компоненти, що гарантувало об’єктивну загальну оцінку.

2) Ці міста є досить різноманітними як географічно, так і за розмірами, зл дає змогу провести порівняльний аналіз та зробити висновки на основі побудованих моделей.

На рисунку 3.12, 3.13 представлені результати моделювання для міста Берлін: тренувальна та тестова вибірка із використанням 2-х алгоритмів МГУА - Ria та Mia. Ria у переважній більшості випадків показував значно кращі показники точності, тому він був узятий за основу для подальшого аналізу моделей. Коефіцієнт детермінації R^2 для моделі Берліну склав 81% і 79% для навчальної та тренувальної вибірки відповідно (алгоритм Ria). Аналітичний вигляд навчений моделі представлено на рисунку 3.13. Тут показана лише частина, повне рівняння моделі наведено у Додатку А.

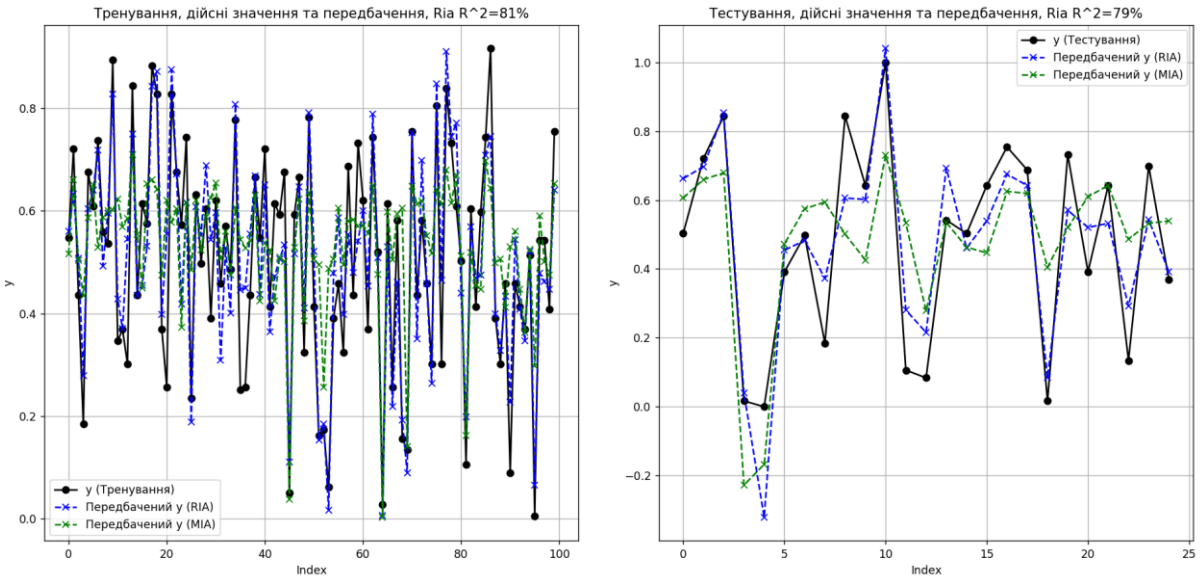

Рисунок 3.12 – результати моделювання AQI для
міста Берлін, 164 лаги

```
f48 = 0.2442*x40 + 1.0699*f47 - 0.1687*x40*f47 - 0.1717*x40^2 - 0.0125*f47^2 - 0.0614
f49 = 0.0022*x40 + 1.008*f48 - 0.0031*x40*f48 - 0.0009*x40^2 - 0.0065*f48^2 - 0.0023
f50 = - 0.0003*x40 + 1.0001*f49 + 0.0003*x40*f49 + 0.0002*x40^2 - 0.0002*f49^2 + 3.13133e-05
f51 = - 4.07326e-05*x40 + f50 + 4.49598e-05*x40*f50 + 2.2151e-05*x40^2 - 1.04075e-05*f50^2 + 7.184e-06
f52 = - 4.4727e-06*x40 + f51 + 4.94916e-06*x40*f51 + 2.42671e-06*x40^2 - 1.00421e-06*f51^2 + 8.25945e-07
f53 = - 4.80368e-07*x40 + f52 + 5.31719e-07*x40*f52 + 2.60547e-07*x40^2 - 1.0584e-07*f52^2 + 8.92451e-08
f54 = - 5.14353e-08*x40 + f53 + 5.69363e-08*x40*f53 + 2.78968e-08*x40^2 - 1.1303e-08*f53^2 + 9.56389e-09
f55 = - 5.50511e-09*x40 + f54 + 6.09392e-09*x40*f54 + 2.98577e-09*x40^2 - 1.20931e-09*f54^2 + 1.02374e-09
f56 = - 5.89176e-1*x40 + f55 + 6.52193e-1*x40*f55 + 3.19547e-1*x40^2 - 1.29419e-1*f55^2 + 1.09566e-1
f57 = - 6.30554e-11*x40 + f56 + 6.97994e-11*x40*f56 + 3.4199e-11*x40^2 - 1.38506e-11*f56^2 + 1.17261e-11
f58 = - 6.74947e-12*x40 + f57 + 7.47121e-12*x40*f57 + 3.66076e-12*x40^2 - 1.48226e-12*f57^2 + 1.25532e-12
f59 = - 7.20969e-13*x40 + f58 + 7.97919e-13*x40*f58 + 3.91081e-13*x40^2 - 1.58087e-13*f58^2 + 1.33754e-13
f60 = - 7.62517e-14*x40 + f59 + 8.48116e-14*x40*f59 + 4.12463e-14*x40^2 - 1.79382e-14*f59^2 + 1.42525e-14
y = - 8.14385e-15*x40 + f60 + 9.09695e-15*x40*f60 + 4.48639e-15*x40^2 - 8.13041e-16*f60^2 + 1.53701e-15
```

Рисунок 3.13 - аналітичний вигляд частини МГУА
моделі для міста Берлін

Повна модель Берліна містить близько 20 вхідних значущих параметрів - як основних параметрів трафіку, швидкості вітру, опадів тощо, так і їх лагів, які вносять не менший, а подеколи навіть більший внесок, ніж основні параметри, вказуючи на значний ефект саме відкладеного впливу.

Аналітична форма моделей (приклад рисунок 3.13) була використана для знаходження відсоткового внеску кожної групи параметрів у AQI, щоб показати, наскільки сильною є значущість того чи іншого параметру в залежності від міста.

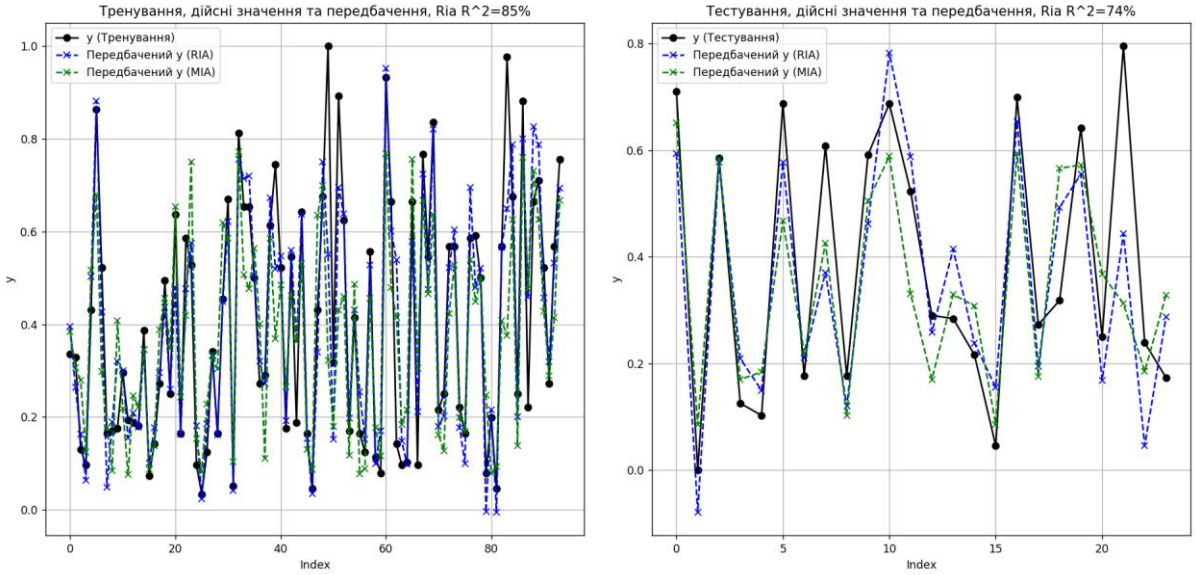

Рисунок 3.14 – результати моделювання AQI для міста Лондон на всіх наявних даних (20 днів), 202 лаги

Обрахунок проводився чисельним методом центральної різниці. Знаходилася часткова похідна для кожного із значущого параметру моделі на основі наявного аналітичного рівняння для кожної точки у дата-сеті й потім проводилося підсумовування по усим точках для певної змінної. Провівши цю операцію для всіх значущих змінних моделі, можна визначити відсотковий внесок кожної з них у фінальний показник AQI. Результати аналізу моделей представлені нижче.

З наведених результатів (рисунок 3.15) можна зробити висновки, що вирішальним чинником на AQI для Берліну є саме швидкість вітру. На це вказували й дані кореляційного аналізу дані у попередньому розділі. Від'ємний знак вказує на протилежний ефект - тобто при збільшенні швидкості вітру показник AQI зменшується.

Показовим є також від'ємний знак для опадів та температури. Опади мають “очисний ефект” і їх достатня кількість істотно впливає на якість повітря в місті. Транспорт очікувано посідає одну з ключових позицій та у впливі на рівень AQI. Щодо решти параметрів - температури, хмарності та вологості, вони мають деякий вплив у кілька відсотків, але відносно слабкий і не настільки виражений, ніж попередні чинники й напевно обумовлюються значною мірою конвекційними потоками, взаємодією сонячного світла та вологи із забруднювачами.

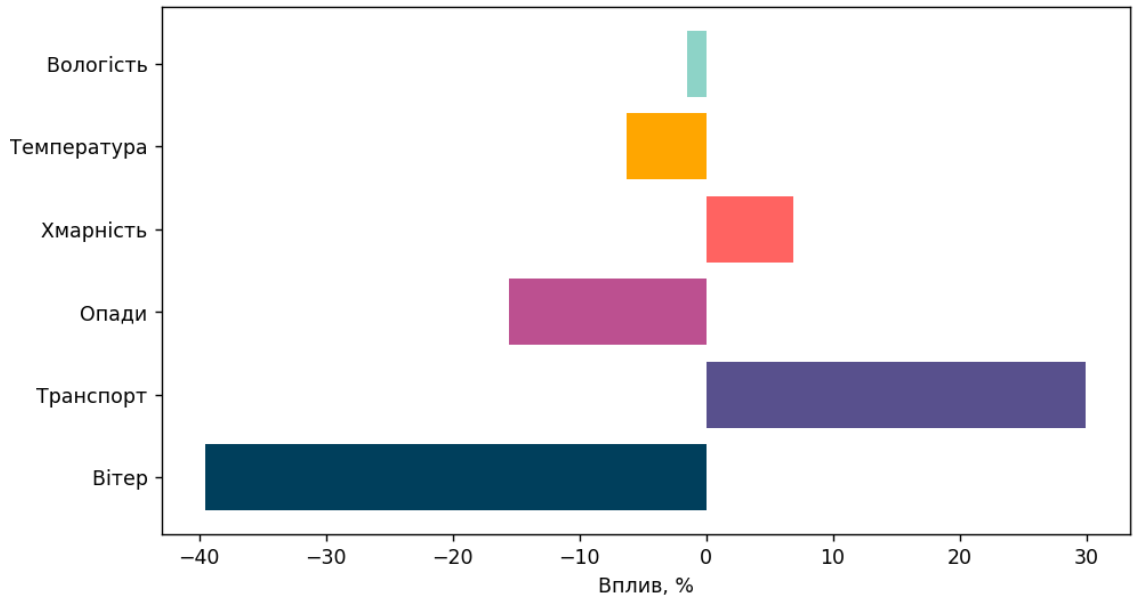

Рисунок 3.15 – Відсотковий вплив груп вхідних параметрів моделі разом з їх лагами на AQI, Берлін

Дані для деяких інших міст наводяться нижче:

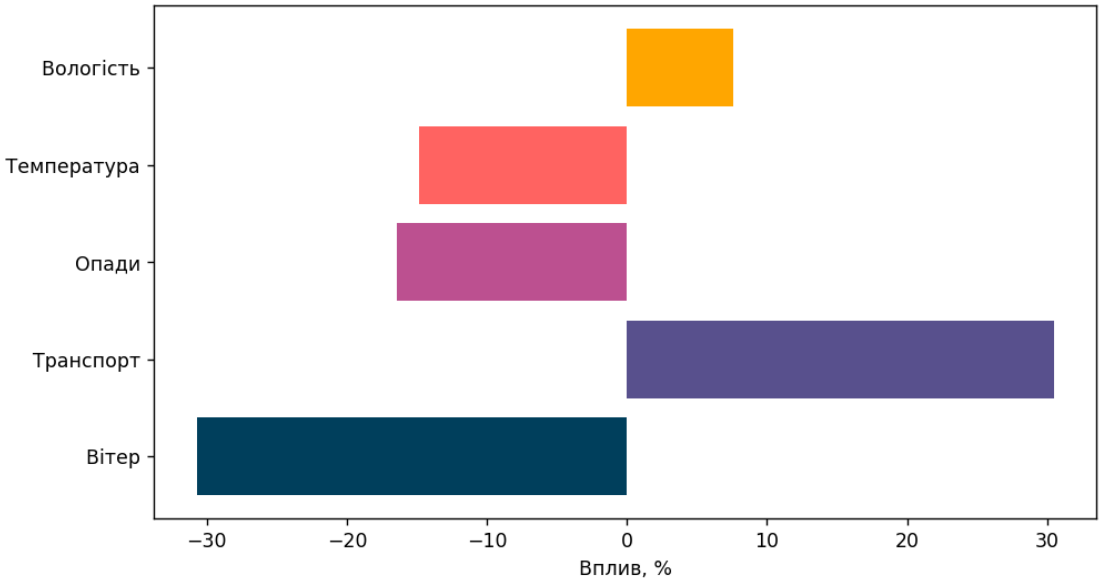

Рисунок 3.16 – Відсотковий вплив груп вхідних параметрів моделі разом з їх лагами на AQI, Париж

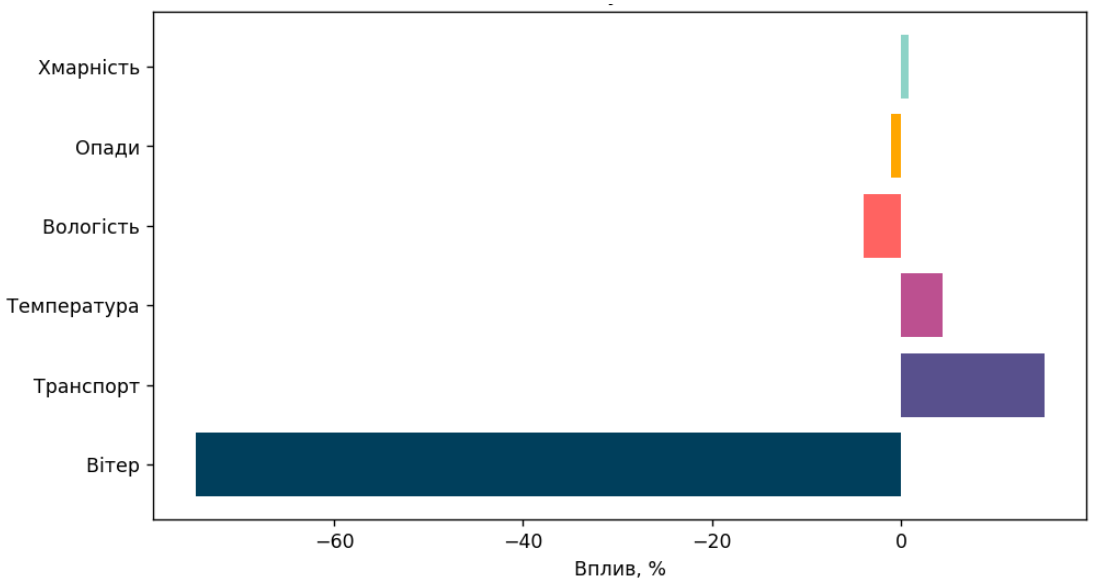

Рисунок 3.17 – Відсотковий вплив груп вхідних параметрів моделі разом з їх лагами на AQI, Лондон

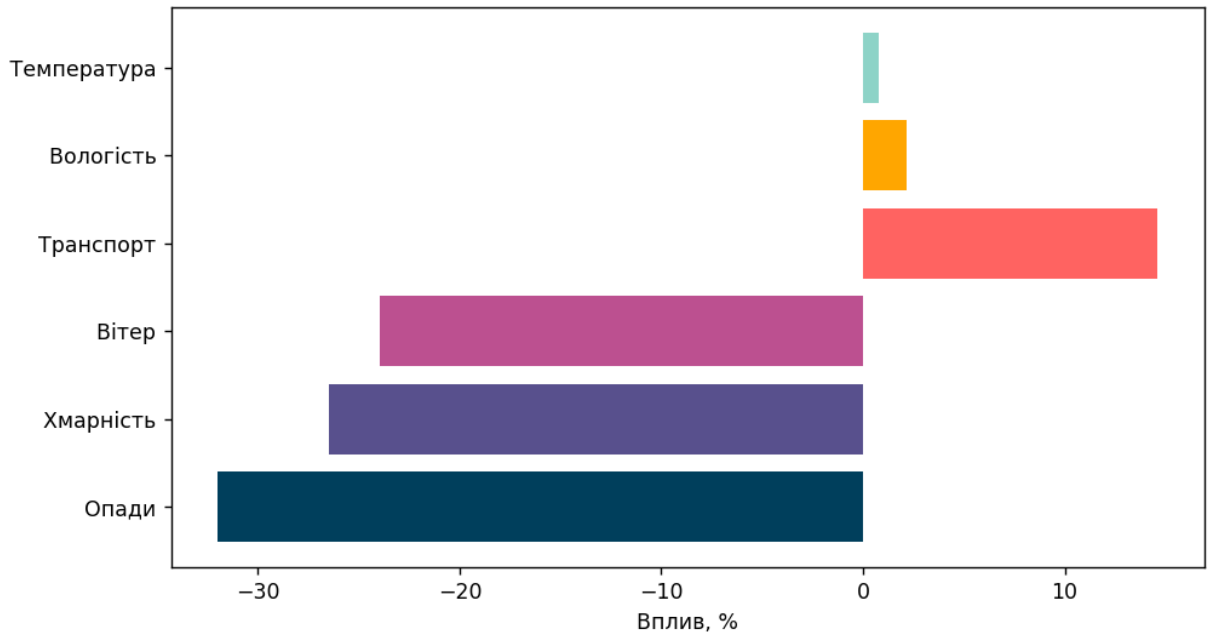

Рисунок 3.18 – Відсотковий вплив груп вхідних параметрів моделі разом з їх лагами на AQI, Амстердам

Загальні висновки, які можна зробити із наведених даних, обчислених на основі відповідних моделей, є настіпні:

1. вирішальним чинником, що впливає на AQI є швидкість вітру. Більша швидкість вітру призводить сумарно до зменшення рівня AQI, хоча у структурі лагів для вітру є складові із позитивним знаком, що може вказувати на підвищення пилової складової при сильному вітрові у перші години.

2. Рівень трафіку є також ключовим із впливу на AQI. Цей вплив істотно залежить від розміру міста. Як видно на рисунку 3.17, у Лондоні, для прикладу, очевидно через значні розміри міста та, як наслідок, значним часом необхідним для розсіювання, вплив варіацій трафіка на AQI менше 20%. Деякі лаги моделі мають від'ємний внесок у AQI, що може вказувати на взаємодію автомобільних викидів із іншими забруднювачами, а також присутність інших чинників, які не враховувала модель.

3. Рівень опадів належить до основних поряд із швидкістю вітру та трафіком. Хоча у перші години рівень забруднення повітря може зрости через

опади, що може бути пояснене здійманням пилових частинок, сукупний внесок лагів показника опадів все ж має від'ємне значення, що свідчить про “очисний” характер опадів, особливо у значній кількості (рисунок 3.18, Амстердам).

4. Очевидними є зв’язки між деякими чинниками, наприклад хмарність та рівень опадів (рисунок 3.18, Амстердам). Тому для отримання більш точних результатів моделювання потрібно збір даних та побудова моделей на їх основі за значно більший проміжок часу, протягом якого було б більше різноманітних метеоумов, які модель змогла б розрізнити.

ВИСНОВКИ

У даній дипломній роботі була проведена робота з аналізу та кластеризації близько 25 різних міст світу за агрегованим рівнем трафіку TomTom індексом. Було запропоновано та обґрунтовано використання агрегованого квадратичного TomTom індексу для порівняльного аналізу даних міст, проведена кластеризація за визначеним рівнем та зроблені висновки.

Окрім цього, іншою частиною роботи було дослідження впливу як метеорологічних чинників: швидкості вітру, рівня опадів, температури тощо, так і рівня трафіку на загальний рівень забрудненості AQI шляхом побудови моделей для кількох міст різного розміру та розташованих у різних кліматичних зонах з використанням алгоритмів МГУА та генетичних алгоритмів. Шляхом використання чисельних методів був обчислений відсотковий вплив кожного із чинників на результуюче значення QAI. Зроблені висновки про вплив кожного із чинників.

Усі проведені розрахунки були виконані на основі дата-сетів, отриманих з допомогою написаного власними силами мікросервісу мовою Javascript із наступною його контейнеризацією у Docker та розгортанням на віртуальному сервері, де він працював протягом близько 20 днів. Даний сервіс використовував відкритий API 3 сервісів - Google Maps, WQAI, WAPI - для отримання даних у режимі реального часу із збереженням отриманих даних у хмарній NoSQL базі даних Firebase.

Вихідний код мікросервісу, а також скрипти Python, що використовувалися для обробки даних, побудові моделей та візуалізації отриманих результатів, знаходиться у відкритому репозиторії, що наведено у додатку.

Використання алгоритмів МГУА побудови моделей на основі метеорологічних даних навіть за умов обмеженої кількості даних показало свою

високу ефективність. Точність моделей на текстовому наборі даних перевищувала 70%.

Також було підтверджено, що при значній кількості вхідних параметрів - десятки, сотні, істотно покращити показники моделі можна за допомогою використання генетичного алгоритму.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бузанич В. В. Математическое моделирование экологических процессов. М.: Издательство МГТУ, 2012. 352 с.

2. Корякин В. А., Петров С. П. Методы машинного обучения в задачах анализа больших данных. М.: Издательство МГУ, 2016. 280 с.

3. Мостовой Д. П., Мартынюк А. Г. Прогнозирование загрязнения атмосферного воздуха на основе методов машинного обучения. // Экология и безопасность, 2020. №2. С. 11–19.

4. Судаков В. М. Методы машинного обучения для экологического мониторинга. СПб.: Политехника, 2019. 318 с.

Фурсов А. В., Коваленко Ю. П. Прогнозирование временных рядов с помощью методов машинного обучения. // Вестник информационных технологий, 2017. №3. С. 41–48

Babovic V. Modeling of Environmental Systems: Advances and Challenges. // Environmental Modelling & Software, 2005. Vol. 20, No. 8. P. 1163–1172.

5. Bishop C. M. Pattern Recognition and Machine Learning. Springer, 2006. 738 p.

6. Carslaw D. C. The Openair Manual — Open-Source Tools for Analysing Air Pollution Data. Manual for Tools in R, 2017.

7. Coello Coello C. A., Lamont G. B., Van Veldhuizen D. A. Evolutionary Algorithms for Solving Multi-Objective Problems. Springer, 2007.

8. Deb K. Multi-Objective Optimization using Evolutionary Algorithms. Wiley, 2001. 518 p.

9. Fayyad U., Piatetsky-Shapiro G., Smyth P. From Data Mining to Knowledge Discovery in Databases. // AI Magazine, 1996. Vol. 17, No. 3. P. 37–54.

10. Goldberg D. E. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, 1989.

11. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed. Springer, 2009. 745 p.

12. Helbing D. Traffic and related self-driven many-particle systems. Review of Modern Physics, 2001.

13. Ivakhnenko A. G. The Group Method of Data Handling: A Rival of the Method of Stochastic Approximation. // Soviet Automatic Control, 1968. Vol. 13, No. 3. P. 43–56.

14. Mitchell M. An Introduction to Genetic Algorithms. MIT Press, 1998.

15. National Institute of Strategic Studies. URL: https://www.gmdh.net

16. Seinfeld J. H., Pandis S. N. Atmospheric Chemistry and Physics: From Air Pollution to Climate Change. John Wiley & Sons, 2016.

17. Sivanandam S. N., Deepa S. N. Introduction to Genetic Algorithms. Springer-Verlag, 2008.

18. Zhang Y. Machine Learning Approaches for Air Quality Prediction. Journal of Environmental Management, 2017. Vol. 193. P. 56–68.

ДОДАТОК А
АНАЛІТИЧНА МОДЕЛЬ МГУА ДЛЯ МІСТА БЕРЛІН

```
f1 = 1.0796*x14 - 0.689*x47 - 0.0693*x14*x47 - 1.7637*x14^2 + 0.4689*x47^2 + 0.6009

f2 = 0.2726*x81 + 0.8473*f1 - 0.0622*x81*f1 - 0.4758*x81^2 + 0.1819*f1^2 + 0.0297

f3 = 1.2547*x23 + 1.7277*f2 - 1.5284*x23*f2 - 0.783*x23^2 - 0.3242*f2^2 - 0.3437

f4 = - 0.1104*x16 - 0.4094*f3 + 0.7866*x16*f3 - 0.3746*x16^2 + 1.105*f3^2 + 0.3757

f5 = - 0.1547*x160 + 1.2393*f4 - 0.5061*x160*f4 + 0.4004*x160^2 + 0.1565*f4^2 - 0.1262

f6 = - 0.0983*x35 + 0.6609*f5 + 0.402*x35*f5 - 0.0527*x35^2 + 0.2675*f5^2 + 0.0723

f7 = - 0.4254*x50 + 0.979*f6 + 0.2583*x50*f6 + 0.2353*x50^2 - 0.0819*f6^2 + 0.1006

f8 = - 0.7113*x23 + 0.2145*f7 + 1.1425*x23*f7 + 0.2879*x23^2 + 0.4883*f7^2 + 0.2768

f9 = 0.0646*x143 + 0.7894*f8 + 0.3777*x143*f8 - 0.2607*x143^2 - 0.0367*f8^2 + 0.0718

f10 = - 0.1376*x68 + 0.656*f9 + 0.5449*x68*f9 + 0.0536*x68^2 + 0.137*f9^2 + 0.082

f11 = - 0.3559*x89 + 0.9285*f10 + 0.1814*x89*f10 + 0.2538*x89^2 - 0.0147*f10^2 + 0.0885

f12 = 0.1623*x160 + 0.9452*f11 + 0.0704*x160*f11 - 0.1917*x160^2 + 0.0567*f11^2 - 0.0122

f13 = 0.4639*x36 + 0.9327*f12 - 0.1354*x36*f12 - 0.3857*x36^2 + 0.1368*f12^2 - 0.0818

f14 = - 0.1013*x50 + 0.9794*f13 + 0.1398*x50*f13 + 0.0146*x50^2 - 0.0266*f13^2 + 0.0276

f15 = 0.0653*x82 + 1.0446*f14 - 0.0202*x82*f14 - 0.1061*x82^2 - 0.0569*f14^2 - 0.0033

f16 = - 0.0651*x114 + 0.9115*f15 + 0.0593*x114*f15 - 0.0423*x114^2 + 0.0476*f15^2 + 0.0531

f17 = 0.1474*x154 + 1.1322*f16 - 1.0003*x154*f16 + 0.2596*x154^2 - 0.1185*f16^2 - 0.0193

f18 = 0.0156*x17 + 1.0012*f17 + 0.2791*x17*f17 - 0.012*x17^2 - 0.0572*f17^2 - 0.0285

f19 = - 0.5581*x12 + 1.0893*f18 + 0.318*x12*f18 + 0.4525*x12^2 - 0.1202*f18^2 + 0.0526

f20 = 0.2789*x17 + 1.3603*f19 - 0.3225*x17*f19 - 0.0679*x17^2 - 0.2231*f19^2 - 0.1495
```

f21 = - 0.0914*x68 + 1.0283*f20 + 0.1423*x68*f20 + 0.046*x68^2 - 0.0999*f20^2 + 0.0124

f22 = - 0.2944*x50 + 1.0374*f21 + 0.0945*x50*f21 + 0.1885*x50^2 - 0.083*f21^2 + 0.0614

f23 = - 0.0654*x67 + 0.9948*f22 - 0.0996*x67*f22 + 0.1683*x67^2 + 0.0163*f22^2 + 0.0121

f24 = 0.0698*x160 + 0.943*f23 + 0.015*x160*f23 - 0.0822*x160^2 + 0.0516*f23^2 + 0.0088

f25 = 0.3682*x43 + 1.0828*f24 - 0.2617*x43*f24 - 0.1637*x43^2 + 0.0372*f24^2 - 0.1111

f26 = 0.1527*x82 + 1.1911*f25 - 0.2418*x82*f25 - 0.0392*x82^2 - 0.091*f25^2 - 0.0778

f27 = 0.1113*x156 + 1.223*f26 - 0.2033*x156*f26 + 0.0434*x156^2 - 0.0595*f26^2 - 0.129

f28 = 0.0133*x94 + 1.1399*f27 - 0.1898*x94*f27 + 0.0943*x94^2 - 0.0423*f27^2 - 0.049

f29 = 0.1981*x68 + 1.1043*f28 - 0.1832*x68*f28 - 0.0755*x68^2 - 0.0212*f28^2 - 0.0685

f30 = 0.1119*x82 + 1.1071*f29 - 0.1524*x82*f29 - 0.0383*x82^2 - 0.0434*f29^2 - 0.0493

f31 = 0.23*x48 + 0.9671*f30 - 0.118*x48*f30 - 0.1838*x48^2 + 0.06*f30^2 - 0.0302

f32 = 0.1114*x160 + 1.0077*f31 - 0.0148*x160*f31 - 0.1129*x160^2 + 0.0094*f31^2 - 0.0133

f33 = 0.1087*x141 + 1.0205*f32 - 0.032*x141*f32 - 0.0717*x141^2 + 0.0002*f32^2 - 0.0378

f34 = 0.1469*x142 + 0.9926*f33 - 0.009*x142*f33 - 0.1378*x142^2 + 0.0051*f33^2 - 0.0251

f35 = 0.0071*x130 + 1.0457*f34 + 0.0858*x130*f34 - 0.0029*x130^2 - 0.0687*f34^2 - 0.0245

f36 = - 0.0337*x82 + 1.0649*f35 - 0.0739*x82*f35 + 0.0552*x82^2 - 0.0426*f35^2 - 0.006

f37 = - 0.4275*x9 + 0.7885*f36 + 0.2995*x9*f36 + 0.2882*x9^2 + 0.1212*f36^2 + 0.1225

f38 = 0.0815*x12 + 1.361*f37 - 0.1951*x12*f37 + 0.0886*x12^2 - 0.237*f37^2 - 0.1249

f39 = - 0.5932*x55 + 0.9656*f38 + 0.1631*x55*f38 + 0.5022*x55^2 - 0.0723*f38^2 + 0.136

f40 = 0.2378*x18 + 0.8268*f39 + 0.0855*x18*f39 - 0.2664*x18^2 + 0.1254*f39^2 + 9.63716e-05

f41 = 0.0757*x61 + 1.0767*f40 - 0.0649*x61*f40 - 0.0121*x61^2 - 0.0464*f40^2 - 0.0382

f42 = 0.049*x94 + 1.0214*f41 - 0.1149*x94*f41 + 0.0261*x94^2 + 0.0266*f41^2 - 0.0215

f43 = 0.064*x37 + 0.9894*f42 - 0.0064*x37*f42 - 0.098*x37^2 + 0.0069*f42^2 - 0.0026

f44 = 0.1181*x23 + 1.018*f43 - 0.0402*x23*f43 - 0.1336*x23^2 - 0.0081*f43^2 - 0.0216

f45 = 0.0045*x61 + 1.0241*f44 - 0.0159*x61*f44 + 0.0246*x61^2 - 0.0193*f44^2 - 0.009

f46 = 0.111*x160 + 1.0196*f45 - 0.04*x160*f45 - 0.0857*x160^2 + 0.0174*f45^2 - 0.0283

f47 = - 0.0262*x157 + 1.0605*f46 - 0.0359*x157*f46 + 0.0552*x157^2 - 0.0325*f46^2 - 0.0233

f48 = 0.2442*x40 + 1.0699*f47 - 0.1687*x40*f47 - 0.1717*x40^2 - 0.0125*f47^2 - 0.0614

f49 = 0.0022*x40 + 1.008*f48 - 0.0031*x40*f48 - 0.0009*x40^2 - 0.0065*f48^2 - 0.0023

f50 = - 0.0003*x40 + 1.0001*f49 + 0.0003*x40*f49 + 0.0002*x40^2 - 0.0002*f49^2 + 3.13133e-05

f51 = - 4.07326e-05*x40 + f50 + 4.49598e-05*x40*f50 + 2.2151e-05*x40^2 - 1.04075e-05*f50^2 + 7.184e-06

f52 = - 4.4727e-06*x40 + f51 + 4.94916e-06*x40*f51 + 2.42671e-06*x40^2 - 1.00421e-06*f51^2 + 8.25945e-07

f53 = - 4.80368e-07*x40 + f52 + 5.31719e-07*x40*f52 + 2.60547e-07*x40^2 - 1.0584e-07*f52^2 + 8.92451e-08

f54 = - 5.14353e-08*x40 + f53 + 5.69363e-08*x40*f53 + 2.78968e-08*x40^2 - 1.1303e-08*f53^2 + 9.56389e-09

f55 = - 5.50511e-09*x40 + f54 + 6.09392e-09*x40*f54 + 2.98577e-09*x40^2 - 1.20931e-09*f54^2 + 1.02374e-09

f56 = - 5.89176e-1*x40 + f55 + 6.52193e-1*x40*f55 + 3.19547e-1*x40^2 - 1.29419e-1*f55^2 + 1.09566e-1

f57 = - 6.30554e-11*x40 + f56 + 6.97994e-11*x40*f56 + 3.4199e-11*x40^2 - 1.38506e-11*f56^2 + 1.17261e-11

f58 = - 6.74947e-12*x40 + f57 + 7.47121e-12*x40*f57 + 3.66076e-12*x40^2 - 1.48226e-12*f57^2 + 1.25532e-12

f59 = - 7.20969e-13*x40 + f58 + 7.97919e-13*x40*f58 + 3.91081e-13*x40^2 - 1.58087e-13*f58^2 + 1.33754e-13

```
f60 = - 7.62517e-14*x40 + f59 + 8.48116e-14*x40*f59 + 4.12463e-14*x40^2 - 1.79382e-14*f59^2 + 1.42525e-14

y = - 8.14385e-15*x40 + f60 + 9.09695e-15*x40*f60 + 4.48639e-15*x40^2 - 8.13041e-16*f60^2 + 1.53701e-15
```

ДОДАТОК Б
GITHUB РЕПОЗИТОРІЙ З ВИХІДНИМ КОДОМ

Вихідний код розробленого сервісу для збору метеорологічних даних та рівня забрудненості разом зі скриптами Python для побудови моделей, обробки та візуалізації даних