МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«Допущений до захисту»
Завідувач кафедри
_____ проф. Приходько С.Б.
«___» _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти «магістр»

на тему: Нелінійна регресійна модель для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter, та розробка програми для її реалізації.

Виконав: студент групи 6151м
_____ Іванов Д.О.
(підпис, ПІБ)
Керівник роботи:
завідувач кафедри ПЗАС, проф.
(посада, науковий ступень вчене звання)
_____ Приходько С.Б.
(підпис, ПІБ)

Миколаїв – 2024 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____проф. С.Б.Приходько
(
« 14 » жовтня 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «магістр»

Студенту Іванов Денис Олексійович _____
(Прізвище, ім'я, по батькові)

1. Тема роботи: Нелінійна регресійна модель для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter, та розробка програми для її реалізації. _____

Керівник роботи завідувач кафедри ПЗАС, проф. Приходько Сергій Борисович _____

Затверджені наказом ректора № 1070-УЧ від « 11 » _____10_____ 2024 р.

2. Термін подання роботи: 09.12.2024 р._____

3. Вихідні дані по роботі: _____

4. Перелік питань, що належать до розробки (найменування розділів) _____

Вступ (Актуальність теми. Зв'язок роботи з науковими програмами, планами, темами. Мета і завдання дослідження. Об'єкт дослідження. Предмет дослідження. Методи дослідження. Наукова новизна одержаних результатів. Практичне значення одержаних результатів. Особистий внесок здобувача. Апробація результатів досліджень**.** Публікації.); Огляд літератури та обґрунтування необхідності проведення досліджень за обраною темою; Викладення результатів власних досліджень з висвітленням того нового, що пропонується; Розділ з розробки програмного забезпечення; Організаційно-економічний розділ; Розділи з охорони праці та охорони навколишнього середовища; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма і методика випробувань програмного забезпечення)

5. Перелік презентаційних матеріалів Титульний слайд, Мета та завдання дослідження, Об'єкт та Предмет дослідження, Наукова новизна одержаних результатів, Практичне значення одержаних результатів, Особистий внесок здобувача, Апробація результатів досліджень та Публікації, Аналіз існуючих методів оцінювання кількості строк коду мультимедійних застосунків для Android, Нелінійна регресійна модель для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter, Аналіз вимог

до програмного забезпечення, Архітектура програмного забезпечення, Діаграма варіантів використання, Технічний проект програмного забезпечення, Вибір інструментів розробки, Реалізація та Тестування програмного забезпечення, Результати розробки, Висновки.

6. Консультанти розділів кваліфікаційної (магістерської) роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 14.10.2024 р.

К А Л Е Н Д А Р Н И Й П Л А Н

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу ВСТУП	16.10.2024	НС*
2.	Підготовка розділу з огляду літератури та обгрунтування необхідності проведення досліджень за обраною темою	18.10.2024	НС*
3.	Підготовка розділу (ів) за результатами власних досліджень	21.10.2024	НС*
4.	Підготовка розділу з розробки програмного забезпечення	27.11.2024	
5.	Підготовка організаційно-економічного розділу	29.11.2024	
6.	Підготовка розділу з охорони праці	02.12.2024	
7.	Підготовка розділу з охорони навколишнього середовища	04.12.2024	
8.	Підготовка розділу ВИСНОВКИ	05.12.2024	
9.	Оформлення списку використаних джерел та додатків	06.12.2024	
10.	Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версії роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.)	09.12.2024	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку)
11.	Підготовка презентації та доповіді	13.12.2024	
12.	Попередній захист роботи на засіданні кафедри ПЗАС	16.12.2024	
13.	Подання на кафедру ПЗАС електронних версії наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді	19.12.2024	

* - за результатами наукового стажування (НС), яке було з 02.09.2024 по 13.10.2024 р.

Студент _____ Іванов Д.О
(підпис) (ПІБ)

Керівник роботи _____ Приходько С.Б.
(підпис) (ПІБ)

РЕФЕРАТ

Іванов Денис Олексійович

«Нелінійна регресійна модель для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter, та розробка програми для її реалізації. »

Кваліфікаційна (магістерська) робота на здобуття освітнього рівня магістра зі спеціальності 121 «Інженерія програмного забезпечення». Національний університет кораблебудування імені адмірала Макарова. Миколаїв, 2024 р.

Обсяг роботи: 107 стор., 10 табл., 17 рис., 35 використаних джерел, 5 додатків.

Актуальність теми: Дослідження зосереджується на важливій проблемі оцінювання кількості строк коду для мультимедійних застосунків, розроблених на Flutter. У зв’язку зі зростаючою популярністю Flutter для кросплатформної розробки, розробка точних інструментів прогнозування обсягу коду є надзвичайно актуальною. Це дозволяє не лише підвищити ефективність планування, а й оптимізувати розробницькі процеси, що є важливим аспектом розвитку програмного забезпечення.

Мета та завдання дослідження: Метою дослідження є створення нелінійної регресійної моделі для оцінювання кількості строк коду мультимедійних застосунків на Flutter шляхом розробки та впровадження нелінійної регресійної моделі. Основними завданнями є:

– Аналіз існуючих підходів до прогнозування обсягу коду.

– Розробка моделі, яка враховує специфіку розробки застосунків на Flutter.

– Реалізація програмного забезпечення для автоматизації оцінювання.

– Тестування моделі на реальних проєктах.

Об'єкт дослідження: процес оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter.

Предмет дослідження: Предметом дослідження є нелінійна регресійної моделі для оцінювання кількості строк коду мультимедійних застосунків на , що використовує метрики CBO (Coupling Between Object Classes), WMC (Weighted Methods per Class) та DIT (Depth of Inheritance Tree).

Методи дослідження: Для розробки моделі застосовано методи регресійного аналізу, математичного моделювання, статистичних обчислень та аналізу даних. Емпіричне тестування проводилося з використанням прикладів реальних застосунків для оцінки точності моделі.

Наукова новизна: Удосконалено нелінійну регресійну модель для оцінювання кількості строк коду мультимедійних застосунків для Android, розроблених на платформі Flutter, котра на відміну від існуючих моделей для Kotlin та PHP застосунків, враховує специфічні особливості Flutter, шляхом визначення нових параметрів моделі та застосування нормалізуючого перетворення у вигляді десяткового логарифму для оцінювання метрики LOC на основі метрик CBO, WMC та DIT, що забезпечує достовірне оцінювання мультимедійних застосунків розроблених на платформі Flutter.

Практична значущість: Модель і програмне забезпечення на її основі забезпечують розробників ефективним інструментом для прогнозування строк коду, що сприяє оптимізації використання ресурсів, підвищенню продуктивності команд та полегшує процес управління проєктами.

Апробація результатів дослідження: Основні результати дослідження були представлені на V Всеукраїнській науково-практичній інтернет-конференції «Інформаційні технології: моделі, алгоритми, системи», організованій Національним університетом кораблебудування у жовтні 2024 року.

Публікації: За результатами дослідження опубліковано одну наукову статтю у матеріалах конференції.

Ключові слова: нелінійна регресійна модель, оцінювання кількості строк коду, Flutter, мультимедійні застосунки, Android.

REVIEW

Denys Ivanov

«A nonlinear regression model for estimating the number of code lines of multimedia applications for Android created on the Flutter platform and development of a program for its implementation. »

Qualification (master's) thesis for obtaining a master's degree in the specialty 121 “Software Engineering”. Admiral Makarov National University of Shipbuilding. Mykolaiv, 2024.

Volume of work: 107 pages, 10 tables, 17 figures, 35 references, 5 appendices.

Relevance of the topic: This research focuses on the important problem of estimating the number of code lines for multimedia applications developed in Flutter. Due to the growing popularity of Flutter for cross-platform development, the development of accurate code size prediction tools is extremely relevant. This allows not only to increase planning efficiency but also to optimize development processes, which is an important aspect of software development.

The purpose and objectives of the study: The aim of the study is to create a nonlinear regression model for estimating the number of code lines of multimedia applications in Flutter by developing and implementing a nonlinear regression model. The main tasks are:

- Analyzing existing approaches to predicting code size.
- Development of a model that takes into account the specifics of Flutter application development.
- Implementation of software to automate the estimation.
- Testing the model on real projects.

Object of study: the process of estimating the number of code lines of multimedia applications for Android created on the Flutter platform.

Subject of research: The subject of the study is a nonlinear regression model for estimating the number of code lines of multimedia applications on Flutter, using the metrics CBO (Coupling Between Object Classes), WMC (Weighted Methods per Class), and DIT (Depth of Inheritance Tree).

Research methods: Regression analysis, mathematical modeling, statistical calculations, and data analysis were used to develop the model. Empirical testing was conducted using examples of real-world applications to assess the accuracy of the model.

Scientific novelty: A nonlinear regression model for estimating the number of code lines of multimedia applications for Android developed on the Flutter platform has been improved, which, unlike existing models for Kotlin and PHP applications, takes into account the specific features of Flutter, by defining new model parameters and applying a normalizing transformation in the form of a decimal logarithm to estimate the LOC metric based on the CBO, WMC, and DIT metrics, which provides a reliable assessment of multimedia applications developed on the Flutter platform.

Practical significance: The model and the software based on it provide developers with an effective tool for predicting code lifetime, which helps to optimize resource utilization, increase team productivity, and facilitate project management.

Testing of research results: The main results of the study were presented at the V All-Ukrainian Scientific and Practical Internet Conference “Information Technologies: Models, Algorithms, Systems” organized by the National University of Shipbuilding in October 2024.

Publications: Based on the results of the study, one scientific article was published in the conference proceedings.

Keywords: nonlinear regression model, estimation of the number of code lines, Flutter, multimedia applications, Android.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CBO – Coupling between Objects;

DIT – Depth of Inheritance Tree;

KLOC – thousands of lines of code;

WMC – Weighted Methods for Class.

ВСТУП

Актуальність теми. З кожним роком збільшується кількість мобільних застосунків, які розробляються за допомогою кросплатформених технологій. Основними причинами для цього є: прагнення оптимізувати процес розробки, зменшити витрати на підтримку різних платформ та прискорити вихід продукту на ринок. І великі технологічні компанії, і порівняно невеликі команди розробників все частіше обирають Flutter [1] як інструмент для створення мобільних додатків.

На даний момент розробка мобільних застосунків на Flutter у багатьох компаній вважається одним з перспективних напрямків діяльності. Перехід від нативної розробки до кросплатформеної – це напрям, який стрімко розвивається. Той, хто володіє сучасними інструментами розробки реалізує свої ідеї, може створювати якісний продукт.

Flutter застосунок – це програма, написана з використанням фреймворку від Google, де один і той же код може виконуватися на різних платформах. Логіка Flutter застосунків базується на мові програмування Dart [2], що забезпечує високу продуктивність та гнучкість розробки. Можна впевнено казати, що успіх програмних проектів на Flutter залежить від правильної оцінки їх складності та розміру. Для оцінки кількості рядків коду допомагає розробникам прогнозувати обсяг необхідних ресурсів і оцінювати складність проекту, що полегшує планування та розподіл ресурсів. Слід зауважити, що у даній роботі в якості розміру Flutter застосунку розуміється кількість рядків вихідного коду (KLOC) [3].

Оцінювання кількості строк коду (LOC) є ключовим у розробці програмного забезпечення. Воно дозволяє визначати розмір проекту,

прогнозувати ресурси та час, необхідні для завершення, а також оцінювати складність програмного продукту. LOC допомагає планувати ресурси, моніторити прогрес розробки та виявляти проблемні ділянки. Воно є індикатором якості коду: надмірний обсяг може свідчити про дублювання чи складність, що потребують оптимізації. Крім того, LOC використовується для порівняння проектів, оцінки продуктивності команд і вибору технологій. До теперішнього часу існує певна проблема в оцінці кількості строк коду Flutter застосунків. Для цієї задачі є різні підходи, методи та моделі, у нашому випадку LOC служить основою для побудови регресійної моделі та автоматизації процесу аналізу коду для Flutter-застосунків. Flutter-застосунки демонструють значно інші інтервали значень метрик порівняно з традиційними мовами програмування Kotlin та PHP [4, 5]. Існуючі регресійні моделі для оцінювання кількості строк коду зазвичай базуються на метриках, що характерні для нативної розробки під Android, Kotlin або PHP. Дослідження показують [6], що Flutter має власну специфіку організації коду, включаючи декларативний підхід до побудови інтерфейсу, широке використання віджетів та особливості компонентної архітектури Dart [7].

Мета дослідження: підвищення достовірності оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter шляхом удосконалення нелінійної регресійної моделі для оцінювання кількості строк коду мультимедійних застосунків для Android.

Для досягнення поставленої мети необхідно вирішити наступні **завдання**:

– проаналізувати існуючі моделі для оцінювання кількості строк коду мультимедійних застосунків;

– обґрунтувати необхідність удосконалення регресійної моделі для оцінювання кількості строк коду Flutter застосунків;

– дослідити різні проекти з відкритим вихідним кодом та визначити медіа-застосунки, створені на Flutter;

- отримати метрики;
- побудувати лінійну регресійну модель для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter;
- перейти до нелінійної регресійної моделі для оцінювання кількості строк коду Flutter застосунків;
- розробити програму для оцінювання кількості строк коду Flutter застосунків на основі побудованої моделі.

Об'єкт дослідження: процес оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter.

Предмет дослідження: нелінійна регресійна модель для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter.

Методи дослідження. Для розробки моделі застосовано методи регресійного аналізу, математичного моделювання, статистичних обчислень та аналізу даних. Емпіричне тестування проводилося з використанням прикладів реальних застосунків для оцінки точності моделі.

Наукова новизна. Удосконалено нелінійну регресійну модель для оцінювання кількості строк коду мультимедійних застосунків для Android, розроблених на платформі Flutter, котра на відміну від існуючих моделей для Kotlin та PHP застосунків [4, 5], враховує специфічні особливості Flutter, шляхом визначення нових параметрів моделі та застосування нормалізуючого перетворення у вигляді десяткового логарифму для оцінювання метрики LOC на основі метрик CBO, WMC та DIT [8-10] що забезпечує достовірне оцінювання мультимедійних застосунків розроблених на платформі Flutter.

Практичне значення одержаних результатів: розроблена програма для оцінювання кількості строк коду мультимедійних застосунків для Android, що

створюються на платформі Flutter, дозволила підвищити достовірність відповідних розрахунків.

Особистий внесок здобувача. Всі результати отримані самостійно. В роботі [6], яка опублікована у співавторстві з науковим керівником, автору належить: збір та обробка даних метрик з Flutter-застосунків, проведення порівняльного аналізу діапазонів метрик Kotlin застосункків, побудова нелінійної регресійної моделі та оцінювання кількості строк коду Flutter медіа застосунків.

Апробація результатів досліджень. Основні результати досліджень були оприлюднені на V Всеукраїнській науково-практичній інтернет-конференції «Інформаційні технології: моделі, алгоритми, системи» (30-31 Жовтня 2024), організованій Національним університетом кораблебудування імені адмірала Макарова.

Публікації. За результатами досліджень опубліковано одну наукову працю, а саме матеріали інтернет-конференції.

1 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ДЛЯ ОЦІНЮВАННЯ КІЛЬКОСТІ СТРОК КОДУ МУЛЬТИМЕДІЙНИХ ЗАСТОСУНКІВ ДЛЯ ANDROID ТА ОБГРУНТУВАННЯ НЕОБХІДНОСТІ ПРОВЕДЕННЯ ДОСЛІДЖЕНЬ ЗА ОБРАНОЮ ТЕМОЮ

1.1 Методи для оцінювання кількості строк коду програмного забезпечення

Розмір програмного забезпечення, зокрема кількість строк коду (LOC), є одним із ключових показників, який впливає на складність, вартість та час розробки програмних продуктів. У процесі оцінювання розміру програмного забезпечення існує декілька підходів, кожен із яких має свої переваги та недоліки залежно від платформи, мови програмування та особливостей проекту. Нижче наведено огляд основних методів оцінювання розміру програмного забезпечення, які можуть бути адаптовані для Android-застосунків, зокрема створених на платформі Flutter.

Оцінка розміру програмного забезпечення можлива за допомогою метрик об’єктно-орієнтованого дизайну (Object-Oriented Design Metrics) [11], таких як CBO (Coupling Between Object Classes), WMC (Weighted Methods per Class) та DIT (Depth of Inheritance Tree). Ці метрики підходять для об'єктно-орієнтованих проектів і можуть допомогти в оцінюванні розміру та складності програмного забезпечення. Зокрема, для застосунків на Flutter, які базуються на об’єктно-орієнтованих принципах, вони можуть показати складність взаємодії між компонентами та класами. Наприклад, CBO відображає ступінь взаємозалежності між класами, що може вказувати на труднощі підтримки та тестування програми. WMC оцінює сумарну складність методів у класі, де більша кількість методів та їх складність означають більший обсяг і складність коду. DIT

показує глибину успадкування класу в ієрархії, що може впливати на складність розуміння коду, особливо у великих проектів.

Метод оцінювання кількості строк коду на основі сценаріїв використання (Use Case Points, UCP) [12] також використовується для оцінки складності програмного забезпечення. Він базується на аналізі функціональних вимог через сценарії використання і особливо підходить для проектів із великою кількістю варіантів використання. Однак для застосунків на Flutter цей метод може бути менш ефективним через компонентний підхід у розробці. До переваг UCP належить його висока ефективність для великих проектів, які мають складну функціональність, а до недоліків — необхідність витрат часу на аналіз сценаріїв використання, що може бути недоцільним для невеликих проектів.

Ще одним методом, який може бути корисним для оцінювання кількості строк коду програмного забезпечення, є оцінка об’єму на основі історичних даних (Historical Data Based Estimation) [13]. Цей метод використовує історичні дані з попередніх проектів для прогнозування кількості строк коду нового програмного забезпечення. Наприклад, на основі даних про LOC, CBO, WMC та DIT у попередніх проектах можна зробити приблизні прогнози для майбутніх проектів. Цей метод є особливо корисним для проектів, що мають схожу структуру та архітектуру, наприклад, у випадку розробки на Flutter, де часто використовуються готові компоненти та шаблони.

Функціональні точки (FP) [14]. Модель на основі функціональних точок оцінює розмір програмного забезпечення залежно від функціональних вимог користувача, таких як вхідні дані, вихідні дані, файли та запити. Цей підхід не дозволяє оцінити кількість строк коду, оскільки він не враховує мову програмування він зосереджується на рівні абстракції, пов'язаному з функціональністю, а не з реалізацією.

Аналітичні моделі [15]. Ці моделі використовують математичні формули та алгоритми для оцінки кількості рядків коду. Зазвичай вони враховують складність системи, взаємозв'язки між компонентами та рівень деталізації проєктної документації. Аналітичні моделі можуть бути точнішими, але вимагають глибшого аналізу і деталізації початкових даних.

Лінійні та нелінійні моделі [16,17] також використовуються для прогнозування кількості рядків коду в залежності від складності проєкту та наявності залежностей між параметрами.

Лінійні моделі базуються на припущенні про пропорційну залежність між обсягом коду і певними параметрами проєкту, такими як кількість функціональних вимог або кількість модулів.

Нелінійні моделі дозволяють враховувати складнішу структуру залежностей між параметрами проєкту. Такі моделі можуть містити степеневі, експоненціальні або логарифмічні функції. Нелінійні моделі [17] мають вигляд:

$$LOC = a X^b,$$

де a і b — параметри моделі, які налаштовуються на основі попередніх даних. Нелінійні моделі краще справляються зі складними проєктами, де збільшення кількості функцій призводить до швидшого зростання обсягу коду.

Переваги та обмеження лінійних і нелінійних моделей:

– Лінійні моделі є простими для реалізації і можуть забезпечити швидкі оцінки для проєктів з невеликою кількістю параметрів і лінійною залежністю.

– Нелінійні моделі є точнішими для великих і складних проєктів, але вимагають більше даних для навчання та складніші в обчисленні.

Отже, оцінювання кількості строк коду програмного забезпечення мультимедійних застосунків для Android, створених на платформі Flutter, вимагає комплексного підходу. Комбінація метрик об'єктно-орієнтованого

дизайну, таких як CBO, WMC та DIT, разом із підрахунком LOC надає змогу оцінити не тільки кількість коду, а й складність взаємозв'язків між компонентами програми. Такий підхід є особливо корисним для багатокомпонентних мультимедійних застосунків, де важливо враховувати як кількість коду, так і складність його підтримки та взаємодії між класами.

1.2 Особливості розробки мультимедійних застосунків на платформі Flutter

Платформа Flutter, розроблена компанією Google, є популярним інструментом для створення крос-платформних мобільних застосунків, зокрема мультимедійних. Її основна перевага полягає у можливості розробки додатків для iOS та Android з єдиною кодовою базою, що значно спрощує процес розробки та підтримки. Однак мультимедійні застосунки, які часто включають в себе обробку медіа-контенту та забезпечення інтерактивного користувацького досвіду, вимагають особливого підходу при розробці на Flutter.

Віджети для мультимедіа. Flutter пропонує бібліотеки для відтворення аудіо, відео та анімацій (наприклад, video player), що дозволяє швидко додавати мультимедіа-функціонал [18 - 20].

Оптимізація продуктивності. Застосування кешування, асинхронної обробки та використання інструментів для оптимізації зображень допомагає зберігати ресурси і забезпечувати високу швидкодію застосунків.

Управління станом. Для синхронізації мультимедійних елементів важливим є управління станом, яке реалізується за допомогою Provider, Riverpod, або Bloc, що підходить для роботи з асинхронними процесами[21].

Вплив залежностей на кількості строк коду застосунку. Багато мультимедійних додатків потребують додаткових бібліотек, що може збільшувати загальний розмір і навантаження застосунку.

Крос-платформні особливості. Деякі мультимедійні функції можуть працювати по-різному на Android та iOS, що вимагає додаткової оптимізації для обох платформ.

Таким чином, для успішної розробки мультимедійних застосунків на Flutter потрібно враховувати його специфічні особливості та ефективно використовувати наявні інструменти для оптимізації.

1.3 Обґрунтування необхідності проведення досліджень за обраною темою

Для оцінювання кількості строк коду мультимедійних застосунків для Android, створених на платформі Flutter, було проведено аналіз існуючих моделей і перевірку їхньої придатності для заданих даних. Особливу увагу приділено лінійним регресійним моделям, які зазвичай використовуються для оцінки програмних метрик. Проте аналіз показав, що застосування лінійної регресії в цьому випадку є недоцільним. Високий рівень мультиколінеарності [22] між предикторами свідчить про сильну залежність між змінними. Це створює проблему для інтерпретації моделі, оскільки важко визначити вплив кожного предиктора окремо. Значення метрик відносно від кількості класів (NOC) мають значення VIF: NOM (32,73), RFC (16,24), що значно перевищують критичний поріг 10. Це вказує на сильну кореляцію цих предикторів з іншими змінними. Через високі значення VIF для метрик NOM, RFC, застосування лінійної регресії є проблематичним. Це підтверджує необхідність перегляду

моделі та використання методів, які можуть зменшити вплив мультиколінеарності.

Результати тесту Мардія вказали на значну багатовимірну асиметрію 564,2896 та ексцес 64,7793, що свідчить про відсутність нормального розподілу метрик (CBO, WMC, DIT).

Для підвищення достовірності моделювання було вирішено застосувати нелінійну регресійну модель із використанням нормалізуючого перетворення. Як оптимальне нормалізуюче перетворення обрано десятковий логарифм (log10), який дозволяє зменшити вплив викидів та вирівняти розподіл метрик до умовно нормального. Наприклад, після логарифмічного перетворення тест Мардія показав покращені результати значення асиметрії 30,8969, значення ексцесу = 26,9705, що відповідає критеріям нормальності.

Метрики CBO, WMC та DIT були обрані як предиктори на основі аналізу мультиколіниарності CBO (3,38), WMC (3,13), DIT (4,42):

– CBO (Coupling Between Objects) характеризує взаємозалежність між класами і є показником складності підтримки.

– WMC (Weighted Methods per Class) оцінює сумарну складність методів класу і є ключовим фактором для визначення обсягу коду.

– DIT (Depth of Inheritance Tree) відображає глибину ієрархії наслідування, що впливає на складність коду.

Використовуючи змінні CBO, WMC та DIT як незалежні змінні, а KLOC - як залежну будо побудовано рівняння лінійної регресії :

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \varepsilon, \quad (1.1)$$

Y – KLOC;

X_1 – змінна (CBO);

X_2 – змінна (WMC);

X_3 – змінна (DIT);

$\beta_0, \beta_1, \beta_2, \beta_3$ – коефіцієнти регресії та є оцінками параметрів, які мають відповідно такі значення: 2,2013, 0,5268, 1,0190, -4,0384.

ε – гаусівська випадкова величина з нульовим математичним сподіванням та середньоквадратичним відхиленням яке дорівнює 7,0540.

На рисунку 1.1 зображено розподіл відхилень.

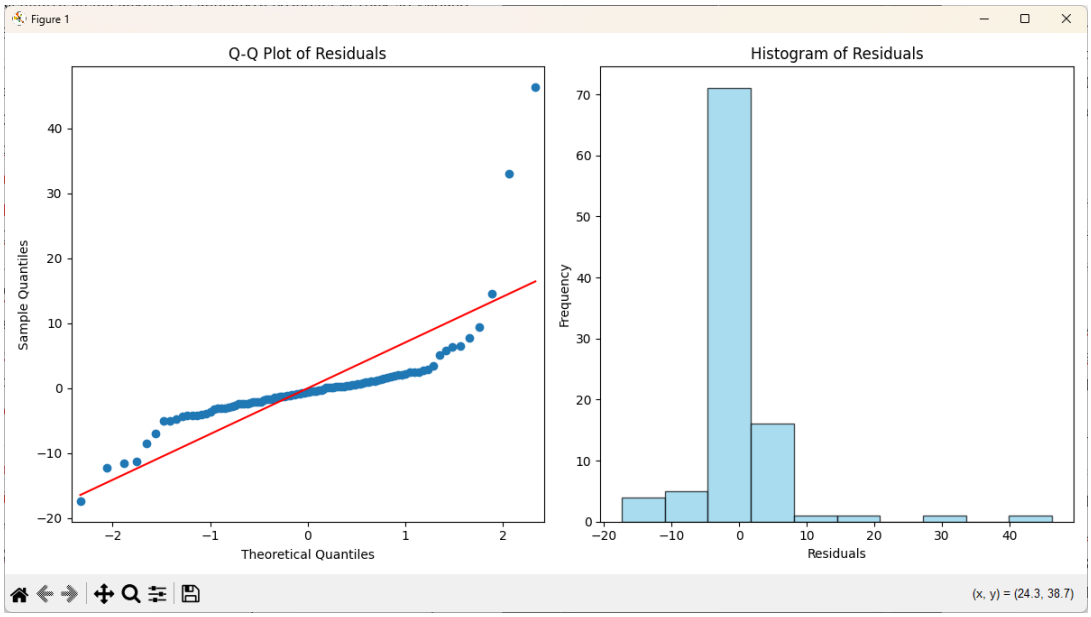

Рисунок 1.1 – Розподіл відхилень

В результаті гіпотеза про нормальній розподіл епсілон відкидається. Це вказує на те що відсутня теоретичне обґрунтування використання лінійної моделі з того що розподіл епсілон відхиляється від нормального в даному випадку.

Також було проведено аналіз існуючих моделей оцінювання KLOC для застосунків на Kotlin та Symfony [4-5].

Нелінійна регресійна модель для фреймворку Symfony [5] використовує нормалізуюче перетворення десяткового логарифму та має параметри:

$\hat{b}_0$= -1,6378, $\hat{b}_1$ = 0,9664, $\hat{b}_2$ = 0,9630, $\hat{b}_3$ = 0,4627.

Нелінійна регресійна модель для Kotlin-застосунків побудована у роботі [4] на основі перетворення у вигляді десяткового логарифму. Модель має вигляд

$$Y = 10^{\varepsilon+\hat{b}_0} X_1^{\hat{b}_1} X_2^{\hat{b}_2} X_2^{\hat{b}_3}, \quad (1.2)$$

де ε дорівнює 0,2366, $\hat{b}_0$, $\hat{b}_1$, $\hat{b}_2$ та $\hat{b}_3$ є оцінками параметрів, які мають відповідно такі значення: -0,2548, -0,4281, 1,1850, -2,5230.

Аналіз показав, що існуючі моделі не можуть бути застосовані до більшості даних з Flutter-застосунків через суттєву різницю в діапазонах метрик:

1. Метрика CBO (Coupling Between Object Classes):

– Flutter: [0,0454, 1,8297] - найнижчі значення серед усіх платформ;

– Kotlin: [0.995, 4,724] - середні значення, ближчі до Flutter.

2. Метрика WMC (Weighted Methods per Class):

– Flutter: [1,5384, 17,9090] - середній діапазон;

– Kotlin: [2,167, 13,020] - майже співпадає з Flutter.

3. Метрика DIT (Depth of Inheritance):

– Flutter: [0,6363, 1,4]- найнижчі значення;

– Kotlin: [2,118, 19,487] - середні значення.

На основі нормалізованих даних було вирішено побудувати нелінійну регресійну модель. Вибір десяткового логарифма як трансформаційного методу обґрунтований наступним:

– Логарифмічне перетворення зменшує вплив великих значень, що важливо для метрик CBO, WMC та DIT із високою варіативністю.

– Цей підхід забезпечує стабільність коефіцієнтів регресії та дозволяє побудувати модель із мінімальною середньою похибкою прогнозів.

Розроблена нелінійна регресійна модель враховує специфіку Flutter-застосунків і залежність KLOC від CBO, WMC, DIT. На відміну від існуючих

моделей для інших платформ (наприклад, Kotlin), вона адаптована до діапазонів метрик, характерних для Flutter, та забезпечує вищу точність оцінки KLOC.

2 РОЗРОБКА НЕЛІНІЙНОЇ РЕГРЕСІЙНОЇ МОДЕЛІ ДЛЯ ОЦІНЮВАННЯ КІЛЬКОСТІ СТРОК КОДУ МУЛЬТИМЕДІЙНИХ ЗАСТОСУНКІВ ДЛЯ ANDROID, ЩО СТВОРЮЮТЬСЯ НА ПЛАТФОРМІ FLUTTER

2.1 Збір та попередня обробка даних метрик Flutter-проектів

Для побудови нелінійної регресійної моделі для оцінювання кількості строк коду мультимедійних застосунків для Android були зібрані дані зі 100 Flutter медіа застосунків з відкритим кодом, розміщених на GitHub. Метрики KLOC, CBO, WMC та DIT на рівні застосунку були отримані за допомогою інструменту Dart Code Metrics [23]. З кожного проекту було отримано наступні відносні значення метрик в залежності від кількості класів:

KLOC – це залежна змінна Y, яку потрібно спрогнозувати за допомогою моделі. CBO – незалежна змінна X_1, WMC – незалежна змінна X_2 та DIT – незалежна змінна X_3 на основі яких буде здійснюватися прогнозування залежної змінної Y.

Для перевірки багатовимірних даних на нормальність був застосований тест Мардія [25], що враховує багатовимірну асиметрію та ексцес. Тест Мардія обчислює два основні параметри Асиметрія та Ексцес. Асиметрія оцінюється на основі відстаней Махаланобіса [26]:

$$d_i^2 = (X_i - \bar{X})^2 S_N^{-1} (X_i - \bar{X}), \tag{2.1}$$

де $\bar{X}$ – вектор середніх значень вибірки для змінних X_1, X_2, X_3, Y;

X_i – вектор спостережень (значення змінних X_1, X_2, X_3, Y, для кожного застосунку);

S_N – коваріаційна матриця.

$$S_N = \frac{1}{N}\sum_{i=1}^{N} (X_i - \bar{X})(X_i - \bar{X})^T. \quad (2.2)$$

У таблиці 2.1 представлено набір даних, де CBO – змінна X_1, WMC – змінна X_2, DIT – змінна X_3, KLOC – змінна Y, D^2 – розрахований квадрат відстаней Махаланобіса.

Таблиця 2.1 – Набір даних

#	URL	KLOC	CBO	WMC	DIT	D^2
1	2	3	4	5	6	7
1	https://github.com/Parth824/multimedia	0,417	0,4	2,4	1	0,8534
2	https://github.com/resulcay/multimedia_app	1,232	0,455	1,545	1	1,4856
3	https://github.com/UmangKaklotar/multimedia_app	0,768	0,5	3,167	1	0,7485
4	https://github.com/IncognitoTabs/multimedia_app	1,96	0,391	7,13	1	0,5705
5	https://github.com/Amirmahdi1380/CINVUplus_multimedia	1,057	0,471	4,941	1	0,2863
6	https://github.com/NeryzaCH/Multimedia-elements_Neryza	1,084	0,222	2,667	1	1,0017
7	https://github.com/multimediayoga/ProjekSkripsiMultimedia	11,894	0,374	6,458	0,872	1,5941
8	https://github.com/rubenAlbuquerque/Artify	3,764	0,438	10,313	1,031	2,8379
9	https://github.com/oapmartins/app-filmes	0,96	0,063	3,556	0,81	4,0813
10	https://github.com/KarimElghamry/chillify	2,041	0,067	6,7	0,933	1,948
11	https://github.com/obnil/flutter_music_app	2,895	0,407	6,681	0,978	0,1872
12	https://github.com/jmshrv/finamp	8,859	0,655	7,612	0,957	1,063
13	https://github.com/Datlyfe/flutter-tunein	1,738	0,311	6,156	0,933	0,3991
14	https://github.com/o-ifeanyi/musicPlayer	3,723	0,286	9,531	0,98	1,9506
15	https://github.com/OpenConsultingGroup/Flutter-Music-App	4,946	0,508	8,889	1,063	2,0358

Продовження таблиці 2.1

1	2	3	4	5	6	7
16	https://github.com/nt4f04uNd/sweyer	15,595	0,75	8,865	0,95	3,1733
17	https://github.com/iamabhishek229313/Flute-Music-Player	0,918	0,28	5	0,88	1,1265
18	https://github.com/Harsh-23/Musify	1,348	0,286	10,714	0,857	3,8219
19	https://github.com/ubuntu-flutter-community/musicpod	14,174	0,599	13,222	0,921	3,9832
20	https://github.com/NimaPayande/Flutter-music-player	0,34	0,4	6,8	1	0,7072
21	https://github.com/KinsomyJS/flutter_netease_music	0,381	0,385	4,077	1	0,3338
22	https://github.com/AIYO77/flutter_cloud_music	20,464	0,284	7,212	0,95	4,673
23	https://github.com/austinried/subtracks	5,599	1	4,763	0,925	6,0247
24	https://github.com/gokadzev/Musify	4,78	0,244	10,422	0,978	2,8045
25	https://github.com/monako97/iCloudMusic	4,004	0,395	4,026	1	0,3009
26	https://github.com/florent37/Flutter-AssetsAudioPlayer	1,707	1,042	13,5	0,792	12,6806
27	https://github.com/mbcorona/flutter_movies	0,939	0,121	3,182	0,939	1,5006
28	https://github.com/mohamadayash22/flutter-movie-app	3,048	0,303	3,006	1	0,6894
29	https://github.com/tusharhow/movie-app	0,621	0,25	2,5	1	0,9664
30	https://github.com/amir-hossein-rajaezadeh/movie_app	2,558	0,476	15,714	0,905	9,8411
31	https://github.com/svprdga/Flutter-Movies-App	0,809	0,275	3,275	1	0,6278
32	https://github.com/mo7amedaliEbaid/movies_riverpod	2,506	0,227	3,567	0,766	4,7877
33	https://github.com/alun1/mu_hua_movie	1,044	0,103	9,793	1,034	5,0982
34	https://github.com/moha-b/Mova	1,586	0,343	3,7	1,029	0,5104

Продовження таблиці 2.1

1	2	3	4	5	6	7
35	https://github.com/Iamzaryab/Flutter-Movie-App-with-Clean-Architecture-and-RiverPod-State-Management	1,73	0,134	3,441	0,819	3,4151
36	https://github.com/Bit-Camp-IO/TMDA-Flutter	7,635	0,287	3,718	1,075	1,8894
37	https://github.com/OmarJ9/movie-finder	2,772	0,519	3,677	1,135	2,3651
38	https://github.com/alyd101/Simple-Flutter-Movie-App	0,431	0,05	2,7	1	2,122
39	https://github.com/mohitnx/MoviesApp	2,837	0,576	2,636	1,182	4,112
40	https://github.com/awesome-academy/movie_db_flutter_bloc_clean	2,195	0,44	3,259	1,026	0,6204
41	https://github.com/lamdv2/flutter_movie_ticket	1,856	0,301	2,055	0,973	1,143
42	https://github.com/mohamedyasser951/MoviesApp	1,063	0,517	1,717	1,4	15,0271
43	https://github.com/sam-nixplay/movies	0,351	0,5	2,333	0,833	3,9149
44	https://github.com/enesakbal/moviegoers	3,921	0,923	4,073	1,214	7,7986
45	https://github.com/dudecoderr/Movie_Booking_App_using_Animation_in_flutter	1,058	0,31	2,517	1	0,8156
46	https://github.com/AtefMMO/Movies-App	1,675	0,304	4,339	1	0,3243
47	https://github.com/HusseinMohamed99/EGY_DEAD	4,327	0,208	3,446	1,042	1,2813
48	https://github.com/mo7amedaliEbaid/movies-app-flutter	4,588	0,456	4,822	0,933	0,3633
49	https://github.com/shebnik/Spotidemo	1,504	0,283	5,755	0,887	0,9114
50	https://github.com/o1298098/Flutter-Movie	36,597	0,275	4,853	0,908	22,4579
51	https://github.com/josemhDev/proyecto_integrado_3	2,136	0,075	6,425	1	2,0235
52	https://github.com/sergio11/quick_reels_flutter	4,482	0,23	5,282	0,866	1,2699
53	https://github.com/ali9653/multimedia_app	0,864	0,129	3,903	1	1,2554
54	https://github.com/jerusalemmoore/capstone	3,206	0,53	5,742	1	0,293
55	https://github.com/ibhavikmakwana/Fluttery-Filmy	0,883	0,29	2,323	1	0,9381
56	https://github.com/bridfish/shuaishuai_film	7,165	0,362	5,889	1,014	0,4035
57	https://github.com/rlechetaup/movies_flutter_2019	0,917	0,361	3,722	1	0,3675
58	https://github.com/mateuss-silva/o-que-assistir	2,106	0,202	3,545	0,788	4,0637
59	https://github.com/Marlon-Paulo-da-Silva/TopFilmes-Flutter	0,419	0,222	1,667	1	1,4575

Продовження таблиці 2.1

1	2	3	4	5	6	7
60	https://github.com/knowbee/filmfan	1,98	0,524	2,798	1,357	12,2347
61	https://github.com/ficiverson/radiocom-flutter	9,006	0,384	8,64	0,767	3,3104
62	https://github.com/silexcorp/Alexander-Playlist	0,503	0,5	4,056	1	0,5087
63	https://github.com/SunsetFrost/Flutter-MediaLibrary	2,103	0,275	2,913	0,986	0,7333
64	https://github.com/bcc-code/bccm-player	3,015	0,803	7,136	0,932	2,5501
65	https://github.com/makineadam/Gramophone	2,153	0,455	4,727	1,136	2,2788
66	https://github.com/ashutoshxhq/media-streaming-app-ui	1,641	0,292	2,167	1	1,0331
67	https://github.com/bipinct/jagnik	1,286	0,5	13,875	1	7,9383
68	https://github.com/Sanotsu/freader-media-player	10,524	0,609	8,177	0,944	1,0642
69	https://github.com/ishandeveloper/Apptitude-Media-Player	0,734	0,313	4,375	1	0,3593
70	https://github.com/shafayathossain/Media-Player-Flutter	0,836	0,313	4,469	1,156	3,1688
71	https://github.com/feilongfl/AVIS	3,681	1,412	6,092	1,008	14,0639
72	https://github.com/iampawan/Flutter-Music-Player	0,409	0,294	5,647	0,882	1,1089
73	https://github.com/deep-gaurav/MusicPiped	3,082	0,471	9,176	1,118	3,9271
74	https://github.com/VN0/BlackHole	5,497	0,475	10,875	1	2,5659
75	https://github.com/amangautam1/flutter-musicplayer	2,671	0,375	6,938	1	0,4281
76	https://github.com/anandnet/Harmony-Music	20,419	0,43	15,104	0,919	6,9528
77	https://github.com/avirias/Grey	3,484	0,525	9,65	1	1,8523
78	https://github.com/devefy/Flutter-Music-Player-UI	0,243	0,333	1,667	1	1,2259
79	https://github.com/felipecastrosales/app_filmes	1,116	0,094	3,734	0,813	3,7167
80	https://github.com/thosakwe/flutter_music_player	0,272	0,5	5	0,833	2,5324
81	https://github.com/StarsWarrior/BlackHole	21,599	0,381	13,497	0,981	6,6835
82	https://github.com/namidaco/namida	63,096	0,295	17,222	0,769	53,5423
83	https://github.com/bukunmialuko/FlutterMusicAppUI	1,415	0,273	2,152	0,939	1,3516
84	https://github.com/moritz-weber/mucke	12,87	1,83	7,839	0,854	31,0335
85	https://github.com/jeromexiong/audio_manager	0,624	0,182	17,909	0,636	21,7879
86	https://github.com/adultcode/Flutter-Web-Music	1,148	0,375	3,375	1,1	1,507

Продовження таблиці 2.1

1	2	3	4	5	6	7
87	https://github.com/xwh817/flutter_music_player	3,927	0,367	7,117	0,958	0,2195
88	https://github.com/shaan-mephobic/The-Phoenix-Project	15,78	0,394	15,058	0,99	7,2488
89	https://github.com/Hamad-Anwar/Neumorphic-Music_Player-Flutter	0,928	0,259	4,185	1	0,5161
90	https://github.com/iamAbhishekkumar/Luna	1,269	0,281	3,875	1	0,4635
91	https://github.com/Ansh-Rathod/Flutter-Musive-app	5,456	0,989	6,25	0,92	5,2449
92	https://github.com/ijinfeng/dreamMusic	11,998	0,341	12,538	0,939	3,5311
93	https://github.com/abuanwar072/Flutter_Music_Player	0,359	0,077	1,538	1	2,3494
94	https://github.com/firgia/FD-Music	1,046	0,156	3,531	0,688	8,6045
95	https://github.com/minikin/audio_player_flutter	0,486	0,893	3,75	0,893	5,5295
96	https://github.com/SayujSujeev/Movie-ticket-App-UI	1,245	0,4	2,55	1	0,7792
97	https://github.com/kleberandrade/movies-flutter	0,35	0,909	3,955	1	4,2537
98	https://github.com/leandrucarvalho/flutter_the_movie_list	0,443	0,25	2,813	0,875	2,0171
99	https://github.com/fleetimee/accesscode-talker	1,741	0,045	2,955	0,909	2,412
100	https://github.com/maickom88/the_batman_app_flutter	0,394	0,105	1,737	1	2,043

Формула для асиметрії та ексцесу Мардіа виглядає наступним чином [25]:

$$\beta_{1,k} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[\left(\mathrm{X}_i - \overline{\mathrm{X}}\right)^T \mathrm{S}_N^{-1}\left(\mathrm{X}_j - \overline{\mathrm{X}}\right)\right]^3, \tag{2.3}$$

$$\beta_{2,k} = \frac{1}{N}\sum_{i=1}^{N}\left[\left(\mathrm{X}_i - \overline{\mathrm{X}}\right)^T \mathrm{S}_N^{-1}\left(\mathrm{X}_i - \overline{\mathrm{X}}\right)\right]^2, \tag{2.4}$$

d^2 - відстаней Махаланобіса (2.1),

$\mathrm{X} - k \times 1$ вектор випадкових величин, $\mathrm{X} = (X_1, X_2, \dots, X_k)^T$,

$\mathrm{S}_N -$ зміщена вибіркова коваріаційна матриця X.

Для $\beta_{1,k}$ потрібно обчислити тестову статистику [23]:

$$\frac{N}{6}\beta_{1,k}, \tag{2.5}$$

де наближений розподіл χ^2 з $k(k+1)(k+2)/6$ ступенями свободи та рівнем значущості $\alpha = 0{,}005$.

Для $\beta_{2,k}$ у якості тестової статистики $z_{2,k}$ використовуємо $1-\alpha$ квантиль нормального закону розподілу з математичним сподіванням $k(k+2)$ та дисперсією $8k(k+2)/N$.

Щоб перевірити відхилення розподілу від нормального, потрібно порівняти значення статистики (2.5) з квантилем $\chi^2_{(k+1)(k+2)/6,\alpha}$, а $\beta_{2,k}$ з $z_{2,k}$. Якщо значення (2.5) більше за квантиль $\chi^2_{(k+1)(k+2)/6,\alpha}$ або, якщо значення $\beta_{2,k}$ більше за $z_{2,k}$, то багатовимірний розподіл даних не є нормальним.

Для перевірки багатовимірних даних на нормальність був застосований тест Мардія [24], що враховує багатовимірну асиметрію та ексцес:

– Тестова статистика для асиметрії Мардія (2.3): 564,2896, критичне значення: 39,9968;

– Ексцес Мардія (2.4): 64,7793, критичне значення: 27,5691.

Результати тесту показали, що розподіл даних не відповідає гаусівському [27], через те, що ексцес більше за критичне значення. Тому, щоб визначити наявність викидів у даних, було виконано нормалізацію даних за допомогою десяткового логарифму.

Тестову статистику для асиметрії Мардія (2.3) на нормалізованих даних було обчислено як 30,8969, і порівняно з критичним значенням 39,9968 Розраховане значення ексцесу Мардія (2.4) становить 26,9705, і воно порівняне з критичним значенням 27,5691.

Проведений аналіз показав, що дані мають значну асиметрію і не відповідають нормальному розподілу, що підтверджено тестом нормальності

Мардія. У зв’язку з цим було вирішено застосувати нелінійну регресійну модель із нормалізуючим перетворенням у вигляді десяткового логарифму, що дозволяє достовірно описати взаємозалежності між метриками.

У зв’язку з цим було вирішено застосувати нелінійну регресійну модель із нормалізуючим перетворенням, що дозволяє точніше описати взаємозалежності між метриками. Використання десяткового логарифму як нормалізуючого перетворення допомогло врахувати асиметричний розподіл, що забезпечило рівномірніший розподіл даних та підвищило достовірність прогнозів. Спочатку потрібно знайти і вилучити викиди із вихідних даних:

При побудові нелінійної регресійної моделі потрібно використовувати метод нормалізуючих перетворень. Спочатку потрібно знайти і вилучити викиди із вихідних даних [24]:

1. Проводимо нормалізацію даних за допомогою десяткового логарифмування значень метрик.

2. Розраховуємо квадрат відстані Махаланобіса (2.1) [26]:

3. Розраховуємо коваріаційну матрицю S_N для 4 змінних на нормалізованих даних (2.2).

4. Знаходимо зворотну матрицю ${S_N}^{-1}$.

5. Для кожного спостереження і розраховуємо квадрат відстані Махаланобіса на нормалізованих даних (2.1).

6. Обчислюємо тестову статистику для кожного спостереження [28]:

$$T_i = (N-k)Nd_i^2/\big((N^2-1)k\big). \tag{2.6}$$

7. Обчислюємо $F_{\text{крит}}$ як [29]:

$$F_{\text{крит}} = F(\alpha, 4, n-4), \tag{2.7}$$

де α – рівень значущості,

З – кількість ступенів вільності,

n – розмір вибірки.

8. Якщо T_i (2.6) $>$ $F_{\text{крит}}$ (2.7), то і–те спостереження є викидом і його вилучаємо з вибірки.

9. Повторяємо крок 2 – 8 поки не буде вилучено усі викиди.

10. Далі потрібно побудувати рівняння нелінійної регресії (1.2) та розрахувати інтервал прогнозування який обчислюється як:

$$10^{\left(\hat{Z}_Y \pm t_{\frac{\alpha}{2},\nu} S_{Z_Y}\left\{1+\frac{1}{N}+\left(\mathbf{z}_X^+\right)^T \mathbf{S}_Z^{-1}\left(\mathbf{z}_X^+\right)\right\}^{\frac{1}{2}}\right)},$$

де $\hat{z}_{y_i}$ – значення, розраховані за рівнянням регресії;

$t_{\alpha/2,n-4}$ – квантіль t-розподілу Стьюдента;

α – рівень статистичної значущості (0,005);

n – кількість елементів в вибірці;

$S_{Z_Y}^2 = \frac{1}{\nu}\sum_{i=1}^{N}(Z_{Y_i} - \hat{Z}_{Y_i})^2$ – залишкова дисперсія.

Якщо є точки даних які виходять за інтервал прогнозування то потрібно вилучити їх із вибірки.

Нижче представлено видалення викидів для нормалізованих даних та розраховані тестові статистиці для кожного кроку.

Ітерація 1 ($F_{\text{крит}} = 3{,}9741$):

– Видалений 85 застосунок TS (4,7813>$F_{\text{крит}}$);

Ітерація 2 (Інтервали):

– Видалений 20 застосунок 0,34 < 0,5067;

– Видалений 50 застосунок 36,597>12,8082;

– Видалений 80 застосунок 0,272<0,3435;

– Видалений 82 застосунок 63,096>59,8748.

Ітерація 3 (Інтервали):

– Видалений 22 застосунок 20,464>15,7548;

– Видалений 67 застосунок 1,286<1,446;

– Видалений 72 застосунок 0,409<0,4534;

– Видалений 97 застосунок 0,35<0,4074.

Ітерація 4 (Інтервали):

– Видалений 21 застосунок 0,381<0,4337;

– Видалений 26 застосунок 1,707<1,9273;

– Видалений 36 застосунок 7,635>6,9316.

Ітерація 5 (Інтервали):

– Видалений 18 застосунок 1,348<1,3542;

– Видалений 62 застосунок 0,503<0,534;

– Видалений 95 застосунок 0,486<0,5903.

Ітерація 6 (Інтервали):

– Видалений 30 застосунок 2,558<3,031;

– Видалений 43 застосунок 0,351<0,3584;

Ітерація 7 (Інтервали):

– Видалений 47 застосунок 4,327>4,0281.

Після видалення викидів за 7 ітерації залишилась вибірка із 82 застосунків. Більше викидів не виявлено, тому можна зробити висновок, що попередня обробка даних завершена.

2.2 Побудова моделі для оцінювання кількості строк коду мультимедійних застосунків для Android

Після попередньої обробки даних і видалення викидів було сформовано вибірку з 82-х застосунків. Надалі дані були розділені на навчальну та тестову вибірки для побудови й оцінювання моделі. Навчальна вибірка – 60% від загальної вибірки (49-и застосунків), використана для побудови моделі. Тестова вибірка – 40% від загальної вибірки (33-х застосунків), використана для перевірки розміру моделі. Під час розподілу було враховано, щоб у навчальній вибірці були мінімальні та максимальні значення метрик. У тестовій вибірці залишено значення метрик, близькі до граничних першої вибірки. Це дозволяє забезпечити точність оцінки моделі. У таблиці 2.2 представлено вибірку для побудови моделі.

Таблиця 2.2 – Вибірка для побудови моделі

#	URL	KLOC	CBO	WMC	DIT
1	2	3	4	5	6
1	https://github,com/devefy/Flutter-Music-Player-UI	0,2430	0,3333	1,6667	1,0000
2	https://github,com/StarsWarrior/BlackHole	21,5990	0,3806	13,4968	0,9806
3	https://github,com/fleetimee/accesscode-talker	1,7410	0,0455	2,9545	0,9091
4	https://github,com/moritz-weber/mucke	12,8700	1,8298	7,8389	0,8541
5	https://github,com/abuanwar072/Flutter_Music_Player	0,3590	0,0769	1,5385	1,0000
6	https://github,com/anandnet/Harmony-Music	20,4190	0,4296	15,1037	0,9185
7	https://github,com/firgia/FD-Music	1,0460	0,1563	3,5313	0,6875
8	https://github,com/mohamedyasser951/Movies App	1,0630	0,5167	1,7167	1,4000
9	https://github,com/lamdv2/flutter_movie_ticket	1,8560	0,3014	2,0548	0,9726
10	https://github,com/IncognitoTabs/multimedia_app	1,9600	0,3913	7,1304	1,0000
11	https://github,com/austinried/subtracks	5,5990	1,0000	4,7632	0,9254
12	https://github,com/SayujSujeev/Movie-ticket-App-UI	1,2450	0,4000	2,5500	1,0000
13	https://github,com/OpenConsultingGroup/Flutter-Music-App	4,9460	0,5079	8,8889	1,0635
14	https://github,com/maickom88/the_batman_app_flutter	0,3940	0,1053	1,7368	1,0000
15	https://github,com/KarimElghamry/chillify	2,0410	0,0667	6,7000	0,9333

Продовження таблиці 2.2

1	2	3	4	5	6
16	https://github,com/Ansh-Rathod/Flutter-Musive-app	5,4560	0,9886	6,2500	0,9205
17	https://github,com/AtefMMO/Movies-App	1,6750	0,3036	4,3393	1,0000
18	https://github,com/NeryzaCH/Multimedia-elements_Neryza	1,0840	0,2222	2,6667	1,0000
19	https://github,com/ashutoshxhq/media-streaming-app-ui	1,6410	0,2917	2,1667	1,0000
20	https://github,com/felipecastrosales/app_filmes	1,1160	0,0938	3,7344	0,8125
21	https://github,com/Datlyfe/flutter-tunein	1,7380	0,3111	6,1556	0,9333
22	https://github,com/monako97/iCloudMusic	4,0040	0,3947	4,0263	1,0000
23	https://github,com/deep-gaurav/MusicPiped	3,0820	0,4706	9,1765	1,1176
24	https://github,com/bukunmialuko/FlutterMusicAppUI	1,4150	0,2727	2,1515	0,9394
25	https://github,com/alun1/mu_hua_movie	1,0440	0,1034	9,7931	1,0345
26	https://github,com/shebnik/Spotidemo	1,5040	0,2830	5,7547	0,8868
27	https://github,com/sergio11/quick_reels_flutter	4,4820	0,2297	5,2823	0,8660
28	https://github,com/obnil/flutter_music_app	2,8950	0,4066	6,6813	0,9780
29	https://github,com/leandrucarvalho/flutter_the_movie_list	0,4430	0,2500	2,8125	0,8750
30	https://github,com/oapmartins/app-filmes	0,9600	0,0635	3,5556	0,8095
31	https://github,com/ali9653/multimedia_app	0,8640	0,1290	3,9032	1,0000
32	https://github,com/jerusalemmoore/capstone	3,2060	0,5303	5,7424	1,0000
33	https://github,com/Parth824/multimedia	0,4170	0,4000	2,4000	1,0000
34	https://github,com/alyd101/Simple-Flutter-Movie-App	0,4310	0,0500	2,7000	1,0000
35	https://github,com/ficiverson/radiocom-flutter	9,0060	0,3837	8,6395	0,7674
36	https://github,com/svprdga/Flutter-Movies-App	0,8090	0,2750	3,2750	1,0000
37	https://github,com/Iamzaryab/Flutter-Movie-App-with-Clean-Architecture-and-RiverPod-State-Management	1,7300	0,1339	3,4409	0,8189
38	https://github,com/Hamad-Anwar/Neumorphic-Music_Player-Flutter	0,9280	0,2593	4,1852	1,0000
39	https://github,com/mateuss-silva/o-que-assistir	2,1060	0,2020	3,5455	0,7879
40	https://github,com/mbcorona/flutter_movies	0,9390	0,1212	3,1818	0,9394
41	https://github,com/ijinfeng/dreamMusic	11,9980	0,3408	12,5382	0,9395
42	https://github,com/amangautam1/flutter-musicplayer	2,6710	0,3750	6,9375	1,0000
43	https://github,com/iamabhishek229313/Flute-Music-Player	0,9180	0,2800	5,0000	0,8800
44	https://github,com/enesakbal/moviegoers	3,9210	0,9231	4,0726	1,2137

Продовження таблиці 2.2

1	2	3	4	5	6
45	https://github,com/avirias/Grey	3,4840	0,5250	9,6500	1,0000
46	https://github,com/iamAbhishekkumar/Luna	1,2690	0,2813	3,8750	1,0000
47	https://github,com/bcc-code/bccm-player	3,0150	0,8027	7,1361	0,9320
48	https://github,com/Sanotsu/freader-media-player	10,5240	0,6093	8,1767	0,9442
49	https://github,com/mo7amedaliEbaid/movies_riverpod	2,5060	0,2270	3,5674	0,7660

У таблиці 2.3 представлено вибірку для тестування моделі.

Таблиця 2.3 – Вибірка для тестування моделі

#	URL	KLOC	CBO	WMC	DIT
1	2	3	4	5	6
1	https://github.com/resulcay/multimedia_app	1,2320	0,4545	1,5455	1,0000
2	https://github.com/knowbee/filmfan	1,9800	0,5238	2,7976	1,3571
3	https://github.com/tusharhow/movie-app	0,6210	0,2500	2,5000	1,0000
4	https://github.com/Amirmahdi1380/CINVUplus_multimedia	1,0570	0,4706	4,9412	1,0000
5	https://github.com/SunsetFrost/Flutter-MediaLibrary	2,1030	0,2754	2,9130	0,9855
6	https://github.com/mohitnx/MoviesApp	2,8370	0,5758	2,6364	1,1818
7	https://github.com/moha-b/Mova	1,5860	0,3429	3,7000	1,0286
8	https://github.com/gokadzev/Musify	4,7800	0,2444	10,4222	0,9778
9	https://github.com/josemhDev/proyecto_integrado_3	2,1360	0,0750	6,4250	1,0000
10	https://github.com/adultcode/Flutter-Web-Music	1,1480	0,3750	3,3750	1,1000
11	https://github.com/nt4f04uNd/sweyer	15,5950	0,7500	8,8650	0,9500
12	https://github.com/mohamadayash22/flutter-movie-app	3,0480	0,3034	3,0056	1,0000
13	https://github.com/ishandeveloper/Apptitude-Media-Player	0,7340	0,3125	4,3750	1,0000
14	https://github.com/shaan-mephobic/The-Phoenix-Project	15,7800	0,3942	15,0577	0,9904
15	https://github.com/bridfish/shuaishuai_film	7,1650	0,3623	5,8889	1,0145

Продовження таблиці 2.3

1	2	3	4	5	6
16	https://github.com/Marlon-Paulo-da-Silva/TopFilmes-Flutter	0,4190	0,2222	1,6667	1,0000
17	https://github.com/ibhavikmakwana/Fluttery-Filmy	0,8830	0,2903	2,3226	1,0000
18	https://github.com/awesome-academy/movie_db_flutter_bloc_clean	2,1950	0,4397	3,2586	1,0259
19	https://github.com/shafayathossain/Media-Player-Flutter	0,8360	0,3125	4,4688	1,1563
20	https://github.com/xwh817/flutter_music_player	3,9270	0,3667	7,1167	0,9583
21	https://github.com/VN0/BlackHole	5,4970	0,4750	10,8750	1,0000
22	https://github.com/dudecoderr/Movie_Booking_App_using_Animation_in_flutter	1,0580	0,3103	2,5172	1,0000
23	https://github.com/OmarJ9/movie-finder	2,7720	0,5188	3,6767	1,1353
24	https://github.com/makineadam/Gramophone	2,1530	0,4545	4,7273	1,1364
25	https://github.com/multimediayoga/ProjekSkripsiMultimedia	11,8940	0,3737	6,4579	0,8721
26	https://github.com/ubuntu-flutter-community/musicpod	14,1740	0,5993	13,2219	0,9205
27	https://github.com/jmshrv/finamp	8,8590	0,6549	7,6118	0,9569
28	https://github.com/rubenAlbuquerque/Artify	3,7640	0,4375	10,3125	1,0313
29	https://github.com/rlechetaup/movies_flutter_2019	0,9170	0,3611	3,7222	1,0000
30	https://github.com/feilongfl/AVIS	3,6810	1,4122	6,0916	1,0076
31	https://github.com/o-ifeanyi/musicPlayer	3,7230	0,2857	9,5306	0,9796
32	https://github.com/UmangKaklotar/multimedia_app	0,7680	0,5000	3,1667	1,0000
33	https://github.com/mo7amedaliEbaid/movies-app-flutter	4,5880	0,4556	4,8222	0,9333

Оскільки емпіричні дані дуже рідко бувають нормально розподіленими, для розв'язання поставленої задачі прогнозування рядків коду на основі CBO, WMC та DIT треба побудувати рівняння нелінійної регресії, використовуючи метод нормалізуючих перетворень (десятковий логарифм). Для цього спочатку побудуємо рівняння лінійної регресії (1.1).

Оцінка параметрів моделі $\beta_0, \beta_1, \beta_2, \beta_3$ буде здійснена методом найменших квадратів :

$$\hat{\beta} = (\mathrm{X}^{\mathrm{T}}\mathrm{X})^{-1}\mathrm{X}^{\mathrm{T}}\mathrm{y}, \quad (2.8)$$

де $\hat{\beta}$ – звичайна оцінка найменших квадратів;

X – змінна матричного регресора X;

T – транспонування матриці;

y – вектор значення змінної відгуку.

за допомогою зворотного перетворення отримаємо модель нелінійної регресії:

Рівняння (1.2) відображає залежність між KLOC та метриками CBO, WMC, DIT. Ця залежність визначається через побудовану регресійну модель, яка дозволяє здійснювати оцінювання на основі вказаних параметрів

Після обчислень були отримані наступні значення коефіцієнтів:

β_0 = -0,2182, β_1 = 0,4599, β_2 = 1,1427, β_3= -1,1017, ε – гаусівська випадкова величина з нульовим математичним сподіванням та середньоквадратичним відхиленням, оцінка якого дорівнює 0,2485

Таким чином, остаточна модель має вигляд наступного рівняння:

$$\hat{Y} = 10^{-0,2182} {X_1}^{0,4599} {X_2}^{1,1427} {X_2}^{-1,1017}.$$

Розраховуємо показники якості для побудованої нелінійної регресійної моделі. Аналіз якості та перевірка адекватності побудованої моделі буде виконана за допомогою критеріїв якості R^2, MMRE та PRED.

Результати оцінки для навчальної та тестової вибірок:

– Навчальна вибірка: R^2 = 0,6019, MMRE = 0,5043, PRED(0.25) = 0,3265.

– Тестова вибірка: R^2 = 0,5594, MMRE = 0,4712, PRED(0.25) = 0,2727.

Розроблена нелінійна регресійна модель продемонструвала задовільну якість прогнозування як на навчальній, так і на тестовій вибірках.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ КІЛЬКОСТІ СТРОК КОДУ МУЛЬТИМЕДІЙНИХ ЗАСТОСУНКІВ ДЛЯ ANDROID, ЩО СТВОРЮЮТЬСЯ НА ПЛАТФОРМІ FLUTTER

3.1 Вимоги до програмного забезпечення

На основі проведеного аналізу предметної області та нелінійної регресійної моделі було сформульовано вимоги до програмного забезпечення для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter програмне забезпечення отримало назву «FlutterMediaMetrics»

Технічне завдання на розробку програмного забезпечення «FlutterMediaMetrics» представлено у повному обсязі у Додатку А.

"FlutterMediaMetrics" розроблено для полегшення процесів планування та управління створенням ПЗ для роботи з мультимедіа. Програмне забезпечення автоматизує обчислення оцінювання кількості строк коду мультимедійних застосунків, які розробляються на Flutter.

Програма "FlutterMediaMetrics" призначена для аналізу та оцінки метрик мультимедійних проектів, які створюються за допомогою Flutter. Основні функції включають підрахунок і візуалізацію таких характеристик, як CBO, WMC, DIT та рядків коду (SLOC), а також побудову регресійної моделі для прогнозування розміру майбутніх проектів.

Програмне забезпечення повинно забезпечувати наступні функції:

1. Робота з даними

– Зберігання метрик (SLOC, CBO, WMC, DIT) для застосунків у форматі CSV.

– Перегляд списку застосунків користувача.

– Редагування списку застосунків.

2. Побудова моделей та оцінка її якості

– Побудова регресійної моделі для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter.

Аналіз якості моделі на основі статистичних показників, включаючи:

– коефіцієнт детермінації (R^2),

– середню відносну похибку (MMRE),

– рівень прогнозування PRED(0.25).

3. Прогнозування розміру у SLOC для нових додатків з використанням метрик CBO, WMC та DIT

Вхідні дані:

– Список метрик для Flutter-проектів у форматі CSV.

– SLOC (Source Lines of Code)

– CBO (Coupling Between Objects)

– WMC (Weighted Methods per Class)

– DIT (Depth of Inheritance Tree)

Вихідні дані:

– Параметри побудованої лінійної моделі.

– Параметри нелінійної регресійної моделі.

– Показники якості нелінійної регресійної моделі.

– Прогнозні значення розміру у SLOC.

– Допустимі діапазони значень CBO, WMC та DIT

3.2 Архітектура програмного забезпечення

Для створення програмного забезпечення було використано архітектурний шаблон MVC (Model-View-Controller) [30]. Цей підхід був обраний для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter, завдяки його здатності розділяти завдання та покращувати підтримку, масштабованість і повторне використання коду. MVC розбиває систему на три основні компоненти: модель, представлення та контролер. Ось як вони функціонують:

Модель відповідає за представлення даних і бізнес-логіки системи. Вона забезпечує доступ до даних, пов'язаних з проектами, метриками та параметрами побудованої регресії, а також їх маніпулювання та зберігання в базі даних, підтримуючи цілісність і узгодженість цих даних.

Контролер діє як посередник між моделлю та представленням. Він обробляє вхідні дані від користувача, взаємодіє з моделлю для оновлення стану системи та оновлює представлення відповідно до результатів взаємодії з моделлю, забезпечуючи логіку взаємодії між цими двома компонентами.

Взаємодія між компонентами може бути проілюстрована діаграмами компонентів і макетами UML, які відображають структурну організацію системи та взаємодію між компонентами. Використання архітектури MVC надає кілька переваг, включаючи чітке розділення відповідальностей, модульність, повторне використання коду, легке додавання нових функцій і полегшене модульне тестування окремих компонентів.

На рисунку 3.1 представлено загальну діаграму компонентів програмного забезпечення для оцінювання розміру медіа застосунків, що створюються за допомогою фреймворку Flutter.

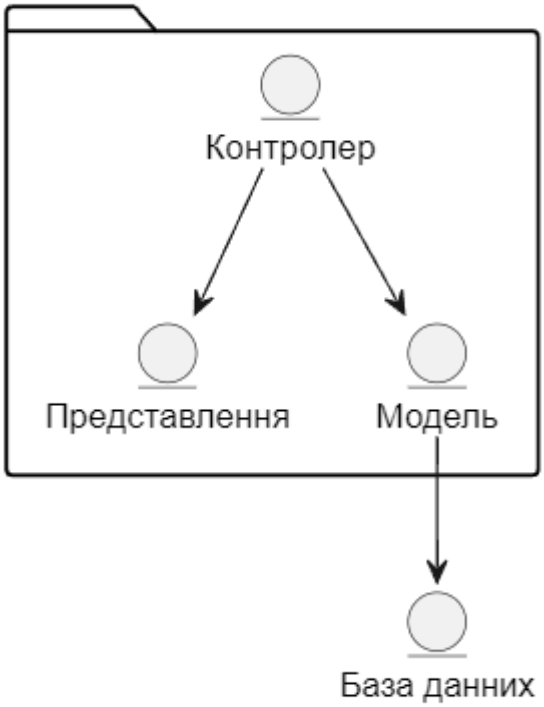

Рисунок 3.1 – Діаграма компонентів програмного забезпечення для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

У цій діаграмі зображено архітектуру MVC програмного забезпечення та її співвідношення з локальною базою даних. Контролер взаємодіє як з представленням, так і з моделлю. Модель зберігається у базі даних. Рисунок 3.2 показує послідовність взаємодії між компонентами.

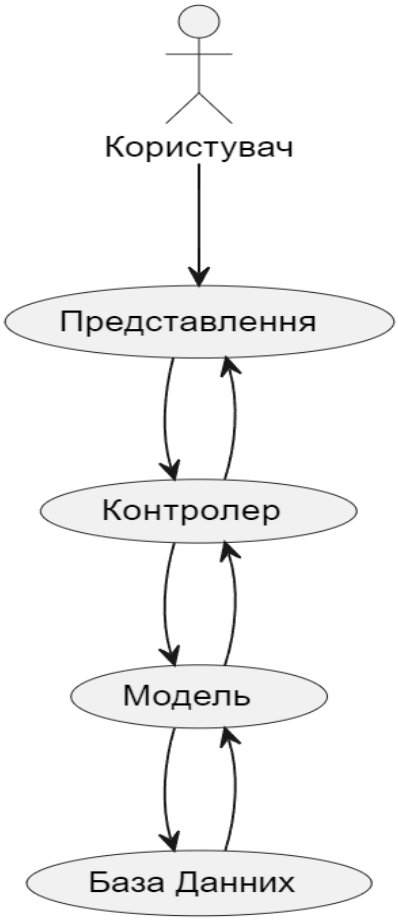

Рисунок 3.2 – Діаграма взаємодії компонентів програмного забезпечення для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Ця діаграма зображує архітектуру взаємодії компонентів, яка представляє наступне:

– Користувач взаємодіє з Представленням, яке є інтерфейсом користувача.

– Представлення передає введені користувачем дані Контролеру і отримує оновлений стан для відображення.

– Контролер вирішує потік даних між Представленням та Моделлю. Він отримує дані від Представлення, опрацьовує їх і передає у Модель, а також приймає оновлені дані від Моделі й передає їх у Представлення.

– Модель містить логіку додатку та зберігає його стан. Вона взаємодіє з Базою Даних для зчитування та запису даних.

– База Даних зберігає постійні дані.

3.3 Модель предметної області програмного забезпечення

Оцінювання кількості строк коду спрямовані на поліпшення процесу розробки, допомагаючи аналізувати та управляти якістю свого коду, а також прогнозувати обсяг робіт і ресурсів, необхідних для проектування та реалізації програмних проектів на платформі Flutter.

На рисунку 3.3 представлено модель предметної області для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Дана модель предметної області описує систему для побудови регресійної моделі з метою оцінювання кількості строк коду мультимедійних застосунків для

A

n Основні сутності моделі:

d Проект – представляє програмний проект, що аналізується. Має атрибути,

я які відповідають за різні кількісні метрики проекту: SLOC, CBO, WMC, DIT.

o Модель – математична регресійна модель, яка будується для прогнозування

ї розміру програмного забезпечення на основі метрик проекту. Має коефіцієнти

d регресії та методи для побудови і використання моделі.

, Метрики – перелік можливих метрик програмного проекту, які виступають незалежними змінними при побудові регресії.

щ Якість - критерій якості побудованої моделі (R^2, *MMRE*, *PRED*).

o

c

Рисунок 3.3 – Модель предметної області для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Зв'язки між сутностями відображають, що:

– Проект має метрики

– Модель використовує ці метрики як вхідні дані

– Модель оцінює кількость строк коду мультимедійних застосунків

– Модель розраховує коефіцієнти

3.4 Use Case модель програмного забезпечення

Діаграма варіантів використання описує функціональне призначення програмного забезпечення, тобто те, що програма повинна робити. Діаграми варіантів використання розроблюються для того, щоб:

– визначити загальні кордони та предметну галузь майбутнього програмного забезпечення;

– визначити функціональні вимоги до проектованого програмного забезпечення;

– розробити початкову концептуальну модель майбутнього програмного забезпечення;

– підготувати початкову документацію для взаємодії між всіма учасниками розробки програмного забезпечення.

На рисунку 3.4 представлена Use Case модель, що описує варіанти використання системи оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter. В якості учасників буде – Користувач.

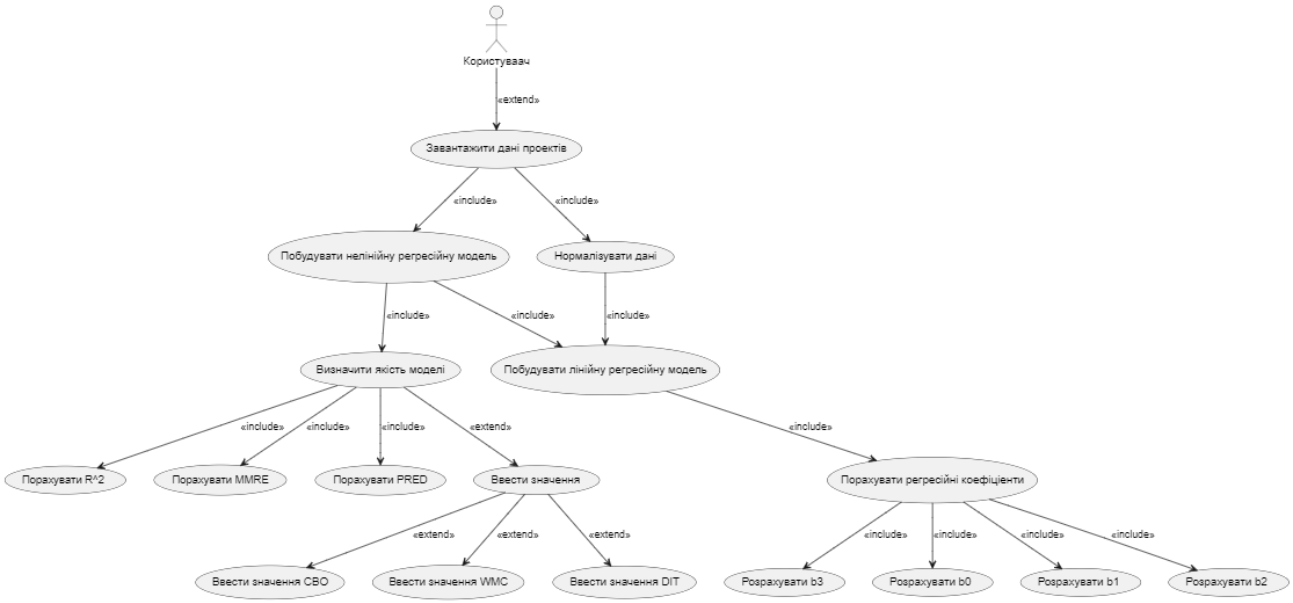

Рисунок 3.4 - Use Case діаграма для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter.

Ця Use Case діаграма моделює основні процеси, пов'язані з оцінюванням кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter.

В моделі виділено такі ключові процеси:

Завантажити данні проєкта - завантаження вихідних даних для подальшого аналізу.

Нормалізувати дані - первинна обробка та підготовка набору даних; включається в процес імпорту даних.

Побудувати лінійну регресійну модель – побудова лінійну регресійну модель на основі підготовлених даних. Включає розрахунок коефіцієнтів регресії.

Побудувати нелінійну регресійну модель – створення моделі нелінійної регресії на основі лінійної моделі.

Порахувати регресійні коефіцієнти – розрахунок $\beta_0, \beta_1, \beta_2, \beta_3$ за формулою (2.8)

Визначення якості моделі – оцінка достовірності та адекватності отриманої регресійної моделі за допомогою критеріїв: R^2, *MMRE, PRED(0.25).*

Ввести значення - отримання прогнозних значень якості проектів на основі побудованої регресійної моделі за допомогою метрик CBO, WMC та DIT.

3.5 Побудова діаграми діяльності

У процесі розроблення програмного забеспечення "FlutterMediaMetrics", призначеного для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter, були підготовлені декілька артефактів для формалізації функціоналу та архітектури системи. Зокрема, було сформовано діаграму діяльності для графічного відображення основних сценаріїв використання програмного забезпечення.

На рисунку 3.5 представлено діаграму діяльності «FlutterMediaMetrics» для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Рисунок 3.5 – Діаграма діяльності «FlutterMediaMetrics» для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Цей процес аналізу даних розпочинається з імпорту проектних даних, необхідних для подальшого дослідження. Перш ніж продовжити, важливо перевірити, чи дані нормалізовані. Якщо нормалізація ще не була проведена, застосовується метод нормалізуючих перетворень, такий як десятковий логарифм.

β0, β1,β2,β3.

Побудувати лінійну регресійну модель на основі підготовлених даних яка включає розрахунок коефіцієнтів регресії.

Потім побудувати нелінійну регресійну модель на основі лінійної.

Потім проводиться оцінка якості побудованої моделі.

На завершення здійснюються прогнози на основі розробленої регресійної моделі, після чого процес завершується.

3.6 Ескізний проект програмного забезпечення

Був підготовлений концептуальний проект програмного забезпечення після аналізу вимог до нього. Рисунки 3.6 – 3.8 включають прототипи вікон програми.

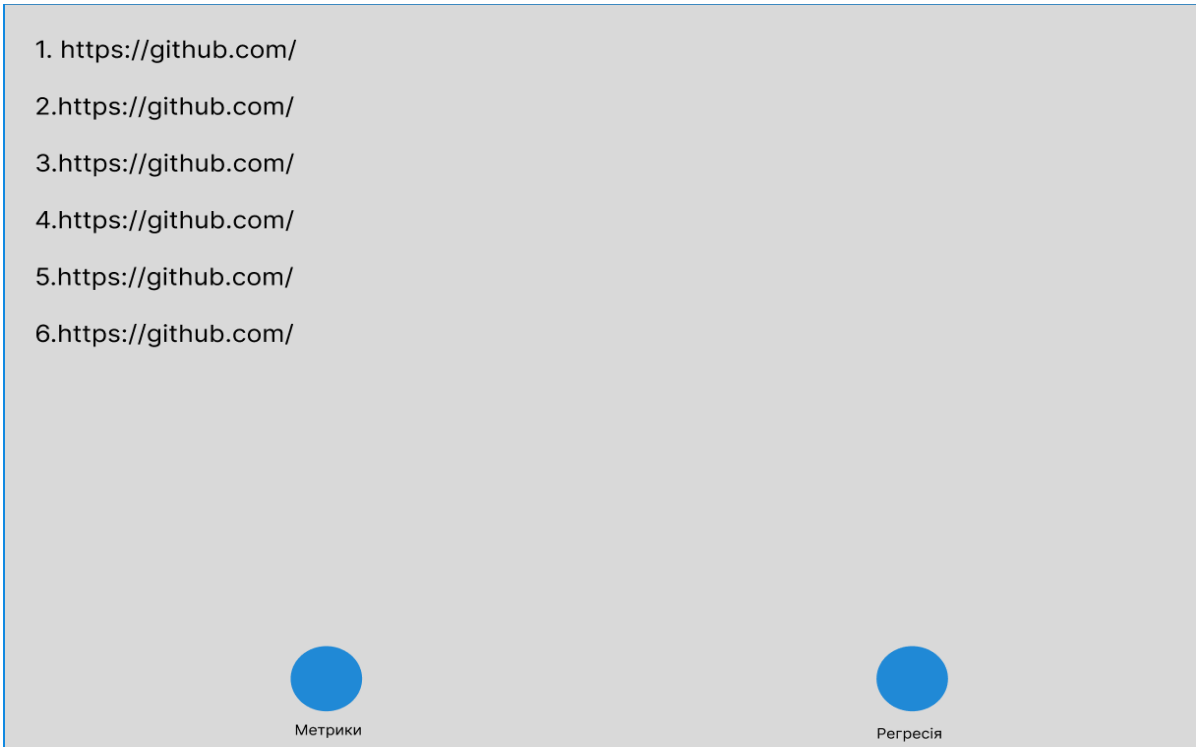

Рисунок 3.6 – Прототип головного екрана «FlutterMediaMetrics» для для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Головне вікно має нижню панель з опціями «Метрики», «Регресія». Вище панелі знаходяться завантаженні мультимедійні проекти. Кожен проект пронумерований і містить URL послання.

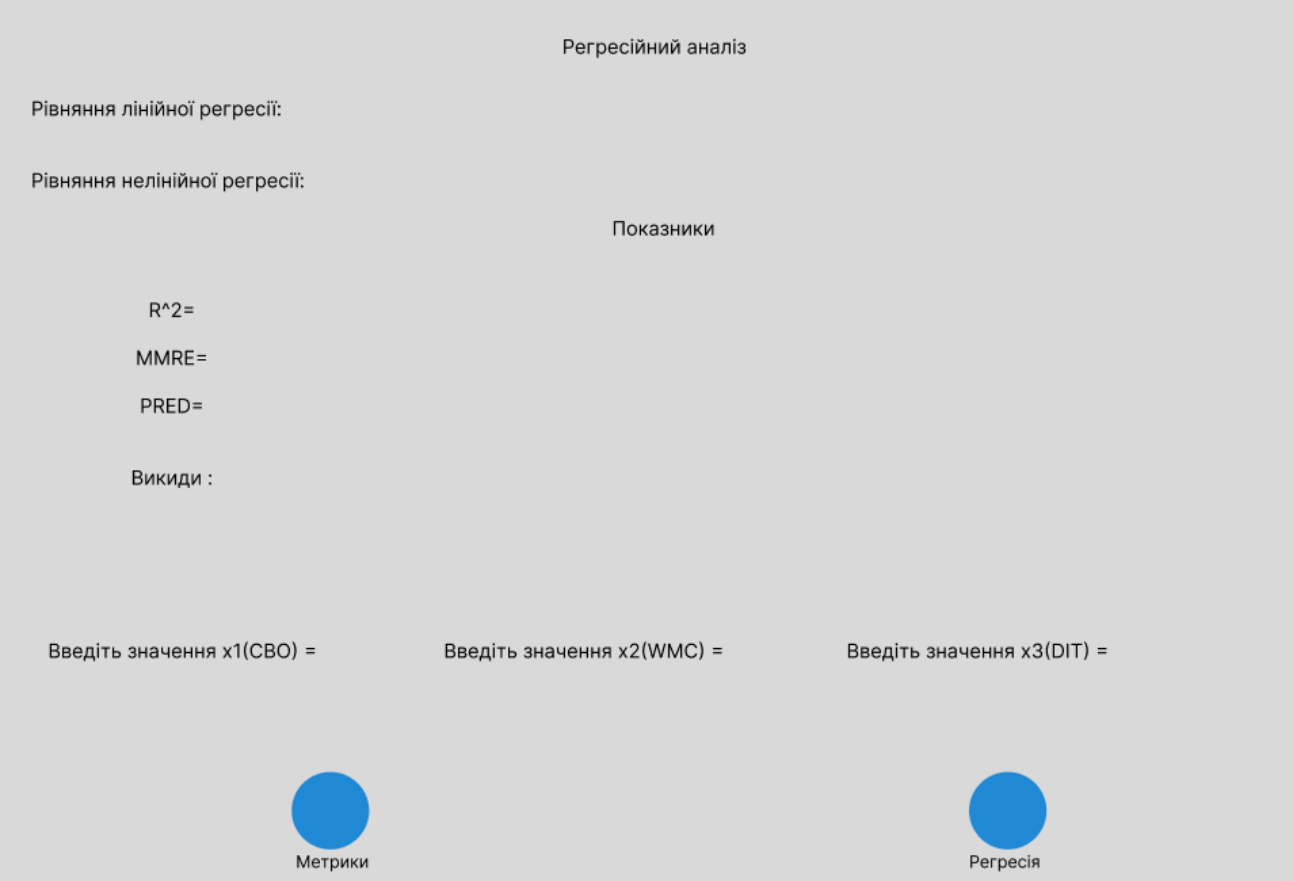

Рисунок 3.7 – Прототип екрана «Регресія» «FlutterMediaMetrics» для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Вкладка «Регресія» показує рівень лінійної та не лінійної регресійної моделі. Нижче наведено метрики якості нелінійної моделі: R^2, середня відносна похибка *MMRE* та відсоток передбачень у допустимому діапазоні PRED. Далі йде підрахунок кількості класів та кількості методів.

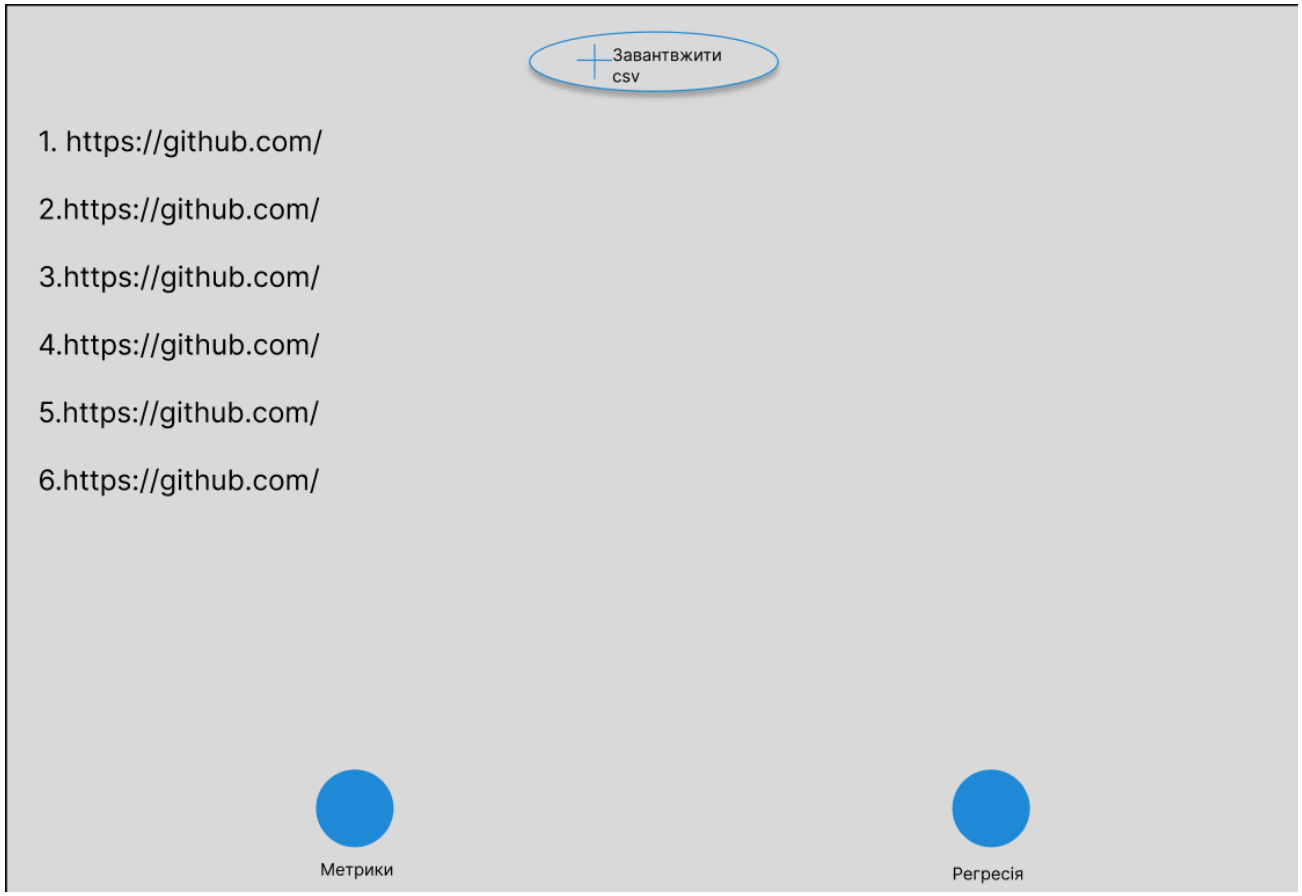

Рисунок 3.8 – Прототип екрана «Метрики» для завантаження наступних даних «FlutterMediaMetrics» для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

На вкладці «Метрики» на верхній частині панелі знаходиться кнопка «Завантажити csv» завдяки якій можна завантажити наступні дані.

3.7 Технічний проект програмного забезпечення

Після детального вивчення початкового проекту успішно розроблено повноцінний технічний проект для програмного забезпечення "FlutterMediaMetrics". Це програмне забезпечення призначене для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter. Для повного розуміння функціональності системи потрібно створити дві моделі: статичну, що відображається через діаграму класів. На рисунку 3.9 зображено статичну модель «FlutterMediaMetrics» у вигляді діаграми класів.

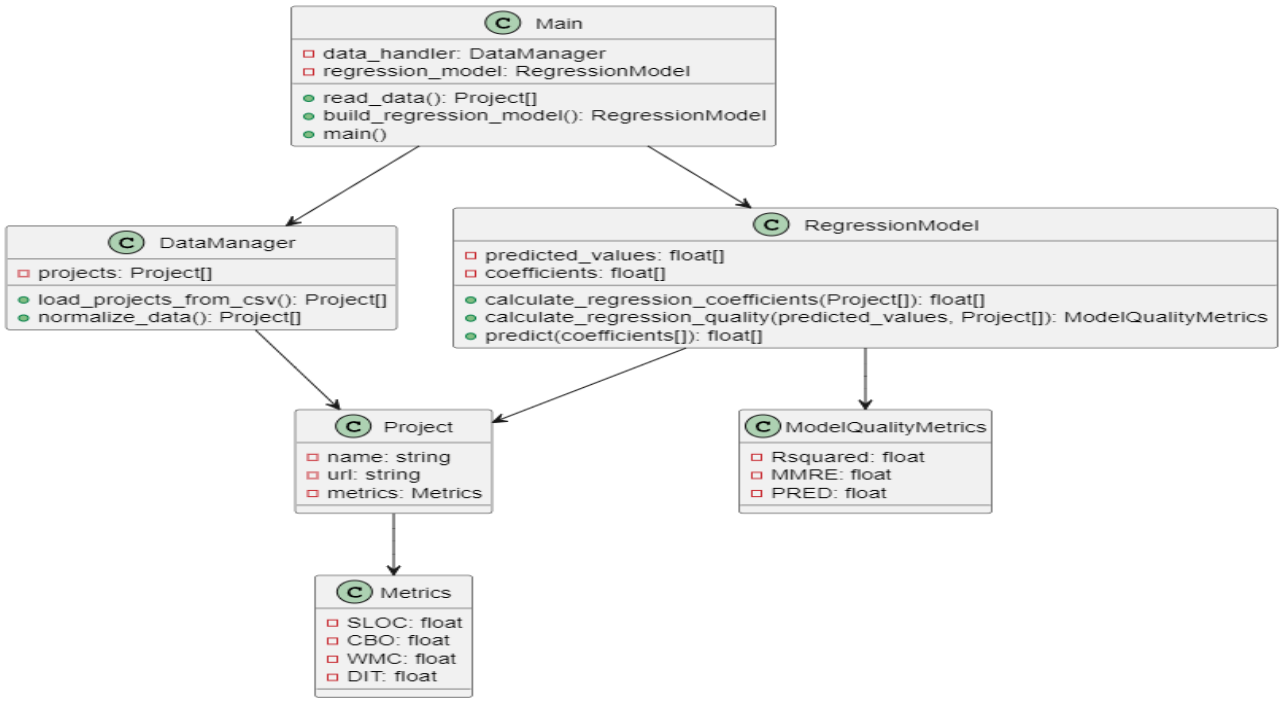

Рисунок 3.9 – Діаграма класів «FlutterMediaMetrics» для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Ця діаграма UML класів відображає систему, яка аналізує дані проектів у сфері програмного забезпечення та створює регресійну модель для передбачення певної величини. Класи:

– Main: Цей клас виступає як вхідна точка програми, відповідаючи за збір даних про проекти, побудову регресійної моделі та виконання основної логіки.

– Project: представляє окремий проект програмного забезпечення з інформацією про назву, URL-адресу та метрики, що зберігаються в об'єкті класу "Metrics".

– Metrics: Цей клас зберігає різні метрики проекту, такі як кількість класів, методів та рядки коду у тисячах.

– RegressionModel: використовується для побудови регресійних моделей та прогнозування за їх допомогою.

– ModelQualityMetrics: Цей клас зберігає показники якості моделі, такі як коефіцієнт детермінації (R^2), середню абсолютну похибку відновлення (MMRE) та прогнозоване значення (PRED).

– DataManadger: Цей клас завантажує файл у форматі csv та нормалізує дані.

3.8 Реалізація програмного забезпечення

Процес реалізації програмного забезпечення "FlutterMediaMetrics" базувався на вимогах, визначених у попередніх етапах проекту, та дотримувався сучасних принципів розробки, що забезпечують надійність, масштабованість і зручність використання. Впровадження функціональності здійснювалося з використанням передових технологій та інструментів, які підвищили ефективність розробки.

Для реалізації програми було обрано мову програмування Dart у зв'язці з фреймворком Flutter, що дозволило створити додаток. Для локального зберігання даних використовувалася база даних SQLite. До проєкту також інтегровано низку бібліотек:

– provider – для управління станом додатку,
– csv – для обробки файлів у форматі CSV,
– file_picker – для зчитування файлу.

Програма побудована на основі шаблону MVC (Model-View-Controller).

Модель забезпечує управління даними та їх збереження, представлення включає інтерфейс користувача, а контролер відповідає за логіку роботи програми та взаємодію між іншими компонентами. Такий підхід зробив додаток більш гнучким і модульним.

Основні етапи реалізації

– Робота з даними

Було реалізовано імпорт даних із CSV-файлів, перевірку їхньої коректності та збереження у базі SQLite. Крім цього, передбачена функція нормалізації метрик для підготовки їх до аналізу.

2. Побудова регресійних моделей

Програмне забезпечення автоматично обчислює коефіцієнти для лінійної та нелінійної регресійних моделей, оцінюючи їх якість за такими показниками, як R^2, MMRE та PRED(0.25).

– Інтерфейс користувача

Створено адаптивний інтерфейс, що забезпечує зручність роботи. Головний екран містить список проєктів із функціями додавання, редагування записів.

– Оцінювання строк коду мультимедійних застосунків

Користувач може вводити нові метрики для оцінювання кількості строк коду мультимедійних застосунків для Android. Реалізовано функції обробки даних і використання побудованої моделі.

У результаті розробки було створено функціональне програмне забезпечення "FlutterMediaMetrics". Код програмного забезпечення представлено у додатку Б. На рисунку 3.10 представлено головну сторінку застосунку "FlutterMediaMetrics"

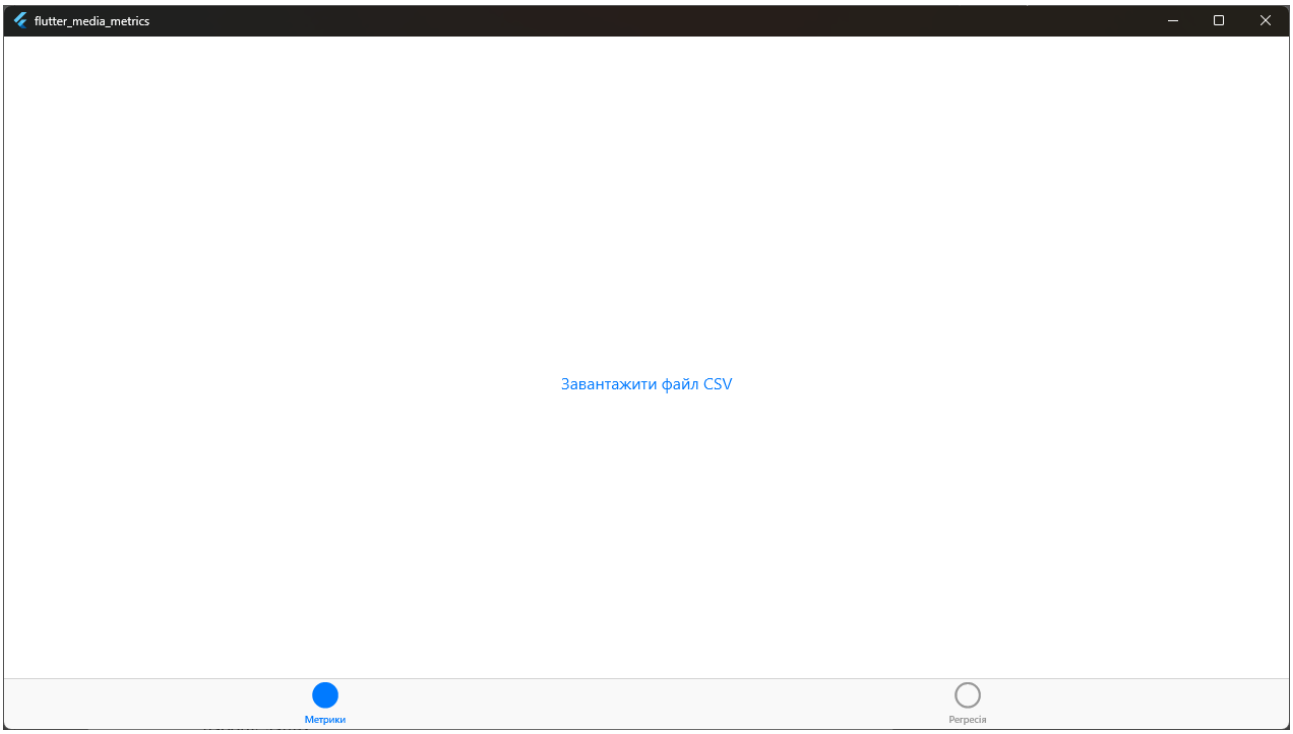

Рисунок 3.10 – Головна сторінка застосунку

На головній сторінці можна побачити кнопку із завантаженням даних із .csv файлу та компонети меню

– Метрики – перегляд завантажених проектів

– Регресія – перегляд побудованих лінійної та нелінійної регресійних моделей, її коефіцієнтів та показників якості.

На рисунку 3.11 представлено вкладку Регресія (побудована лінійна та нелінійна регресійна модель із завантажених даних).

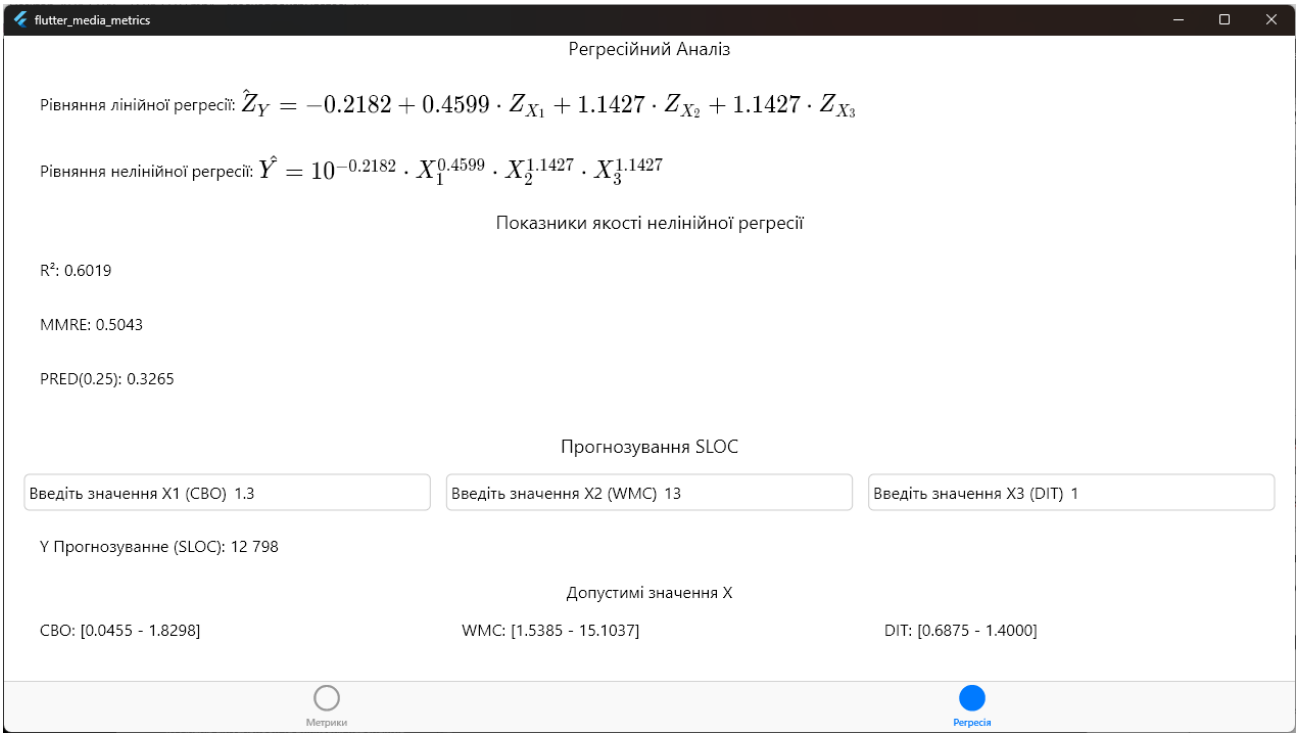

Рисунок 3.11 – Вкладка Регресія застосунку "FlutterMediaMetrics"

Можна побачити коефіцієнти $\beta_0, \beta_1, \beta_2, \beta_3$ моделі склали відповідно -0,2182, 0,4599, 1,1427. Нелінійна модель має показник якості $R^2 = 0{,}6019$, $MMRE = 0{,}5043$ та $PRED(0.25) = 0{,}3265$. Нижче користувач може ввести CBO[0,0455 – 1,8298], WMC[1,5385 – 15,1037], та DIT [0,6875 – 1,4000]. Нижче можна побачити значення Y прогнозоване для заданих значень. Опис програмного забезпечення «FlutterMediaMetrics» представлено у додатку В.

На рисунку 3.12 представлено вкладку «Метрики» завдяки якій можна додати нові проекти для оцінювання строк коду мультимедійних застосунків для Android.

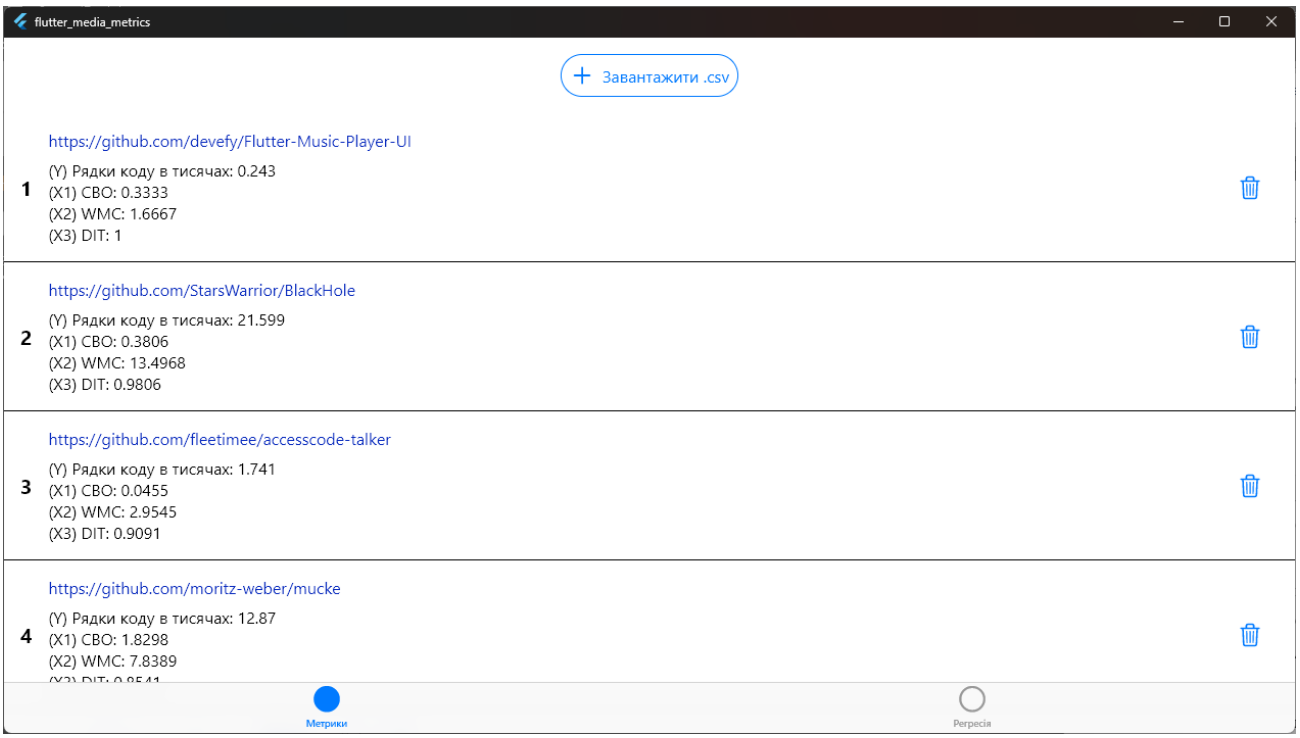

Рисунок 3.12 – Вкладка Метрики застосунку "FlutterMediaMetrics"

На вкладсі «Метрики» в верхній частині екрану можна побачити кнопку «Завантажити .csv» завдяки якій можна завантажити наступні проекти.

Інструкція користувача програмного забезпечення представлена у додатку Г.

3.9 Випробування програмного забезпечення

Програму та методику випробувань розробленого програмного забезпечення описано у Додатку Д. Проведемо випробування розробленого програмного забезпечення згідно з методикою описаної у додатку Д.

Під час випробування програмного забезпечення "FlutterMediaMetrics" для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter були проведені наступні випробування для перевірки вимог до функціональних характеристик:

- Завантаження даних з CSV файлу;
- Побудова лінійної регресійної моделі;
- Побудова нелінійної регресійної моделі;
- Здійснення прогнозів;

У таблиці 3.1 представлено випробування програмного забезпечення.

Таблиця 3.1 – випробування програмного забезпечення

№	Дія	Очікуваний результат
1	Завантаження даних з CSV файлу	Дані успішно завантажені та відображаються у вкладці «Матриця».
2	Побудова лінійної регресійної моделі	Регресійна модель успішно побудована та відображається у вкладці «Регресія».
3	Побудова нелінійної регресійної моделі	Регресійна модель успішно побудована та відображається у вкладці «Регресія».
4	Здійснення прогнозів	Прогнози успішно здійснюються та відображаються у вкладці «Prediction».
5	Обробка некоректних даних або порожнього файлу	Програма видає відповідне попередження або повідомлення про помилку

Результат роботи програми для оцінювання для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter наведено на рисунку 3.13-3.15.

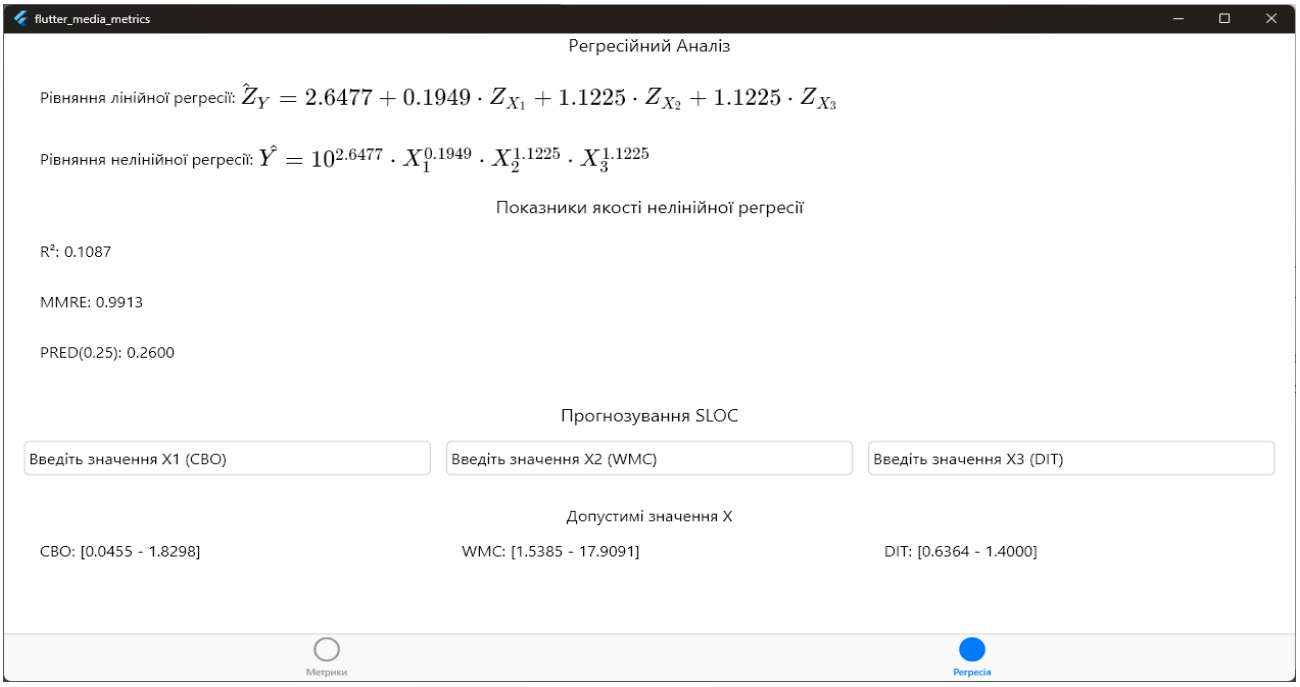

Рисунок 3.13 – Вкладка «Регресія» застосунку "FlutterMediaMetrics"

Якщо користувач вводить неправильні дані, то система виводить повідомлення про помилку, що приведене на рисунку 3.14

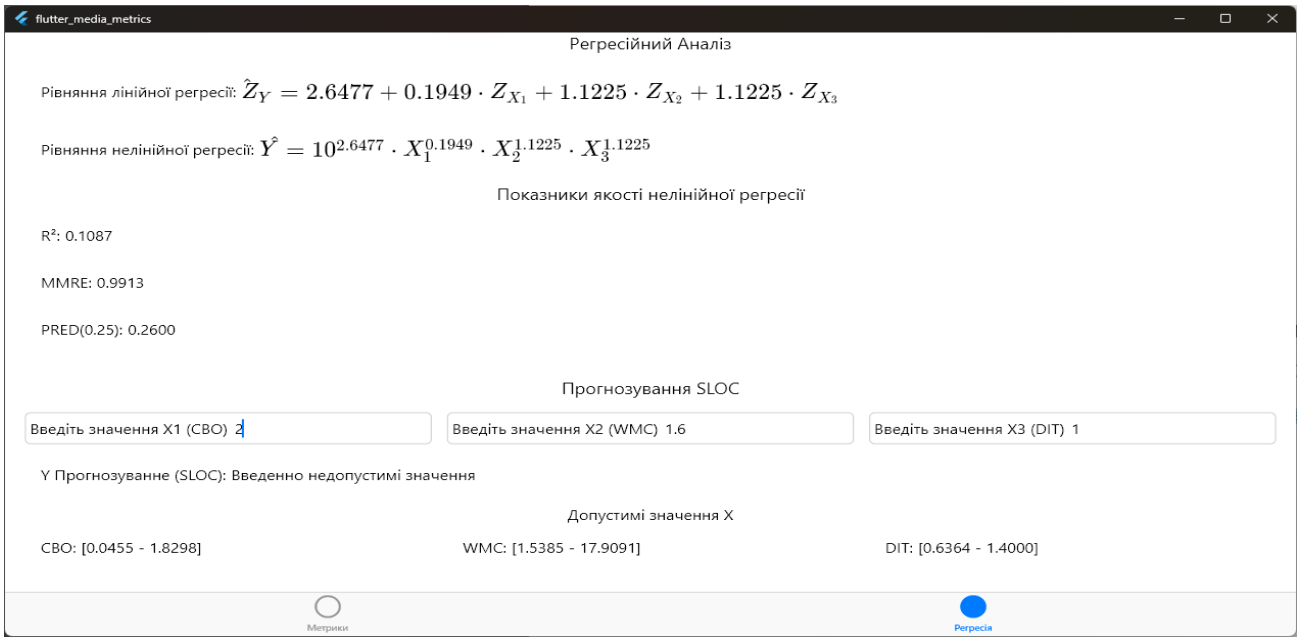

Рисунок 3.14 – Вкладка «Регресія» для прогнозування SLOC з повідомленням про помилку

Якщо користувач вводить дані вірно, то система виводить результат розрахунків, що приведено на рисунку 3.15

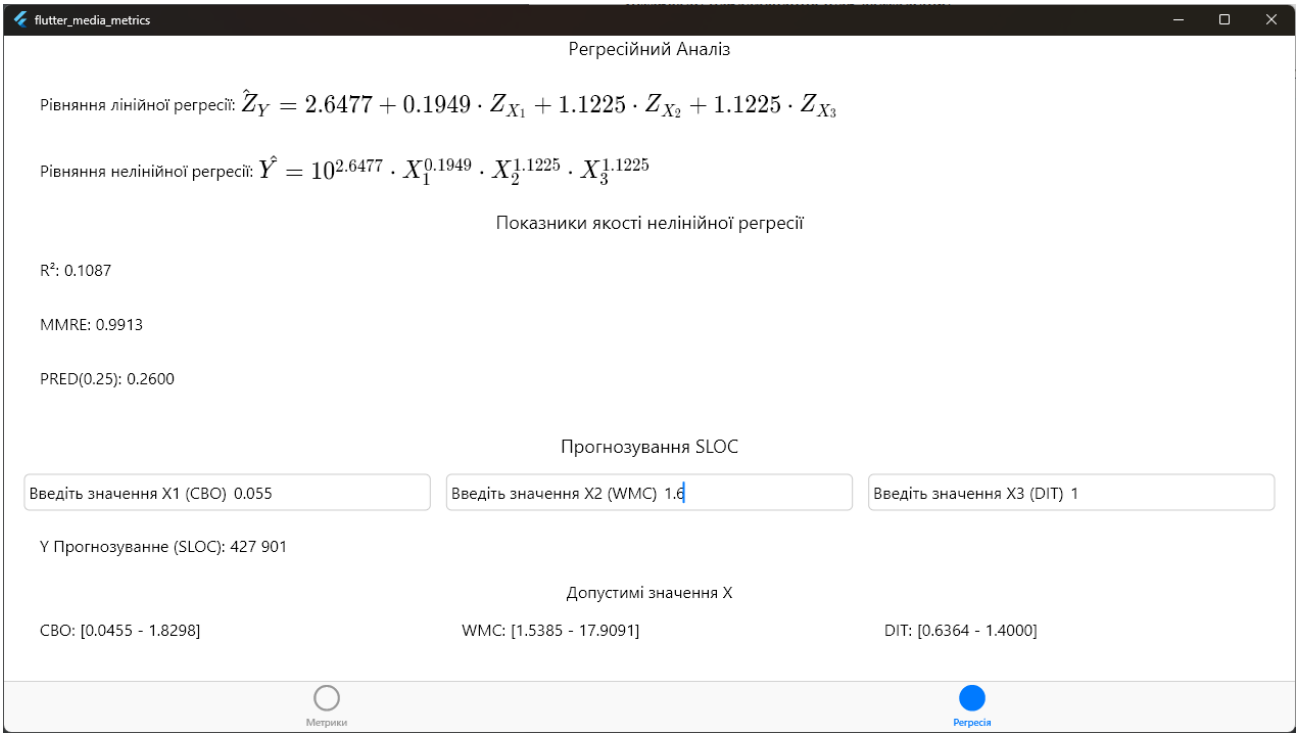

Рисунок 3.15 – Вкладка «Регресія» програмного забезпечення з результатами розрахунків для прогнозування SLOC

Під час випробування програмного забезпечення "FlutterMediaMetrics" всі функціональні вимоги були успішно протестовані. Дані завантажувалися з CSV файлу без помилок, модель будувалася належним чином, прогнози здійснювалися успішно.

Таким чином, програмне забезпечення "FlutterMediaMetrics" успішно пройшло всі випробування на функціональність і може бути використане для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter.

4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВІД РОЗРОБКИ І ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ КІЛЬКОСТІ СТРОК КОДУ МУЛЬТИМЕДІЙНИХ ЗАСТОСУНКІВ ДЛЯ ANDROID, ЩО СТВОРЮЮТЬСЯ НА ПЛАТФОРМІ FLUTTER

4.1 Аналіз актуальності та ринкових перспектив

Розробка програмного забезпечення для оцінювання кількості строк коду мультимедійних застосунків для Android на платформі Flutter є актуальною через зростаючий попит на автоматизацію оцінки складності програмних продуктів. Проект спрямований на підвищення ефективності розробки та планування програмного забезпечення завдяки використанню точних прогнозних моделей. У цьому розділі проаналізовано витрати на розробку програмного забезпечення та потенційні економічні вигоди від його впровадження.

4.2 Розрахунок витрат на розробку

Для розробки та впровадження програмного забезпечення необхідно оцінити як прямі витрати, пов’язані з розробкою, так і супутні витрати.

Таблиця 4.1 – Витрати на допоміжні матеріали

Пункти витрат	Сума, грн
Канцелярські товари	1000
Ліцензії на програмне забезпечення	12 000
Оплата інтернет-з'єднання	3000
Друк документів та презентацій	2 000
Інші витрати	2000
Разом	20 000

У таблиці 4.2 представлено витрати на розробку програмного забезпечення.

Таблиця 4.2 – Витрати на розробку програмного забезпечення

Пункти витрат	Сума, грн
Зарплата програміста (1 особа, 3 місяці)	60 000
Зарплата аналітика (1 особа, 2 місяці)	30 000
Зарплата тестувальника (1 особа, 2 місяці)	20 000
Оплата серверів та хмарних сервісів	15 000
Накладні витрати (офіс, комунальні тощо)	10 000
Інші витрати	5 000
Разом	140 000

Загальні витрати на проект становлять 160 000 грн.

4.3 Доходи від впровадження ПЗ

Економія ресурсів та часу:

Використання розробленого ПЗ дозволяє зменшити час на оцінку складності коду на 30%. Для одного розробника (зарплата 20 000 грн/місяць) це дає економію 6 000 грн на місяць.

Підвищення точності планування:

Покращення прогнозів скорочує ризик перевитрат на 10%, що, у середньому, економить 20 000 грн на кожному проекті.

Інші переваги:

ПЗ сприяє зниженню помилок, полегшує управління проектами, що позитивно впливає на ефективність роботи.

4.4 Розрахунок показників економічної ефективності

Чиста приведена вартість (NPV):

Програмне забезпечення буде реалізовано за ліцензійною моделлю з ціною однієї ліцензії 130 грн. Прогнозований обсяг продажів на перший рік становить 600 ліцензій. Очікуваний дохід: 130 × 600 = 78 000 грн.

Приведена вартість NPV розраховується як [31]:

$$NPV = \sum_{t=1}^{3} \frac{CF_t}{(1+0.1)^t} - C_0, \quad (4.1)$$

де $C_0 = 160000$ грн,

$CF_t = 78000$ грн,

$t = 1,2,3,$

$r = 0.1.$

$NPV \approx 179000$ грн.

1. IRR становить приблизно 24%, що перевищує ставку дисконту 10%.
2. Термін окупності (Payback Period):

   Загальні витрати окупаються за 1.5 року.
3. Індекс прибутковості (PI) [32]:

$$PI = \frac{\sum_{t=1}^{3} \frac{CF_t}{(1+0.1)^t}}{C_0} \approx 1.12. \quad (4.2)$$

Економічний аналіз підтверджує доцільність розробки та впровадження програмного забезпечення. Позитивне значення NPV, висока IRR та короткий період окупності свідчать про високу ефективність проекту. Очікується, що впровадження ПЗ сприятиме покращенню управління проектами, скороченню витрат на розробку та підвищенню точності оцінок.

5 ОХОРОНА ПРАЦІ

5.1 Вступ

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних та лікувально-профілактичних заходів і засобів, спрямованих на збереження здоров’я та працездатності людини в процесі трудової діяльності [33]. Охорона праці є комплексною науковою дисципліною, яка базується на теоретичних положеннях природничих (фізика, хімія, біологія, медицина), математичних (математичне моделювання, теорія ймовірностей, математична статистика), суспільних (економіка, соціологія, психологія, право) і технічних наук (електротехніка, електроніка, автоматика, системологія, ергономіка, інженерна психологія, технічна естетика, наукова організація праці). Особливо тісно охорона праці пов'язана з безпекою життєдіяльності, екологією, науковою організацією праці, ергономікою, інженерною психологією та технічною естетикою. У них єдина мета – сприяти збереженню здоров'я та працездатності людини, підвищенню її продуктивності праці, усуненню або зменшенню впливу на неї шкідливих і небезпечних виробничих чинників. У той же час усі вони підходять до досягнення поставленої мети з різних боків і на різних рівнях, а саме:

1. Безпека життєдіяльності вивчає загальні закономірності виникнення небезпек, їх властивості та особливості впливу на людину, наслідки такого впливу, а також способи та засоби захисту життя та здоров'я людини й середовища її проживання від реальних та потенційних небезпек.

2. Наукова організація праці займається вивченням, розробкою та впровадженням у практику раціональної побудови трудового процесу, за якої забезпечується висока продуктивність праці, створюються умови для збереження здоров'я працівників, збільшується період їх трудової діяльності.

3. Ергономіка досліджує, розробляв та дає рекомендації щодо конструювання, виготовлення та експлуатації технічних засобів, які забезпечують людині в процесі праці необхідні зручності, зберігають її сили, працездатність та здоров'я.

4. Інженерна психологія вивчає взаємодію людини з новою технікою і з'ясовує функціональні можливості людини в трудових процесах з метою створення таких умов праці, за яких зберігаються високі психофізіологічні можливості людини.

5. Технічна естетика встановлює залежність умов та результатів праці від архітектурного, конструктивного та художнього вирішення знарядь праці, робочих місць, дільниць, цехів, санітарно-побутових та інших допоміжних приміщень, усього, що оточує людину на виробництві.

Інваріантною основою змісту вищеназваного є система “людина – середовище”, де під середовищем розуміють як природне середовище (біосфера), так і штучне середовище (техносфера, соціальна сфера). Охорона праці розглядає штучне середовище – виробництво, об’єкти виробничої діяльності, що конкретизує техносферу [34].

5.2 Санітарно-гігієнічні вимоги до виробничого середовища програміста

Санітарно-гігієнічні вимоги до процесу роботи людини у приміщені з персональним комп`ютером визначають: мікроклімат; степінь освітленості;

рівень шуму і вібрації; іонізуюче та неіонізуюче випромінювання [35].

Мікроклімат

Мікроклімат на робочих місцях з персональним комп`ютером нормується ДЕСТ 12.1.005-88 і описаний у таблиці 5.1.

Таблиця 5.1 – Нормовані параметри мікроклімату для приміщень з персональним комп`ютером

Пора року	Категорія робіт	Темпе-ратура	Відносна вологість, %	Швидкість руху повітря, м/с, не більше
Холодна пора року	Легка – Іа	22-24	40-60	0,1
	Легка – Іб	21-23	40-60	0,1
Тепла пораро ку	Легка – Іа	23-25	40-60	0,1
	Легка – Іб	22-24	40-60	0,1

Освітлення

1) Робочі місця з персональними комп`ютерами повинні мати природне і штучне освітлення.

2) Вікна приміщень з персональними комп`ютерами повинні мати орієнтацію на північ або на північний схід, з КПО 1,5%. Дозволяється експлуатація персональних комп`ютерів у приміщеннях без природного освітлення за узгодженням з органами Держнаглядохоронпраці та санітарно-епідеміологічними установами.

3) Рівень освітленості на робочому місці в зоні розміщення документів має бути від 300 лк до 500лк;

4) Штучне освітлення виконується, як правило, люмінесцентними лампами. Допускається для місцевого освітлення використовувати лампи розжарювання.

5) Місцеве освітлення не повинно створювати блисків на екрані персональних комп`ютерів, а його освітленість повинна не перевищувати 300 лк.

Рівні шуму

Рівні шуму на робочих місцях з персональними комп`ютерами визначені ДсанПіН 3.3.2-007-98 в залежності від його тональності та виду трудової діяльності користувачів персональних комп`ютерів (програмісти, оператори, оператори комп'ютерного набору). Рівні звукового тиску описані у таблиці 5.2.

Таблиця 5.2 – Рівні звукового тиску в дБ в октавних смугах частот, Гц

Вид трудової діяльності	Рівні звукового тиску в дБ в октавних смугах із середньогеометричними частотами, Гц									
Робочі місця	31,5	63	125	250	500	1000	2000	4000	8000	Рівні звуку, еквівалентні рівні звуку, дБА/дБА екв.
Програмісти	86	71	61	54	49	45	42	40	38	50
Оператори в залах обробки інформації на ПК та оператори комп'ютерного набору	96	83	74	68	63	60	57	55	54	65
В приміщеннях для розташування шумних агрегатів	103	91	83	77	73	70	68	66	64	75

Для нормування шуму застосовуються шумопоглинальні засоби, визначені спеціальними інженерно-акустичними розрахунками.

Рівні вібрації

Рівні вібрації не повинні перевищувати допустимих норм, визначених ДсанПіН 3.3.2-007-98.

Рівні іонізуючого випромінювання

Рівні іонізуючого випромінювання повинні відповідати НРБУ–97. Потужність експозиційної дози рентгенівського випромінювання на відстані 5 см від екрану ПК повинна бути не більше 0,1 мбер/година (100 мкР/година).

Рівні неіонізуючого випромінювання

Рівні неіонізуючого випромінювання повинні відповідати вимогам ДЕСТ 12.1.006, СН №3206-85 «Гранично допустимі рівні магнітного поля частотою 50 Гц, СН №4557-88 «Санітарні норми ультрафіолетового випромінювання у виробничих приміщеннях».

5.3 Ергономічні вимоги до робочого місця програміста

Проектування робочих місць, забезпечених відеотерміналами, відноситься до числа важливих проблем ергономічного проектування в області обчислювальної техніки.

Робоче місце користувача персонального комп`тера повинно відповідати вимогам ДЕСТ 12.2.032-78.

Робоче місце і взаємне розташовує всіх його елементів повинне відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має

також характер роботи. Зокрема, при організації робочого місця програміста повинні бути дотримані наступні основні умови: оптимальне розміщення устаткування, що входить до складу робочого місця і достатній робочий простір, що дозволяє здійснювати всі необхідні рухи і переміщення.

Ергономічними аспектами проектування відеотермінальних робочих місць, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до того, що розташовує документів на робочому місці (наявність і розміри підставки для документів, можливість різного розміщення документів, відстань від очей користувача до екрану, документа, клавіатури і т.д.), характеристики робочого крісла, вимоги до поверхні робочого столу, можливість регулювання елементів робочого місця. Головними елементами робочого місця програміста єстіл і крісло. Основним робочим положенням є положення сидячи.

Робоча поза сидячи викликає мінімальне стомлення програміста. Раціональне планування робочого місця передбачає чіткий порядок і постійність розміщення предметів, засобів праці і документації. Те, що потрібне для виконання робіт частіше, розташоване в зоні легкої досяжності робочого простору.

Моторне поле – простір робочого місця, в якому можуть здійснюватися рухові дії людини.

Максимальна зона досяжності рук – це частина моторного поля робочого місця, обмеженого дугами, описуваними максимально витягнутими руками при русі їх в плечовому суглобі.

Оптимальна зона – частина моторного поля робочого місця, обмеженого дугами, що описуються передпліччями при русі в ліктьових суглобах з опорою в точці ліктя і з відносно нерухомим плечем.

Оптимальне розміщення предметів праці і документації в зонах досяжності:

– дисплей розміщується в центрі;

– системний блок розміщується в передбаченій ніші столу;

– клавіатура – в зоні попереду;

– «миша» – в зоні справа;

– сканер в зоні зліва;

– принтер знаходиться в зоні справа.

Документація: необхідна при роботі – в зоні легкої досяжності долоні, а у висувних ящиках столу – література, невживана постійно.

Для комфортної роботи стіл повинен задовольняти наступним умовам :

– висота столу повинна бути вибрана з урахуванням можливості сидіти вільно, в зручній позі, при необхідності спираючись на підлокітники;

– нижня частина столу повинна бути сконструйована так, щоб програміст міг зручно сидіти, не був вимушений підтискати ноги;

– поверхня столу повинна володіти властивостями, що виключають появу відблисків в полі зору програміста;

– конструкція столу повинна передбачати наявність висувних ящиків (не менше 3 для зберігання документації, лістингів, канцелярських обладнань).

– висота робочої поверхні рекомендується в межах 680-760 мм. Висота поверхні, на яку встановлюється клавіатура, повинна бути біля 650 мм.

Велике значення надається характеристикам робочого крісла. Так, висота сидіння над рівнем підлоги, що рекомендується, знаходиться в межах 420-550мм. Поверхня сидіння м'яка, передній край закруглює, а кут нахилу спинки – регульований. безпека оператор виробниче приміщення

Необхідно передбачати при проектуванні можливість різного розміщення документів: збоку від відеотерміналу, між монітором і клавіатурою і т.п. Крім того, у випадках, коли відеотермінал має низьку якість зображення, наприклад помітні мигтіння, відстань від очей до екрану роблять більше (біля 700 мм), ніж

відстань від ока до документа (300-450 мм). Взагалі при високій якості зображення на відеотерміналі відстань від очей користувача до екрану, документа і клавіатури може бути рівним.

Положення екрану визначається:

– відстанню прочитування (0,6-0,7 м);

– кутом прочитування, напрямом погляду на 20 нижче горизонталі до центру екрану, причому екран перпендикулярний цьому напряму.

Повинна також передбачатися можливість регулювання екрану:

– по висоті +3 см;

– по нахилу від -10 до +20 щодо вертикалі;

– в лівому і правом напрямах.

Велике значення також надається правильній робочій позі користувача. При незручній робочій позі можуть з'явитися болі в м'язах, суглобах і сухожиллях. Вимоги до робочої пози користувача відеотерміналу наступні:

– голова не повинна бути нахилена більш ніж на 20;

– плечі повинні бути розслаблені;

– лікті – під кутом 80-100;

– передпліччя і долоні рук – в горизонтальному положенні.

Причина неправильної пози користувачів обумовлена наступними чинниками: немає хорошої підставки для документів, клавіатура знаходиться дуже високо, а документи – низько, нікуди покласти руки і кисті, недостатній простір для ніг.

В цілях подолання вказаних недоліків даються загальні рекомендації: краще пересувна клавіатура; повинні бути передбачені спеціальні пристосування для регулювання висоти столу, клавіатури і екрану, а також підставка для рук.

Істотне значення для продуктивної і якісної роботи на комп'ютері мають розміри знаків, густину їх розміщення, контраст і співвідношення яскравості

символів і фону екрану. Якщо відстань від очей оператора до екрану дисплея складає 60-80 см, то висота знака повинна бути не менше 3 мм, оптимальне співвідношення ширини і висоти знака складає 3:4, а відстань між знаками – 15-20% їх висоти. Співвідношення яскравості фону екрану і символів – від 1:2 до 1:15.

Під час користування комп'ютером медики радять встановлювати монітор на відстані 50-60 см від очей. Фахівці також вважають, що верхня частина відеодисплея повинна бути на рівні очей або трохи нижче. Коли людина дивиться прямо перед собою, його очі відкриваються ширше, ніж коли він дивиться вниз. За рахунок цього площа огляду значно збільшується, викликаючиобезводнення очей. До того ж якщо екран встановлений високо, а очі широко відкриті, порушується функція моргання. Це значить, що очі не закриваються повністю, не омиваються слізною рідиною, не одержують достатнього зволоження, що приводить до їх швидкої стомлюваності.

Створення сприятливих умов праці і правильне естетичне оформлення робочих місць на виробництві має велике значення як для полегшення праці, такі для підвищення його привабливості, позитивно впливаючою на продуктивність праці.

6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Загальні положення

Охорона навколишнього середовища є важливою складовою сталого розвитку, що передбачає заходи для збереження природних ресурсів, зменшення забруднення та підтримання екологічної рівноваги. У сфері інформаційних технологій цей аспект стає дедалі важливішим, оскільки розвиток цифрових технологій і зростання масштабів обчислювальних потужностей супроводжуються значним екологічним впливом.

У межах цього проєкту аналізуються потенційні екологічні ризики, пов’язані з розробкою, тестуванням та експлуатацією програмного забезпечення для оцінювання кількості рядків коду мультимедійних застосунків для Android, створених на платформі Flutter. Мета полягає у визначенні основних джерел впливу на довкілля та розробці рекомендацій для їхньої мінімізації.

6.2 Аналіз потенційного впливу

Процес реалізації програмного забезпечення має кілька аспектів, що можуть впливати на стан природного середовища:

1. Енергоспоживання:

- Використання обчислювальних ресурсів під час компіляції, тестування та виконання програмного забезпечення.
- Робота серверів для хостингу застосунків і забезпечення їхньої

доступності.

- Зростання обсягів передавання даних через мережі зв’язку, що підвищує енергоспоживання мережевої інфраструктури.

2. Використання електронних ресурсів:

- Активне застосування комп’ютерної техніки, яка вимагає регулярного оновлення.

- Висока продуктивність обладнання, необхідна для роботи із сучасними застосунками, зокрема графічні процесори.

3. Відходи електронного обладнання:

- Утворення електронного сміття через модернізацію апаратних засобів і швидке моральне зношення техніки.

- Неналежна утилізація електронних відходів, що може забруднювати ґрунти та водні ресурси.

6.3 Заходи з охорони навколишнього середовища

Для зменшення негативного впливу на довкілля під час розробки програмного забезпечення пропонуються такі заходи:

1. Оптимізація енергоспоживання:

- Використання енергоефективного обладнання.

- Оптимізація алгоритмів і скорочення часу виконання задач, що зменшує витрати електроенергії.

2. Мінімізація цифрових відходів:

- Переробка або правильна утилізація застарілого обладнання.

- Використання віртуалізації для зниження потреби у фізичних серверах.

3. Використання відновлюваних джерел енергії:

- Перехід на хмарні сервіси, які підтримують екологічні ініціативи, наприклад, Google Cloud із програмою Carbon Free Energy.

4. Екологічна обізнаність команди:

- Організація тренінгів і заходів для підвищення рівня знань співробітників про екологічні аспекти їхньої діяльності.

6.4 Оцінка ефективності заходів

Для досягнення сталого розвитку слід регулярно проводити аудит енергоспоживання, аналізувати вплив програмного забезпечення на довкілля та виявляти нові можливості для покращення екологічних показників. Це сприятиме створенню прозорої системи контролю екологічної відповідальності.

Розробка програмного забезпечення, яка враховує екологічні аспекти, є важливим кроком у забезпеченні сталого розвитку суспільства. Мінімізація впливу на довкілля досягається завдяки оптимізації енергоспоживання, правильній утилізації електронного обладнання та використанню відновлюваних джерел енергії. Впровадження екологічно орієнтованих підходів дозволяє не лише знизити негативний вплив на природу, а й підвищити рівень відповідальності команди розробників.

Таким чином, інтеграція екологічних принципів у розробку програмного забезпечення не лише зберігає природні ресурси, але й сприяє підвищенню ефективності та сталості цифрових рішень.

ВИСНОВКИ

У межах кваліфікаційної роботи була розв’язана науково-практична задача оцінювання кількості строк коду мультимедійних застосунків для Android, створених на платформі Flutter, шляхом розробки та впровадження удосконаленої нелінійної регресійної моделі. Розв’язання цієї задачі сприяє підвищенню ефективності планування ресурсів у розробці програмного забезпечення та оптимізації процесів проєктного управління.

Основні результати роботи:

– Проведено аналіз існуючих моделей оцінювання кількості строк коду програмного забезпечення. Встановлено, що традиційні моделі, створені для мов програмування Kotlin та PHP, не враховують специфіку застосунків, розроблених на Flutter, що підтверджено емпіричними дослідженнями.

– Удосконалено нелінійну регресійну модель, яка враховує особливості архітектури та метрики Flutter-застосунків (CBO, WMC, DIT). Нормалізуюче перетворення у вигляді десяткового логарифму значно зменшило вплив викидів та забезпечило більш точне моделювання кількості строк коду.

– Розроблено програмне забезпечення FlutterMediaMetrics, яке реалізує запропоновану модель, забезпечуючи автоматизацію процесу оцінювання кількості строк коду. Програмне забезпечення успішно протестоване на реальних проєктах, що підтвердило його ефективність і точність.

– Застосування розробленої моделі та програмного забезпечення дозволяє знизити похибки оцінювання, поліпшити розподіл ресурсів та підвищити продуктивність команд розробників.

Наукова новизна роботи полягає в створенні та апробації удосконаленої регресійної моделі, адаптованої до особливостей платформи Flutter. Практична

значущість результатів підтверджується можливістю їхнього використання в процесах планування й управління розробкою кросплатформних застосунків.

Технічне завдання представлене в додатку А. Текст коду програми представлений в додатку Б. Опис програми представлений в додатку В. Інструкція користувача – в додатку Г. Програма і методика випробувань програмного забезпечення у додатку Д.

Рекомендації щодо подальшого використання результатів дослідження: запропоновану модель можна інтегрувати в інструменти керування проєктами розробки програмного забезпечення, що працюють з кросплатформними технологіями. Подальші дослідження можуть бути спрямовані на розширення набору метрик та оптимізацію моделі для більш складних програмних продуктів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Flutter. Build apps for any screen [Електронний ресурс]. – URL: https://flutter.dev/ (дата звернення: 02.10.2024).

2. Dart Programming language | Dart [Електронний ресурс]. – URL: https://dart.dev/ (дата звернення: 06.10.2024).

3. Sheldon R. Metrics KLOC [Електронний ресурс]. – URL: https://www.techtarget.com/whatis/definition/KLOC-thousands-of-lines-of-code дата звернення: 02.10.2024).

4. Приходько С.Б., Кольцов А.В., Грабовський Є.В. Нелінійна регресійна модель для оцінювання метрики RFC застосунків з відкритим кодом на Kotlin // Інформаційні технології: моделі, алгоритми, системи : матеріали IV-ї Всеукраїнської науково-практичної інтернет-конференції (ITMAS–2023), Миколаїв, 2023. – С. 25–26. URL: https://itconf.nuos.edu.ua/2023/wp-content/uploads/sites/6/NUOS-ITMAS-2023-Proceedings.pdf.

5. Латанська Л.О., Кольцов А.В. Нелінійна регресійна модель для оцінювання кількості строк коду web-застосунків, що створюються з використанням PHP фреймворку Symfony // Інформаційні технології: моделі, алгоритми, системи : матеріали III-ї Всеукраїнської науково-практичної інтернет-конференції (ITMAS–2022), Миколаїв, 2022. – С. 10–11. URL: http://itconf.nuos.edu.ua/2022/wpcontent/uploads/sites/5/NUOS-ITMAS-2022-Proceedings.pdf.

6. Приходько С.Б., Іванов Д.О. Нелінійна регресійна модель для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter // Інформаційні технології: моделі, алгоритми, системи : матеріали V-ї Всеукраїнської науково-практичної інтернет-конференції

(ITMAS–2024), Миколаїв, 2024. – С. 57–58. – URL: https://itconf.nuos.edu.ua/2024/wp-content/uploads/sites/7/NUOS-ITMAS-2024-Proceedings.pdf.

7. Flutter architectural overview [Електронний ресурс]. – URL: https://docs.flutter.dev/resources/architectural-overview (дата звернення: 22.10.2024).

8. Coupling Between Object Classes | DCM - Code Quality Tool for Flutter Developers [Електронний ресурс]. – URL: https://dcm.dev/docs/metrics/class/coupling-between-object-classes/ (дата звернення: 08.10.2024).

9. Weighted Methods Per Class | DCM - Code Quality Tool for Flutter Developers [Електронний ресурс]. – URL: https://dcm.dev/docs/metrics/class/weighted-methods-per-class/ (дата звернення: 08.10.2024).

10. Depth of Inheritance Tree | DCM - Code Quality Tool for Flutter Developers [Електронний ресурс]. – URL: https://dcm.dev/docs/metrics/class/depth-of-inheritance-tree/ (дата звернення: 08.10.2024).

11. Rodriguez, D., Harrison, R. An Overview of Object-Oriented Design Metrics [Electronic resource] / D. Rodriguez, R. Harrison. - RUCS/2001/TR/A. - Reading: University of Reading, Department of Computer Science, 2001. - 26 p. - Access mode: https://www.cc.uah.es/drg/b/RodHarRama00.English.pdf (accessed 08.10.2024).Use Case Points, UCP [Електронний ресурс]. – URL: https://www.tutorialspoint.com/estimation_techniques/estimation_techniques_use_case_points.htm (дата звернення: 08.10.2024).

12. Najadat, H., Alsmadi, I., Shboul, Y. Predicting Software Projects Cost Estimation Based on Mining Historical Data // Research Article Predicting Software Projects Cost Estimation Based on Mining Historical Data. Jordan University of

Science and Technology, Yarmouk University. Дата отримання: 18 листопада 2011 р. Дата прийняття: 16 січня 2012 р. [Електронний ресурс]. URL: https://onlinelibrary.wiley.com/doi/10.5402/2012/823437 (дата звернення: 10 грудня 2024 р.).

13. Function Point Analysis [Електронний ресурс]. – URL: https://www.fingent.com/blog/function-point-analysis-introduction-and-fundamentals/ (дата звернення: 02.10.2024).

14. Coras F., Domingo-Pascual J., Lewis D., Cabellos-Aparicio A. An Analytical Model for Loc/ID Mappings Caches / Universitat Politècnica de Catalunya (BarcelonaTECH), Barcelona, Spain; Cisco Systems, San Jose, CA, USA. URL:https://www.semanticscholar.org/reader/66c11184547ac4af1ea7fa76081422080e206dbb (дата звернення: 02.10.2024).

15. Barb, A. S., Neill, C. J., Sangwan, R. S., & Piovoso, M. J. (2014). A statistical study of the relevance of lines of code measures in software projects. Innovations in Systems and Software Engineering, 10(4), 243–260. DOI: 10.1007/s11334-014-0231-5. Retrieved from https://www.researchgate.net/publication/267761547_A_statistical_study_of_the_relevance_of_lines_of_code_measures_in_software_projects. (дата звернення: 22.10.2024).

16. Prykhodko S. B., Prykhodko N. V., Koltsov A. V. A nonlinear regression model for early LOC estimation of open-source Kotlin-based applications // Radio Electronics, Computer Science, Control. 2024. No 1. P. 45–58. URL: https://ric.zp.edu.ua/article/view/300970/293102 (дата звернення: 02.10.2024).

17. Widgets | Flutter [Електронний ресурс]. – URL: https://docs.flutter.dev/ui/widgets (дата звернення: 02.10.2024).

18. Flutter Audio_players [Електронний ресурс]. – URL: https://pub.dev/packages/audioplayers (дата звернення: 02.10.2024).

19. Flutter Animate [Електронний ресурс]. – URL: https://pub.dev/packages/flutter_animate (дата звернення: 02.10.2024).

20. Riverpod Flutter [Електронний ресурс]. – URL: https://riverpod.dev/ (дата звернення: 02.10.2024).

21. Detecting Multicollinearity Using Variance Inflation Factors – URL: https://online.stat.psu.edu/stat462/node/180/ (дата звернення: 02.10.2024).

22. DCM – Code Quality Tool for Flutter Developers [Електронний ресурс]. – URL: https://dcm.dev/ (дата звернення: 08.10.2024).

23. S. Prykhodko, N. Prykhodko, L. Makarova, and K. Pugachenko, “Detecting Outliers in Multivariate Non-Gaussian Data on the basis of Normalizing Transformations”, in Proceedings of the 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON) «Celebrating 25 Years of IEEE Ukraine Section», May 29 – June 2, 2017, Kyiv, Ukraine, 2017, pp. 846-849. https://doi.org/10.1109/UKRCON.2017.8100366

24. Koizumi K., Sumikawa T., Pavlenko T. Measures of multivariate skewness and kurtosis in high-dimensional framework // SUT Journal of Mathematics. 2014. Vol. 50, No. 2. P. 483–511. URL: https://people.kth.se/~pavlenko/SUTJM14.pdf (accessed: 10.12.2024).

25. Mahalanobis Distance [Електронний ресурс]. – URL: https://www.machinelearningplus.com/statistics/mahalanobis-distance/ (дата звернення: 22.10.2024).

26. Chen, James. Normal Distribution: What It Is, Uses, and Formula. Updated March 13, 2024. Reviewed by Khadija Khartit, fact-checked by Suzanne Kvilhaug. Available at: https://www.investopedia.com/terms/n/normaldistribution.asp#:~:text=Normal%20distribution%2C%20also%20known%20as,%22bell%20curve%22%20when%20graphed. (дата звернення: 08.10.2024).

27. Afifi A.A., Azen S.P. Statistical analysis: a computer-oriented approach. New York; London: Academic Press, 1972.

28. F Distribution [Електронний ресурс]. – URL: https://www.sciencedirect.com/topics/mathematics/f-distribution (дата звернення: 12.10.2024).

29. MVC [Електронний ресурс]. URL: https://developer.mozilla.org/en-US/docs/Glossary/MVC (дата звернення: 02.10.2024).

30. NPV [Електронний ресурс]. URL: https://finances.in.ua/shcho-take-chysta-potochna-vartist-npv/ (дата звернення: 02.10.2024).

31. Profitability index (PI) [Електронний ресурс]. URL: https://library.if.ua/book/128/8395.html (дата звернення: 02.10.2024).

32. Про охорону праці: Закон України від 21.11.2002 р. № 229-IV. Дата оновлення: 04.02.2021. URL: https://zakon.rada.gov.ua/laws/main/2694-12#Text. (дата звернення: 17.05.2023).

33. Поняття, мета і завдання охорони праці. URL: https://studfile.net/preview/9010003/ (дата звернення: 17.05.2023).

34. Санітарні норми мікроклімату виробничих приміщень (ДСН 3.3.6.042- 99). URL: https://zakon.rada.gov.ua/rada/show/va042282-99#Text (дата звернення: 17.05.2023).

ДОДАТОК А – ТЕХНІЧНЕ ЗАВДАННЯ НА РОЗРОБКУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ «FlutterMediaMetrics»

Вступ

Повна назва програмного продукту: Програмне забезпечення «FlutterMediaMetrics» для визначення розміру медіа застосунків, що створюються з використанням фреймворку Flutter.

Коротка назва: «FlutterMediaMetrics»

Сфера застосування: компанії, що займаються розробкою програмного забезпечення з використанням фреймворку Flutter.

1. Підстави для розробки

Підставою для розробки «FlutterMediaMetrics» є наказ на затвердження тем кваліфікаційних робіт магістрів зі спеціальності 121 Інженерія програмного забезпечення в Національному університеті кораблебудування ім. адмірала Макарова.

2. Призначення розробки

2.1 Експлуатаційне призначення

Експлуатаційним призначенням є спрощення процесів планування та керування розробкою програмного забезпечення для роботи з медіа та автоматизація обчислення передбачуваних розмірів проектів, які створюються за допомогою фреймворку Flutter.

2.2 Функціональне призначення

"FlutterMediaMetrics" призначений для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter. Його функції включають перегляд завантажених метрик SLOC, CBO, WMC та DIT, а також створення регресійної моделі для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter.

3. Вимоги до програмного забезпечення

3.1.Вимоги до функціональних характеристик

3.1.1Вимоги до складу виконуваних функцій

– ведення інформації про метрики мультимедійних застосунків у форматі

– п

е – редагування списку застосунків;

р – побудова лінійної регресійної моделі методом найменших квадратів;

е – побудова нелінійної регресійної моделі методом зворотного перетворення;

л – перегляд оцінювання кількості строк коду у SLOC за допомогою я CBO,WMC та DIT.

д 3.1.2 Вимоги до організації вхідних та вихідних даних

Дані про метрики та модель зберігаються у локальній базі даних. СУБД забезпечує розмежування прав доступу до даних – надає користувачу права на и питання та запис.

в Вхідні дані:

а – файл у форматі CSV, що містять метрики Flutter-проектів: SLOC, ф CBO,WMC та DIT;

т – метрики CBO,WMC та DIT для оцінювання кількості строк коду а мультимедійних застосунків для Android, що створюються на платформі Flutter.

ж Вихідні дані:

е – список завантажених проектів та їх поділ на дві вибірки для побудови та н тестування моделі;

и – викиди;

х – параметри побудованої лінійної моделі;

– параметри нелінійної регресійної моделі;

м

е

т

– показники якості нелінійної регресійної моделі;

– показники якості тестової вибірки для побудованої моделі;

– допустимі інтервали значень метрик CBO,WMC та DIT;

– прогнозування оцінки кількості строк коду.

3.2.Вимоги до надійності

3.2.1 Вимоги до забезпечення надійного (сталого) функціонування програми

Передбачити обробку ситуацій некоректного вводу інформації з інформуванням користувача про шляхи усунення.

3.2.2 Час відновлення після відмови

Вимоги до часу відновлення після відмови не пред’являються.

3.2.3 Відмови через некоректні дії оператора

Відмова можлива при наступних умовах:

– внаслідок закриття програми під час аналізу, в цьому випадку користувачеві показується додаткове вікно, чи впевнений він що хоче закрити програму.

3.3. Умови експлуатації

3.3.1 Кліматичні умови експлуатації

Кліматичні умови експлуатації, при яких повинні забезпечуватися задані характеристики, повинні відповідати вимогам, що пред'являються до технічних засобів в частині умов їх експлуатації.

3.3.2 Вимоги до чисельності та кваліфікації користувача

Користувач повинен мати базові навички роботи з комп’ютером.

3.4 Вимоги до складу та параметрів технічних засобів

Для роботи із програмним забезпеченням необхідний наступний склад технічних засобів з мінімальними параметрами:

– CPU: Intel Pentium 1Ghz;

– RAM: 1 GB;

– HDD: 5 GB вільного місця.

Інтернет-з’єднання. Програмне забезпечення вимагає доступу до Інтернету.

3.5 Вимоги до інформаційної та програмної сумісності

Програма повинна працювати на ліцензійній версії операційної системи Windows, починаючи з Windows 7. Вихідний код програми має бути написаний на мові Dart, з використанням інтегрованого середовища розробки Microsoft Visual Studio Code. Для розробки використовується Flutter Framework версії 2.18.6 або вище.

В якості браузера використовувати Chrome, FireFox або Opera.

3.6 Вимоги до маркування та пакування

Продукт буде упакований в електронний архів і мати найменування FlutterMediaMetrics.rar

4. Вимоги до програмної документації

Склад програмної документації повинен включати в себе:

– технічне завдання на розробку програмного забезпечення;

– програму та методику випробувань;

– керівництво користувача;

– опис програми;

– код програми.

5. Техніко-економічні показники

У даній кваліфікаційній роботі техніко-економічні показники не розраховуються.

6. Стадії та етапи розробки

Стадії та етапи розробки представлені у таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки

№ п/п	Назва етапів роботи створення програмного забезпечення	Кінцеві терміни виконання
1.	Аналіз вимог до ПЗ	10.09.2024
2.	Розробка архітектури ПЗ	21.09.2024
3.	Розробка проекту ПЗ	16.10.2024
4.	Реалізація та тестування ПЗ	26.10.2024
5.	Випробування ПЗ	1.11.2024

7. Порядок контролю та приймання

Контроль за аналізом та проектуванням кожної окремої частини програмного забезпечення на кожному етапі з урахуванням вимог, визначених у технічному завдані. Кожна стадія розробки повинна бути представлена в зазначені строки та узгоджена із замовником.

Прийом проводяться відповідно до програми і методики випробувань і документують за допомогою протоколу проведення випробувань. У разі знаходження помилок підчас прийому програмного виробу складається акт про знайдені помилки, який підписуються представниками замовника і розробника і затверджуються керівником організації – замовника та організації – розробника. Розробник повинен на протязі не більше ніж 2 тижнів виправити зазначені помилки і оповістити замовника про повторне проведення перевірки.

ДОДАТОК Б – КОД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ «FlutterMediaMetrics»

main.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/gestures.dart';
import 'package:flutter_media_metrics/constants.dart';
import 'package:flutter_media_metrics/providers/app_navigation_provider.dart';
import 'package:flutter_media_metrics/providers/app_theme_provider.dart';
import 'package:flutter_media_metrics/providers/regression_model_provider.dart';
import 'package:flutter_media_metrics/services/data_manager.dart';
import 'package:flutter_media_metrics/services/utils.dart';
import 'package:flutter_media_metrics/ui/pages/home/home_page.dart';
import 'package:provider/provider.dart';

void main() {
  runApp(const MainApp());
}

class MainApp extends StatelessWidget {
  const MainApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MultiProvider(
      providers: [
        Provider(create: (_) => DataManager()),
        Provider(create: (_) => Utils()),
        ChangeNotifierProvider(
          create: (_) => RegressionModelProvider(),
        ),
        ChangeNotifierProvider(
          create: (_) => AppNavigationProvider(),
        ),
        ChangeNotifierProvider(
          create: (_) => AppThemeProvider(context),
        ),
```

```
      ],
      child: Builder(
        builder: (context) => CupertinoApp(
          theme: const CupertinoThemeData(
            brightness: Brightness.light,
            textTheme: CupertinoTextThemeData(
              textStyle: TextStyle(
                fontSize: 16,
                color: CupertinoColors.black,
              ),
            ),

          ),
          scrollBehavior: const CupertinoScrollBehavior().copyWith(
            dragDevices: {
              PointerDeviceKind.touch,
              PointerDeviceKind.mouse,
            },
          ),
          title: appName,
          debugShowCheckedModeBanner: false,
          home: const HomePage(),
        ),
      ),
    );
  }
}
```

regression_view.dart

```
import 'dart:math';

import 'package:flutter/cupertino.dart';
import 'package:flutter_media_metrics/providers/regression_model_provider.dart';
import 'package:flutter_media_metrics/ui/pages/home/widgets/metrics_card.dart';
import
'package:flutter_media_metrics/ui/pages/home/widgets/prediction_widget.dart';
import 'package:provider/provider.dart';
```

```
class MinMax {
  MinMax(this.min, this.max);

  final double min;
  final double max;
}

class MinMaxX {
  MinMaxX({
    required this.x1,
    required this.x2,
    required this.x3,
  });

  final MinMax x1;
  final MinMax x2;
  final MinMax x3;
}

class RegressionView extends StatefulWidget {
  const RegressionView({super.key});

  @override
  State<RegressionView> createState() => _RegressionViewState();
}

class _RegressionViewState extends State<RegressionView> {
  late MinMaxX minMaxX;

  String getLinearEquation(String b0, String b1, String b2, String b3) {
    // ignore: prefer_interpolation_to_compose_strings
    return r'\hat{Z_Y} = ' +
        b0 +
        ' + ' +
        b1 +
        r' \cdot Z_{X_1} + ' +
        b2 +
        r' \cdot Z_{X_2} + ' +
        b3 +
        r' \cdot Z_{X_3} ';
```

```
}

String getNonlinearEquation(String b0, String b1, String b2, String b3) {
  // ignore: prefer_interpolation_to_compose_strings
  return r'\hat{Y} = 10^{' +
      b0 +
      r'} \cdot X_1^{' +
      b1 +
      r'} \cdot X_2^{' +
      b2 +
      r'} \cdot X_3^{' +
      b3 +
      '}';
}

@override
Widget build(BuildContext context) {
  final provider = context.watch<RegressionModelProvider>();
  minMaxX = MinMaxX(
    x1: MinMax(
      provider.x1Data.reduce(min),
      provider.x1Data.reduce(max),
    ),
    x2: MinMax(
      provider.x2Data.reduce(min),
      provider.x2Data.reduce(max),
    ),
    x3: MinMax(
      provider.x3Data.reduce(min),
      provider.x3Data.reduce(max),
    ),
  );

  final b0 = provider.coefficients[0].toStringAsFixed(4);
  final b1 = provider.coefficients[1].toStringAsFixed(4);
  final b2 = provider.coefficients[2].toStringAsFixed(4);
  final b3 = provider.coefficients[2].toStringAsFixed(4);
  return Padding(
    padding: const EdgeInsets.symmetric(horizontal: 20),
    child: ListView(
```

```
      children: [
        Text(
          'Регресійний Аналіз',
          style: CupertinoTheme.of(context).textTheme.textStyle.copyWith(
              fontSize: 18,
            ),
          textAlign: TextAlign.center,
        ),
        const SizedBox(height: 20),
        MetricsCard(
          title: 'Рівняння лінійної регресії',
          value: getLinearEquation(b0, b1, b2, b3),
          isEquation: true,
        ),
        const SizedBox(height: 20),
        MetricsCard(
          title: 'Рівняння нелінійної регресії',
          value: getNonlinearEquation(b0, b1, b2, b3),
          isEquation: true,
        ),
        const SizedBox(height: 16),
        quality(provider),
        const SizedBox(height: 20),
        const PredictionWidget(),
        _availableFactorsRange(),
        const SizedBox(height: 40),
      ],
    ),
  );
}

Widget quality(RegressionModelProvider provider) => Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Align(
          child: Text(
            'Показники якості нелінійної регресії',
            style: CupertinoTheme.of(context).textTheme.textStyle.copyWith(
                fontSize: 18,
              ),
```

```
          textAlign: TextAlign.center,
        ),
      ),
      const SizedBox(height: 16),
      MetricsCard(
        title: 'R²',
        value: provider.modelQuality.rSquared.toStringAsFixed(4),
      ),
      const SizedBox(height: 16),
      MetricsCard(
        title: 'MMRE',
        value: provider.modelQuality.mmre.toStringAsFixed(4),
      ),
      const SizedBox(height: 16),
      MetricsCard(
        title: 'PRED(0.25)',
        value: provider.modelQuality.pred.toStringAsFixed(4),
      ),
      const SizedBox(height: 16),
    ],
  );

Widget _availableFactorsRange() {
  String formatNumber(double number) => number.toStringAsFixed(4);

  final minX1 = formatNumber(minMaxX.x1.min);
  final maxX1 = formatNumber(minMaxX.x1.max);

  final minX2 = formatNumber(minMaxX.x2.min);
  final maxX2 = formatNumber(minMaxX.x2.max);

  final minX3 = formatNumber(minMaxX.x3.min);
  final maxX3 = formatNumber(minMaxX.x3.max);

  return Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      const Align(
        child: Text(
          'Допустимі значення X',
```

```
            style: TextStyle(fontSize: 16),
          ),
        ),
        const SizedBox(height: 8),
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
          children: [
            Flexible(
              child: MetricsCard(
                title: 'CBO',
                value: '[$minX1 - $maxX1]',
              ),
            ),
            const SizedBox(width: 16),
            Flexible(
              child: MetricsCard(
                title: 'WMC',
                value: '[$minX2 - $maxX2]',
              ),
            ),
            const SizedBox(width: 16),
            Flexible(
              child: MetricsCard(
                title: 'DIT',
                value: '[$minX3 - $maxX3]',
              ),
            ),
          ],
        ),
      ],
    );
  }
}
```

ДОДАТОК В – ОПИС ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ «FlutterMediaMetrics»

Програмне забезпечення «FlutterMediaMetrics» розроблено для оцінювання кількості строк коду (LOC) мультимедійних застосунків для платформи Android, створених за допомогою фреймворку Flutter. Воно дозволяє автоматизувати процес аналізу, базуючись на запропонованій нелінійній регресійній моделі, та забезпечує точність і ефективність оцінювання.

Функціональні можливості:

1. Завантаження метрик.

Програма працює з файлами формату CSV, що містять значення метрик: KLOC (Thousands of Lines of Code), (Weighted Methods per Class), CBO (Coupling Between Object Classes), DIT (Depth of Inheritance Tree).

3. Побудова прогнозів:

– Розрахунок параметрів моделі, коефіцієнтів b_0, b_1, b_2, b_3 (2.8).

– Використання нелінійної регресійної моделі для оцінювання LOC на основі метрик CBO, WMC, DIT.

3.Тестування моделі:

– Визначення показників якості побудованої моделі, таких як R^2, *MMRE*, *PRED (0,25)*.

– Оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter

Мова розробки та технології:

«FlutterMediaMetrics» розроблено на платформі Flutter з використанням мови програмування Dart. Мова програмування Dart – основна мова Flutter, яка підтримує декларативний стиль програмування і забезпечує високу

продуктивність застосунків. Dart використовується для реалізації всіх функцій програмного забезпечення, включаючи збір метрик, аналіз даних і побудову моделей. Фреймворк Flutter – кросплатформний фреймворк від Google, що дозволяє створювати додатки для Android, iOS, Windows, macOS та вебплатформ. Використовується для реалізації графічного інтерфейсу користувача та інтеграції модулів.

Принципи роботи:

– Користувач завантажує файл із метриками у форматі CSV.

– На основі наданих метрик будується нелінійна регресійна модель

– Користувач може отримати оцінку кількості строк коду

Технічні характеристики:

Системні вимоги:

– Операційна система: Windows.

Мінімальні вимоги:

– Процесор із частотою 1 ГГц або вище.

– Оперативна пам’ять: 512 МБ і більше.

– Дисковий простір: 100 МБ.

Функціональні особливості:

– Підтримка роботи з великими наборами даних.

– Швидке виконання обчислень завдяки алгоритмам у Dart.

Переваги використання:

– Підвищена достовірність: Завдяки адаптованій нелінійній регресійній моделі програмне забезпечення враховує специфіку організації коду в проєктах на платформі Flutter, що забезпечує високу точність прогнозів.

– Ефективне планування: Отримані оцінки кількості строк коду дозволяють розробникам і менеджерам проєктів краще планувати ресурси, часові рамки та етапи реалізації.

– Економія ресурсів: Зменшується потреба у залученні великої кількості аналітиків та витратах часу на розробку власних інструментів для аналізу коду.

Програмне забезпечення «FlutterMediaMetrics» рекомендовано для використання розробниками, менеджерами проектів і командами, які працюють з кросплатформними застосунками, де важлива ефективність прогнозування та оптимізація ресурсів.

ДОДАТОК Г – ІНСТРУКЦІЯ КОРИСТУВАЧА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1. Завантаження та встановлення

– Завантажте виконуваний файл FlutterMediaMetrics.

– Розпакуйте архів та запустіть файл FlutterMediaMetrics.exe

2. Завантаження даних

– Натисніть кнопку «Завантажте CSV файл».

– Виберіть файл у форматі CSV з даними про кількість класів та рядків коду.

– Дані будуть автоматично завантажені та оброблені.

На рисунку Г.1 представлено завантаження даних.

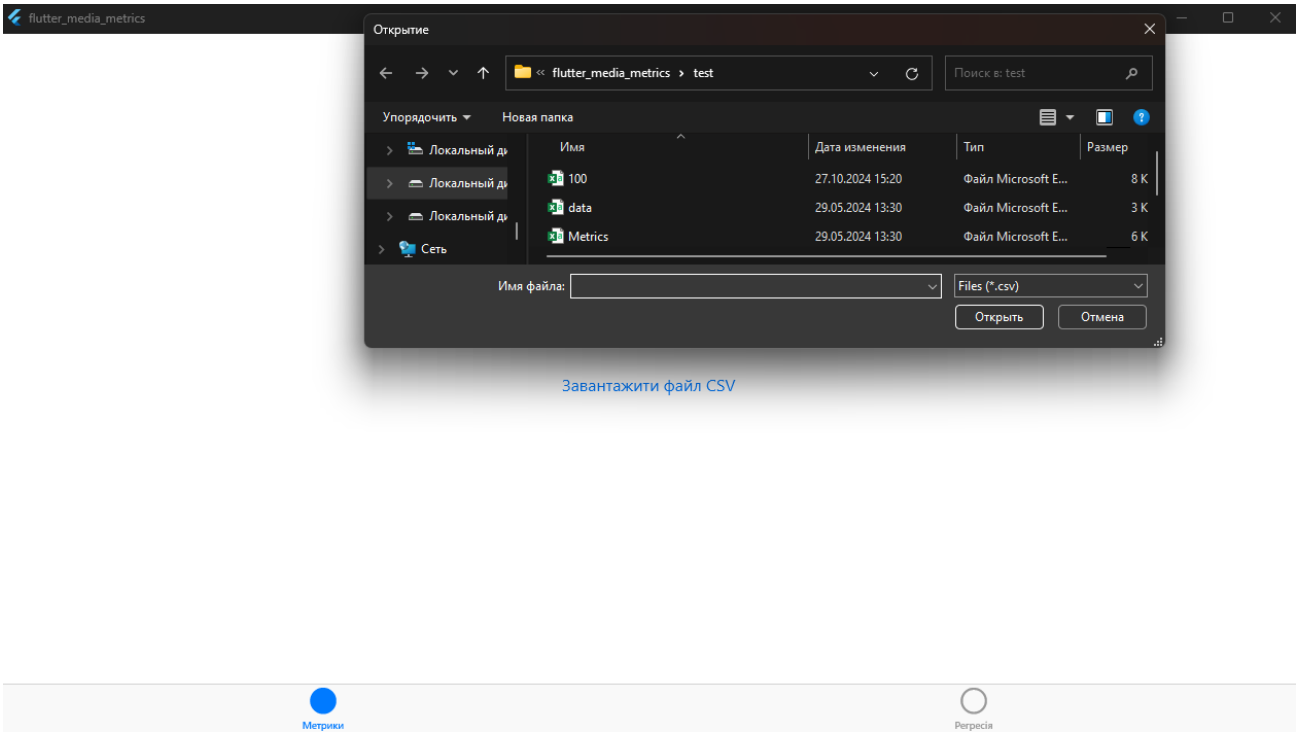

Рисунок Г.1 – Завантаження даних у «FlutterMediaMetrics»

3. Перегляд та аналіз даних

– Використовуйте нижнє меню для переходу між різними вкладками з даними.

– Вкладка «Метрики» – перегляд завантажених проектів.

– Вкладка «Регресія» – перегляд побудованої лінійної регресійної моделі, її коефіцієнтів та статистик якості.

4. Використання побудованої моделі

– Перейдіть до вкладки «Регресія».

– Введіть бажану кількість класів в поле «Введіть значення X1 (CBO)».

– Введіть бажану кількість методів в поле «Введіть значення X2 (WMC)».

– Введіть бажану кількість класів в поле «Введіть значення X3 (DIT)».

– Змінюйте значення та переглядайте прогноз рядків коду.

На рисунку Г.2 представлено використання побудованої моделі.

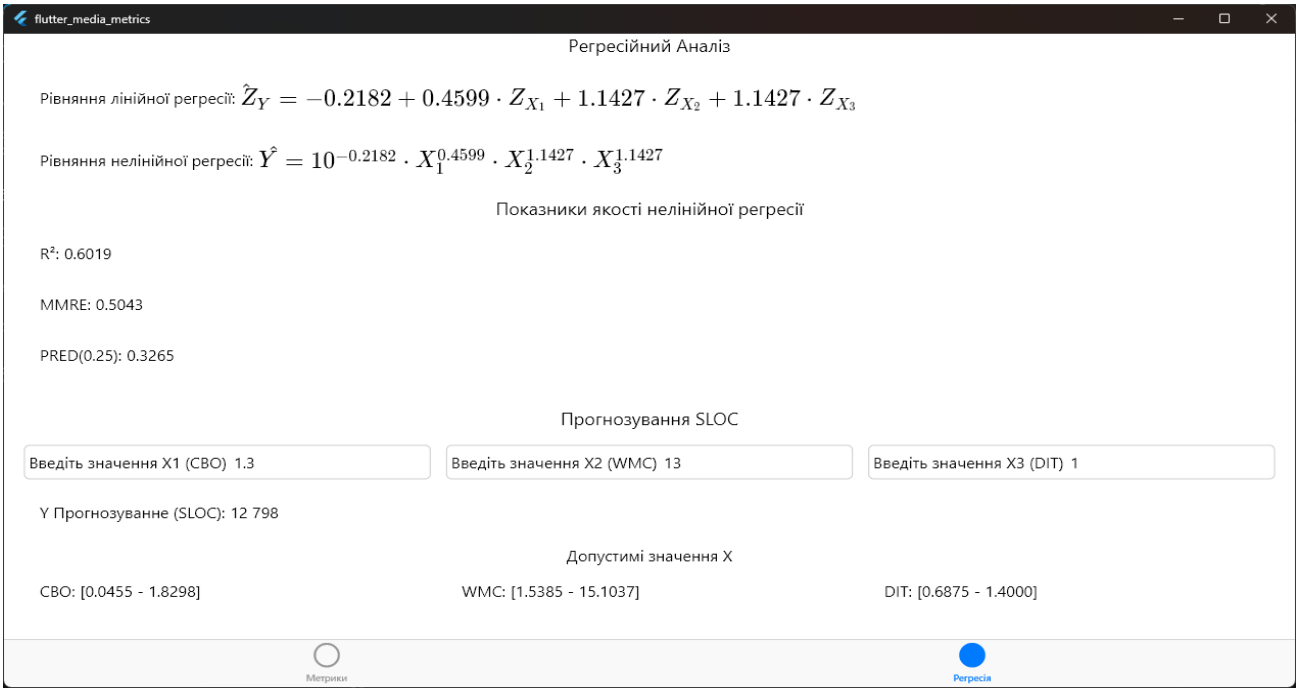

Рисунок Г.2 – Використання побудованої моделі у «FlutterMediaMetrics»

ДОДАТОК Д – ПРОГРАМА І МЕТОДИКА ВИПРОБУВАНЬ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ «FlutterMediaMetrics»

1. Об’єкт випробування

Об'єктом випробування є програмне забезпечення «FlutterMediaMetrics» для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter. Програмне забезпечення використовує нелінійну регресійну модель, яка базується на метриках CBO, WMC та DIT для оцінювання строк коду.

2. Мета випробування

Метою випробування програмного забезпечення «FlutterMediaMetrics» є перевірка функціональної відповідності, достовірності прогнозів і стабільності роботи програми відповідно до заявлених технічних і функціональних характеристик. Конкретні цілі випробування:

– Оцінку функціональної відповідності програмного забезпечення заявленим вимогам.

– Перевірку коректності розрахунків.

– Тестування продуктивності та стійкості до навантажень.

– Виявлення та аналіз можливих дефектів і недоліків у роботі.

3. Вимоги до програмного забезпечення

Програмне забезпечення «FlutterMediaMetrics» має відповідати вимогам, викладеним у технічному пункті (Додаток А). Перевіряються такі аспекти:

– Завантаження та обробка метрик.

– Побудова та тестування регресійної моделі.

– Генерація прогнозів і аналізу даних.

4. Вимоги до програмної документації

Програмна документація повинна містити такі документи:

– Технічне завдання.

– Опис програми.

– Інструкцію користувача.

– Програму та методику випробувань.

5. Засоби і порядок випробувань

5.1 Технічні засоби, що використовуються під час випробувань

– CPU:AMD Ryzen 5 5600x 6-core.

– RAM: 32 GB 3200MHz.

– SSD: Samsung SSD 500GB.

– Ethernet: TP-LINK

5.2 Програмні засоби, що використовуються під час випробувань

– Windows 11 Home x64 23H2.

– Visual Studio Code 1.95.3.0.

– Visual Studio Community 2022 17.11.5.

– Flutter 3.24.5.

– Dart 3.5.4.

5.3 Порядок проведення випробувань

Випробування програми складаються з таких етапів:

– Перевірити наявності та відповідності документації.

– Зробити тестування функціональних характеристик, зокрема основних модулів, заготовлених сценаріїв і обробки різних типів даних.

6 Методи випробувань

6.1 Методика проведення перевірки вимог до програмної документації

Перевірка документації починається з аналізу її структури та змісту. Оцінюється, чи відповідає документація визначеним стандартам і чи містить усі

необхідні елементи, такі як опис функцій, структура даних, методика тестування. Особлива увага приділятиметься перевірці точності та повноти описів програмних модулів, методів взаємодії між ними та алгоритмів роботи. Також буде проаналізоване оформлення документації: наявність заголовків, списків, зображень, схем, нумерації сторінок, а також відповідність стилю написання технічним вимогам. Документи повинні бути зрозумілими, послідовними та зручними для читання.

6.2 Методика проведення перевірки вимог до функціональних характеристик програмного продукту

Тестування функціональних характеристик програмного забезпечення «FlutterMediaMetrics» охоплює кілька ключових сценаріїв:

– Завантаження метрик з CSV-файлів. Цей тест перевіряє коректність зчитування даних метрик CBO, WMC і DIT із CSV-файлів. Оцінюється, чи правильно програма обробляє вхідні дані, зокрема, перевіряється відповідність форматів та коректність імпорту.

– Побудова моделі. На цьому етапі тестується коректність обчислення параметрів моделі (b_0, b_1, b_2, b_3), які використовуються для прогнозування кількості строк коду (LOC). Перевіряється, чи відповідають розрахунки математичній моделі та заданим технічним вимогам.

– Тестування моделі. Виконується оцінка якості побудованої моделі. Застосовуються метрики, такі як коефіцієнт детермінації (R^2), середньоквадратична похибка (MSE) та точність прогнозу (PRED(0.25)). Це дозволяє оцінити, наскільки ефективно модель виконує свої функції на тестовій вибірці.

– Розрахунок інтервалів прогнозу. У цьому сценарії перевіряється функція обчислення довірчих інтервалів для прогнозованих значень. Оцінюється достовірності інтервалів та відповідність їх розрахунків очікуваним результатам.

– Обробка некоректних даних. Цей тест спрямований на перевірку стійкості програми до роботи з некоректними або порожніми даними. Аналізується, чи вміє програма виявляти помилки у вхідних файлах, повідомляти про них користувача та уникати аварійного завершення роботи.

У таблиці Д.1 представлено перелік випробувань.

Таблиця Д.1 – Перелік випробувань основних функціональних можливостей

№	Дія	Очікуваний результат
1	Завантаження метрик із файлу CSV	Дані метрик (SLOC, CBO, WMC, DIT) успішно завантажено без помилок
2	Побудова нелінійної регресійної моделі на основі навчальної вибірки	Модель побудовано, параметри (b_0, b_1, b_2, b_3) розраховано
3	Тестування моделі	Показники R^2, MMRE, PRED(0.25) відповідають очікуваним значенням
4	Оцінювання кількісті строк коду на основі метрик CBO, WMC та DIT	Програма класифікує оцінку строк коду у проектах
5	Обробка некоректних даних або порожнього файлу	Програма видає відповідне попередження або повідомлення про помилку

В результаті проведення випробувань «FlutterMediaMetrics» для оцінювання кількості строк коду мультимедійних застосунків для Android, що створюються на платформі Flutter необхідно перевірити його функціональні характеристики.