НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ

Кафедра математичних методів системного аналізу

До захисту допущено:

Завідувач кафедри
_____ Оксана ТИМОЩУК

«_____»_____2024 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системний аналіз і управління»

спеціальності 124 «Системний аналіз»

на тему: «Дослідження очікуваної тривалості життя населення за допомогою методів машинного навчання»

Виконав:
студент IV курсу, групи КА-05
Низовець Микола Віталійович _____

Керівник:
доцент, к.ф.-м.н., доцент кафедри ММСА,
Шубенкова Ірина Анатоліївна _____

Консультант з економічного розділу:
доцент кафедри ТПЕ, к.е.н., доц.
Рощина Надія Василівна _____

Консультант з нормоконтролю:
доцент кафедри ММСА, к.ф.-м.н.
Статкевич Віталій Михайлович _____

Рецензент:
професор кафедри ШІ, д.т.н., професор,
Данилов Валерій Якович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут прикладного системного аналізу

Кафедра математичних методів системного аналізу

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 124 «Системний аналіз»

Освітньо-професійна програма «Системний аналіз і управління»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Оксана ТИМОЩУК
«___»_____2024 р.

ЗАВДАННЯ

на дипломну роботу студенту

Низовцю Миколі Віталійовичу

1. Тема роботи «Дослідження очікуваної тривалості життя населення за допомогою методів машинного навчання», керівник роботи Шубенкова Ірина Анатоліївна, к.ф.-м.н., доцент, затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом роботи 11.06.2024

3. Вихідні дані до роботи: дані про країни світу із сайту Kaggle.

4. Зміст роботи: огляд предметної області, методи дослідження, огляд даних та їх попередня обробка, розробка і оцінка моделей, функціонально-вартісний аналіз.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): графіки, результати обробки даних, критерії оцінювання якості моделі.

6. Консультанти розділів роботи*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н.В., доц., к.е.н.		

* Якщо визначені консультанти. Консультантом не може бути зазначено керівника дипломної роботи.

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Вибір теми і постановка дослідження	21.04.2024	Виконано
2	Збір та аналіз літератури	21.04.2024	Виконано
3	Формулювання задачі дослідження	28.04.2024	Виконано
4	Аналіз актуальності задачі дослідження	28.04.2024	Виконано
5	Розробка програмного продукту	05.05.2024	Виконано
6	Аналіз та впорядкування результатів	12.05.2024	Виконано
7	Оформлення пояснювальної записки	19.05.2024	Виконано
8	Оформлення дипломної роботи та аналіз отриманих результатів	19.05.2024	Виконано

Студент Микола НИЗОВЕЦЬ

Керівник Ірина ШУБЕНКОВА

РЕФЕРАТ

Дипломна робота містить : 120 с., 20 табл., 29 рис., 2 дод., 28 джерел.

ОЧІКУВАНА ТРИВАЛІСТЬ ЖИТТЯ, РЕГРЕСІЙНІ МОДЕЛІ, ЛІНІЙНА РЕГРЕСІЯ, МАШИННЕ НАВЧАННЯ, PYTHON, ELASTIC NET, PRINCIPAL COMPONENT REGRESSION, RANDOM FOREST, SUPPORT VECTOR MACHINES.

Об’єктом дослідження є очікувана тривалість життя населення.

Предметом дослідження є методи машинного навчання – регресійні моделі, такі як лінійна регресія, метод опорних векторів для регресії, метод випадкового лісу, principal component regression, elastic net.

Метою дипломної роботи є розробка моделей машинного навчання, які зможуть опановувати дані щодо очікуваної тривалості життя населення, знайти фактори, які впливають на цей показник найбільше, а також прогнозувати показник.

В даній роботі проведено дослідження показника очікуваної тривалості життя та основних факторів, що впливають на нього. Для побудови дослідження були використані методи машинного навчання та аналізу даних, проведено роботу для визначення можливостей по покращенню роботи стандартних моделей та визначено кращу модель для реалізації прогнозування на основі наведених даних.

Програмна реалізація була розроблена на мові програмування Python.

ABSTRACT

Thesis contains: 120 p., 20 tables, 29 fig., 2 append., 28 references.

LIFE EXPECTANCY, REGRESSION MODELS, LINEAR REGRESSION, MACHINE LEARNING, PYTHON, ELASTIC NET, PRINCIPAL COMPONENT REGRESSION, RANDOM FOREST, SUPPORT VECTOR MACHINES.

The research topic: «Research of population life expectancy using machine learning methods»

The object of the study is the life expectancy of the population.

The subject of the research is the set of machine learning methods - regression models, such as linear regression, support vector machine for regression, random forest method, principal component regression, elastic net.

The purpose of the thesis is to develop machine learning models that can process data on life expectancy, find the factors that affect this indicator the most, and predict the indicator.

In this research, we study life expectancy and the main factors that affect it. Machine learning and data analysis methods were used to build the study, work was done to identify opportunities to improve the performance of standard models, and the best model for implementing forecasting based on the data was determined.

The program was written in Python programming language.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ООН – Організація Об’єднаних Націй.

ІЛР – Індекс Людського Розвитку.

ВВП – Валовий внутрішній продукт.

ШІ – Штучний інтелект.

МН – Машинне навчання.

RMSE – Root Mean Square Error.

MAE – Mean Absolute Error.

SVR – Support Vector Regression.

RF – Random Forest.

PCR – Principal Component Regression.

PCA – Principal Component Analysis.

МНК – Метод найменших квадратів.

ВСТУП

Дослідження очікуваної тривалості життя населення є одним з найважливіших аспектів демографічної науки і має великий соціально-економічний вплив. Очікувана тривалість життя є індикатором здоров'я нації та відображає загальний рівень медичної допомоги, якості життя та соціально-економічного розвитку країни. Вивчення цього показника дозволяє урядам і організаціям краще розуміти потреби населення, прогнозувати розвиток соціальних і економічних програм, а також розробляти ефективні стратегії для покращення якості життя.

Актуальність дослідження обумовлена стрімкими змінами в демографічній структурі багатьох країн, що спричинені старінням населення, зміною рівня смертності та народжуваності, а також впливом глобальних епідемій, таких як COVID-19. Ці фактори вимагають постійного моніторингу та аналізу для адаптації політик і стратегій охорони здоров'я.

Методи машинного навчання відкривають нові можливості для дослідження очікуваної тривалості життя завдяки їхній здатності обробляти великі обсяги даних та виявляти складні закономірності. Традиційні методи статистичного аналізу часто обмежені у своїй здатності враховувати багатофакторний вплив різноманітних чинників, таких як соціально-економічні умови, генетичні фактори, екологічні впливи та поведінкові аспекти. У свою чергу, машинне навчання дозволяє створювати більш точні та адаптивні моделі, які можуть передбачати тривалість життя з урахуванням широкого спектру даних.

В сучасному світі, де всі дані стають доступними, саме методи машинного навчання можуть відіграти визначну роль у вирішенні задач різного характеру, а також прогнозуванні параметрів. Завдяки цим методам можна врахувати всі важливі і складні взаємозв’язки між різними факторами, що впливають на очікувану тривалість життя населення.

Отже, метою даної дипломної роботи є розробка моделей машинного навчання, які зможуть опановувати дані щодо очікуваної тривалості життя населення, знайти фактори, які впливають на цей показник найбільше, а також прогнозувати показник.

У першому розділі буде проведений загальний аналіз та знайомство з досліджуваною областю. Огляд сучасних проблем, які постають перед питанням очікуваної тривалості життя, а також дослідження основних компонент задач машинного навчання.

В другому розділі будуть детально розглядані теоретичні відомості щодо методів, які будуть застосовуватися, метрик, а також інструментів, які будемо використовувати.

В третьому розділі буде огляд безпосередньо практичної частини дослідження, а саме розробка та порівняння результатів створених моделей задля цього дослідження.

Четвертий розділ буде присвячений функціонально-вартісному аналізу програмного продукту.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ. ЗБІР ДАНИХ

1.1 Актуальність проблеми

Очікувана тривалість життя є одним із найважливіших показників здоров’я та добробуту населення. Більш того цей показник є найважливішим для оцінки здоров’я населення, оскільки він є більш ширшим, ніж смертність новонароджених та дітей, який оцінює лише смертність певної вікової групи [1]. Одним з найпоширеніших показників очікуваної тривалості життя є очікувана тривалість життя при народженні. Завдяки ньому можна припустити, що вікові показники смертності для відповідного року будуть застосовані протягом усього життя людей, які народилися в цьому році [2].

По суті, оцінка проектує вікові коефіцієнти смертності за певний період на все життя населення, яке народилося протягом цього періоду. Ця ознака є важливим інтегральним демографічним показником, що слугує для поколінь, що живуть нині, і означає передбачувану кількість років майбутнього життя людини, яка досягла даного віку. Зокрема його значимість відіграє вагому роль у формуванні досліджень загальної якості та рівня життя населення в країні, Індекс Людського Розвитку (ІЛР) тощо [3].

Згідно визначенню Всесвітньої організації охорони здоров’я, очікувана тривалість життя при народженні виводиться з таблиць очікуваної тривалості життя і базується на статево-вікових коефіцієнтах смертності. Значення очікуваної тривалості життя при народженні, отримані від ООН, відповідають оцінкам на середину року і узгоджуються з відповідними середніми п'ятирічними демографічними прогнозами ООН щодо народжуваності за середнім сценарієм [4].

На рисунку 1.1 можна побачити інформацію про очікувану тривалість життя країн нашої планети за 2022 рік.

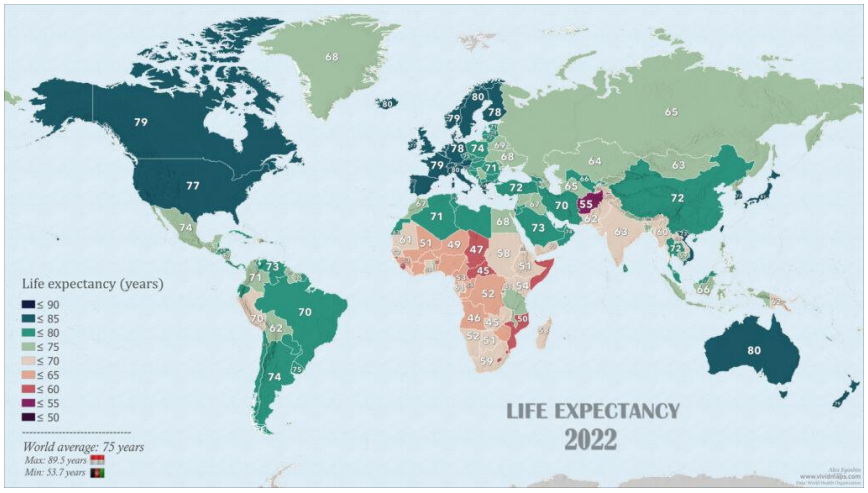


Рисунок 1.1 - Очікувана тривалість життя населення планети Земля за 2022 рік

Цей показник використовується для важливих задач стратегічного планування та політики всіх країн світу. Серед основних галузей є планування охорони здоров'я та розподіл ресурсів. Уряди та організації охорони здоров'я використовують його для прогнозування потреб населення в охороні здоров'я, включаючи попит на медичні послуги, заклади та персонал.

До прикладу з 1990 року в США спостерігалось більш швидке зниження смертності від раку молочної залози в порівнянні з іншими країнами. В 2006 році пройшло моделювання зниження смерності від раку молочної залози в США, під час якого науковці прийшли до висновку, що близько двох третин зниження було зумовлено покращенням діагностики і скринінгом захворювання. Що означало, що система охорони здоров'я США перевершила систему охорони здоров'я інших країн, що і стало однією з причин, чому дослідники виявили, що рак має менший вплив на тенденції очікуваної тривалості життя в США, ніж в більшості інших країн [5].

Економічний розвиток: вища тривалість життя часто корелює з кращими економічними показниками. За графіком кореляції між очікуваною

тривалістю життя при народженні та ВВП на душу населення, представленим порталом Our World in Data [6], можна помітити, що частіше за все вищий показник очікуваної тривалості життя буде означати кращий показник ВВП на душа населення країни, що в свою чергу буде означати про кращий економічний стан країни. Графічний вигляд розподілу на рисунку 1.2.

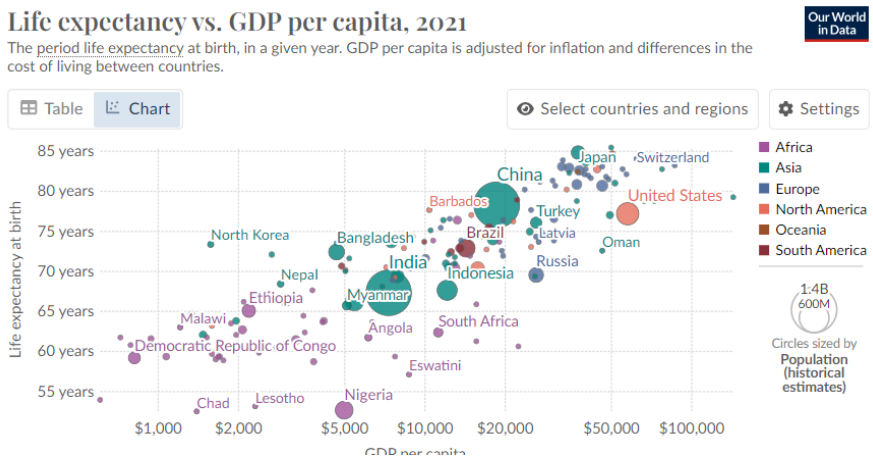


Рисунок 1.2. - Кореляція очікуваної тривалості життя і ВВП на душу населення

Також дослідники зі Стокгольмського Інституту Бізнес Досліджень Андреас Берг і Тереза Нільссон у 2009 році на прикладі 92 країн з 1970 по 2005 рік дослідили зв’язок між трьома компонентами глобалізації та очікуваною тривалістю життя. А саме економічним, соціальним і політичним. Вони виявили, що економічна глобалізація мала дуже сильний позитивний вплив на очікувану тривалість життя навіть після поправок на дохід, споживання продуктів харчування, грамотність населення, кількість лікарів тощо [7].

Якість життя: оцінка очікуваної тривалості життя відображає загальну якість життя в суспільстві. Більша очікувана тривалість життя часто відповідає кращим умовам життя, доступу до охорони здоров’я, освіти, санітарії та іншим факторам, що сприяють добробуту населення. Тому збільшення очікуваної тривалості життя є головною метою для урядів і політиків, які прагнуть підвищити рівень життя свого населення.

Актуальність проблеми дослідження очікуваної тривалості життя населення з використанням методів машинного навчання полягає в її

потенціалі для вирішення критичних проблем громадського здоров'я та прийняття рішень на основі фактичних даних. Дослідження, опубліковане в журналі Nature Communications під назвою "Прогнозування глобальних спалахів інфекційних захворювань за допомогою глибокого навчання" [8], дає уявлення про важливість використання методів машинного навчання для розуміння динаміки здоров'я населення. Це дослідження демонструє важливість моделювання для прогнозування спалахів інфекційних захворювань у всьому світі. Використовуючи великі масиви даних та передові алгоритми машинного навчання, дослідження показало можливість передбачати поширення інфекційних захворювань з безпрецедентною точністю та швидкістю.

Розуміння складної взаємодії цих факторів та їхнього впливу на тривалість життя має вирішальне значення для урядів країн, організацій охорони здоров’я та соціальних організацій. Використовуючи методи машинного навчання, можна виявляти складні закономірності, визначати фактори ризику та розробляти прогностичні моделі для прогнозування тенденцій тривалості життя в конкретних групах населення.

Щодо ситуації в Україні, за останні 30 років зросла тривалість життя в Україні за даними дослідження Державної служби статистики у 2021 році. Для чоловіків цей показник складає 66 років, в той час як для жінок 76 років. Що ж про середню очікувану тривалість життя, в 2019 році склала 72,01 років. Цікаво, що ці ж самі показники у 1990 році складали 70 років: 74 – для жінок, 65 – для чоловіків [9].

Однак через розпочатим Росією повномасштабним вторгненням на територію України, цей показник значно знизився. Станом на серпень 2023 року кількість населення нашої країни становив 36,3 млн осіб, з яких на підконтрольній українській владі території 31,5 млн. В той же час за оцінкою Інституту демографії та проблем якості життя НАН України, найближчі 26 років населення України може скоротитися до 25,2 млн осіб. Про це йдеться

в проєкті Стратегії демографічного розвитку України на період до 2040 року [10].

Серед демографічних викликів та загрозами розвитку України, дослідники виявили високий рівень передчасної смертності. В першу чергу це стосується чоловіків, що пов’язано зі збройною агресією Російської Федерації, яка спричинила численні смерті, як військових так і цивільного населення. Як висновок терористичних дій Росії, середня очікувана тривалість життя скоротилась з 66,4 року та 76,2 року у 2020 році до 57,3 і 70,9 років у 2023 році для чоловіків та жінок відповідно.

Важливо також відмітити той факт, що народжуваність порівняно з 2021 роком знизилась на 45%. За даними Мін’юсту, в Україні впродовж 2023 року народилося 187387 немовлят, що є найменшим показником за час незалежності. Про це повідомляє Опендатабот із посиланням на дані Мін’юсту [11]. Приклад статистики можна побачити на рисунку 1.3.

Рисунок 1.3 - Народжуваність в Україні 2010-2023

Тому на меті у Мінсоцполітики є ухвалення Стратегії демографічного розвитку України, яка може бути розроблена, враховуючи міжнародний досвід, задля забезпечення гідного рівня і якості життя населення, а також поверненню громадян з-за кордону та відтворення населення.

Впровадження даної Стратегії дозволить покращити параметри демографічної динаміки, в результаті якого чисельність населення України становитиме на 1 січня 2041 року 33,9 млн осіб, а на 1 січня 2051-го – 31,6 млн осіб [10].

Як можемо побачити, актуальність теми є досить вагомою, як в Україні так і закордоном. Показник очікуваної тривалості життя є важливим індикатор задля створення нових стратегій розвитку, як цілих країн, так і

окремих галузей. Саме завдяки цій оцінці можна ідентифікувати і спрогнозувати рівень життя і якість життя населення в майбутньому. Зокрема, використовуючи методи машинного навчання для дослідження можна зрозуміти складну взаємодію вирішальних факторів та їхнього впливу, а найголовніше перейняти досвід інших країн.

1.2 Пошук та попередня обробка даних

Попередня обробка даних є важливим етапом будь-якого дослідження та прогнозу даних. Зокрема цей етап може слугувати підвищенням точності аналізу, покращення продуктивності алгоритмів, зниження ризику помилок та отримання кращих результатів.

Задля підвищення швидкості аналізу методами інтелектуального аналізу, дослідники та науковці зазвичай спочатку очищають вихідні дані, що включає в себе заповнення пустих значень, очищення від шумів, видалення нерелевантної інформації, тощо.

Якість даних відповідає за точність, повноту, узгодженість та релевантність розв’язуваної задачі. Кількість даних – це кількість і різноманітність даних, які охоплюють можливі сценарії та варіації, з якими можуть зіткнутися моделі. Погана якість та кількість даних може призвести до упередженості, надмірної або недостатньої пристосованості моделей, які погано працюють з новими та небаченими даними. Задля вирішення даної проблеми, перед застосуванням алгоритмів машинного навчання слід виконати очищення, попередню обробку, доповнення та перевірку даних. Слід використовувати відповідні методи для розбиття даних і вибірки для створення репрезентативних і збалансованих навчальних, тестових та валідаційних наборів.

Одними з основних помилок, з якими можна зіштовхнутися під час дослідження є:

- пропуски у даних;
- аномальні значення;

- шум;
- невідповідний стан;
- суперечливість.

Аби уникнути або виправити ці проблеми, під час дослідження можна користуватися різними рішеннями. До прикладу для виправлення аномальних значень, тобто значень, які разюче відрізняються від інших в наборі даних, можна замінити значення на найближче граничне.

Пропуски у даних є однією з найпоширеніших проблем під час аналізу будь-якого масиву даних. Оскільки інформація через ті чи інші причини може бути не записана, то прогнозування на таких даних буде неякісним і неточним. Задля цього можна замінити дані шляхом апроксимації даних, заміни на середнє значення.

Шум означає відхил даних від середнього значення даних, який не несе особливо важливої додаткової інформації для дослідження. Аби уникнути цього можна використовувати різні авторегресійні методи, які застосовуються зазвичай при аналізі часових рядів, а також спектральний аналіз, який дозволяє відсікти високочастотні складові даних.

Невідповідний стан даних – це більш широке та загальне поняття самої проблеми. Дані можуть бути записані в іншому форматі, наприклад текстовому замість числового, дублювання тощо. Такі проблеми краще за все вирішувати і аналізувати безпосередньо в середовищі програмування, де буде можливість завдяки різним інструментам та бібліотекам відслідкувати невідповідність даних і виправляти їх.

В свою чергу суперечливість даних полягає в тому, звідки була взята інформація, яка експертна оцінка була проведена для них, і яке кінцеве джерело суперечливості. Це можна зробити ще на етапі пошуку інформації, і брати свої дані з перевірених джерел.

Важливість підбору і якості даних для завдання є надважливим етапом, адже на меті стоїть задача виявити взаємозв’язки між різними показниками, аби відстежити очікувану тривалість життя населення. Чим якісніше та

«чистіші» будуть наші дані, тим краще нам вдасться побудувати модель машинного навчання (МН), яка зможе прогнозувати даний показник.

1.3 Етапи розробки моделі машинного навчання

Найголовнішими та найважливішими етапами розробки певної моделі МН є:

1) процес підготовки даних;
2) визначення найважливіших ознак для заданої задачі;
3) визначення типу вирішуваної задачі, конструювання алгоритму;
4) тренування алгоритму та конфігурація гіперпараметрів;
5) тестування, покращення задля отримання кращих результатів.

Якість представлення даних в МН є найважливішим на початку будь-якого розробки. Попередня обробка даних є важливим етапом будь-якого дослідження та прогнозу даних. Зокрема цей етап може слугувати підвищенням точності аналізу, покращення продуктивності алгоритмів, зниження ризику помилок та отримання кращих результатів [12].

Задля підвищення швидкості аналізу методами інтелектуального аналізу, дослідники та науковці зазвичай спочатку очищають вихідні дані, що включає в себе заповнення пустих значень, очищення від шумів, видалення нерелевантної інформації, тощо [13].

Варто зазначити, що якість роботи алгоритму сильно залежить від тих ознак, які дослідники обрали в даній роботі. Отже, надважливою задачею є визначити і перевірити, які ознаки не несуть цінності для заданого дослідження. На прикладі даного дослідження не буде важливим столиця обраної держави та офіційна мова, саме тому для алгоритму ці дані будуть нерелевантні, і відповідно їх можна відкинути.

Задля оцінки важливості та пошуку закономірностей між даними можна використати різні підходи, наприклад кореляційний аналіз,

інформаційне значення, візуалізація даних, рекурсивне виключення ознак тощо.

Наступним етапом є визначення типу вирішуваної задачі. Серед задач МН можна виділити такі як класифікація, кластеризація, регресія, ранжування та знаходження аномалій. Вибір типу прямо залежить від цілей та мети дослідження. Це допоможе отримати якомога точніші результати, адже визначення типу задачі буде впливати на якість роботи моделі та її показників.

Для подальшої розробки, перед навчанням даних треба розбити всі відфільтровані і почищені дані на тренувальну та тестувальну вибірки. Зазвичай розбиття вибірки йде у співвідношенні 80% та 20% відповідно. Важливим етапом є визначення оцінки результатів, шляхом вибору метрик якості, які допоможуть нам обирати найкращу з моделей. Також з їх допомогою можна покращувати результати. Серед задач регресії найпопулярнішими метриками є Середньо-квадратична помилка MSE, середньо-абсолютна похибка MAE, коефіцієнт детермінації R^2.

Після цього модель вчиться на навчальних даних з метою вивчення закономірностей або побудови прогнозів. Пізніше йде оцінка продуктивності моделі на валідаційних даних для визначення її точності та генералізованості. Після чого можна покращувати значення метрик, задля отримання більш кращих та якісніших результатів роботи моделі.

1.4 Постановка задачі

Актуальність проблеми дослідження прогнозування очікуваної тривалості життя несе неабиякий вплив на оцінку якості життя населення. Можливість прогнозування, пошуку тенденцій та дослідження досвіду багатьох країн світу дає сформувати урядам держав, всесвітнім організаціям та спілками і окремим людям уявлення про майбутні перешкоди та закономірності, з якими треба буде зіштовхнутися у майбутньому. Це дає

змогу сформувати стратегію розвитку багатьох галузей життєдіяльності людини, базуючись на основних ознаках, які впливають на зміну показника очікуваної тривалості життя.

Враховуючи розмаїття наявних моделей та методів для прогнозування, стоїть задача вибору та обгрунтування найкращої з них. Рекомендується спробувати декілька різних моделей та методів, порівняти їхні результати та обрати ті, які найкраще будуть відповідати визначеним проблемам.

Отже, для якісного дослідження очікуваної тривалості життя населення і виявленню найважливіших факторів впливу на цей показник, необхідно вирішити наступні задачі.

1. Дослідити і зібрати дані, які будуть актуальні і релевантні для заданої мети і цілей. Також підготувати та обробити ці дані, визначивши найважливіші аспекти для подальшої роботи.
2. Дослідити та розробити найкращі методи машинного навчання з метою виявлення основних факторів, які впливають на показники очікуваної тривалості життя.
3. Оцінити ефективність розроблених алгоритмів та моделей, використовуючи визначені оцінки ефективності та точності.

1.5 Висновки до розділу 1

У першому розділі було визначено поняття очікуваної тривалості життя населення. Також було розглянуто вплив цього показника на різні галузі життєдіяльності людини. Вдалося вияснити найбільш поширені приклади застосування саме цієї ознаки. Ще раз підсумуємо, що очікувана тривалість життя є одним із найбільш поширеним і найважливішим показників здоров’я та добробуту населення.

Не менш важливим етапом було приділення належної уваги проблемам очікуваної тривалості життя в та рівню смертності в Україні. Зокрема розглянута ситуація в нашій країні до початку повномасштабного вторгнення

та після. Були приведені прогнози окремих державних установ, щодо зміни показника очікуваної тривалості життя у майбутньому.

У розділі більш детально розглянуто різноманіття існуючих методів машинного навчання, та їх детальні етапі розробки. Важливу частину зайняло розглядання етапів роботи та підготовки даних до дослідження, ідентифікація найбільш впливових ознак для визначеної задачі та визначення найбільш популярних метрик для оцінки моделі.

Останній підрозділ присвячений формулюванню постановки задачі та визначенню приблизного покрокового алгоритму даного дослідження.

РОЗДІЛ 2 ОГЛЯД ІСНУЮЧИХ МЕТОДІВ МАШИННОГО НАВЧАННЯ ТА ЇХ ВИБІР

2.1 Основні поняття терміну машинне навчання

Машинне навчання стало невід'ємною частиною аналізу даних, прогнозування та прийняття рішень в сучасному світі. Завдяки своєму інноваційному підходу до аналізу і інтерпретації складних даних, машинному навчанню вдалося замінити звичні способи аналізу. При цьому зосередившись на виконанні завдань прогнозування, класифікації, кластеризації тощо. Вдалося це зробити завдяки використанню наборів технік та алгоритмів, завдяки яким комп'ютери здатні навчитися на основі даних, виділяючи значущі закономірності задля вирішення задач .

Можна впевнено стверджувати, що дана сфера знаходиться на перетині статистики, комп'ютерних наук та штучного інтелекту [14]. Варто зазначити, що алгоритми та методи машинного навчання є всього лиш одним з етапів вирішення поставленої задачі. Найважливішою задачею машинного навчання є інтерпретація наявних даних і застосування їх до поставленої задачі [15].

Одними з основних компонентів машинного навчання є наступні.

1. Дані - сировина для машинного навчання. Саме на даних модель має навчитися виявляти різні закономірності та зв'язки. Дані можуть бути представлені в різних форматах, таких як текст, зображення, відео та аудіо. Якість та релевантність даних є визначальними для успішності машинного навчання.
2. Ознаки - одиниці інформації, які описують дані. Обравши правильні ознаки, модель здатна краще оцінити та зрозуміти взаємозв'язки та закономірності в наборі даних.
3. Алгоритм - математичне підґрунтя завдяки якій ведеться процес навчання на даних. Вибір найкращого методу є головним аспектом у

швидкості роботи та розмірі готової моделі. Також завдяки алгоритмам можна покращувати свою модель, оптимізуючи продуктивність та налаштовуючи параметри. Машинне навчання використовує багато математичних концепцій та принципів. Серед них найосновнішими є математичний аналіз, теорія ймовірності і математична статистика, лінійна алгебра, теорія оптимізації.

Малюнок основних компонентів МН можна побачити на рисунку 2.1:

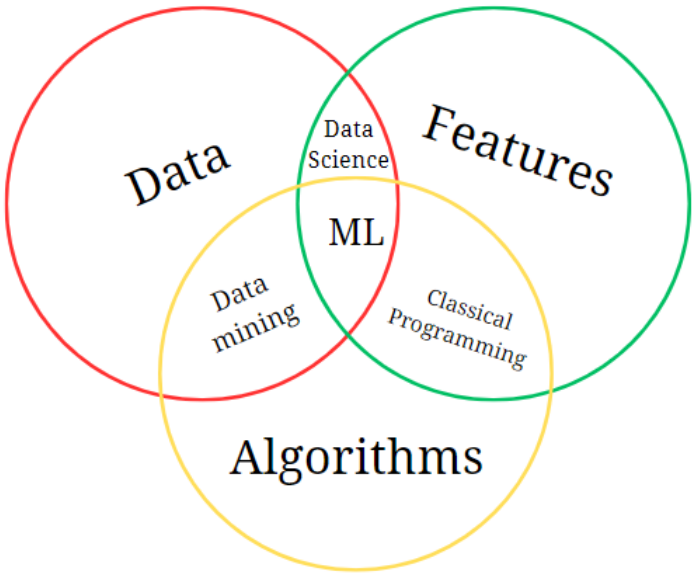


Рисунок 2.1 - Основні компоненти МН

В загальному випадку прийнято розрізняти чотири підходи до машинного навчання. Залежно від типу задачі, машинне навчання можна поділити на чотири типи (рис. 2.2).

1. Навчання з учителем (Supervised Learning). Цей метод вбачає навчання на основі маркованих даних, де для кожного прикладу вхідних даних є відомими відповідні вихідні мітки або цільові значення. Такий підхід може без допомоги людини надати відповідь для нового об’єкту. Даний тип навчання застосовується для завдань класифікації та регресії.
2. Навчання без вчителя (Unsupervised Learning). В цьому методі модель навчається на невідмічених даних, без наявності цільових міток або відповідей. Модель шукає внутрішні закономірності або структури даних без заздалегідь визначених цілей. Основною метою є виявлення

прихованих шаблонів, кластерів або структур у наборі даних. Прикладами таких задач є кластеризація, зниження розмірності, задача оцінки щільності.

3. Навчання з підкріпленням (Reinforcement Learning). В цьому методі модель навчається приймати послідовні дії в певному середовищі з метою максимізації певної нагороди або досягнення певної цілі. В даному методі немає заздалегідь підготовлених даних, модель повинна взаємодіяти з середовищем, пробуючи різні дії та вивчаючи, які з них приводять до бажаних результатів.
4. Навчання з частковим залученням вчителя. Це метод МН, в якому модель навчається на датасеті з частково міченими і частково неміченими даними. Дані, які не мають міток, використуються для покращення моделі шляхом використання їх структури або внутрішньої інформації.

Рисунок 2.2 - Класифікація методів МН

2.2 Алгоритми машинного навчання

Вибір методів машинного навчання є важливим кроком для якісного вирішення поставленої задачі. До нього варто підійти з відповідною зваженістю і ретельністю, адже різні моделі можуть працювати з різними типами даних. В цій роботі були обрані дані, що мають числові показники цільової змінної, а саме очікувана тривалість життя, було прийнято рішення розглянути п’ять алгоритмів машинного навчання: лінійна регресія, метод опорних векторів, Elastic Net, Random Forest та метод головних компонентів.

2.2.1 Лінійна регресія

Лінійна регресія є одним з найпростіших і найпоширеніших алгоритмів машинного навчання для задач регресії. Це метод, який використовується для прогнозування безперервної цільової змінної (залежної змінної) на основі однієї або декількох вхідних ознак (незалежних змінних) [16].

У лінійній регресії зв’язок між вхідними ознаками X і цільовою змінною у вважається лінійним. Математично, загальна регресійна модель представлена у формулі (2.1).

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \mathcal{E}, \quad (2.1)$$

де y – цільова змінна;

$(\beta_1, \beta_2, \dots, \beta_n)$ – коефіцієнти регресії, що відповідають за зсуви та напрями прямої;

$(x_1, x_2, \dots, x_n)$ – незалежні змінні;

$\mathcal{E}$ - похибка, яка відображає різницю між фактичними та прогнозованими значеннями.

На рисунку 2.3 можна побачити приклад графіку простої лінійної регресії з однією незалежною зміною.

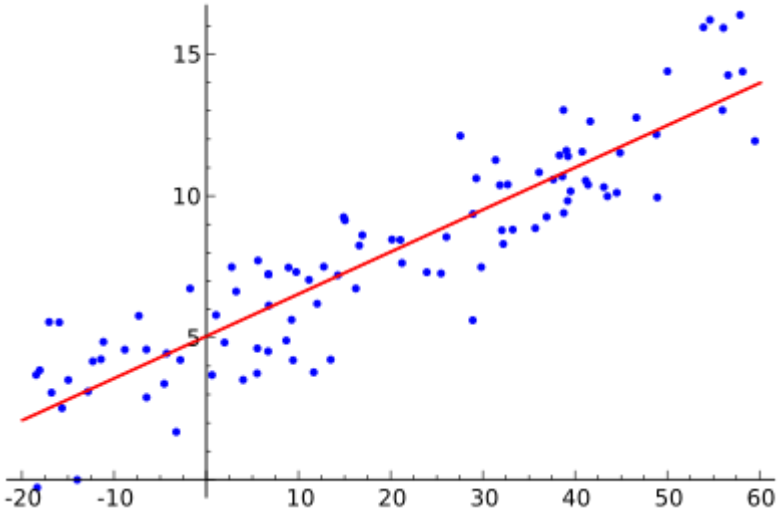


Рисунок 2.3 - Приклад простої лінійної регресії з однією незалежною зміною

Завдання лінійної регресії полягає у підборі таких параметрів, які дозволять використовувати рівняння для прогнозування цільової змінної. Для визначення цих параметрів застосовуватимемо метод найменших квадратів.

Серед переваг лінійної регресії можна виділити простоту інтерпретації, ефективність при великих наборах даних, швидкість навчання та передбачення, а також низький ризик перенавчання. Оскільки це є одним з найбільш зрозумілих методів в машинному навчанні, інтерпретація коефіцієнтів моделі дає змогу зрозуміти, як кожна вхідна ознака впливає на цільову змінну. Важливим для оперативності є її швидкість, адже вона має швидкі властивості до навчання, а також тенденцію до меншого ризику перенавчання порівняно з більш складними методами.

Однак з недоліків варто зазначити обмежену спроможність та чутливість до викидів та неспроможність робити складні прогнози. Проблема полягає в тому, дана модель передбачає лише лінійну залежність між вхідними ознаками та цільовою зміною, чого може бути недостатньо для моделювання складних, не лінійних зв’язків. Також задачею буде якісно підготувати дані, адже ця модель чутлива до викидів, і може бути вразлива

до наявності неправильних даних, які відрізняються від загальної закономірності.

2.2.2 Метод опорних векторів для задачі регресії

Метод опорних векторів для задачі регресії (SVR) – це простий алгоритм аналізу даних для задач класифікації та регресії. На меті даного методу є пошук гіперплощини або набору гіперплощин в N-мірному просторі (N – кількість функцій), яка максимально розділяє точки даних різних класів. Для задач регресії цю мету можна сформувати, як пошук функції, яка апроксимує зв'язок між вхідними змінними та безперервною цільовою змінною, мінімізуючи при цьому помилку прогнозування [17].

Гіперплощина визначається за допомогою опорних векторів. Це зразки, які найближче знаходяться до границі розподілення. Вони визначають положення та орієнтацію гіперплощини.

Алгоритм прагне знайти гіперплощину, яка найкраще відповідає точкам даних у неперервному просторі. Це досягається шляхом відображення вхідних змінних у високорозмірний простір ознак і знаходження гіперплощини, яка максимізує відстань між гіперплощиною і найближчими точками даних. Також вона мінімізує помилку прогнозування.

На рисунку 2.4 можна побачити структуру SVR.

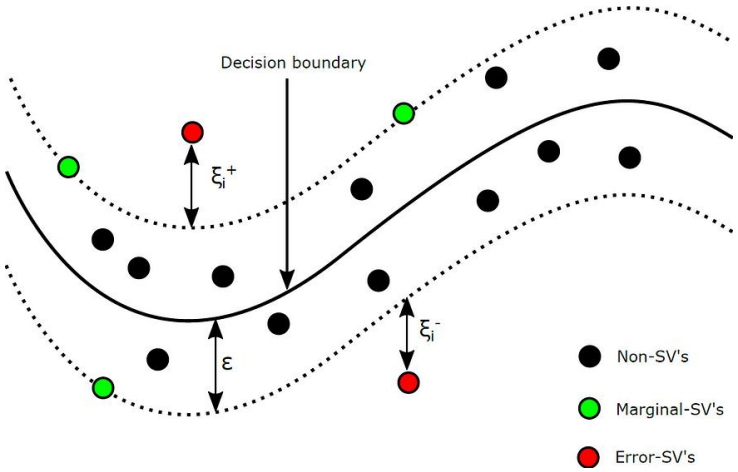


Рисунок 2.4 - Структура SVR

SVR може обробляти нелінійні зв’язки між вхідними змінними і цільовою змінною, використовуючи функцію ядра для відображення даних у просторі вищої розмірності. Це робить SVR потужним інструментом для задачі регресії, де можуть існувати складні взаємозв’язки між вхідними змінними та цільовою змінною.

Даний метод може використовувати різні ядерні функції, наприклад лінійну, поліноміальну, радіальну базисну функцію. Це дозволяє алгоритму моделювати складні взаємозв’язки між змінними.

Перевагами SVR можна виділити високу точність прогнозування, навіть у випадку складних та нелінійних даних. Також стійкість до шуму та інтерпретованість, що дає краще зрозуміти зв’язок між ознаками та цільовою функцією.

З недоліків зазначимо високу обчислювану складність, особливо для задач з великим набором даних, а також чутливість до вибору похибки, адже точність буде залежати від вибору похибки.

2.2.3 Random Forest

Random Forest (Випадковий ліс, RF) є ансамблевою моделлю, яка відома своєю універсальністю. Як і попередній метод, RF використовується як для задач регресії, так і для задач класифікації.

Суть алгоритму полягає в створенні багатьох дерев рішень, кожне з яких побудоване на випадково вибраних підмножинах даних. Потім модель агрегує результати всіх цих дерев рішень, щоб зробити загальний прогноз для невидимих точок даних. Таким чином, вона може обробляти більші набори даних і фіксувати складніші асоціації, ніж окремі дерева рішень.

Отож окремі моделі дерев рішень будуються за допомогою методу “begging”. Вона передбачає випадковий вибір підмножин навчальних даних і побудову з них менших дерев рішень [18].

Після цього етапу, йде об’єднання менших моделей, щоб сформувати модель випадкового лісу, яка виводить єдине значення прогнозу. Цей метод допомагає зменшити дисперсію та підвищити точність, об’єднуючи прогнози декількох дерев рішень.

Приклад алгоритму RF можна побачити на рисунку 2.5.

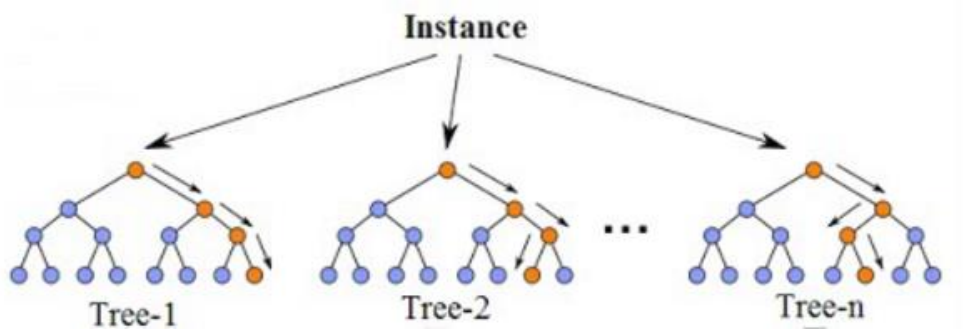


Рисунок 2.5 - Приклад алгоритму Random Forest

При використанні цього алгоритму важливо налаштувати такі гіперпараметри, як кількість дерев в лісі, максимальна глибина кожного дерева та мінімальна кількість вибірок, щоб досягти найкращої продуктивності моделі.

Серед переваг RF варто виділити високу точність, завдяки комбінації декількох дерев рішень. Важливість кожної ознаки для прогнозування, а також той факт, що RF менш схильні до перенавчання в порівнянні з окремими деревами рішень.

Однак з недоліків треба наголошувати на високій обчислювальній складності, особливо для великих наборів даних. Це пов’язано з необхідністю побудови та навчання багатьох дерев рішень. А також інтерпретованість, бо через складну структуру важко визначити, як саме кожна вхідна ознака може впливати на результат RF може бути схильний до «чорної скриньки» - ситуації, коли відомо, що модель працює ефективно, але не можна точно розглянути, як саме вона приймає свої рішення. В деяких випадках це може бути неприйнятним, особливо, якщо потрібна детальна інтерпретація результатів.

2.2.4 Principal Component Regression

Метод головних компонент регресії (Principal Component Regression) – це статистичний метод регресійного аналізу, який використовується для зменшення розмірності набору даних шляхом проектування його на підпростір меншої розмірності. Це робиться шляхом знаходження набору ортогональних (некорельованих) лінійних комбінацій вихідних змінних, що охоплюють дисперсію в даних. Головні компоненти використовуються як предиктор в регресійній моделі замість вихідних змінних [19].

PCR часто використовується як альтернативна лінійній регресії, особливо, коли кількість змінних велика або коли змінні є корельованими. Різниця полягає в тому, що PCR використовує головні компенети як предикторні змінні для регресійного аналізу замість вхідних даних. Використовуючи цей метод, можна зменшити кількість змінних у моделі та покращити інтерпретованість і стабільність результатів регресії.

Даний алгоритм працює в три етапи.

1. Застосування методу головних компонент (PCA) для генерації головних компонент зі змінних предикторів. При цьому кількість головних компонент має відповідати кількості вихідних ознак p.
2. Після цього варто зберігти перші k головних компонент, які пояснюють більшу частину дисперсії (де $k < p$), де k визначається шляхом перехресної перевірки.
3. Після йде побудова лінійної регресії, з використанням МНК.

Це може бути візуалізовано, як на рисунку 2.6.

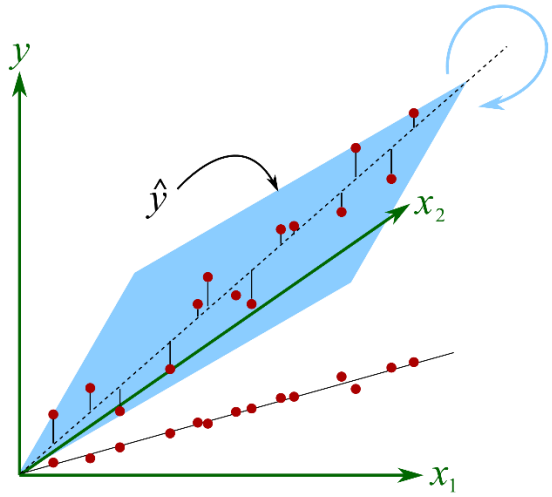


Рисунок 2.6 - Principal Component Regression

Ідея полягає в тому, що менша кількість головних компонент відображає більшу частину варіабельності даних і скоріш за все зв'язок з цільовою змінною. Тому використовується лише підмножина головних компонент.

Перевагами методу є те, що він за рахунок свого алгоритму призводить до кращої продуктивності ніж стандартна лінійна регресія, яка навчана на вихідних ознаках. Також PCR допомагає усунути мультиколінеарність у даних, видаляючи головні компоненти, пов'язані з малими власними значеннями.

З недоліків можна виділити можливість втрати інформації під час зменшення розмірності даних, оскільки не всі головні компоненти можуть відображати важливі варіативності в даних. Також чутливість до вибору кількості компонентів k та той факт, що модель не враховує нелінійні зв'язки.

2.2.5 Elastic Net

Elastic Net Regression – це метод регресії, який поєднує в собі два підходи: регуляризацію L1 (Lasso) і L2 (Ridge). Цей метод намагається збалансувати між простотою моделі (через використання меньшої кількості змінних) і її точністю (через уникнення перенавчання) [20].

Працює цей алгоритм таким чином, що спочатку потрібно визначити функцію втрат, яка вимірює різницю між фактичним і прогнозованими значеннями цільової змінної. L1 регуляризація сприяє стисненню неважливих ознак до нуля, тим самим забезпечуючи відбір ознак. L2 регуляризація штрафує великі значення коефіцієнтів, допомагаючи уникнути перенавчання.

Пізніше йде підбір оптимальних гіперпараметрів: α (який керує балансом між L1 та L2 регуляризацією) і λ (який визначає силу регуляризації). Це можна зробити шляхом крос-валідації, або інших методів оптимізації. Ну а далі йде навчання моделі і подальша її оцінка.

Перевагою даного методу є універсальність, адже він дозволяє поєднувати регуляризацію L1 і L2, що робить його адаптивним до різних наборів даних і проблем. Надійність є важливою ознакою методу, адже вона менш схильна до проблем пов’язаних з мультиколінеарністю. Також, контролюючи силу L1 та L2 можна оптимізувати модель до рівня, коли вона зможе добре узагальнювати і демонструвати меншу надмірну пристосованість.

Однак налаштування гіперпараметрів є не найкращою стороною алгоритму, адже вибір відповідних параметрів регуляризації λ може бути складним завдання і часто вимагає перехресної перевірки для оптимізації продуктивності. Також варто відмітити інтенсивність обчислень моделі, що є складнішою ніж лінійна регресія, через додаткові члени у функції витрат.

2.3 Метрики для оцінки моделі

Метрики оцінювання моделі машинного навчання є важливим інструментом для визначення ефективності та якості моделі. Вони надають об’єктивний спосіб порівняти різні моделі або налаштування однієї моделі, щоб зрозуміти, наскільки добре вони виконуються на конкретній задачі. Таким чином, використовуючи метрики, можна покращити якість моделей,

вибрати найкращу для конкретної задачі та зрозуміти, як моделі приймають рішення.

Важливо пам’ятати, що не існує єдиної «найкращої» метрики оцінювання. Вибір метрик залежить від конкретної задачі, типу моделі та доступних даних. Тому важливо використовувати декілька різних метрик для отримання всебічної оцінки.

В даній роботі було прийнято рішення використовувати наступні метрики:

- MAE (Mean Absolute Error) – середня абсолютна помилка;
- RMSE (Root Mean Square Error) – середньоквадратична помилка прогнозу;
- R-squared – коефіцієнт детермінації.

Розглянемо більш детально, що означає кожен з них [21].

MAE – це широко використовувана метрика для оцінки продуктивності моделей регресії в машинному навчанні. Вона вимірює середнє значення абсолютних помилок між прогнозованими та фактичними значеннями цільової змінної. Чим менше значення MAE, тим краще модель працює [22].

Математична формула наведена в формулі (2.2).

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|, \quad (2.2)$$

де N – кількість прикладів у наборі даних;

y_i – фактичне значення цільової змінної для і-го прикладу;

$\hat{y}_i$ – прогнозоване значення цільової змінної для і-го прикладу.

Як зазначалося вище, MAE має прості інтерпретації. Низьке значення MAE (близько до 0) вказує на те, що модель робить точні прогнози, а фактичні значення цільової змінної близькі до прогнозованих. Високе значення MAE (більше 0) вказує на те, що модель робить неточні прогнози, а фактичні значення цільової змінної значно відрізняються від прогнозованих.

Отже, MAE – це важлива метрика, яка дозволяє оцінити точність моделі регресії, зокрема виміряти наскільки добре прогнози моделі відповідають фактичним значенням.

RMSE – це ще одна ключова метрика для вимірювання точності моделі в задачах регресії. Вона вимірює середнє значення квадратів відхилень між прогнозованими значеннями моделі та фактичними значеннями цільової змінної, і після цього вираховує квадратний корінь цього значення [23].

RMSE обраховується завдяки формулі (2.3).

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}, \quad (2.3)$$

де N – кількість прикладів у наборі даних;

y_i – фактичне значення цільової змінної для і-го прикладу;

$\hat{y}_i$ – прогнозоване значення цільової змінної для і-го прикладу.

Також відомо, що RMSE є квадратним коренем MSE (Mean Squared Error) тобто середньоквадратичної похибки. Цю метрику не варто окремо обраховувати, знаючи значення RMSE можна легко порахувати значення MSE. Тому для зручності можна також навести альтернативну формулу RMSE наведену у формулі (2.4):

$$RMSE = \sqrt{MSE} \quad (2.4)$$

RMSE вимірюється в тих самих одиницях, що і цільова змінна, що робить його досить легко інтерпретованим. Він вказує середнє відхилення прогнозованих значень від фактичних значень у тих же одиницях, що і цільова змінна.

Аналогічно до MSE, чим нижче показник, тим модель робить точніші прогнози. Відповідно чим більше, тим модель робить неточні прогнози, а фактичні значення цільової змінної значно відрізняються від прогнозованих.

R-squared (R^2) – це метрика, яка використовується для вимірювання того наскільки добре модель підходить до даних. Вона вказує на частку варіації у цільовій змінній, яка пояснюється моделлю. Показник R^2 знаходиться в

діапазоні від 0 до 1, де 0 означає, що модель не пояснює жодної варіації у даних, а 1 – що модель пояснює всю варіацію у даних [24].

Формула R^2 приймає вигляд:

$$R^2 = 1 - \frac{SS_{res}}{SS_{\text{tot}}}, \quad (2.5)$$

де SS_{res} – сума квадратів помилок (відхилень) між фактичними значеннями (y_i) та прогнозованими значеннями ($\hat{y}_i$). Зокрема ця сума обраховується формулою (2.6).

$$SS_{res} = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2, \quad (2.6)$$

де N – кількість прикладів у наборі даних;

y_i – фактичне значення цільової змінної для і-го прикладу;

$\hat{y}_i$ – прогнозоване значення цільової змінної для і-го прикладу.

SS_{tot} – загальна сума квадратів відхилень між фактичними значеннями (y_i) та середнім значенням ($\bar{y}$). Приклад обчислення наведений у формулі (2.7).

$$SS_{tot} = \sum_{i=1}^{N} (y_i - \bar{y})^2, \quad (2.7)$$

де N – кількість прикладів у наборі даних;

y_i – фактичне значення цільової змінної для і-го прикладу;

$\bar{y}$ – прогнозоване значення цільової змінної для і-го прикладу.

R^2 вказує на те, яка частина варіації і цільової змінної пояснюється моделлю. Чим ближче значення до 1, тим краще модель пояснює варіацію даних. Якщо значення близьке до 0, то модель погано підходить до даних і не пояснює їхню варіативність [25].

2.4 Вибір засобів для програмного продукту

Вибір засобів для програмного продукту є важливою частиною планування роботи, адже від цього буде повністю залежати функціональність проекту, час розробки, що є дуже важливим для задач МН, зручність та доступність.

2.4.1 Мова програмування Python 3

Для реалізації поставленої задачі в даній роботі було обрано мову програмування Python. Мова програмування Python стала однією з найпопулярніших мов у сфері машинного навчання з численними причинами, які підтримують її вибір для розв'язання задач в цій області [26].

Історія Python в машинному навчанні почалась в 1990-х роках, коли вона почала набирати популярність серед дослідників та вчених. Одним із головних принципів цієї мови є читабельність коду. Творець Python Гвідо ван Россум підкреслював, що програма має бути зрозумілою для читача, навіть якщо це дещо уповільнює її виконання. Зрозумілість, чіткість і простота є ключовими аспектами [27].

Також серед переваг мови програмування Python варто виділити наступні властивості.

1. Простота та читабельність. Python 3 відомий своїм лаконічним синтаксисом та читабельним кодом. Це робить його легко доступним для початківців, що полегшує вивчення та співпрацю з іншими розробниками. Ця простота також полегшує розробку та відкладку програм в машинному навчанні.
2. Продуктивність та масштабованість. Python 3 може ефективно обробляти великі обсяги даних, що робить його придатним для задач машинного навчання з великими наборами даних. Його масштабованість дозволяє використовувати його на різних платформах від персональних комп'ютерів до потужних серверів.
3. Велика кількість бібліотек. Python 3 має розгалужену екосистему бібліотек, спеціально розроблених для машинного навчання. Наприклад, Scikit-learn, TensorFlow, PyTorch, Keras, pandas і багато інших. Ці інструменти дозволяють розробникам використовувати готові рішення для швидкої і ефективної роботи з даними та моделями машинного навчання. Це дає великий плюс у роботі з даними.

Бібліотеки дозволяють легко і ефективно проводити операції з даними, такі як фільтрація, об’єднання, групування та аналіз, що є важливими кроками у процесі підготовки даних для машинного навчання.

4. Доступність навчальних документацій та різноманітних матеріалів. Python має багато відкритих порталів, а також ресурсів, де легко можна знайти потрібну тобі інформацію, документацію, курс тощо. Це допомагає ефективно знайти помилку або підказку в написанні коду.
5. Широке застосування: Python використовується не лише в машинному навчанні, але й в багатьох інших сферах, таких як веб-розробка, наукові дослідження, робота з базами даних та інше. Це означає, що розробники, які володіють цією мовою програмування, можуть легко переходити з одного проекту на інший.

Задля вирішення поставленої задачі в даній роботі було обрано Jupyter Notebook за середовище розробки моделей машинного навчання. Це вибір зумовлений наступними перевагами.

1. Інтерактивність. JN дозволяє виконувати код Python рядок за рядком, що робить його ідеальним для експериментів та досліджень. Можна побачити результат коду відразу після його виконання, що полегшує процес розробки та налагодження.
2. Легка інтерпретація результатів. JN ідеально підходить для створення звітів та презентацій, оскільки можна вставляти код, текст, візуалізації та пояснення результатів в одному документі, що дозволяє легко демонструвати та пояснювати результати роботи моделі.
3. Підтримка великої кількості мов програмування: Jupyter Notebook підтримує багато мов програмування, включаючи Python, R, Julia та інші. Це дозволяє обирати мову програмування, яка найбільше підходить для конкретної задачі машинного навчання.
4. Легка візуалізація даних: Jupyter Notebook інтегрується з багатьма бібліотеками візуалізації даних, такими як Matplotlib, Seaborn, Plotly та інші. Це дозволяє легко створювати і візуалізувати графіки, діаграми та

інші візуальні елементи для аналізу даних та відображення результатів моделі.

2.4.2 Бібліотека Scikit-learn

Scikit-learn (також відома як sklearn) - це відкрита бібліотека машинного навчання для мови програмування Python. Вона надає простий та ефективний спосіб реалізації алгоритмів машинного навчання для класифікації, регресії, кластеризації, виявлення аномалій, вимірювання рівня точності та багато іншого. Розглянемо кілька ключових аспектів бібліотеки scikit-learn.

1. Простота використання: Scikit-learn створена з урахуванням простоти використання та зрозумілого інтерфейсу. Вона має чітко визначені методи та параметри, що дозволяє швидко та легко створювати, навчати та оцінювати моделі машинного навчання.
2. Багатий функціонал: Scikit-learn надає широкий спектр алгоритмів машинного навчання, включаючи методи класифікації, регресії, кластеризації, вимірювання рівня точності, виявлення аномалій, вибір найкращих параметрів та багато іншого.
3. Інтеграція з іншими бібліотеками: Scikit-learn легко інтегрується з іншими популярними бібліотеками Python, такими як NumPy, pandas, Matplotlib, Seaborn та інші, що дозволяє легко виконувати операції з даними, візуалізувати їх та ефективно створювати моделі машинного навчання.
4. Документація та підтримка: Scikit-learn має високоякісну документацію, яка містить розгорнуті приклади використання та описи методів. Крім того, бібліотека активно підтримується та регулярно оновлюється, що забезпечує користувачам доступ до новітніх методів та алгоритмів машинного навчання.
5. Широкий вибір моделей та алгоритмів: Scikit-learn містить в собі велику кількість реалізованих алгоритмів машинного навчання, що

охоплюють багато різних методів та моделей. Це дозволяє розробникам вибирати найбільш підходящі методи для конкретних завдань та даних.

Однак варто зазначити, що все ж таки дана бібліотека не реалізує все, що пов’язано з машинним навчанням. До прикладу вона не має підтримки для задач нейронних мереж, мереж Кохонена, навчання з асоціативним правилам та навчанням з підкріпленням.

2.4.3 Бібліотека Seaborn

Seaborn - це високорівнева бібліотека для візуалізації даних на мові програмування Python, яка базується на бібліотеці Matplotlib. Вона надає інструменти для створення привабливих та інформативних візуалізацій даних з меншими зусиллями, ніж Matplotlib. Розглянемо деякі ключові характеристики бібліотеки Seaborn.

1. Простота використання: Seaborn надає простий та зрозумілий інтерфейс для створення візуалізацій даних. Її функції часто вимагають менше коду порівняно з Matplotlib, що робить процес візуалізації більш швидким та простим.
2. Привабливий дизайн графіків: Seaborn надає стилізовані графіки за замовчуванням, що дозволяє створювати привабливі та професійно виглядаючі візуалізації даних без необхідності великого обсягу роботи.
3. Великий вибір типів графіків: Seaborn підтримує багато різних типів графіків, таких як scatter plots, line plots, bar plots, violin plots, box plots та багато інших. Це дозволяє ефективно візуалізувати різні аспекти даних.
4. Інтеграція з pandas: Seaborn ідеально підходить для візуалізації даних, які зберігаються у форматі DataFrame з бібліотеки pandas. Вона підтримує простий інтерфейс для роботи з даними з pandas та автоматично застосовує певні оптимізації для покращення продуктивності.

5. Можливості статистичного аналізу: Seaborn надає інструменти для візуалізації статистичних даних, таких як діаграми розсіювання з лініями тренду, розподілів, кореляційних матриць та інших. Це дозволяє швидко вивчати взаємозв'язки між різними змінними у даних.

2.4.4 Бібліотека NumPy

Бібліотека NumPy (numeric Python) – це основний інструмент для обчислень на мові програмування Python. Вона надає високошвидкісні структури даних, а також функції для роботи з цими даними. Серед ключових аспектів функціональності бібліотеки NumPy.

1. Масиви: Одним з основних об'єктів у NumPy є масиви (ndarray), які представляють собою N-вимірні масиви даних одного типу. Масиви numpy дозволяють ефективно зберігати та оброблювати великі обсяги даних.
2. Векторизація операцій: NumPy дозволяє виконувати операції над масивами ефективно та швидко завдяки векторизації. Замість виконання ітерацій над кожним елементом, можна використовувати операції між масивами, що робить код більш зрозумілим та ефективним.
3. Броадкастинг: NumPy підтримує броадкастинг, що дозволяє виконувати операції між масивами різних розмірностей та форм. Це спрощує роботу з даними та робить код більш гнучким.
4. Математичні та статистичні функції: NumPy містить широкий вибір математичних та статистичних функцій для роботи з масивами даних. Ці функції дозволяють виконувати операції, такі як обчислення середнього значення, медіани, стандартного відхилення, сортування та багато іншого. Зокрема функції лінійної алгебри, оптимізації, випадкових чисел, тощо.
5. Індексація та зрізи: NumPy підтримує різні способи індексації та вибору даних з масивів, включаючи звичайну індексацію, зрізи,

булеву індексацію та інші. Це дозволяє легко отримувати доступ до певних частин даних та виконувати операції з ними.

6. Інтеграція з іншими бібліотеками: NumPy легко інтегрується з іншими популярними бібліотеками для обробки даних та візуалізації, такими як Pandas, Matplotlib, Seaborn та інші.

2.4.5 Бібліотека Pandas

Бібліотека Pandas - це потужний інструмент для обробки та аналізу даних на мові програмування Python. Вона надає структури даних та інструменти для роботи з ними, що дозволяє легко та ефективно виконувати операції з даними. Розглянемо деякі ключові характеристики та функціональність бібліотеки Pandas.

1. DataFrame та Series: Основними структурами даних в Pandas є DataFrame і Series. DataFrame - це двовимірна таблиця даних, а Series - одновимірний масив даних з індексами. Вони дозволяють легко організовувати та маніпулювати даними.
2. Завантаження та збереження даних: Pandas надає зручні інструменти для завантаження та збереження даних з різних джерел, таких як CSV файли, Excel файли, бази даних SQL, HTML сторінки та інші.
3. Індексація та вибір даних: Pandas має потужні механізми індексації, які дозволяють легко вибирати та отримувати доступ до певних частин даних. Це включає звичайну індексацію, зрізи, булеву індексацію, маскування та інші методи.
4. Операції з даними: Pandas надає широкий набір операцій для роботи з даними, таких як сортування, фільтрація, групування, агрегація, об'єднання, розбиття та багато інших.
5. Маніпуляція даними: Pandas дозволяє легко додавати, видаляти, змінювати та обробляти дані за допомогою різних методів та функцій.

6. Візуалізація даних: Хоча Pandas не є бібліотекою для візуалізації даних, вона може легко інтегруватися з іншими бібліотеками, такими як Matplotlib та Seaborn, для створення візуалізацій даних.

2.5 Висновки до розділу 2

У даному розділі було розглянуто основні поняття і термінології машинного навчання. Було наведено основні розгалуження та типи видів задач машинного навчання. Серед типів задач було виділено задачі навчання з учителем (Supervised Learning), навчання без вчителя (Unsupervised Learning) та навчання з частковим залученням вчителя. Також було визначено три основні компоненти машинного навчання, серед яких дані, ознаки і алгоритми.

Пізніше досліджується вибір алгоритмів машинного навчання. Задля вирішення поставленої задачі, а саме дослідження очікуваної тривалості життя населення було обрано наступні види моделей машинного навчання: лінійна регресія, метод опорних векторів, Elastic Net, Random Forest та Principal Component Regression.

Під час цього етапу було розглянуто принципи роботи кожного з алгоритмів, наведено математичні формули та визначення. Серед іншого, було визначено переваги і недоліки кожного з методів.

Після цього вагому частину дослідження було приділено обґрунтуванню вибору мови програмування та середовищу розробки. Було обрано мову програмування Python 3, а середовище розробки Jupyter Notebook. Виділивши найосновніші переваги даного вибору, було пояснено вибір бібліотек для вирішення поставленої проблеми.

Серед бібліотек було прийнято рішення використовувати бібліотеки Scikit-lear, Seaborn, NumPy та Pandas. Було наведено короткі відомості про кожну з бібліотек окремо, а також окреслено їхні основні переваг та функціонал роботи згідно поставленної задачі.

Метою наступного розділу буде розробка вищезазначених моделей МН, а саме моделі лінійної регресія, метод опорних векторів, Elastic Net, Random Forest та Principal Component Regression. В ході розробки, варто буде розглянути, яка з моделей буде працювати краще на обраному наборі даних, який буде стосуватися даного дослідження, а саме очікувана тривалість життя населення. Порівняння якості роботи моделей будемо виконувати шляхом перевірки метрик, вищезазначених в другому розділі.

РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМІВ. АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

3.1 Огляд та обробка набору даних

3.1.1 Опис датасету

Для розробки власних моделей прогнозування очікуваної тривалості життя було обрано набір даних із відкритого джерела на сайті Kaggle [28].

В обраному датасеті міститься комплексний набір даних, який містить велику кількість інформації про всі країни світу за 2023 рік, що охоплює широкий спектр показників та атрибутів. Датасет є універсальним для багатьох досліджень, адже містить в собі демографічну статистику, економічні показники, екологічні фактори, показники охорони здоров’я, статистику освіти та багато іншого. Завдяки тому, що представлена кожна країна, цей набір даних пропонує повну глобальну картину різних аспектів життя націй, що дозволяє проводити поглиблений аналіз і порівняння між країнами.

Датасет містить інформацію про всі країни світу. Зокрема в обраному наборі даних 195 записів і 35 стовпців. Зокрема у датасеті є інформація про демографічні показники кожної країни, економічні, екологічні показники, показники рівня освіченості населення, соціальні показники (табл. 3.1).

Таблиця 3.1 – Атрибути набору даних та їхній опис

Country	Назва країни
Density (P/Km2)	Щільність населення, виміряна в особах на квадратний кілометр
Abbreviation	Абревіатура або код країни
Agricultural Land (%)	Відсоток земельної площі, що використовується для сільськогосподарських цілей

Продовження таблиці 3.1

Land Area (Km2)	Загальна площа території країни у квадратних кілометрах
Armed Forces Size	Чисельність збройних сил у країні
Birth Rate	Кількість народжень на 1 000 населення за рік
Calling Code	Міжнародний телефонний код країни
Capital/Major City	Назва столиці або великого міста
CO2 Emissions	Викиди вуглекислого газу в тоннах
CPI	Індекс споживчих цін, показник інфляції та купівельної спроможності
CPI Change (%)	Відсоткова зміна індексу споживчих цін порівняно з попереднім роком
Currency_Code	Код валюти, що використовується в країні
Fertility Rate	Середня кількість дітей, народжених жінкою протягом життя
Forested Area (%)	Відсоток земельної площі, вкритої лісами
Gasoline_Price	Ціна бензину за літр у місцевій валюті
GDP	Валовий внутрішній продукт, загальна вартість товарів і послуг, вироблених у країні
Gross Primary Education Enrollment (%)	Загальний коефіцієнт охоплення початковою освітою
Gross Tertiary Education Enrollment (%)	Загальний коефіцієнт охоплення вищою освітою
Infant Mortality	Кількість смертей на 1000 живонароджених у віці до одного року
Largest City	Назва найбільшого міста країни
Life Expectancy	Очікувана тривалість життя населення

Продовження таблиці 3.1

Maternal Mortality Ratio	Кількість материнських смертей на 100 000 живонароджених
Minimum Wage	Рівень мінімальної заробітної плати в місцевій валюті
Official Language	Офіційна мова (мови), якою (якими) розмовляють у країні
Out of Pocket Health Expenditure (%)	Відсоток загальних витрат на охорону здоров'я, що сплачуються фізичними особами
Physicians per Thousand	Кількість лікарів на тисячу осіб
Population	Загальна кількість населення країни
Population: Labor Force Participation (%)	Відсоток населення, що входить до складу робочої сили
Tax Revenue (%)	Податкові надходження як відсоток від ВВП
Total Tax Rate	Загальне податкове навантаження у відсотках від комерційного прибутку
Unemployment Rate	Відсоток робочої сили, яка є безробітною
Urban Population	Відсоток населення, що проживає в містах
Latitude	Координата місцезнаходження країни за широтою
Longitude	Координата довготи розташування країни

3.1.2 Аналіз і попередня обробка наявних даних

Як зазначалося раніше, аналіз і попередня обробка даних є важливими етапами будь-якого дослідження, зокрема побудови якісних моделей машинного навчання. Якість даних напряму впливає на якість роботи моделей, тому очистка і аналіз даних є критично важливими, адже під час них видаляються шум, помилки та неповнота даних, гарантуючи, що модель навчається на достовірній інформації.

Для того, щоб підготувати дані для побудови моделей, спочатку треба перевірити, які типи даних має кожен атрибут. Побачити результат цієї перевірки можна на рисунку 3.1.

```
Country                                    object
Density\n(P/Km2)                           object
Abbreviation                               object
Agricultural Land( %)                      object
Land Area(Km2)                             object
Armed Forces size                          object
Birth Rate                                float64
Calling Code                              float64
Capital/Major City                         object
Co2-Emissions                              object
CPI                                        object
CPI Change (%)                             object
Currency-Code                              object
Fertility Rate                            float64
Forested Area (%)                          object
Gasoline Price                             object
GDP                                        object
Gross primary education enrollment (%)     object
Gross tertiary education enrollment (%)    object
Infant mortality                          float64
Largest city                               object
Life expectancy                           float64
Maternal mortality ratio                  float64
Minimum wage                               object
Official language                          object
Out of pocket health expenditure           object
Physicians per thousand                   float64
Population                                 object
Population: Labor force participation (%)  object
Tax revenue (%)                            object
Total tax rate                             object
Unemployment rate                          object
Urban_population                           object
Latitude                                  float64
Longitude                                 float64
```

Рисунок 3.1 - Типи даних датасету

Як можемо побачити, більшість даних мають тип object, що істотно може ускладнювати вирішення поставленої задачі. Отже, було прийнято рішення змінити тип даних всіх колонок на тип float, що дасть змогу зробити дослідження простішим.

Після цього варто було визначити кількість пропущених значень в кожній з колонок, для того, щоб заповнити пропуски. На рисунку 3.2 можна побачити кількість пропущених значень в кожній з ознак набору даних.

```
Missing Values:
 Country                                    0
Density\n(P/Km2)                            0
Abbreviation                                7
Agricultural Land( %)                       7
Land Area(Km2)                              1
Armed Forces size                          24
Birth Rate                                  6
Calling Code                                1
Capital/Major City                          3
Co2-Emissions                               7
CPI                                        17
CPI Change (%)                             16
Currency-Code                              15
Fertility Rate                              7
Forested Area (%)                           7
Gasoline Price                             20
GDP                                         2
Gross primary education enrollment (%)      7
Gross tertiary education enrollment (%)    12
Infant mortality                            6
Largest city                                6
Life expectancy                             8
Maternal mortality ratio                   14
Minimum wage                               45
Official language                           1
Out of pocket health expenditure            7
Physicians per thousand                     7
Population                                  1
Population: Labor force participation (%)  19
Tax revenue (%)                            26
Total tax rate                             12
Unemployment rate                          19
Urban_population                            5
Latitude                                    1
Longitude                                   1
```

Рисунок 3.2 - Кількість пропущених значень

Для числових значень було використано заповнення пропусків середнім значенням. Це один з простих та поширених методів заповнення пропусків, який полягає в заміні відсутніх значень середнім значенням. Такий спосіб є простим, швидким, а також інтерпретованим. Для категоріальних же значень, пропуски були заповнені модою.

Отже, оскільки обрані дані містять багато різноманітної інформації про кожну з країн, було прийнято рішення дещо змінити вихідний датасет, а саме виключити з нього 7 колонок, які б не несуть важливості для дослідження очікуваної тривалості життя. Серед цих атрибутів: Country, Abbreviation, Calling Code, Capital/Major City, Currency Code, Largest City та Official Language.

Щодо змінної Life Expectancy, гістограма очікуваної тривалості життя, яка зображена на рисунку 3.3, демонструє майже нормальний розподіл з невеликою лівою асиметрії й піком близько 75-80 років. У більшості країн світу очікувана тривалість життя становить від 60 до 85 років. Цей перекіс підкреслює нерівність у результатах охорони здоров'я у світі, вказуючи на те, що хоча багато країн досягли прогресу в підвищенні середньої очікуваної тривалості життя, частина з них все ще не дотягує до цього показника.

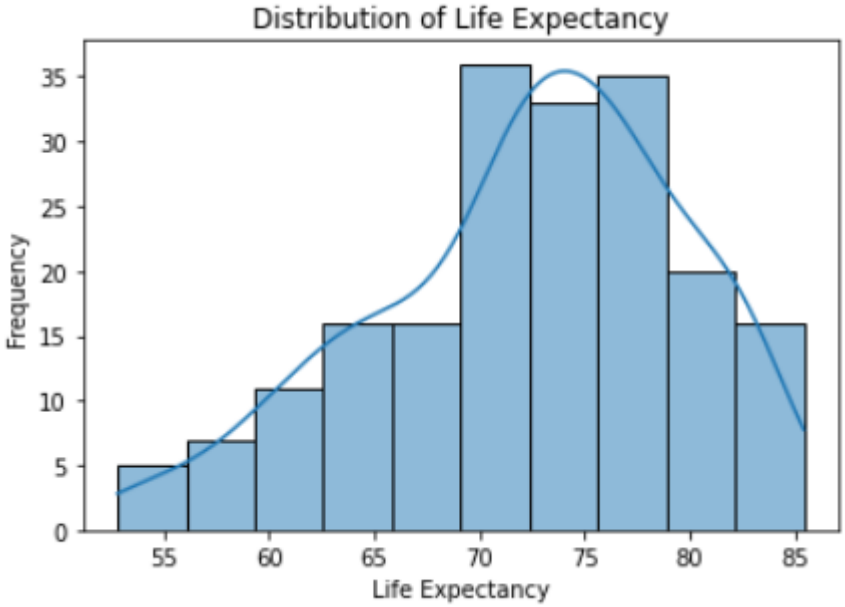


Рисунок 3.3 - Розподіл очікуваної тривалості життя

Для залежної змінної, Life Expectancy, можна обчислити описову статистику. Це може покращити розуміння даних перед візуалізацією та аналізом. Обчислення трьох мір центральної тенденції: середнє значення, моду і медіану. Результати можна побачити на рисунку 3.4:

```
Central tendency:
Life expectancy mean: 72.27967914438503
Life expectancy median: 73.2
Life expectancy mode: 76.5
```

Рисунок 3.4 - Центральна тенденція очікуваної тривалості життя

Також обчислимо три міри дисперсії, а саме стандартне відхилення, розмах і міжквартильний розмах (рис. 3.5):

```
Dispersion:
Life expectancy standard deviation: 7.483660617677969
Life expectancy range: 32.60000000000001
Interquartile range: 10.5
```

Рис. 3.5 - Дисперсія очікуваної тривалості життя

Отже, підсумовуючи можна вивести наступну інформацію про зміну Life expectancy (рис. 3.6):

- кількість записів (count);
- середнє значення (mean);
- стандартне відхилення (std);
- мінімальне значення (min);
- квартилі (0.25, 0.5, 0.75);
- максимальне значення (min);

```
Summary:
 count    187.000000
mean      72.279679
std        7.483661
min       52.800000
25%       67.000000
50%       73.200000
75%       77.500000
max       85.400000
```

Рис. 3.6 - Загальні відомості про зміну очікуваної тривалості життя

Отже, середня очікувана тривалість життя становить приблизно 72,2 роки. В середньому очікувана тривалість життя відхиляється від цього середнього значення приблизно на 7,5 років - стандартне відхилення. Між найвищою та найнижчою очікуваною тривалістю життя різниця становить приблизно 32,6 року - розмах. Середні 50% очікуваної тривалості життя охоплюють діапазон 10,5 років - інтерквартильний діапазон. Підсумок показує, що максимальна очікувана тривалість життя становить приблизно 85,4 роки, а мінімальна очікувана тривалість життя - приблизно 52,8 роки. Як

виявилося, ці країни – це Сан-Марино і Центральна Африканська Республіка відповідно.

Також важливо зазначити, що в усіх наступних етапах вибірка розбивається у співвідношенні 80% до 20%.

Наступним важливим кроком перед розробкою моделей є встановлення кореляційної залежності між залежною змінною (очікувана тривалість життя) та незалежними змінними (фактори, які впливають на залежну змінну). Для цього необхідно побудувати кореляційну матрицю, яка буде складатися з коефіцієнтів кореляції між змінною очікувана тривалість життя і іншими стовпцями, які залишилися в наборі даних. Це все зроблено для того, аби зосередитися на найбільш важливих атрибутах в даних.

Отримані результати представлені на рисунках 3.7 і 3.8.

Life expectancy	1.000000
Gross tertiary education enrollment (%)	0.708210
Physicians per thousand	0.675590
Minimum wage	0.462103
Latitude	0.460774
Tax revenue (%)	0.321470
Gasoline Price	0.244495
GDP	0.175577
Co2-Emissions	0.118737
Gross primary education enrollment (%)	0.093897
Armed Forces size	0.072699
Urban_population	0.070342
Density\n(P/Km2)	0.055198
Land Area(Km2)	0.054772
Forested Area (%)	0.019361
Population	0.008802
Unemployment rate	-0.043416
Longitude	-0.070463
CPI Change (%)	-0.143779
Population: Labor force participation (%)	-0.148990
CPI	-0.174632
Total tax rate	-0.183695
Agricultural Land(%)	-0.226174
Out of pocket health expenditure	-0.311387
Maternal mortality ratio	-0.816951
Fertility Rate	-0.846317
Birth Rate	-0.865804
Infant mortality	-0.915081

Рис. 3.7 - Коефіцієнти кореляції

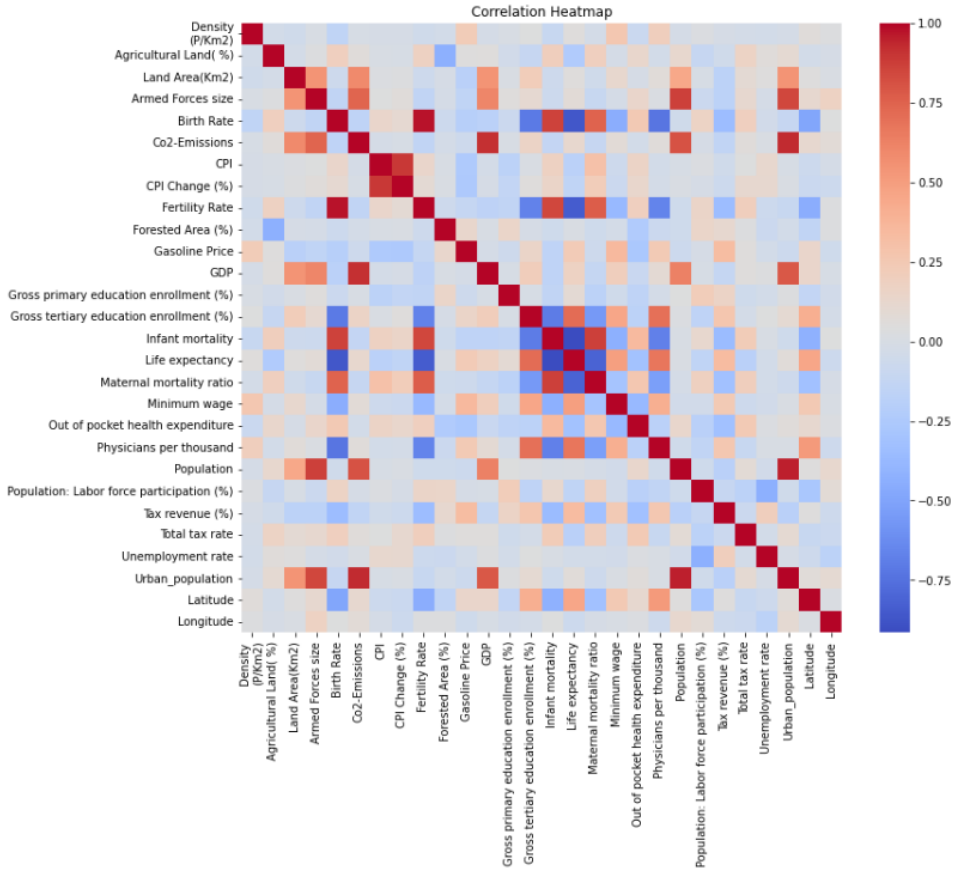


Рис. 3.8 - Теплова карта, яка відображає коефіцієнти кореляції

Після цього етапу було встановлено поріг значення кореляції по модулю. Поріг кореляції в даному дослідженні - 0.3, і під цей критерій попали дані, наведені в таблиці 3.2.

Таблиця 3.2 - Показники з найвищими за модулем коефіцієнтами кореляції

Загальний коефіцієнт охоплення вищою освітою	0.714287
Кількість лікарів на тисячу осіб	0.675590
Мінімальна заробітня плата	0.496110
Широта	0.460774
Податкові надходження як відсоток від ВВП	0.340034

Продовження таблиці 3.2

Відсоток загальних витрат на охорону здоров'я, що сплачуються фізичними особами з власної кишені	- 0.314840
Коефіцієнт материнської смертності	-0.816951
Середня кількість дітей, народжених жінкою протягом життя	-0.846317
Коефіцієнт народжуваності	-0.865804
Дитяча смертність	-0.915081

3.2 Навчання моделей

Як було зазначено раніше, в даній роботі будуть розглядатися наступні моделі машинного навчання: лінійна регресія, метод опорних векторів, Elastic Net, Random Forest та Principal Component Regression. З-поміж цих методів буде обраний найкращий.

3.2.1 Лінійна регресія

Модель лінійної регресії цінується своєю простотою та легкістю аналізу, що робить її ідеальною базовою моделлю. Для оцінки надійності моделі було використано 5-кратну перехресну перевірку, яка допомагає підтвердити стабільність моделі на різних підмножинах даних.

Після навчання моделі отримали результати, наведені в таблиці 3.3.

Таблиця 3.3 – Результати роботи моделі лінійної регресії

RMSE	MAE	R^2
3,19	2,46	0,82

R-квадрат 0,82 означає, що 82% дисперсії очікуваної тривалості життя можна пояснити ознаками, використаними в моделі. Це вказує на сильний зв'язок між предикторами та цільовою змінною, що свідчить про хорошу відповідність моделі.

Значення RMSE 3,19 свідчить про те, що в середньому прогнози моделі знаходяться в межах 3,19 років від фактичної очікуваної тривалості життя. Враховуючи, що очікувана тривалість життя зазвичай коливається від 50 до 90 років, похибка в 3,19 років є відносно невеликою, що свідчить про хорошу точність прогнозу.

MAE 2,46 означає, що в середньому абсолютна різниця між прогнозованою та фактичною очікуваною тривалістю життя становить 2,46 роки. Це також вказує на хороший рівень точності, оскільки прогнози моделі близькі до фактичних значень.

Завдяки 5-кратній перехресній перевірці, з’ясовано (табл. 3.4), що RMSE має значення 2,68, що краще за попереднє для тестового набору даних. Це свідчить про те, що модель працює стабільно добре на різних підмножинах даних. Значення R^2 0.84 свідчить про те, що модель пояснює 84% дисперсії очікуваної тривалості життя, що підтверджує стабільність і точність моделі на різних підмножинах даних.

Таблиця 3.4 – Результати роботи моделі лінійної регресії після 5-кратної перехресної перевірки

RMSE	MAE	R^2
2.68	2.04	0.84

Ефективність моделі була підтверджена графіком залежності фактичної очікуваної тривалості життя від очікуваної тривалості життя, де

точки даних близько підходили до лінії ідеального прогнозу (рис. 3.9). Однак графік залишків висвітлив області для вдосконалення; дисперсія залишків свідчить про те, що можуть існувати варіації очікуваної тривалості життя, не включені в модель, можливо, через культурні, регіональні або політичні чинники, які потребують додаткового дослідження.

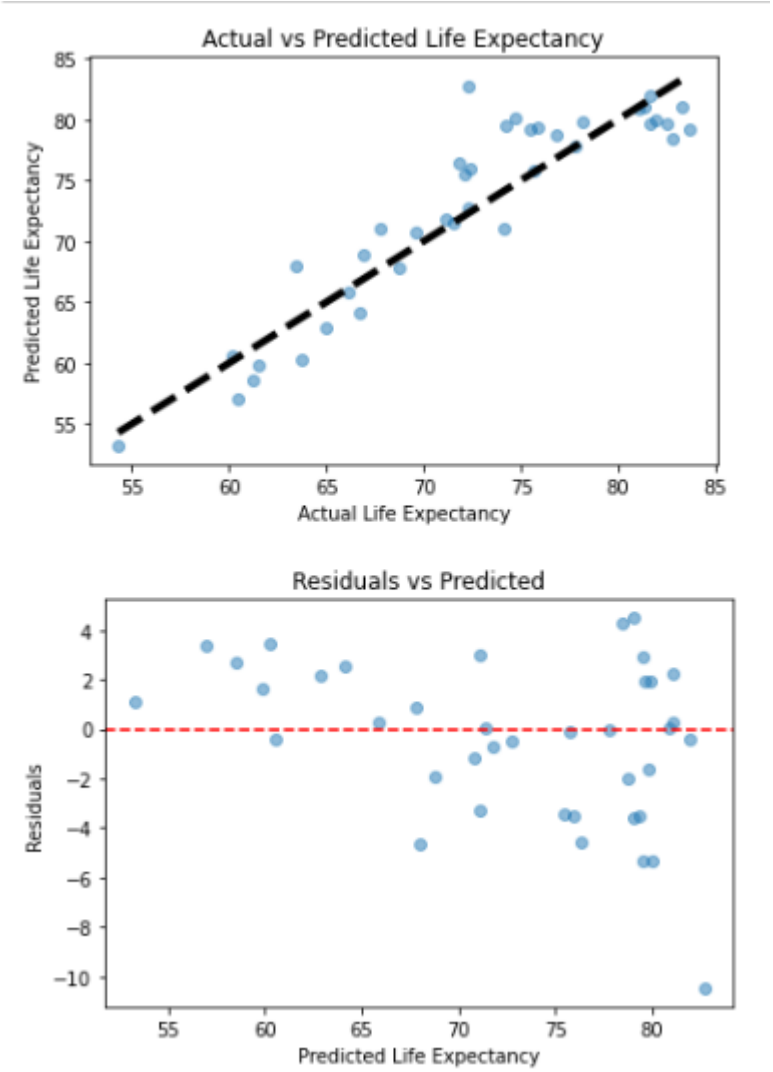


Рисунок 3.9 – Графік залежності фактичного і очікуваного значення і графік залишків моделі лінійної регресії

3.2.2 Метод опорних векторів для задачі регресії

Для SVR треба було дослідити декілька функцій ядра, щоб визначити, яка найкраще відображає складність набору даних. Серед них були: лінійне ядро, поліноміальне, ядро радіального базису та сигмоїдне ядро. Зрештою, як

можна побачити по таблиці 3.5, було обрано лінійне ядро через його хороші результати метрик, надійність, простоту та ефективність, особливо враховуючи лінійні зв’язки у вибраних ознаках.

SVR модель вчилась на тих самих ретельно відібраних ознаках, що й лінійна регресійна модель.

Таблиця 3.5 – Порівняння метрик різних ядер функцій

Тип ядра	R^2	RMSE	MAE
Лінійне	0.82	3.20	2.53
Поліноміальне	0.7	4.11	2.94
Радіального базису	0.74	3.83	2.97
Сигмоїдне	0.72	3.97	3.28

Аналогічно, якщо оцінювати без перехресної перевірки, SVR з лінійним ядром дала RMSE 3.20 і MAE 2.53, що майже відповідає показникам лінійної регресійної моделі зі значенням R^2 0.82.

Однак під час 5-кратної перехресної перевірки (табл. 3.6) SVR дещо покращила над лінійною регресією. RMSE зменшилося до 2.64, а середнє квадратичне відхилення - до 2.0228, що свідчить про дещо кращу точність прогнозування.

Таблиця 3.6 – Результати роботи моделі SVR після 5-кратної перехресної перевірки

RMSE	MAE	R^2
2.64	2.02	0.86

Варто зазначити, що середнє значення R^2 для SVR під час перехресної перевірки становило 0.86, що вище, ніж для лінійної регресії, що свідчить про її кращу здатність пояснювати варіабельність очікуваної тривалості життя між різними країнами. Стандартні відхилення для RMSE і MAE також були

нижчими – 2.64 і 2.02 відповідно, що свідчить про меншу варіабельність і вищу узгодженість прогнозів SVR, ніж лінійної регресії.

Порівнюючи результати SVR та лінійної регресії, очевидно, що обидві моделі працюють однаково без перехресної перевірки, з майже однакові значення R^2. Однак SVR дещо випереджає лінійну регресію, враховуючи перехресну перевірку не лише за середніми показниками ефективності (RMSE, MAE та R^2), але й за узгодженістю та надійністю прогнозів на різних підмножинах даних, про що свідчать нижчі стандартні відхилення. Це свідчить про те, що SVR може бути більш надійною моделлю для прогнозування очікуваної тривалості життя, особливо коли маєш справу з різноманітними та змінними даними.

Також на рисунку 3.10 зображено графіки залежності фактичної очікуваної тривалості життя від очікуваної тривалості життя, а також графік залишків.

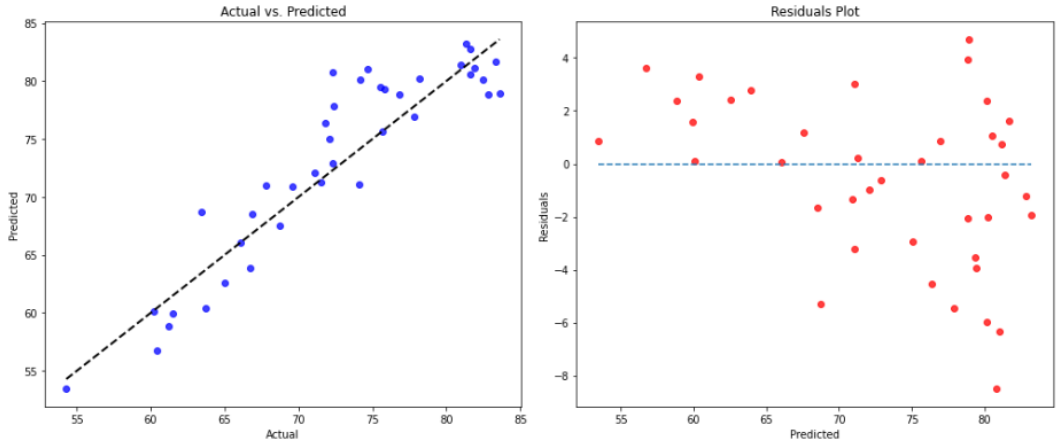


Рисунок 3.10 – Графік залежності фактичного і очікуваного значення і графік залишків моделі SVR

3.2.3 Random Forest

Для початку був проведений додатковий аналіз ознак, щоб вибрати набір, який призведе до найкращого прогнозування очікуваної тривалості

життя. Важливість ознаки визначалася за допомогою початкової моделі Random Forest для кожної ознаки набору даних.

Для відбору набору ознак, який потім використовувався для навчання моделі випадкового лісу, на додаток до оцінок важливості ознак, було використано статистичний кореляційний аналіз. Поєднання цих методів було необхідним, оскільки деякі важливі ознаки, такі як загальний рівень охоплення вищою освітою (%) та мінімальна заробітна плата, суттєво корелювали з очікуваною тривалістю життя, але не входили до числа найкращих кандидатів при оцінці важливості ознак. Результат оцінок важливості ознак можна побачити на рисунку 3.11.

Infant mortality	0.658916
Maternal mortality ratio	0.189326
Birth Rate	0.043786
Fertility Rate	0.014007
Physicians per thousand	0.010939
Gasoline Price	0.008161
Longitude	0.006679
GDP	0.006356
CPI Change (%)	0.005672
Gross tertiary education enrollment (%)	0.005100
Gross primary education enrollment (%)	0.004687
Out of pocket health expenditure	0.004461
Minimum wage	0.004318
Latitude	0.003673
Population: Labor force participation (%)	0.003493
CPI	0.003372
Agricultural Land(%)	0.003291
Total tax rate	0.003232
Unemployment rate	0.002987
Tax revenue (%)	0.002928
Co2-Emissions	0.002522
Density\n(P/Km2)	0.002358
Forested Area (%)	0.002265
Population	0.002207
Urban_population	0.002004
Land Area(Km2)	0.001875
Armed Forces size	0.001387

Рисунок 3.11 – Оцінки важливості ознак моделі SVM

Таким чином, розглядаючи комбінацію двох оцінок, було визначено набір надійних ознак, що зменшило складність моделі та запобігло надмірному підганянню даних.

Налаштування гіперпараметрів також було використано для подальшої оптимізації випадкового лісу за допомогою GridSearchCV, який визначив найкращі параметри для випадкового лісу при перегляді тестових даних для підвищення точності. 5-кратна перехресна перевірка була використана для

тестування моделі на різних підмножинах даних, щоб забезпечити узагальнюваність та узгодженість моделі.

Отже, можемо ознайомитися з результатами роботи моделі, наведеними в таблиці 3.7.

Таблиця 3.7 – Результати роботи моделі Random Forest

RMSE	MAE	R^2
2,9	2,3	0,85

Модель досить добре показала себе на невидимих тестових даних з оцінкою RMSE 2,9. MAE склало 2,3 роки, що також свідчить про хороші прогнози. Високий показник R^2 0,85 показує, що модель добре справляється з дисперсією даних.

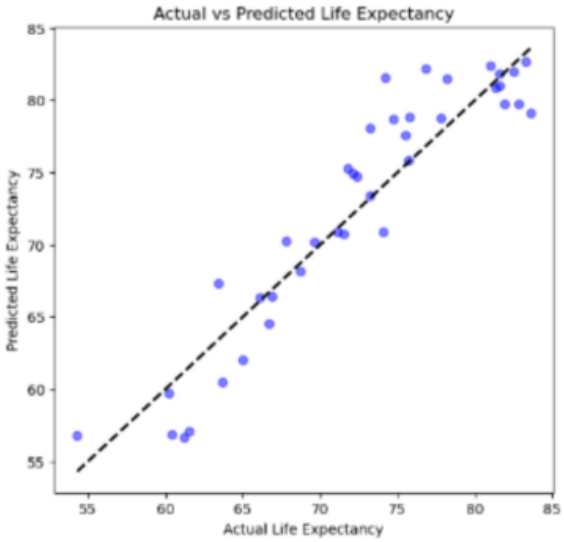
Результати 5-кратної перехресної перевірки можна побачити в таблиці 3.8. Вони ще більше підтверджують валідність моделі із середніми значеннями 2,82, 2,12 та 0,84 для RMSE, MAE та R^2 відповідно. Стандартне відхилення між оцінками для кожної складки було відносно низьким - 0,42 для RMSE і 0,32 для MAE. Ці результати вказують на те, що використана модель випадкового лісу є послідовною у прогнозуванні очікуваної тривалості життя на різних підмножинах даних, що робить її придатною для узагальнення.

Таблиця 3.8 – Результати роботи моделі Random Forest після 5-кратної перехресної перевірки

RMSE	MAE	R^2
2.82	2.12	0.84

На рисунку 3.12 показано, що прогнозовані значення близькі до фактичних значень очікуваної тривалості життя. Разом з графіком залишків, дозволяє зробити висновок, що помилки, зроблені моделлю, є випадковими, оскільки з графіків не видно жодних системних закономірностей. Відбір ознак було завершено з використанням комбінації статистичної кореляції та

оцінки важливості ознак з початкової моделі випадкового лісу для визначення більш надійного набору ознак для прогнозування очікуваної тривалості життя. Це дозволяє моделі випадкового лісу краще працювати з невидимими даними і бути узагальненою.



(a) Plot of Random Forest

(b) Residuals of Random Forest

Рисунок 3.12 – Графік залежності фактичного і очікуваного значення і графік залишків моделі Random Forest

3.2.4 Principal Component Regression

Як було пояснено раніше, PCR це метод, який використовується для зменшення розмірності набору даних шляхом проектування його на

підпростір меншої розмірності. Отже, було проведено аналіз головних компонент (PCA): перші дві головні компоненти пояснюють 39,6% дисперсії (рис. 3.13). Перша головна компонента пояснює близько 34,2% дисперсії. На другу компоненту припадає ще 11,4%, в результаті чого сукупна дисперсія пояснюється приблизно на 45,7%. Десята компонента пояснює приблизно 81,16% дисперсії в наборі даних.

Щоб вибрати кількість компонентів для регресії за методом головних компонент (PCR), зазвичай шукається точка, в якій кумулятивна пояснена дисперсія досягає розумного порогу, наприклад, 80-90%. У цьому випадку використання перших 10 головних компонент може бути достатньо для балансу між зменшенням розмірності та збереженням інформації.

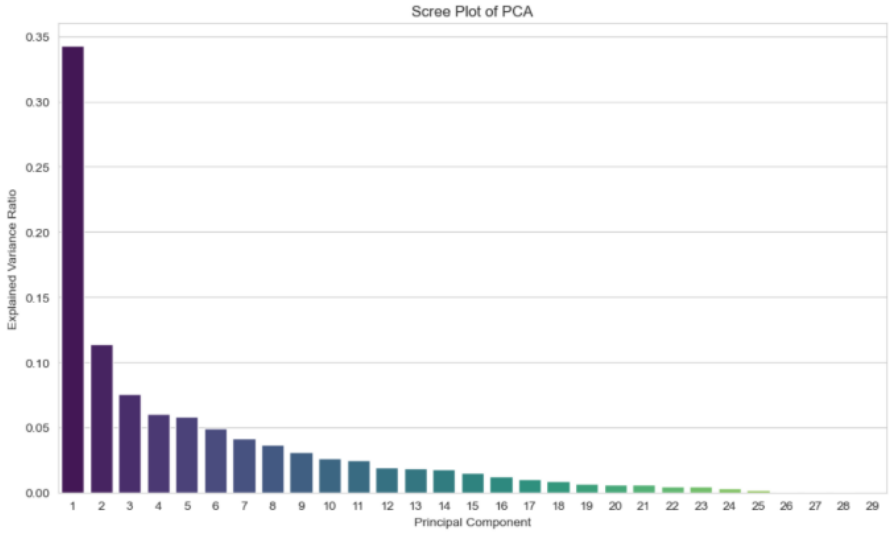


Рисунок 3.13 – Головні компоненти PCA

Отже, перший основний компонент (PC1) знаходиться під сильним впливом змінних, пов'язаних зі здоров'ям населення та демографічними показниками, таких як народжуваність, дитяча смертність, очікувана тривалість життя та коефіцієнт материнської смертності. Він являє собою загальний фактор здоров'я та демографії.

На другу головну компоненту (PC2) сильно впливають такі змінні, як чисельність збройних сил, викиди CO2 та кількість населення міста, що

свідчить про те, що вона відображає аспекти масштабу країни та її промислової активності.

PC3 має сильний зв'язок з такими економічними показниками, як CPI та зміна CPI Change (%), що відображає аспекти економічної стабільності та інфляції.

PC4 – PC10 демонструють змішаний внесок різних інших соціально-економічних та екологічних факторів.

Основні спостереження показали, що на першу головну компоненту впливають медичні та демографічні змінні, а на другу - показники промислової діяльності, такі як викиди CO2.

Змінні, які постійно демонструють сильне навантаження на перші кілька компонентів, є більш впливовими і можуть бути пріоритетними в регресійній моделі. Наприклад, такі змінні, як народжуваність, очікувана тривалість життя, міське населення та чисельність населення можуть бути особливо важливими для подальшого моделювання.

Очікувана тривалість життя має сильну негативну кореляцію з коефіцієнтом народжуваності (-0,87) та середньою кількістю дітей, народжених жінкою протягом життя (-0,85), що вказує на те, що вищі показники народжуваності та фертильності пов'язані з нижчою очікуваною тривалістю життя, що може відображати більш широкі соціально-економічні фактори.

Як було зазначено раніше PCA використовується для зменшення розмірності набору даних, що спрощує модель без втрати важливої інформації. Кількість компонентів, що залишаються, базується на поясненій дисперсії, яка показує, скільки інформації фіксує кожен компонент.

Отже, отримано результати, наведені в таблиці 3.9.

Таблиця 3.9 – Результати роботи моделі PCR

RMSE	MAE	R^2
0.078	0.061	0.89

Для оптимізації моделі в даному методі, можемо змінювати кількість головних компонент і спостерігати за впливом на RMSE. Середньоквадратична помилка (MSE) для моделі PCR з використанням оптимальних 20 головних компонент на тестових даних становить приблизно 0,006. Це покращення порівняно з початковою моделлю, яка використовувала 20 компонентів і мала MSE близько 0,003. Це вдвічі краще, ніж початкова модель, що свідчить про те, що оптимізована модель дає кращі прогнози зі зменшеною похибкою. В кінці було зроблено візуалізаційне порівняння (рис. 3.14).

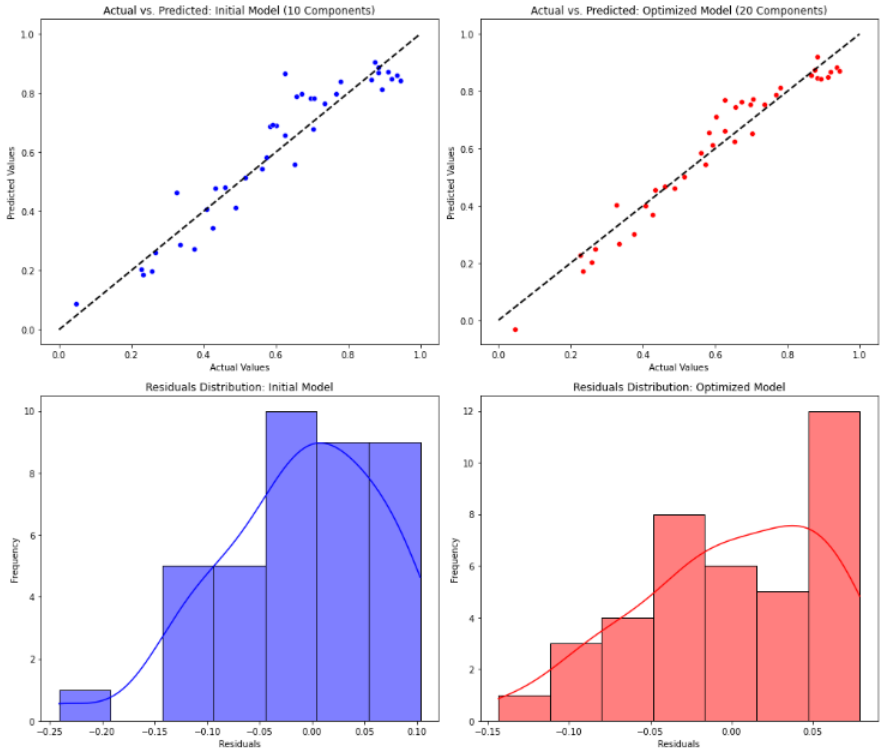


Рисунок 3.14 – Порівняння результатів з 10 і 20 компонентами PCA

Оптимізована модель з 20 компонентами (на рисунку червона) явно забезпечує кращу відповідність даним, про що свідчить як більш тісне вирівнювання фактичного та прогнозованого графіків, так і більш бажаний розподіл залишків. Таке покращення демонструє корисність використання перехресної перевірки та оптимізації компонентів для досягнення найкращої якості моделі.

Далі після перехресної перевірки маємо наступні результати наведені в таблиці 3.10.

Таблиця 3.10 – Результати роботи моделі PCR після 5-кратної перехресної перевірки

RMSE	MAE	R^2
0.063	0.045	0.92

Отже, враховуючи вищезазначені факти, можна стверджувати, що оптимізована модель PCR з 20 компонентами забезпечує кращий баланс складності та точності прогнозування, що призводить до значного зниження MSE. Це демонструє ефективність використання перехресної валідації та оптимізації компонентів для побудови більш точних прогнозних моделей.

3.2.5 Elastic Net

Під час процесу розробки було проведено пошук за попередньо визначеною сіткою гіперпараметрів, включаючи значення α коефіцієнта та L1-коефіцієнта. Це дозволило визначити оптимальну комбінацію гіперпараметрів, яка максимізувала прогностичну ефективність моделі.

Результати роботи моделі після навчання показали (табл. 3.11) що середнє RMSE становить 2.91, MAE 2.39 і R^2 рівний 0.85.

Таблиця 3.11 – Результати роботи моделі Elastic Net

RMSE	MAE	R^2
2.91	2.39	0.85

Використання 5-кратної перехресної перевірки дозволило нам оцінити стабільність та узагальнюваність моделі (табл. 3.12). Однак модель Elastic Net продемонструвала дещо нижчі показники ефективності порівняно з іншими методами, що використовувалися в подібних дослідженнях. Таку розбіжність

у результатах можна пояснити намаганням параметра регуляризації максимально узагальнити модель, що потенційно призводить до компромісу між складністю моделі та точністю прогнозування.

Таблиця 3.12 – Результати роботи моделі Elastic Net після 5-кратної перехресної перевірки

RMSE	MAE	R^2
3.46	2.33	0.77

Рисунок 3.15 показує ефективність моделі у прогнозуванні цільової змінної. Незважаючи на порівняно нижчу ефективність, модель Elastic Net надала цінну інформацію про фактори, що впливають на очікувану тривалість життя. Модель сприяє кращому розумінню динамічного взаємозв'язку між різноманітними факторами, що впливають на очікувану тривалість життя, підкреслюючи важливість конкретних характеристик та їхніх відповідних коефіцієнтів

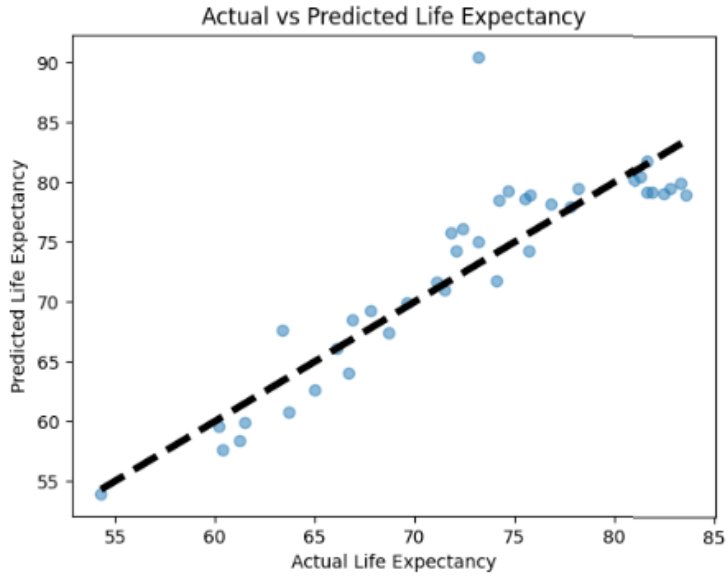


Рисунок 3.15 - Графік залежності фактичного і очікуваного значення моделі Elastic Net

3.3 Аналіз отриманих результатів

Провівши дослідження 5 різних моделей машинного навчання, можна скласти порівняльну таблицю (табл. 3.13), яка містить метрики кожної з моделей.

Таблиця 3.13 – Метрики всіх використаних моделей

Метрика	Лінійна регресія	SVR	RF	PCR	Elastic Net
RMSE	3.19	3.2	2.9	0.056	2.91
MAE	2.46	2.53	2.3	0.047	2.39
R^2	0.82	0.82	0.85	0.94	0.85

Також було проведено тестування моделей використовуючи 5-кратну перехресну перевірку, яка допомагає підтвердити стабільність моделі на різних підмножинах даних. Вона дозволяє використовувати всі наявні дані для тренування моделі, що може покращити її продуктивність. Результати наведені в таблиці 3.14.

Таблиця 3.14 – Метрики всіх використаних моделей після 5-кратної перехресної перевірки

Метрика	Лінійна регресія	SVR	RF	PCR	Elastic Net
RMSE	2.68	2.64	2.82	0.062	3.46
MAE	2.04	2.02	2.12	0.045	2.33
R^2	0.84	0.86	0.84	0.91	0.77

Усі використані моделі отримали показники RMSE, MAE та R^2, схожі до базової лінійної регресійної моделі.

Основними показниками високої очікуваної тривалості життя є вища освіта, мінімальна заробітна плата та кількість лікарів. На противагу цьому, індикаторами низької очікуваної тривалості життя є високий рівень народжуваності, рівень дитячої смертності та рівень фертильності. Хоча це

лише індикатори, а не причини (цей аналіз не може довести причинно-наслідковий зв'язок, лише кореляцію), індикаторами високої тривалості життя є показники багатших і розвиненіших країн.

Результати 5-кратних перехресних перевірок усіх методів виявилися подібними до відповідних 10-кратних перевірок, що свідчить про те, що моделі можна узагальнювати і вони можуть давати хороші результати з новими даними. Результати також вказують на те, що підхід до багатофакторного аналізу добре прогнозує очікувану тривалість життя, що робить моделі більш надійними. Вибрані ознаки також добре узгоджуються з відомими показниками очікуваної тривалості життя, включаючи вищу освіту та ресурси охорони здоров'я.

Варто відмітити, що найгірше себе проявила у роботі модель Elastic Net. Можемо припустити, що це пов’язано з тим, що цей метод потребує налаштування двох основних параметрів регуляризації α (параметр регуляризації) і l1_ratio (співвідношення між L1 і L2 регуляризацією). Якщо ці параметри не оптимізовані належним чином, модель може працювати гірше. Інші моделі, наприклад SVR і RF можуть бути більш стійкими до неправильних налаштувань параметрів. Також з причин можна виділити велику кількість ознак в датасеті, через ймовірний шум, дана модель може бути більш уразлива в порівнянні з іншими.

Найкращі ж результати показала модель PCR. Подальша робота має бути спрямована саме на неї та інші методи, що використовують PCA. Комбінація PCA і лінійної регресії в цьому методі дозволяє зменшувати розмірність даних, зберігаючи основні компоненти, які пояснюватимусь найбільшу частину варіації в даних. Зменшення розмірності може призвести до моделей, які краще узагальнюють нові дані, оскільки вони не перенавчаються на конкретні особливості тренувального набору даних.

Також варто відмітити, що вибір головних компонент дає змогу сфокусуватися на найбільш інформативних частинах даних і ігнорувати шум.

Завдяки цьому методу, було використано не всі колонки даних, які залишились після підготовки даних, а використати лише ті компоненти, які збирають найбільш важливі кореляції даних. Було взято 20 компонентів, які мають значно важливішу інформацію для побудови моделі і які пояснюють набор даних. Також була проведена хороша оптимізація роботи моделі і крос-валідація, що покращила ще більше результати.

3.4 Висновки до розділу 3

У даному розділі було показано більш детально процеси огляду і опису датасету, передобробку даних та побудови кожної з моделей машинного навчання. Дивлячись на метрики якості роботи моделей, не важко помітити, що більшість з моделей були обрані правильно, адже на виході отримали задовільну точність. Найгірше справилась модель Elastic Net. Однак звісно варто відмітити модель, яка спрацювала краще, а саме PCR. Ця модель краще за все себе показала через комбінацію PCA і лінійної регресії, що дозволяє зменшувати розмірність даних, зберігаючи основні компоненти, які пояснюватимуть найбільшу частину варіації в даних.

Для подальшого покращення продуктивності моделей машинного навчання можна розглянути збільшення об’єму вибірки, пошуки і дослідження нових більш сучасних алгоритмів для визначення найкращих параметрів моделей.

РОЗДІЛ 4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У цьому розділі буде проведено оцінку ключових характеристик майбутнього програмного продукту, що спеціалізується на дослідженні очікуваної тривалості життя.

Дана реалізація буде сприяти проведенню усіх необхідних досліджень, що дасть змогу якісно дослідити питання не лише в Україні, проте у всьому світі.

У даному дослідженні показано різні варіанти реалізації для забезпечення найбільш коректної та оптимальної стратегії вибору, що має вплив на економічні фактори та сумісність з майбутнім програмним продуктом. Для цього застосовувався апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) передбачає собою технологію, що дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. ФВА проводиться з метою виявлення резервів зниження витрат за рахунок ефективніших варіантів виробництва, кращого співвідношення між споживчою вартістю виробу та витратами на його виготовлення. Для проведення аналізу використовується економічна, технічна та конструкторська інформація.

Алгоритм функціонально-вартісного аналізу включає в себе визначення послідовності етапів розробки продукту, визначення повних витрат (річних) та кількості робочих часів, визначення джерел витрат та кінцевий розрахунок вартості програмного продукту.

4.1 Постановка задачі проектування

У роботі застосовується метод ФВА для проведення техніко-економічного аналізу розробки системи прогнозу стійкості фінансових

показників. Оскільки рішення стосовно проектування та реалізації компонентів, що розробляється, впливають на всю систему, кожна окрема підсистема має її задовольняти. Тому фактичний аналіз представляє собою аналіз функцій програмного продукту, призначеного для збору, обробки та проведення аналізу даних по компанії.

Технічні вимоги до програмного продукту є наступні:

- функціонування на персональних комп'ютерах із стандартним набором компонентів;
- зручність та зрозумілість для користувача;
- швидкість обробки даних та доступ до інформації в реальному часі;
- можливість зручного масштабування та обслуговування;
- мінімальні витрати на впровадження програмного продукту.

4.2 Обґрунтування функцій програмного продукту

Головна функція F_0 – розробка можливого програмного продукту, яка дозволяє аналізувати різні характеристики, що безпосередньо впливають на стійкість підприємства. Беручи за основу цю функцію, можна виділити наступні:

F_1 – вибір самої програми.

F_2 – якісний аналіз даних.

F_3 – графічні показники.

Кожна з цих функцій має декілька варіантів реалізації:

Функція F_1:

а) Python.

б) R.

Функція F_2.

а) Використання вбудованих бібліотек.

б) Власна реалізація функцій.

Функція F_3:

а) Використання шаблонних графіків.

б) Створення своїх.

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1).

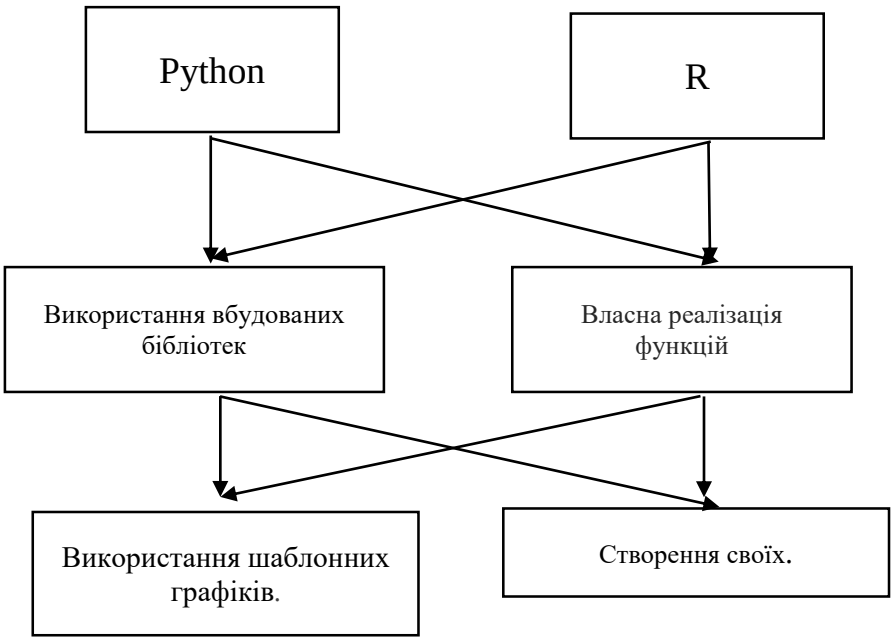


Рисунок 4.1 – Морфологічна карта

Морфологічна карта відображає множину всіх можливих варіантів основних функцій. Позитивно-негативна матриця показана в таблиці 4.1.

Таблиця 4.1 - Позитивно-негативна матриця

Функції	Варіанти реалізації	Переваги	Недоліки
F_1	*А*	Універсальність та ширше застосування	Менша специфічність для статистики та аналізу даних
	Б	Доступна в реалізації програма для різних обчислень	Важка інтеграція з іншими мовами програмування

Продовження таблиці 4.1

F_2	*А*	Простота написання програми	Обмежений спектр застосування
	Б	Підходить для будь-якої необхідної задачі	Складність в написанні програми
F_3	*А*	Загальноприйняті позначення	Обмежений спектр використання
	Б	При виконанні власних досліджень краще може передавати висновки	Необхідно достатньо багато часу для написання програми для побудови та знаходження всього необхідного в задачі

На основі аналізу отриманої матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція F_1: Перевагу даємо універсальності. Для спрощення роботи по написанню коду варіант Б має бути відкинутий.

Функція F_2*:* Перевага надається простоті написання програми, тобто варіанту із вбудованими бібліотеками, для спрощення написання коду варіант Б не був розглянутий.

Функція F_3*:* Реалізація першого варіанту є сприйнятливою для програми. Це варіант А.

Таким чином, будемо розглядати такий варіанти реалізації ПП:

$$F_1\text{а} - F_2\text{а} - F_3\text{а}$$

$$F_1\text{а} - F_2\text{а} - F_3\text{б}$$

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обґрунтування системи параметрів програмного продукту

На основі даних, розглянутих вище, визначаються основні параметри вибору, які будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

- *X1* – швидкодія мови програмування;
- *X2* – об’єм пам’яті для обчислень та збереження даних;
- *X3* – час навчання даних;
- *X4* – потенційний об’єм програмного коду.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію програмного продукту, як показано у таблиці 4.2.

Таблиця 4.2 - Основні параметри програмного продукту

Назва Параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	оп/мс	60	80	110
Об’єм пам’яті	X2	Мб	60	50	30
Час попередньої обробки даних	X3	мс	80	70	60
Потенційний об’єм програмного коду	X4	кількість рядків коду	35	25	20

За даними таблиці 4.3 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

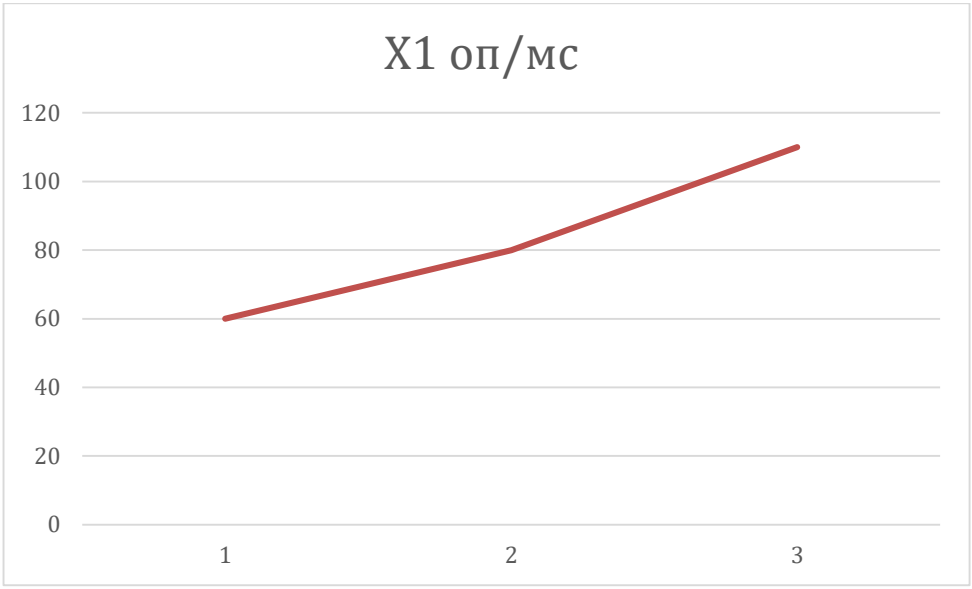


Рисунок 4.2 – X1, швидкодія мови програмування

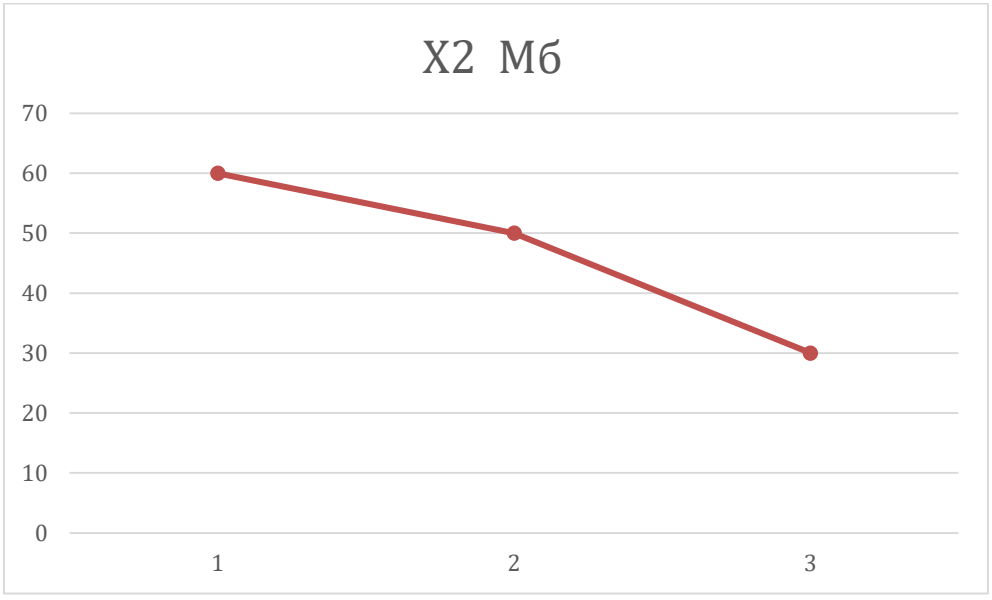


Рисунок 4.3 – X2, об’єм пам’яті

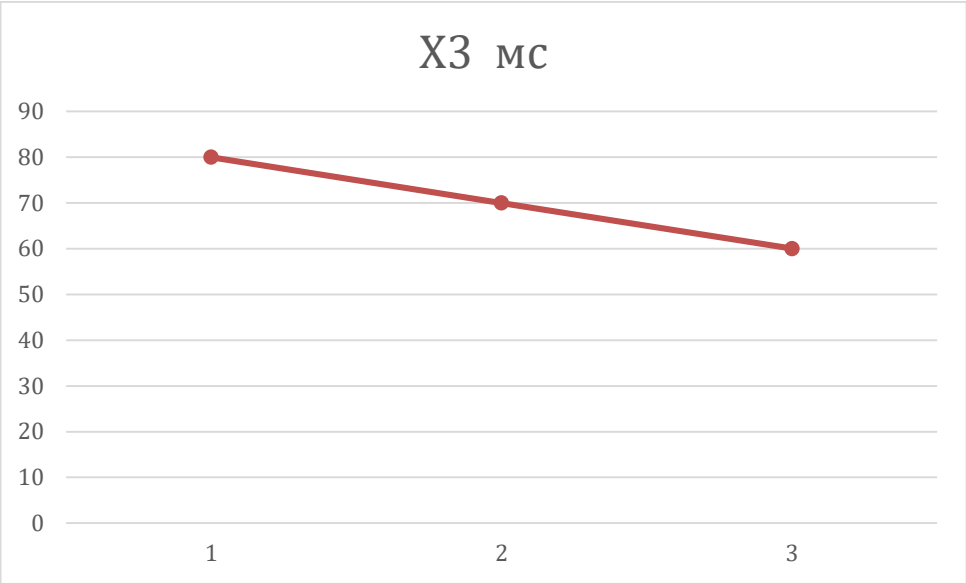


Рисунок 4.4 – X3, час попередньої обробки даних

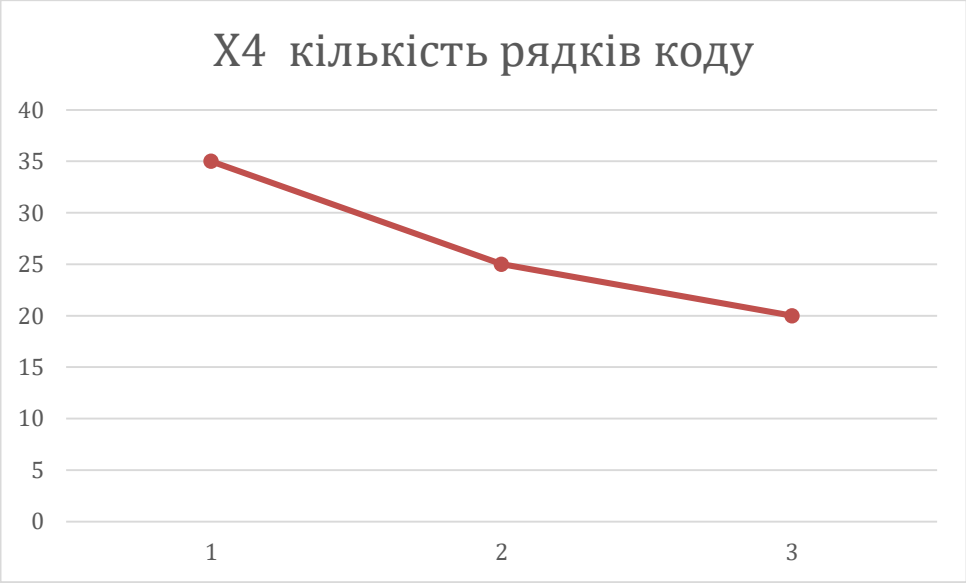


Рисунок 4.5 – X4, потенційний об’єм програмного коду

4.4 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 4.3.

Таблиця 4.3 - Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	3	3	4	3	3	4	3	23	5,5	30,25
X2	Об’єм пам’яті	Mb	1	2	2	1	2	3	2	13	-4,5	20,25
X3	Час попередньої обробки даних	мс	4	4	3	4	4	2	4	25	7,5	56,25
X4	Потенційний об’єм програмного коду	Кількість рядків коду	2	1	1	2	1	1	1	9	-8,5	72,25
	Разом		10	10	10	10	10	10	10	70	0	179

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70, \quad (4.1)$$

де N – число експертів,

n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17{,}5 \quad (4.2)$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T. \quad (4.3)$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 179. \quad (4.4)$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 179}{7^2(4^3 - 4)} = 0{,}73 > W_k = 0{,}67. \quad (4.5)$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 - Попарне порівняння параметрів.

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	>	>	>	>	>	>	>	>	1,5
X1 і X3	<	<	>	<	<	>	<	<	0,5
X1 і X4	>	>	>	>	>	>	>	>	1,5
X2 і X3	<	<	<	<	<	>	<	>	0,5
X2 і X4	<	>	<	<	>	<	<	<	1,5
X3 і X4	<	<	<	<	<	<	<	<	1,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases}. \quad (4.6)$$

З отриманих числових оцінок переваги складемо матрицю А=$\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i} \quad (4.7)$$

$$b_i = \sum_{i=1}^{N} a_{ij} \quad (4.8)$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \tag{4.9}$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j \tag{4.10}$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 - Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1	1,5	0,5	1,5	4,5	0,28	16,25	0,27	59,125	0,27
X2	0,5	1	0,5	1,5	3,5	0,22	12,25	0,21	44,875	0,21
X3	1,5	1,5	1	1,5	5,5	0,34	21,25	0,36	77,875	0,36
X4	1,5	1,5	1,5	1	2,5	0,16	9,25	0,16	34,125	0,16
Всього:					16	1	59	1	216	1

4.5 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2* (Об’єм пам’яті), *X3* (час попередньої обробки даних) та *X4* (потенційний об’єм програмного коду) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X1* (швидкість роботи мови програмування) обрано не найгіршим.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j}\,, \qquad (4.11)$$

де n – кількість параметрів;

$K_{вi}$– коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 4.6 - Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	А	Х1	100	27	0,27	7,29
F3	А	Х2	90	24	0,21	5,04
	Б	Х3	30	20	0,36	7,2
F4	А	Х4	28	29	0,16	4,64

За даними з таблиці 4.6 за формулою:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}], \qquad (4.12)$$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 7{,}29 + 5{,}04 + 4{,}46 = 16{,}79\,;$$

$$K_{K2} = 7{,}29 + 7{,}2 + 4{,}46 = 18{,}95.$$

Як видно з розрахунків, кращим є 2 варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.6 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як:

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М}, \quad (4.13)$$

де T_P – трудомісткість розробки ПП;

$K_П$ – поправочний коефіцієнт;

$K_{СК}$ – коефіцієнт на складність вхідної інформації;

K_M – коефіцієнт рівня мови програмування;

$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;

$K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму

1, трудомісткість дорівнює: $T_Р$ =39 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $К_П$ = 1,9. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $К_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $К_{СТ}$ = 0,9. Тоді загальна трудомісткість програмування першого завдання дорівнює:

$$T_1 = 39 \cdot 1{,}9 \cdot 0{,}7 = 51{,}87 \text{ людино-днів.}$$

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $T_Р$ =27 людино-днів, $К_П$ =0,9, $К_{СК}$ = 1, $К_{СТ}$ =0.8:

$$T_2 = 27 \cdot 0{,}9 \cdot 0{,}8 = 19{,}44 \text{ людино-днів.}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$$T_I = (51{,}87 + 19{,}44 + 4{,}8 + 19{,}44) \cdot 8 = 764{,}4 \text{ людино-годин.}$$

$$T_{II} = (51{,}87 + 19{,}44 + 6{,}91 + 19{,}44) \cdot 8 = 781{,}28 \text{ людино-годин.}$$

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 22000 грн., один аналітик в області даних з окладом 25000. Визначимо середню зарплату за годину за формулою:

$$С_ч = \frac{М}{T_m \cdot t} грн., \quad (4.14)$$

де М – місячний оклад працівників;

T_m – кількість робочих днів тиждень;

t – кількість робочих годин в день.

$$С_ч = \frac{22000 + 22000 + 25000}{3 \cdot 21 \cdot 8} = 136{,}16 \ \ грн. \quad (4.15)$$

Тоді, розрахуємо заробітну плату за формулою:

$$С_{ЗП} = С_ч \cdot Т_i \cdot К_Д, \quad (4.16)$$

де $С_ч$– величина погодинної оплати праці програміста;

$Т_i$ – трудомісткість відповідного завдання;

$К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП} = 136{,}9 \cdot 764{,}4 \cdot 1.2 = 125575{,}63$ грн.

II. $С_{ЗП} = 136{,}9 \cdot 781{,}28 \cdot 1.2 = 128348{,}68$ грн.

Відрахування на єдиний соціальний внесок становить 22%:

I. $С_{ВІД} = С_{ЗП} \cdot 0{,}22 = 125575{,}63 \cdot 0{,}22 = 27626{,}6$ грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0{,}22 = 128348{,}68 \cdot 0{,}22 = 28236{,}7$ грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 22000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 22000 \cdot 0,2 = 52800 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 52800 \cdot (1 + 0.2) = 63360 \text{ грн.}$$

Відрахування на соціальний внесок:

$$С_{ВІД} = С_{ЗП} \cdot 0.22 = 63360 \cdot 0,22 = 13939,2 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 18000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.4 \cdot 0.25 \cdot 18000 = 6300 \text{ грн.,}$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.4 \cdot 18000 \cdot 0.08 = 2016 \text{ грн.,}$$

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 12 – 16) \cdot 8 \cdot 0,4 =$$

$$= 745,6 \text{ години,}$$

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 745,6 \cdot 0,2 \cdot 0,3 \cdot 5,23 = 233,96 \text{ грн.,}$$

де $N_С$ – середньо-споживча потужність приладу;

$К_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 18000 \cdot 0,67 = 12060 \text{ грн.}$$

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н, \qquad (4.17)$$

$С_{ЕКС}$ = 63360 + 13939,2 + 6300 + 2016 + 233,96 + 12060 = 97909,16 грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$C_{М\text{-}Г} = C_{ЕКС} / T_{ЕФ} = 97909{,}16 / 745{,}6 = 131{,}31 \text{ грн/год.}$$

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$C_М = C_{М-Г} \cdot T, \quad (4.18)$$

I. $C_М = 131{,}31 \cdot 764{,}4 = 100373{,}36$ грн.

II. $C_М = 131{,}31 \cdot 781{,}28 = 102589{,}88$ грн.

Накладні витрати складають 67% від заробітної плати:

$$C_Н = C_{ЗП} \cdot 0{,}67, \quad (4.19)$$

I. $C_Н = 125575{,}63 \cdot 0{,}67 = 84135{,}67$ грн.

II. $C_Н = 128348{,}68 \cdot 0{,}67 = 85993{,}62$ грн.

Отже, вартість розробки ПП за варіантами становить:

$$C_{ПП} = C_{ЗП} + C_{ВІД} + C_М + C_Н, \quad (4.20)$$

I. $C_{ПП} = 125575{,}63 + 27626{,}6 + 100373{,}36 + 84135{,}6 = 337711{,}19$ грн.

II. $C_{ПП} = 128348{,}68 + 28236{,}7 + 102589{,}88 + 85993{,}62 = 345168{,}88$ грн.

4.7 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕР j} = К_{Кj} / С_{Фj}, \quad (4.21)$$

$$К_{ТЕР1} = 16,79 / 337711,19 = 4,9717 \cdot 10^{-5},$$

$$К_{ТЕР2} = 18,95 / 345168,88 = 5,49 \cdot 10^{-5}.$$

Як бачимо, найбільш ефективним є другий варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР2} = 5,49 \cdot 10^{-5}$.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є другий варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР2} = 5,49 \cdot 10^{-5}$.

Цей варіант реалізації програмного продукту має такі параметри:

– вибір програмного продукту – Python;

– реалізація важливої постановки з допомогою вбудованих функцій;

– використання стандартного інтерфейсу для побудови значень.

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, швидку реалізацію програми та доступний функціонал для роботи.

4.8 Висновки до розділу 4

В даному розділі було проведено повний функціонально-вартісний аналіз програмного продукту. Також було знайдено оцінку основних функцій програмного продукту.

В результаті виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, було визначено та проведено оцінку основних функцій програмного продукту, а також знайдено параметри, які його характеризують.

На основі аналізу вибрано варіант реалізації програмного продукту.

ВИСНОВКИ

В ході виконання дипломної роботи було проведено дослідження очікуваної тривалості життя населення за допомогою методів машинного навчання. Дане дослідження є дуже актуальним, адже показник очікуваної тривалості життя є одним з найважливіших аспектів демографічної науки і має неабиякий соціально-економічний вплив. Саме цей показник є індикатором здоров'я нації і її системи охорони здоров'я, відображає загальний рівень медичної допомоги, займає важливе місце при розрахунку якості життя та соціально-економічних показників. Саме дослідження очікуваної тривалості життя несе важливе значення для урядів країн та всесвітніх організацій, адже дозволяє краще зрозуміти потреби населення, дозволяє прогнозувати розвиток соціальних і економічних програм, а також допомагає у розробці різноманітних стратегій для покращення показника якості життя.

Метою дипломної роботи було розробити оптимальні моделі машинного навчання, які зможуть опановувати дані щодо очікуваної тривалості життя населення, а також допомагати знаходити закономірності, що висвітлювали б найважливіші фактори, які впливають на цей показник.

Для досягнення заданої мети спершу було детально досліджено актуальність даної проблеми, визначено поняття очікуваної тривалості життя, а також виконано пошук і ознайомлення з відповідною літературою.

Після цього було детально вивчено проблему збору даних і їх попередню обробку задля того, щоб розробити ефективні і якісні моделі. Потім було досліджено основні етапи розробки моделей машинного навчання.

В другому розділі було приділено увагу огляду існуючих методів машинного навчання, вибору моделей і методів, які допоможуть у дослідженні, а також ознайомлення з основними інструментами роботи.

Виділивши основні поняття машинного навчання, було детально розібрано п'ять методів машинного навчання, серед яких: лінійна регресія,

метод опорних векторів для задач регресії, random forest, elastic net та principal component regression. Більш того, виділено основні метрики якості роботи моделей, які допомагають у розробці, як-от: середня абсолютна помилка, середньоквадратична помилка прогнозу та коефіцієнт детермінації.

Серед засобів для програмного продукту було обрано мову програмування Python 3, а також декілька важливих бібліотек, наприклад scikit-learn, seaborn, numpy і pandas.

В третьому розділі перш за все, було проведено пошук та вибір відповідних даних. Було обрано набір даних із відкритого джерела на сайті Kaggle, що містить велику кількість інформації про країни світу за 2023 рік, що охоплює широкий спектр показників та атрибутів. Було отримано набір даних у яких є 195 записів і 25 стовпців. Після вибору датасету, було розглянуто дані, проведена передобробка даних, наведені основні статистики і визначення кореляцій між даними.

Найважливішим етапом роботи було розробка моделей та аналіз отриманих результатів. Аналізуючи метрики якості роботи моделей, можна побачити, що більшість з них були обрані правильно, оскільки результати показують задовільну точність. З точки зору порівняння розроблених п’яти методів, найгірші результати показала модель Elastic Net. Водночас, найкраще себе зарекомендувала модель PCR. Її успішність пояснюється комбінацією PCA та лінійної регресії, що дозволяє зменшити розмірність даних, зберігаючи основні компоненти, які пояснюють найбільшу частку варіації в даних.

Основними показниками високої очікуваної тривалості життя є вища освіта, мінімальна заробітна плата та кількість лікарів. На противагу цьому, індикаторами низької очікуваної тривалості життя є високий рівень народжуваності, рівень дитячої смертності та рівень фертильності.

Додатковим до основного дослідження став функціонально-вартісний аналіз програмного продукту. Було визначено та проведено оцінку основних

функцій програмного продукту, а також знайдено параметри, які його характеризують. На основі результатів аналізу було вибрано найкращий варіант реалізації програмного продукту

У майбутньому для покращення продуктивності моделей машинного навчання варто розглянути збільшення обсягу вибірки, а також пошук і дослідження нових сучасних алгоритмів для визначення оптимальних параметрів моделей.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Life expectancy. *Wikipedia*. URL: https://en.wikipedia.org/wiki/Life_expectancy (дата звернення: 16.04.2024).
2. Державна служба статистики (Держстат). URL: https://ukrstat.gov.ua/gend_rivnist/metadata_gr/06/meta/6.1.pdf (дата звернення: 16.04.2024).
3. HDI (Human Development Index) for United Nations. URL: https://hdr.undp.org/data-center/human-development-index#/indicies/HDI (дата звернення: 18.04.2024).
4. World Health Organization – Indicator Metadata Registry List – Life expectancy at birth (years). URL: https://www.who.int/data/gho/indicator-metadata-registry/imr-details/65 (дата звернення: 18.04.2024).
5. Glei DA, Meslé F, Vallin J. National Research Council; Panel on Understanding Divergent Trends in Longevity in High-Income Countries; Committee on Population; Division of Behavioral and Social Sciences and Education. Diverging trends in life expectancy at age 50: A look at causes of death. In: Crimmins EM, Preston SH, Cohen B, editors. International Differences in Mortality at Older Ages: Dimensions and Sources. Washington, DC: The National Academies Press; 2010. pp. 17–67.
6. Life expectancy vs. GDP per capita, 2021 by Our World in Data. URL: https://ourworldindata.org/grapher/life-expectancy-vs-gdp-per-capita
7. Andreas Bergh, Therese Nilsson. Good for Living? On the Relation between Globalization and Life Expectancy by Andreas Bergh and Therese Nilsson. 2010. DOI: https://doi.org/10.1016/j.worlddev.2010.02.020
8. Sangwon Chae,Sungjun Kwon, Donghyun Lee. Predicting Infectious Disease Using Deep Learning and Big Data. International Journal Environ. Res. Public Health, 2018 Aug; 15(8), 1596. DOI: 10.3390/ijerph15081596

9. Яка середня тривалість життя в Україні. Дані Держстату. *Суспільне.* URL: https://suspilne.media/133692-aka-seredna-trivalist-zitta-v-ukraini-dani-derzstatu/ (дата звернення 19.04.2024).
10. СТРАТЕГІЯ демографічного розвитку України на період до 2040 року. *Міністерство Соціальної Політики України.* URL: https://www.msp.gov.ua/projects/870/ (дата звернення 19.04.2024).
11. На 32% менше дітей народилося у 2023 році в Україні. *Опендатабот.* URL: https://opendatabot.ua/analytics/newborn-2023 (дата звернення 20.04.2024).
12. Hadeel S. Obaid; Saad Ahmed Dheyab; Sana Sabah Sabry. The Impact of Data Pre-Processing Techniques and Dimensionality Reduction on the Accuracy of Machine Learning. *Published in 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON) on 13-15 March 2019*. DOI: 10.1109/IEMECONX.2019.8877011
13. Carlos Vladimiro Gonzalez Zelaya, Towards Explaining the Effects of Data Preprocessing on Machine Learning. *Published in 2019 IEEE 35th International Conference on Data Engineering (ICDE) on 08-11 April 2019.* DOI: 10.1109/ICDE.2019.00245
14. Недашківська Н. І. Інтелектуальний Аналіз Даних: практикум [Електронний ресурс], Київ, КПІ, 2021, С. 3-5. URL: https://ela.kpi.ua/server/api/core/bitstreams/13204a0c-2996-4d2a-959d-9e9f2a3895d6/content
15. Мюллер А., Гвидо С. Введение в машинное обучение с помощью Python. М.: O'Reilly Media, 2017. 392 с.
16. Liniar regression. *Wikipedia.* URL: https://en.wikipedia.org/wiki/Linear_regression (дата звернення: 24.04.2024).

17. Support Vector Regression Tutorial for Machine Learning. *Analytics Vidhya*. URL: https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/ (дата звернення: 24.04.2024).
18. Random Forest Regression – How it Helps in Predictive Analytics? *Analytixlabs*. URL: https://www.analytixlabs.co.in/blog/random-forest-regression/ (дата звернення 25.04.2024).
19. Principal Component Regression — Clearly Explained and Implemented. *Towards Data Science*. URL: https://towardsdatascience.com/principal-component-regression-clearly-explained-and-implemented-608471530a2f (дата звернення 26.04.2024).
20. C. De Mol, E. De Vito, L. Rosasco Elastic-Net Regularization in Learning Theory. Cornell University. DOI: https://doi.org/10.48550/arXiv.0807.3423
21. Chugh A. MAE, MSE, RMSE, Coefficient of Determination, Adjusted R Squared — Which Metric is Better? Dec 8, 2020. URL: https://medium.com/analytics-vidhya/mae-mse-rmse-coefficient-ofdetermination-adjusted-r-squared-which-metric-is-better-cd0326a5697e
22. Mean absolute error. *Wikipedia*. URL: https://en.wikipedia.org/wiki/Mean_absolute_error (дата звернення: 26.04.2024).
23. Root mean square error. *Wikipedia*. URL: https://en.wikipedia.org/wiki/Root_mean_square_deviation (дата звернення: 26.04.2024).
24. Coefficient of determination. *Wikipedia*. URL: https://en.wikipedia.org/wiki/Coefficient_of_determination (дата звернення 26.03.2024).
25. Kumar A. Mean Squared Error or R-Squared – Which one to use? April 4, 2023. URL: https://vitalflux.com/mean-square-error-r-squared-which-one-to-use/
26. Python 3. *Python*. URL: https://www.python.org/download/releases/3.0/ (дата звернення 27.03.2024).

27. Python. *Wikipedia*. URL: https://uk.wikipedia.org/wiki/Python дата звернення 27.03.2024).

28. Датасет: https://www.kaggle.com/datasets/nelgiriyewithana/countries-of-the-world-2023

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

```
#Load Original
import pandas as pd
url = 'https://raw.githubusercontent.com/liyuhao413/filepit/main/world-data-2023.csv'
df = pd.read_csv(url)
df
#Checking Missing Data
# Check for missing values
missing_values = df.isnull().sum()
# Display the missing values for each column
print(missing_values)
# Columns to convert to float
columns_to_convert = ['Density\n(P/Km2)', 'Agricultural Land( %)', 'Land Area(Km2)',
                      'Armed Forces size', 'Birth Rate', 'Co2-Emissions', 'Forested Area (%)',
                      'CPI', 'CPI Change (%)', 'Fertility Rate', 'Gasoline Price', 'GDP',
                      'Gross primary education enrollment (%)', 'Gross tertiary education enrollment
(%)',
                      'Infant mortality', 'Life expectancy', 'Maternal mortality ratio', 'Minimum wage',
                      'Out of pocket health expenditure', 'Physicians per thousand', 'Population',
                      'Population: Labor force participation (%)', 'Tax revenue (%)', 'Total tax rate',
                      'Unemployment rate', 'Urban_population', 'Latitude', 'Longitude']
# Convert columns using a lambda function
df[columns_to_convert] = df[columns_to_convert].applymap(lambda x: float(str(x).replace(',',
'').replace('$', '').replace('%', '')))
# List of columns with missing values
columns_with_missing = df.columns[df.isnull().any()]
# Impute numerical columns with mean
numerical_columns = df.select_dtypes(include=['float64'])
numerical_columns = numerical_columns.columns[numerical_columns.isnull().any()]
df[numerical_columns] = df[numerical_columns].fillna(df[numerical_columns].mean())
# Impute categorical columns with mode
categorical_columns = df.select_dtypes(include=['object'])
categorical_columns = categorical_columns.columns[categorical_columns.isnull().any()]
df[categorical_columns] =
df[categorical_columns].fillna(df[categorical_columns].mode().iloc[0])

# Verify if all missing values are handled
missing_counts = df.isnull().sum()
print(missing_counts)
#Remove Unnecessary Columns that is unrelated to life expectancy
columns_to_drop = ['Abbreviation', 'Calling Code', 'Capital/Major City',
                   'Currency-Code', 'Largest city', 'Official language',
                   ]
# Drop the identified columns from the DataFrame
df = df.drop(columns=columns_to_drop)
# Display the first few rows to confirm the drop
df.head()
import seaborn as sns
import matplotlib.pyplot as plt
# Check the distribution of the life expectancy variable
```

```
sns.histplot(df['Life expectancy'], kde=True)
plt.title('Distribution of Life Expectancy')
plt.xlabel('Life Expectancy')
plt.ylabel('Frequency')
plt.show()
import numpy as np
# Select only the numeric columns for correlation
numeric_df = df.select_dtypes(include=[np.number])
# Calculate the correlation matrix
correlation_matrix = numeric_df.corr()
# Consider only the correlations with 'Life expectancy'
life_expectancy_correlations = correlation_matrix['Life
expectancy'].sort_values(ascending=False)
print(life_expectancy_correlations)
# Visualize the correlations with a heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
#focus on a correlation above 0.5
from sklearn.model_selection import train_test_split
# Function to select features based on correlation threshold
def select_features_by_correlation(dataframe, target, threshold=0.5):
    numeric_df = df.select_dtypes(include=[np.number])
    # Calculate the correlation matrix
    correlation_matrix = numeric_df.corr()
    # Find features with correlation above the threshold
    high_correlation = correlation_matrix[target][abs(correlation_matrix[target]) > threshold]
    selected_features = high_correlation.index.drop(target).tolist()
    return selected_features
# Define the correlation threshold
correlation_threshold = 0.3
# Use the function to select features
selected_features = select_features_by_correlation(df, 'Life expectancy', correlation_threshold)
# Create the feature matrix (X) and target variable (y)
X = df[selected_features]
y = df['Life expectancy']
# Split the data into training and testing sets (using an 80-20 split)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Output the selected features
print(f"Selected features with correlation threshold > {correlation_threshold}:")
print(selected_features)
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.metrics import mean_absolute_error
# Initialize the Linear Regression model
lin_reg = LinearRegression()
# Fit the model on the training data
lin_reg.fit(X_train, y_train)
# Predict life expectancy on the test data
y_pred = lin_reg.predict(X_test)
# Evaluate the model using R-squared, RMSE, and MAE metrics
```

```
r2 = r2_score(y_test, y_pred)
rmse = mean_squared_error(y_test, y_pred, squared=False)
mae = mean_absolute_error(y_test, y_pred)

print(f'R-squared (R2): {r2:.2f}')
print(f'Root Mean Squared Error (RMSE): {rmse:.2f}')
print(f'Mean Absolute Error (MAE): {mae:.2f}')
# Plotting actual vs predicted values
plt.scatter(y_test, y_pred, alpha=0.5)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=4)
plt.xlabel('Actual Life Expectancy')
plt.ylabel('Predicted Life Expectancy')
plt.title('Actual vs Predicted Life Expectancy')
plt.show()
# Plot residuals
residuals = y_test - y_pred
plt.scatter(y_pred, residuals, alpha=0.5)
plt.title('Residuals vs Predicted')
plt.xlabel('Predicted Life Expectancy')
plt.ylabel('Residuals')
plt.axhline(y=0, color='red', linestyle='--')
plt.show()
from sklearn.model_selection import cross_val_score
from sklearn.metrics import make_scorer, r2_score, mean_absolute_error
# Define the number of folds for the cross-validation
k = 10
# Initialize the Linear Regression model
lin_reg = LinearRegression()
# Create scorers for RMSE, MAE, and R-squared
scorer_rmse = make_scorer(lambda y_true, y_pred: mean_squared_error(y_true, y_pred,
squared=False))
scorer_mae = make_scorer(mean_absolute_error)
scorer_r2 = make_scorer(r2_score)
# Perform k-fold cross-validation for each metric
cv_scores_rmse = cross_val_score(lin_reg, X, y, cv=k, scoring=scorer_rmse)
cv_scores_mae = cross_val_score(lin_reg, X, y, cv=k, scoring=scorer_mae)
cv_scores_r2 = cross_val_score(lin_reg, X, y, cv=k, scoring=scorer_r2)

# Calculate the mean and standard deviation for each metric
mean_rmse = cv_scores_rmse.mean()
std_rmse = cv_scores_rmse.std()
mean_mae = cv_scores_mae.mean()
std_mae = cv_scores_mae.std()
mean_r2 = cv_scores_r2.mean()
std_r2 = cv_scores_r2.std()
# Print out the results
print(f'RMSE: {mean_rmse:.2f} (std: {std_rmse:.2f})')
print(f'MAE: {mean_mae:.2f} (std: {std_mae:.2f})')
print(f'R-squared: {mean_r2:.2f} (std: {std_r2:.2f})')
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error
```

```
from sklearn.model_selection import train_test_split
# Using the provided function structure and selected features
def select_features_by_correlation(dataframe, target, threshold=0.3):
    numeric_df = dataframe.select_dtypes(include=[np.number])
    correlation_matrix = numeric_df.corr()
    high_correlation = correlation_matrix[target][abs(correlation_matrix[target]) > threshold]
    selected_features = high_correlation.index.drop(target).tolist()
    return selected_features
# Selecting features based on the correlation threshold of 0.3
selected_features = select_features_by_correlation(df, 'Life expectancy', 0.3)
# Create the feature matrix (X) and target variable (y)
X = df[selected_features]
y = df['Life expectancy']
# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Scaling the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
# Initialize and train the Support Vector Regression model with a linear kernel
svr_model = SVR(kernel='linear')
svr_model.fit(X_train_scaled, y_train)
# Predicting and evaluating the model using RMSE
y_pred = svr_model.predict(X_test_scaled)
rmse = mean_squared_error(y_test, y_pred, squared=False)
# Output the selected features and RMSE
selected_features, rmse
from sklearn.metrics import r2_score, mean_absolute_error
# Function to train and evaluate the model using different kernels
def train_and_evaluate_svr(X_train, X_test, y_train, y_test, kernels):
    results = {}
    for kernel in kernels:
        svr = SVR(kernel=kernel)
        svr.fit(X_train, y_train)
        y_pred = svr.predict(X_test)
        # Calculate evaluation metrics
        rmse = mean_squared_error(y_test, y_pred, squared=False)
        r2 = r2_score(y_test, y_pred)
        mae = mean_absolute_error(y_test, y_pred)
        results[kernel] = {'RMSE': rmse, 'R2': r2, 'MAE': mae}
    return results
# Kernels to test
kernels = ['linear', 'poly', 'rbf', 'sigmoid']
# Train and evaluate the model using different kernels
svr_results = train_and_evaluate_svr(X_train_scaled, X_test_scaled, y_train, y_test, kernels)
svr_results
import matplotlib.pyplot as plt
# Predict using the best performing model (linear kernel)
best_svr = SVR(kernel='linear')
best_svr.fit(X_train_scaled, y_train)
y_pred_linear = best_svr.predict(X_test_scaled)
# Calculating residuals
```

```
residuals = y_test - y_pred_linear
# Plotting Actual vs. Predicted
plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
plt.scatter(y_test, y_pred_linear, alpha=0.75, color='blue')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Actual vs. Predicted')
# Plotting Residuals
plt.subplot(1, 2, 2)
plt.scatter(y_pred_linear, residuals, alpha=0.75, color='red')
plt.hlines(y=0, xmin=y_pred_linear.min(), xmax=y_pred_linear.max(), linestyles='dashed')
plt.xlabel('Predicted')
plt.ylabel('Residuals')
plt.title('Residuals Plot')
plt.tight_layout()
plt.show()
from sklearn.model_selection import cross_val_score
from sklearn.metrics import make_scorer, mean_squared_error, mean_absolute_error, r2_score
from sklearn.svm import SVR
# Define the number of folds for the cross-validation
k = 5
# Initialize the Support Vector Regression model with linear kernel
svr_linear = SVR(kernel='linear')
# Scaling all data since SVR is sensitive to unscaled data
X_scaled = scaler.fit_transform(X)  # Using all data for cross-validation, thus scaling all at once
# Create scorers for RMSE, MAE, and R-squared
scorer_rmse = make_scorer(lambda y_true, y_pred: mean_squared_error(y_true, y_pred,
squared=False))
scorer_mae = make_scorer(mean_absolute_error)
scorer_r2 = make_scorer(r2_score)
# Perform k-fold cross-validation for each metric
cv_scores_rmse = cross_val_score(svr_linear, X_scaled, y, cv=k, scoring=scorer_rmse)
cv_scores_mae = cross_val_score(svr_linear, X_scaled, y, cv=k, scoring=scorer_mae)
cv_scores_r2 = cross_val_score(svr_linear, X_scaled, y, cv=k, scoring=scorer_r2)
# Calculate the mean and standard deviation for each metric
mean_rmse = cv_scores_rmse.mean()
std_rmse = cv_scores_rmse.std()
mean_mae = cv_scores_mae.mean()
std_mae = cv_scores_mae.std()
mean_r2 = cv_scores_r2.mean()
std_r2 = cv_scores_r2.std()
# Output the results
mean_rmse, mean_mae, mean_r2,
# Plotting actual vs predicted values
plt.scatter(y_test, y_pred, alpha=0.5)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=4)
plt.xlabel('Actual Life Expectancy')
plt.ylabel('Predicted Life Expectancy')
plt.title('Actual vs Predicted Life Expectancy')
```

```
plt.show()
# Plot residuals
residuals = y_test - y_pred
plt.scatter(y_pred, residuals, alpha=0.5)
plt.title('Residuals vs Predicted')
plt.xlabel('Predicted Life Expectancy')
plt.ylabel('Residuals')
plt.axhline(y=0, color='red', linestyle='--')
plt.show()
# Selecting top features based on the importance scores and correlation analysis
top_features = ['Infant mortality', 'Maternal mortality ratio', 'Birth Rate', 'Fertility Rate',
                'Gross tertiary education enrollment (%)', 'Physicians per thousand', 'Minimum wage']
X_train_top_features = X_train_full[top_features]
# K-Fold Cross-Validation setup
kf = KFold(n_splits=5, shuffle=True, random_state=42)
fold_rmse = []
fold_mae = []
fold_r2 = []
# Perform k-fold cross-validation manually to calculate RMSE for each fold
for train_index, valid_index in kf.split(X_train_top_features):
    X_train, X_valid = X_train_top_features.iloc[train_index],
X_train_top_features.iloc[valid_index]
    y_train, y_valid = y_train_full.iloc[train_index], y_train_full.iloc[valid_index]
    # Train the model
    model = RandomForestRegressor(n_estimators=100, random_state=42)
    model.fit(X_train, y_train)
    # Predict on validation set
    y_pred_valid = model.predict(X_valid)
    rmse = np.sqrt(mean_squared_error(y_valid, y_pred_valid))
    mae = mean_absolute_error(y_valid, y_pred_valid)
    r2 = r2_score(y_valid, y_pred_valid)
    fold_rmse.append(rmse)
    fold_mae.append(mae)
    fold_r2.append(r2)
# Print RMSE, MAE, and R2 for each fold
print("RMSE for each fold:", fold_rmse)
print("Mean RMSE across all folds:", np.mean(fold_rmse))
print("Standard Deviation of RMSE across folds:", np.std(fold_rmse))
print("MAE for each fold:", fold_mae)
print("Mean MAE across all folds:", np.mean(fold_mae))
print("Standard Deviation of MAE across folds:", np.std(fold_mae))
print("R2 for each fold:", fold_r2)
print("Mean R2 across all folds:", np.mean(fold_r2))
print("Standard Deviation of R2 across folds:", np.std(fold_r2))
# Hyperparameter tuning with GridSearchCV using K-Fold Cross-Validation on the selected
features
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10]
}
rf = RandomForestRegressor(random_state=42)
```

```
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=kf,
scoring='neg_mean_squared_error', verbose=2, n_jobs=-1)
grid_search.fit(X_train_top_features, y_train_full)
# Evaluate the best model from GridSearchCV on the unseen test set
best_model = grid_search.best_estimator_
X_test_top_features = X_test[top_features]
y_pred_test = best_model.predict(X_test_top_features)
# Calculate RMSE, MAE, and R2 for the test data
rmse_test = np.sqrt(mean_squared_error(y_test, y_pred_test))
mae_test = mean_absolute_error(y_test, y_pred_test)
r2_test = r2_score(y_test, y_pred_test)
# Print the results for the test data
print(f"Test RMSE: {rmse_test}")
print(f"Test MAE: {mae_test}")
print(f"Test R2 Score: {r2_test}")
import matplotlib.pyplot as plt
# Plotting Actual vs Predicted values
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
plt.scatter(y_test, y_pred_test, alpha=0.5, color='blue', label='Actual vs Predicted')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2, label='Perfect
Prediction')
plt.xlabel('Actual Life Expectancy')
plt.ylabel('Predicted Life Expectancy')
plt.title('Actual vs Predicted Life Expectancy')
# Calculate residuals
residuals = y_test - y_pred_test
# Plotting residuals
plt.subplot(1, 2, 2)
plt.scatter(y_pred_test, residuals, color='red', alpha=0.5, label='Residuals')
plt.axhline(y=0, color='black', linestyle='--')
plt.xlabel('Predicted Life Expectancy')
plt.ylabel('Residuals')
plt.title('Residuals of Predicted Life Expectancy')
plt.legend()
plt.grid(True)
plt.show()
from sklearn.decomposition import PCA
import numpy as np

# Initializing PCA
pca = PCA()
# Applying PCA to the scaled numeric data
pca.fit(data_cleaned[numeric_cols])
# Explained variance ratios
explained_variance_ratio = pca.explained_variance_ratio_
# Cumulative explained variance
cumulative_variance = np.cumsum(explained_variance_ratio)
# Create a DataFrame to display the variance explained by each principal component
pca_results = pd.DataFrame({
    'Principal Component': np.arange(1, len(explained_variance_ratio) + 1),
    'Explained Variance Ratio': explained_variance_ratio,
```

```
    'Cumulative Variance': cumulative_variance
})
print(pca_results)
# Loadings of the principal components
loadings = pd.DataFrame(pca.components_, columns=numeric_cols).T
# Display loadings for the first 10 principal components
loadings_first_10 = loadings.iloc[:, :10]
print(loadings_first_10)
import matplotlib.pyplot as plt
import seaborn as sns
# Setting up the figure and axes
fig, ax = plt.subplots(2, 1, figsize=(10, 12))
# Heatmap of the loadings for the first 10 principal components
plt.figure(figsize=(12, 8))
sns.heatmap(loadings_first_10, cmap='coolwarm', annot=True, fmt=".2f")
plt.title('Heatmap of PCA Component Loadings')
plt.show()
# Correlation of all variables with Co2-Emissions and Life Expectancy
correlations = data_cleaned[numeric_cols].corr()
co2_correlations = correlations['Co2-Emissions'].sort_values(ascending=False)
life_exp_correlations = correlations['Life expectancy'].sort_values(ascending=False)
# Combine the correlations into a single DataFrame for easier comparison
correlation_comparison = pd.DataFrame({
    'Co2-Emissions': co2_correlations,
    'Life Expectancy': life_exp_correlations
})
correlation_comparison.head(10)  # Show top 10 correlations for each
# Setting up the figure for visualization
fig, ax = plt.subplots(1, 2, figsize=(14, 6))
# Scatter plot for Co2-Emissions vs. Armed Forces size
sns.scatterplot(x=data_cleaned['Armed Forces size'], y=data_cleaned['Co2-Emissions'],
ax=ax[0], color='blue')
ax[0].set_title('Co2-Emissions vs. Armed Forces size')
ax[0].set_xlabel('Armed Forces size (scaled)')
ax[0].set_ylabel('Co2-Emissions (scaled)')
# Scatter plot for Life Expectancy vs. Birth Rate
sns.scatterplot(x=data_cleaned['Birth Rate'], y=data_cleaned['Life expectancy'], ax=ax[1],
color='green')
ax[1].set_title('Life Expectancy vs. Birth Rate')
ax[1].set_xlabel('Birth Rate (scaled)')
ax[1].set_ylabel('Life Expectancy (scaled)')
plt.tight_layout()
plt.show()
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
# Extract the first 10 principal components for features and 'Life expectancy' for the target
X_pca = pca.transform(data_cleaned[numeric_cols])[:, :10]
y = data_cleaned['Life expectancy']
# Splitting the data into training and testing sets
X_train_init, X_test_init, y_train_init, y_test_init = train_test_split(X_pca, y, test_size=0.2,
random_state=42)
```

```
# Initialize the Linear Regression model
regressor = LinearRegression()
# Fit the model on the training data
regressor.fit(X_train_init, y_train_init)
# Predict on the test data
y_pred_init = regressor.predict(X_test_init)

# Calculate residuals
resid_init = y_test_init - y_pred_init
# Calculate the Mean Squared Error (MSE) for the test data
mse = mean_squared_error(y_test_init, y_pred_init)
print("Mean Squared Error (MSE):", mse)
# Calculate the Root Mean Squared Error (RMSE)
rmse = mean_squared_error(y_test_init, y_pred_init, squared=False)
print("Root Mean Squared Error (RMSE):", rmse)
# Calculate the Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test_init, y_pred_init)
print("Mean Absolute Error (MAE):", mae)
# Calculate the R-squared value
r_squared = r2_score(y_test_init, y_pred_init)
print("R-squared:", r_squared)
from sklearn.model_selection import cross_val_score
# Define the range of number of components to test
component_range = range(1, 21)  # Testing from 1 to 20 components
results_df = pd.DataFrame()
# Perform cross-validation for each number in the component range
for n_components in component_range:
    # Extract the first n_components principal components for features
    X_pca_n = pca.transform(data_cleaned[numeric_cols])[:, :n_components]
    # Initialize the Linear Regression model
    regressor_n = LinearRegression()
    # Calculate cross-validated MSE scores (using negative MSE for scoring)
    mse_scores = cross_val_score(regressor_n, X_pca_n, y, cv=10,
scoring='neg_mean_squared_error')
    mse_mean = -mse_scores.mean()
    # Calculate RMSE scores from the negative MSE scores
    rmse_scores = cross_val_score(regressor_n, X_pca_n, y, cv=10,
scoring='neg_root_mean_squared_error')
    rmse_mean = -rmse_scores.mean()
    # Calculate MAE scores
    mae_scores = cross_val_score(regressor_n, X_pca_n, y, cv=10,
scoring='neg_mean_absolute_error')
    mae_mean = -mae_scores.mean()
    # Calculate R^2 scores
    r2_scores = cross_val_score(regressor_n, X_pca_n, y, cv=10, scoring='r2')
    r2_mean = r2_scores.mean()
    # Append the results for this number of components to the DataFrame
    iter_df = pd.DataFrame({
        'Number of Components': [n_components],
        'Mean CV MSE': [mse_mean],
        'Mean CV RMSE': [rmse_mean],
        'Mean CV MAE': [mae_mean],
```

```
        'Mean CV R^2': [r2_mean]
    })
    results_df = pd.concat([results_df, iter_df], ignore_index=True)
# Print the results
results_df.sort_values('Mean CV MSE').head()
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
# Selecting the optimal number of components: 20
optimal_components = 20
# Extract the first 20 principal components for features
X_pca_optimal = pca.transform(data_cleaned[numeric_cols])[:, :optimal_components]
# Splitting the data into training and testing sets with the optimal components
X_train_optimal, X_test_optimal, y_train_optimal, y_test_optimal =
train_test_split(X_pca_optimal, y, test_size=0.2, random_state=42)
# Initialize and fit the Linear Regression model with the optimal number of components
regressor_optimal = LinearRegression().fit(X_train_optimal, y_train_optimal)
# Predict on the test data
y_pred_optimal = regressor_optimal.predict(X_test_optimal)
# Calculate residuals
resid_optimal = y_test_optimal - y_pred_optimal

# Calculate the Mean Squared Error (MSE) for the optimal model on test data
mse_optimal = mean_squared_error(y_test_optimal, y_pred_optimal)
print("Mean Squared Error (MSE):", mse_optimal)
# Calculate the Root Mean Squared Error (RMSE)
rmse_optimal = mean_squared_error(y_test_optimal, y_pred_optimal, squared=False)
print("Root Mean Squared Error (RMSE):", rmse_optimal)
# Calculate the Mean Absolute Error (MAE)
mae_optimal = mean_absolute_error(y_test_optimal, y_pred_optimal)
print("Mean Absolute Error (MAE):", mae_optimal)
# Calculate the R-squared value
r_squared_optimal = r2_score(y_test_optimal, y_pred_optimal)
print("R-squared:", r_squared_optimal)
from sklearn.model_selection import cross_val_score, cross_val_predict, KFold
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import numpy as np
# Selecting the optimal number of components: 20
optimal_components = 20
# Extract the first 20 principal components for features
X_pca_optimal = pca.transform(data_cleaned[numeric_cols])[:, :optimal_components]
# Setup the cross-validation framework with k=5 folds
kf = KFold(n_splits=5, shuffle=True, random_state=42)
# Initialize the Linear Regression model
regressor_optimal_cross = LinearRegression()
# Perform cross-validation
scores_mse = -cross_val_score(regressor_optimal_cross, X_pca_optimal, y, cv=kf,
scoring='neg_mean_squared_error')
scores_mae = -cross_val_score(regressor_optimal_cross, X_pca_optimal, y, cv=kf,
scoring='neg_mean_absolute_error')
scores_r2 = cross_val_score(regressor_optimal_cross, X_pca_optimal, y, cv=kf, scoring='r2')
```

```
# Calculate mean and standard deviation of scores
mean_rmse = np.mean(np.sqrt(scores_mse))
std_rmse = np.std(np.sqrt(scores_mse))
mean_mae = np.mean(scores_mae)
std_mae = np.std(scores_mae)
mean_r2 = np.mean(scores_r2)
std_r2 = np.std(scores_r2)
# Print results
print("Cross-validated Root Mean Squared Error (RMSE):", mean_rmse)
print("Standard Deviation of RMSE:", std_rmse)
print("Cross-validated Mean Absolute Error (MAE):", mean_mae)
print("Standard Deviation of MAE:", std_mae)
print("Cross-validated R-squared:", mean_r2)
print("Standard Deviation of R-squared:", std_r2)
import matplotlib.pyplot as plt
import seaborn as sns
fig, ax = plt.subplots(2, 2, figsize=(14, 12))
# Actual vs. Predicted Plot for the initial model
sns.scatterplot(x=y_test_init, y=y_pred_init, ax=ax[0, 0], color='blue')
ax[0, 0].plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=2)
ax[0, 0].set_title('Actual vs. Predicted: Initial Model (10 Components)')
ax[0, 0].set_xlabel('Actual Values')
ax[0, 0].set_ylabel('Predicted Values')
# Actual vs. Predicted Plot for the optimized model
sns.scatterplot(x=y_test_optimal, y=y_pred_optimal, ax=ax[0, 1], color='red')
ax[0, 1].plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=2)
ax[0, 1].set_title('Actual vs. Predicted: Optimized Model (20 Components)')
ax[0, 1].set_xlabel('Actual Values')
ax[0, 1].set_ylabel('Predicted Values')
# Residuals Plot for the initial model
sns.histplot(resid_init, kde=True, ax=ax[1, 0], color='blue')
ax[1, 0].set_title('Residuals Distribution: Initial Model')
ax[1, 0].set_xlabel('Residuals')
ax[1, 0].set_ylabel('Frequency')
# Residuals Plot for the optimized model
sns.histplot(resid_optimal, kde=True, ax=ax[1, 1], color='red')
ax[1, 1].set_title('Residuals Distribution: Optimized Model')
ax[1, 1].set_xlabel('Residuals')
ax[1, 1].set_ylabel('Frequency')
plt.tight_layout()
plt.show()
# Setting up the figure and axes for the plots
fig, ax = plt.subplots(2, 2, figsize=(14, 12))
# Actual vs. Predicted Plot
sns.scatterplot(x=y_test_optimal, y=y_pred_optimal, ax=ax[0, 0], color='blue')
ax[0, 0].plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=2)
ax[0, 0].set_title('Actual vs. Predicted Life Expectancy')
ax[0, 0].set_xlabel('Actual Values')
ax[0, 0].set_ylabel('Predicted Values')
# Residuals Plot
sns.histplot(resid_optimal, kde=True, ax=ax[0, 1], color='green')
ax[0, 1].set_title('Distribution of Residuals')
```

```
ax[0, 1].set_xlabel('Residuals')
ax[0, 1].set_ylabel('Frequency')
# Contribution of Principal Components
pc_contributions = regressor_optimal.coef_
sns.barplot(x=np.arange(1, optimal_components + 1), y=pc_contributions, ax=ax[1, 0],
palette='viridis')
ax[1, 0].set_title('Contribution of Each Principal Component')
ax[1, 0].set_xlabel('Principal Components')
ax[1, 0].set_ylabel('Coefficient Magnitude')
# Distribution of Predicted vs. Actual Values
sns.histplot(y_test_optimal, kde=True, color="blue", label='Actual', ax=ax[1, 1], alpha=0.6)
sns.histplot(y_pred_optimal, kde=True, color="red", label='Predicted', ax=ax[1, 1], alpha=0.6)
ax[1, 1].set_title('Distribution of Actual vs. Predicted Values')
ax[1, 1].set_xlabel('Life Expectancy')
ax[1, 1].set_ylabel('Frequency')
ax[1, 1].legend()
plt.tight_layout()
plt.show()
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
X = data_cleaned.drop(['Life expectancy', 'Country', 'Abbreviation', 'Calling Code',
'Capital/Major City', 'Currency-Code', 'Official language', 'Largest city'], axis=1)
y = data_cleaned['Life expectancy']
X_train_full, X_test, y_train_full, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train_full)
X_train_full = scaler.transform(X_train_full)
X_test = scaler.transform(X_test)
en_initial = ElasticNet(random_state=42, copy_X=True)
en_initial.fit(X_train_full, y_train_full)
y_pred = en_initial.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
print(f"RMSE: {rmse}, MSE: {mse}")
lecnparr = life_expectancy_correlation.to_numpy()
top_features = np.where(np.abs(lecnparr) > 0.05)[0]
columns_to_be_kept = list(set(data_cleaned.columns) & set(X.columns))
top_features = [X.columns.get_loc(name) for name in columns_to_be_kept]
X_train_top_features = X_train_full[:,top_features]
feature_importance = en_initial.coef_
pd.DataFrame(feature_importance, index=columns_to_be_kept, columns=['Importance'])
kf = KFold(n_splits=10, shuffle=True, random_state=42)
fold_rmse = []
for train_index, valid_index in kf.split(X_train_top_features):
    X_train, X_valid = X_train_top_features[train_index], X_train_top_features[valid_index]
    y_train, y_valid = y_train_full.iloc[train_index], y_train_full.iloc[valid_index]
    model = ElasticNet(random_state=42, copy_X=True)
    model.fit(X_train, y_train)
    y_pred_valid = model.predict(X_valid)
    rmse = np.sqrt(mean_squared_error(y_valid, y_pred_valid))
```

```
    fold_rmse.append(rmse)
print("RMSE for each fold:", fold_rmse)
print("Mean RMSE across all folds:", np.mean(fold_rmse))
print("Standard Deviation of RMSE across folds:", np.std(fold_rmse))
import warnings, sys, os
if not sys.warnoptions:
    warnings.simplefilter("ignore")
    os.environ["PYTHONWARNINGS"] = "ignore"
param_grid = {
    'alpha': [0.001, 0.01, 0.1, 1, 10, 100],
    'l1_ratio': [0 ,0.1, 0.3, 0.5, 0.7, 0.9, 1],
    'max_iter': [1000000],
    'selection': ['cyclic', 'random'],
    'tol': [1e-3, 1e-4, 1e-5],
    'positive': [True, False],
}

least_rmse = np.inf
best_params = None
best_model = None
X_test_top_features = X_test[:,top_features]
for kf in range(2, 11):
    kf = KFold(n_splits=kf, shuffle=True, random_state=42)
    en = ElasticNet(random_state=42, copy_X=True)
    grid_search = GridSearchCV(estimator=en, param_grid=param_grid, cv=kf,
scoring='neg_root_mean_squared_error', n_jobs=-1)
    grid_search.fit(X_train_top_features, y_train_full)
    print(f"KFold: {kf.n_splits}, Best Score: {grid_search.best_score_}, Best Params:
{grid_search.best_params_}")
    current_best_model = grid_search.best_estimator_
    y_pred_test = current_best_model.predict(X_test_top_features)
    rmse_test = np.sqrt(mean_squared_error(y_test, y_pred_test))
    print(f"Test RMSE: {rmse_test}")
    if rmse_test < least_rmse:
        least_rmse = rmse_test
        best_params = grid_search.best_params_
        best_model = current_best_model
y_pred = best_model.predict(X_test_top_features)
plt.scatter(y_test, y_pred, alpha=0.5)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=4)
plt.xlabel('Actual Life Expectancy')
plt.ylabel('Predicted Life Expectancy')
plt.title('Actual vs Predicted Life Expectancy')
plt.show()
from sklearn.model_selection import cross_val_score
from sklearn.metrics import make_scorer, r2_score, mean_absolute_error
k = 5
scorer_rmse = make_scorer(lambda y_true, y_pred: mean_squared_error(y_true, y_pred,
squared=False))
scorer_mae = make_scorer(mean_absolute_error)
scorer_r2 = make_scorer(r2_score)
# Perform k-fold cross-validation for each metric
```

```
cv_scores_rmse = cross_val_score(best_model, X, y, cv=k, scoring=scorer_rmse)
cv_scores_mae = cross_val_score(best_model, X, y, cv=k, scoring=scorer_mae)
cv_scores_r2 = cross_val_score(best_model, X, y, cv=k, scoring=scorer_r2)
# Calculate the mean and standard deviation for each metric
mean_rmse = cv_scores_rmse.mean()
std_rmse = cv_scores_rmse.std()
mean_mae = cv_scores_mae.mean()
std_mae = cv_scores_mae.std()
mean_r2 = cv_scores_r2.mean()
std_r2 = cv_scores_r2.std()
# Print out the results
print(f'RMSE: {mean_rmse:.2f} (std: {std_rmse:.2f})')
print(f'MAE: {mean_mae:.2f} (std: {std_mae:.2f})')
print(f'R-squared: {mean_r2:.2f} (std: {std_r2:.2f})')
import optuna
def objective(trial):
    alpha = trial.suggest_float('alpha', 1e-5, 100.0, log=True)
    l1_ratio = trial.suggest_float('l1_ratio', 0.0, 1.0)
    selection = trial.suggest_categorical('selection', ['cyclic', 'random'])
    tol = trial.suggest_float('tol', 1e-5, 1e-1, log=True)
    positive = trial.suggest_categorical('positive', [True, False])
    model = ElasticNet(alpha=alpha, l1_ratio=l1_ratio, selection=selection, tol=tol,
positive=positive, random_state=42, copy_X=True, max_iter=1000000)
    model.fit(X_train_top_features, y_train_full)
    y_pred_test = model.predict(X_test_top_features)
    rmse_test = np.sqrt(mean_squared_error(y_test, y_pred_test))
    mae_test = mean_absolute_error(y_test, y_pred_test)
    r2_test = r2_score(y_test, y_pred_test)
    print(f"RMSE: {rmse_test} | MAE: {mae_test} | R2: {r2_test}")
    return rmse_test
study = optuna.create_study(direction='minimize')
study.optimize(objective, n_trials=100)
best_params = study.best_params
best_score = study.best_value
print("Best parameters:", best_params)
print("Best score:", best_score)
best_model = ElasticNet(**best_params, random_state=42, copy_X=True, max_iter=1000000)
best_model.fit(X_train_top_features, y_train_full)
y_pred = best_model.predict(X_test_top_features)
plt.scatter(y_test, y_pred, alpha=0.5)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=4)
plt.xlabel('Actual Life Expectancy')
plt.ylabel('Predicted Life Expectancy')
plt.title('Actual vs Predicted Life Expectancy')
plt.show()
feature_importance = best_model.coef_
pd.DataFrame(feature_importance, index=columns_to_be_kept, columns=['Importance'])
import numpy as np
from scipy.stats import pearsonr
correlations = []
for i in range(X.shape[1]):
    correlation, _ = pearsonr(X_test[:, i], y_pred)
```

```
    correlations.append(correlation)
for i, correlation in enumerate(correlations):
    print(f"Pearson correlation between input variable {X.columns[i]} and output variable:
{correlation}")
k = 5
# Perform k-fold cross-validation for each metric
cv_scores_rmse = cross_val_score(best_model, X, y, cv=k, scoring=scorer_rmse)
cv_scores_mae = cross_val_score(best_model, X, y, cv=k, scoring=scorer_mae)
cv_scores_r2 = cross_val_score(best_model, X, y, cv=k, scoring=scorer_r2)
# Calculate the mean and standard deviation for each metric
mean_rmse = cv_scores_rmse.mean()
std_rmse = cv_scores_rmse.std()
mean_mae = cv_scores_mae.mean()
std_mae = cv_scores_mae.std()
mean_r2 = cv_scores_r2.mean()
std_r2 = cv_scores_r2.std()
# Print out the results
print(f'RMSE: {mean_rmse:.2f} (std: {std_rmse:.2f})')
print(f'MAE: {mean_mae:.2f} (std: {std_mae:.2f})')
print(f'R-squared: {mean_r2:.2f} (std: {std_r2:.2f})')
```

ДОДАТОК Б ГРАФІЧНІ МАТЕРІАЛИ

ПОСТАНОВКА ЗАДАЧІ

1. Визначити актуальність проблеми і сформулювати задачу
2. Дослідити етапи побудови моделей прогнозування
3. Зібрати та обробити демографічні дані
4. Описати принципи роботи моделей та підходи до її оцінювання
5. Обрати відповідні моделі прогнозування
6. Виконати порівняльний аналіз результатів

2

АКТУАЛЬНІСТЬ ДОСЛІДЖЕННЯ

Очікувана тривалість життя є одним із найважливіших показників здоров’я та добробуту населення. Здебільшого цей показник важливий для таких сфер життя:

- Охорона здоров'я: Визначає ефективність медичних послуг і впливає на планування ресурсів.
- Соціальна політика: Враховується при розробці пенсійних програм і соціального забезпечення.
- Економіка: Впливає на продуктивність праці та економічне планування.
- Освіта: Визначає потребу в освітніх ресурсах для підготовки кваліфікованих кадрів.
- Екологія: Відображає вплив навколишнього середовища на здоров'я населення

Актуальність теми є досить вагомою, як в Україні так і закордоном. Показник очікуваної тривалості життя є важливим індикатор задля створення нових стратегій розвитку, як цілих країн, так і окремих галузей. Саме завдяки цій оцінці можна ідентифікувати і спрогнозувати рівень життя і якість життя населення в майбутньому.

3

ДОСЛІДЖЕННЯ

Дані були взяти з відкритого джерела на сайті Kaggle. Даний датасет містить інформація про демографічні показники кожної країни, економічні, екологічні, соціальні показники, показники рівня освіченості населення за 2023 рік. В обраному наборі даних 195 записів і 35 стовпців.
Приклад даних:

	Country	Density\n(P/Km2)	Abbreviation	Agricultural Land(%)	Land Area(Km2)	Armed Forces size	Birth Rate	Calling Code	Capital/Major City	Co2-Emissions	...	Out of pocket health expenditure	Physicians per thousand	Population	Population: Labor force participation (%)	Tax revenue (%)	Total tax rate	Unemployment rate	Urban_population
0	Afghanistan	60	AF	58.10%	652,230	323,000	32.49	93.0	Kabul	8,672	...	78.40%	0.28	38,041,754	48.90%	9.30%	71.40%	11.12%	9,797,273
1	Albania	105	AL	43.10%	28,748	9,000	11.78	355.0	Tirana	4,536	...	56.90%	1.20	2,854,191	55.70%	18.60%	36.60%	12.33%	1,747,593
2	Algeria	18	DZ	17.40%	2,381,741	317,000	24.28	213.0	Algiers	150,006	...	28.10%	1.72	43,053,054	41.20%	37.20%	66.10%	11.70%	31,510,100
3	Andorra	164	AD	40.00%	468	NaN	7.20	376.0	Andorra la Vella	469	...	36.40%	3.33	77,142	NaN	NaN	NaN	NaN	67,873
4	Angola	26	AO	47.50%	1,246,700	117,000	40.73	244.0	Luanda	34,693	...	33.40%	0.21	31,825,295	77.50%	9.20%	49.10%	6.89%	21,061,025
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
190	Venezuela	32	VE	24.50%	912,050	343,000	17.88	58.0	Caracas	164,175	...	45.80%	1.92	28,515,829	59.70%	NaN	73.30%	8.80%	25,162,368
191	Vietnam	314	VN	39.30%	331,210	522,000	16.75	84.0	Hanoi	192,668	...	43.50%	0.82	96,462,106	77.40%	19.10%	37.60%	2.01%	35,332,140
192	Yemen	56	YE	44.60%	527,968	40,000	30.45	967.0	Sanaa	10,609	...	81.00%	0.31	29,161,922	38.00%	NaN	26.60%	12.91%	10,869,523
193	Zambia	25	ZM	32.10%	752,618	16,000	36.19	260.0	Lusaka	5,141	...	27.50%	1.19	17,861,030	74.60%	16.20%	15.60%	11.43%	7,871,713
194	Zimbabwe	38	ZW	41.90%	390,757	51,000	30.68	263.0	Harare	10,983	...	25.80%	0.21	14,645,468	83.10%	20.70%	31.60%	4.95%	4,717,305

https://www.kaggle.com/datasets/nelgiriyewithana/countries-of-the-world-2023

4

ДОСЛІДЖЕННЯ

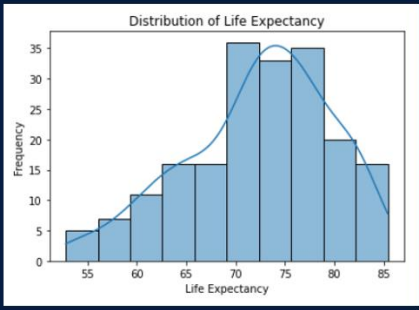
Серед ознак для дослідження були виділені наступні характеристи:

- Щільність населення на км2
- Відсоток земельної площі на сільське госп.
- Загальна площа країни в км2
- Чисельність збройних сил
- Кількість народжених на 1000 населення на рік
- Викиди CO2
- Індекс споживчих цін
- Зміна Індексу споживчих цін з минулим роком
- Рівень фертильності
- Відсоток земельної площі вкритої лісами
- Ціна бензину на літр
- ВВП
- Коефіцієнт охоплення початковою освітою
- Кількість смертей на 1000 народжених
- Очікувана тривалість життя
- Кількість материнських смертей
- Рівень мінімальної зп
- Відсоток витрат на охорону здоров'я (з фіз. особи)
- Кількість лікарів на 1000 населення
- Кількість населення
- Відсоток робочої сили
- Податкові надходження від ВВП
- Загальне побуткове навантаження від комерційного прибутку
- Відсоток безробітньої робочої сили
- Відсоток населення, що проживає в містах

5

ДОСЛІДЖЕННЯ

Якщо казати про змінну очікувана тривалість життя, було проведено наступний аналіз:



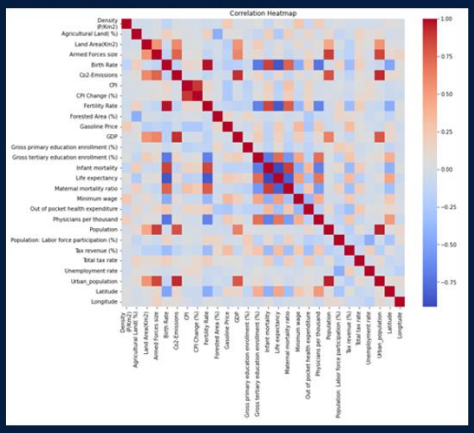

```
Central tendency:
Life expectancy mean: 72.279
Life expectancy median: 73.2
Life expectancy mode: 76.5
```

```
Summary:
 count    187.000000
mean      72.279679
std        7.483661
min       52.800000
25%       67.000000
50%       73.200000
75%       77.500000
max       85.400000
```

6

ДОСЛІДЖЕННЯ

Аналіз коефіцієнтів кореляції




```
Life expectancy                      1.000000
Gross tertiary education enrollment (%)  0.708210
Physicians per thousand              0.675590
Minimum wage                         0.462103
Latitude                             0.460774
Tax revenue (%)                      0.321470
Gasoline Price                       0.244495
GDP                                  0.175577
Co2-Emissions                        0.118737
Gross primary education enrollment (%)  0.093897
Armed Forces size                    0.072699
Urban_population                     0.070342
Density\n(P/Km2)                     0.055198
Land Area(Km2)                       0.054772
Forested Area (%)                    0.019361
Population                           0.008802
Unemployment rate                   -0.043416
Longitude                           -0.070463
CPI Change (%)                      -0.143779
Population: Labor force participation (%)  -0.148990
CPI                                 -0.174632
Total tax rate                      -0.183695
Agricultural Land( %)               -0.226174
Out of pocket health expenditure    -0.311387
Maternal mortality ratio            -0.816951
Fertility Rate                      -0.846317
Birth Rate                          -0.865804
Infant mortality                    -0.915081
```

7

СТВОРЕННЯ ПРОГРАМНОГО ПРОДУКТУ

Мова програмування – Python

Середовище розробки – Jupyter Notebook

Використані бібліотеки:

8

МОДЕЛІ

Для вирішення поставленої задачі було обрано 5 моделей машинного навчання

- Linear regression
- Elastic net
- Principal component regression
- Support vector machine
- Random forest

10

МЕТРИКИ ДЛЯ ОЦІНКИ МОДЕЛЕЙ

Для даного дослідження було обрано наступні метрики оцінки якості моделей:

- **MAE** (Mean Absolute Error) – середня абсолютна помилка. Вимірює середнє значення абсолютних помилок між прогнозованими та фактичними значеннями.
- **RMSE** (Root Mean Square Error) – середньоквадратична помилка прогнозу. Вимірює середнє значення квадратів відхилень між прогнозованими значеннями моделі та фактичними значеннями цільової змінної.
- **R-squared** – коефіцієнт детермінації. Вимірює наскільки добре модель підходить для даних. Вказує на частку варіації у цільовій змінній, яка пояснюється моделлю.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

9

АНАЛІЗ РЕЗУЛЬТАТІВ

Linear regression

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \mathcal{E},$$

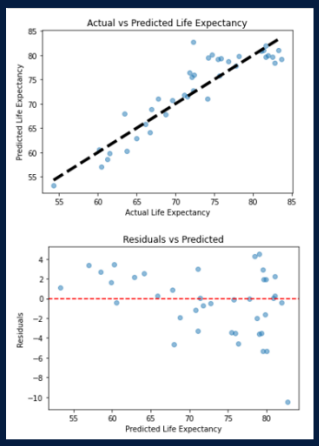
Аналіз результатів моделі

RMSE	MAE	R^2
3,19	2,46	0,82

Аналіз результатів моделі після 5-кратної перевірки

RMSE	MAE	R^2
2.68	2.04	0.84

Фактичні значення	Прогнозовані значення
71,1	71,83
81,6	82
54,3	53,2
63,7	60,29
74,1	71



Графік залежності фактичного і очікуваного значення і графік залишків

11

АНАЛІЗ РЕЗУЛЬТАТІВ

Support vector machine

$f(x) = w^T x + b$ –функція яку треба дослідити

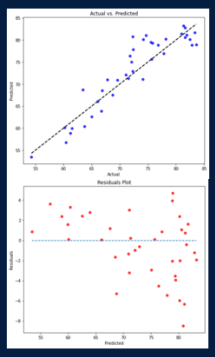
$$\begin{cases} \frac{1}{2}\|w\|^2 + C\Sigma_i \mathcal{E}_i \to min \\ y_i \cdot g(x_i) \geq 1 - \mathcal{E}_i, \forall i \\ \mathcal{E}_i \geq 0, \forall i \end{cases}$$ – оптимізаційна задача

Аналіз метрик різних ядер функцій

Тип ядра	R^2	RMSE	MAE
Лінійне	0.82	3.20	2.53
Поліноміальне	0.7	4.11	2.94
Радіального базису	0.74	3.83	2.97
Сигмоїдне	0.72	3.97	3.28

Аналіз результатів моделі після 5-кратної перевірки

RMSE	MAE	R^2
2.64	2.02	0.86



Графік залежності фактичного і очікуваного значення і графік залишків

Фактичні значення	Прогнозовані значення
71,1	72,07
81,6	82,81
54,3	53,44
63,7	60,4
74,1	71,07

12

АНАЛІЗ РЕЗУЛЬТАТІВ

Random Forest

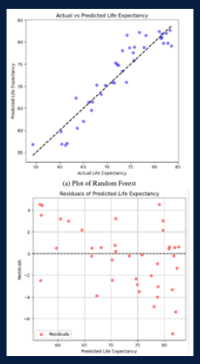
$$\hat{y} = \frac{1}{B}\sum_{b=1}^{B} f_{b(x)}$$ – середнє значення прогнозів всіх дерев

Аналіз результатів моделі

RMSE	MAE	R^2
2,9	2,3	0,85

Аналіз результатів моделі після 5-кратної перевірки

RMSE	MAE	R^2
2.82	2.12	0.84



Графік залежності фактичного і очікуваного значення і графік залишків

Фактичні значення	Прогнозовані значення
71,1	70,88
81,6	81,81
54,3	56,79
63,7	60,5
74,1	70,88

13

АНАЛІЗ РЕЗУЛЬТАТІВ

Principal Component Regression

$\hat{y} = Z_k\beta + b$ - формула регресії, де Z_k - перші k голов. компонент, а β - комп. регресії

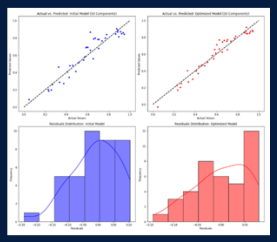
$Z_k = X_{std}P_k$ - формула перших коловних компонентів , де X_{std} - стандарт. дані, а P_k - перші k власних компонент

Аналіз результатів моделі

RMSE	MAE	R^2
0.078	0.061	0.89

Аналіз результатів моделі після 5-кратної перевірки

RMSE	MAE	R^2
0.063	0.045	0.92



Порівняння результатів з 10 і 20 компонентами PCA

Фактичні значення	Прогнозовані значення
71,1	71,05
81,6	81,55
54,3	54,4
63,7	63,9
74,1	74,3

14

АНАЛІЗ РЕЗУЛЬТАТІВ

Elastic net

$\hat{y} = w^T x + b$ - функція прогнозу

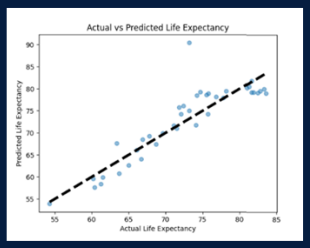
$L(w,b) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda_1\|w\|_1 + \lambda_2\|w\|_2^2$ - функція втрат

Аналіз результатів моделі

RMSE	MAE	R^2
2.91	2.39	0.85

Аналіз результатів моделі після 5-кратної перевірки

RMSE	MAE	R^2
3.46	2.33	0.77



Графік залежності фактичного і очікуваного значення моделі

Фактичні значення	Прогнозовані значення
71,1	68,77
81,6	79,27
54,3	56,63
63,7	61,6
74,1	71,77

15

АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

Метрика	Лінійна регресія	SVR	RF	PCR	Elastic Net
RMSE	3.19	3.2	2.9	0.056	2.91
MAE	2.46	2.53	2.3	0.047	2.39
R^2	0.82	0.82	0.85	0.94	0.85

Метрики всіх використаних моделей

Метрика	Лінійна регресія	SVR	RF	PCR	Elastic Net
RMSE	2.68	2.64	2.82	0.062	3.46
MAE	2.04	2.02	2.12	0.045	2.33
R^2	0.84	0.86	0.84	0.91	0.77

Метрики всіх використаних моделей після 5-кратної перехресної перевірки

Дивлячись на метрики якості роботи моделей, не важко помітити, що більшість з моделей були обрані правильно, адже на виході отримали задовільну точність. Найгірше справилась модель Elastic Net. Однак звісно варто відмітити модель, яка спрацювала краще, а саме PCR. Ця модель краще за все себе показала через комбінацію PCA і лінійної регресії, що дозволяє зменшувати розмірність даних, зберігаючи основні компоненти, які пояснюватимуть найбільшу частину варіації в даних.

Основними показниками високої очікуваної тривалості життя є вища освіта, мінімальна заробітна плата та кількість лікарів. Індикаторами низької очікуваної тривалості життя є високий рівень народжуваності, рівень дитячої смертності та рівень фертильності. Індикаторами високої тривалості життя є показники багатших і розвиненіших країн.

16

ВИСНОВКИ

- Відповідно до дослідження було виконано огляд існуючих моделей та методів прогнозування
- Виконано дослідження щодо основних факторів, які впливають на показник очікуваної тривалості життя. Основними показниками високої очікуваної тривалості життя є вища освіта, мінімальна заробітна плата та кількість лікарів. Індикаторами низької очікуваної тривалості життя є високий рівень народжуваності, рівень дитячої смертності та рівень фертильності.
- Отримано якісно прогнози очікуваної тривалості життя. Зокрема найкращі результати показала модель PCR, адже модель пояснює близько 91% варіації даних, а також середній квадрат різниці між прогнозованою і фактичною тривалістю життя становить 0.062, а середня абсолютна різниця між прогнозованою і фактичною тривалістю життя становить 0.045.

17

ПОДАЛЬШІ ДОСЛІДЖЕННЯ

У майбутньому для покращення продуктивності моделей машинного навчання варто розглянути збільшення обсягу вибірки, а також пошук і дослідження нових сучасних алгоритмів для визначення оптимальних параметрів моделей.

17

ДЯКУЮ ЗА УВАГУ