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
_____
«___» _____ 2021 р.

КВАЛІФІКАЦІЙНАРОБОТА
на здобуття ступеня вищої освіти «магістр»

на тему: Регресійна модель для оцінювання розміру телеграм-ботів, що створюються на Python, та розробка програмного забезпечення для її реалізації

Виконав: студент 6151м групи
Шакула О.В.
(підпис, ПІБ)

Керівник роботи:
доцент, к.ф.-м.н., доцент
(посада, науковий ступень вчене звання)

Латанська Л.О.
(підпис, ПІБ)

Миколаїв – 2021 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
(підпис)
«___» _____ 2021 р.

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

Студенту Шакулі Олександру Валерійовичу
(Прізвище, ім'я, по батькові)

1. Тема роботи: Регресійна модель для оцінювання розміру телеграм-ботів, що створюються на Python, та розробка програмного забезпечення для її реалізації
Керівник роботи Латанська Людмила Олексіївна, к.ф.-м.н., доцент
Затверджені наказом ректора № 1239уч від «13» 10 2021 р.
2. Термін подання роботи: 06.12.2021 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; Проект 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 1.Актуальність теми, 2. Мета і завдання дослідження, 3.Об'єкт і предмет дослідження, 4. Методи дослідження, 5. Наукова новизна одержаних результатів, 6. Практичне значення отриманих результатів, 7. Апробація результатів досліджень, 8. Розподіли вихідних та нормалізованих даних для кількості класів X, 9. Розподіли вихідних та нормалізованих даних для кількості строк коду Y, 10. Нелінійна регресійна модель для оцінювання розміру телеграм-ботів на Python, 11. Лінійна регресійна модель для оцінювання розміру телеграм-ботів на Python, 12. Метрики якості, 13. Контекстна діаграма програмного забезпечення, 14. Концептуальна модель даних ПЗ, 15. STD програмного забезпечення, 16. DFD першого рівня ПЗ оцінювання розміру телеграм-ботів на Python, 17. Деталізація процесу «Прогнозувати» ПЗ оцінювання розміру телеграм-ботів на Python, 18. Логічна модель даних ПЗ оцінювання розміру телеграм-ботів на Python, 19.

Екранні форми ПЗ оцінювання розміру телеграм-ботів на Python, 20.Висновки.

6. Консультанти розділів роботи

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

7. Дата видачі завдання 13.10.2021 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	18.10.2021	НС*
2.	Підготовка розділу з огляду літератури та обґрунтування необхідності проведення досліджень за обраною темою	19.10.2021	НС*
3.	Підготовка розділу (ів) з результатів власних досліджень	22.10.2021	НС*
4.	Підготовка розділу з проекту програмного забезпечення	16.11.2021	
5.	Підготовка організаційно-економічного розділу	19.11.2021	
6.	Підготовка розділу з охорони праці	22.11.2021	
7.	Підготовка розділу з охорони навколишнього середовища	24.11.2021	
8.	Підготовка розділу Висновки	25.11.2021	
9.	Оформлення списку використаних джерел та додатків	29.11.2021	
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.)	06.12.2021	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку
11.	Підготовка презентації та доповіді	09.12.2021	
12.	Попередній захист роботи на засіданні кафедри ПЗАС	10.12.2021	
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	20.12.2021	

* - за результатами наукового стажування (НС), яке було з 01.09.2021 до 10.10.2021 р.

Студент _____ Шакула О.В.
(підпис) (ПІБ)

Керівник роботи _____ Латанська Л.О.
(підпис) (ПІБ)

РЕФЕРАТ

Шакула Олександр Валерійович

Регресійна модель для оцінювання розміру телеграм-ботів, що створюються на Python, та розробка програмного забезпечення для її реалізації

Кваліфікаційна робота на здобуття освітнього рівня магістра зі спеціальності 121 – «Інженерія програмного забезпечення». Національний університет кораблебудування імені адмірала Макарова. Миколаїв, 2021 р.

Обсяг роботи: 103 сторінки, 15 таблиць, 11 рисунків, 19 використаних джерел, 5 додатків.

Актуальність теми. Розмір ПЗ є визначальним для багатьох оцінок програмного продукту. На цей час ще не існує ефективної, універсальної методики оцінювання розміру програмного забезпечення, що розробляється.

Тому удосконалення методів та моделей оцінювання розміру ПЗ взагалі, та в їх числі і телеграм-ботів, що створюються на Python, є актуальною задачею інженерії програмного забезпечення.

Мета і завдання дослідження. Метою роботи є підвищення достовірності оцінювання розміру телеграм-ботів, що створюються на Python, та визначає наступні завдання: виконати порівняльний аналіз методів та моделей для оцінювання розміру ПЗ; обґрунтувати необхідність удосконалення регресійної моделі для оцінювання розміру телеграм-ботів; виконати специфікацію моделі; оцінити параметри, рівень довіри, прогнозні можливості та якість моделі.

Об'єктом досліджень є процес оцінювання розміру телеграм-ботів, що створюються на Python.

Предметом досліджень є однофакторна нелінійна регресійна модель для оцінювання розміру телеграм-ботів, що створюються на Python.

Методи дослідження. При дослідженні використовувалися наступні методи: регресійний аналіз, математична статистика, теорія ймовірності і структурне моделювання.

Наукова новизна одержаних результатів. Удосконалено однофакторну регресійну модель для оцінювання розміру телеграм-ботів, що створюються на Python, за рахунок використання нормалізуючого перетворення десятковий логарифм, що дозволяє підвищити достовірність оцінювання розміру телеграм-ботів порівняно з лінійними моделями.

Практичне значення отриманих результатів. Методики, алгоритми і ПЗ, що були розроблені в кваліфікаційній роботі, можуть бути використані при проектуванні програмного забезпечення для пришвидшення розрахунків та підвищення точності.

Апробація результатів досліджень. Основні положення роботи були оприлюднені на II Всеукраїнській науково-практичній інтернет конференції «Інформаційні технології: моделі, алгоритми, системи» (м. Миколаїв,26-28 жовтня 2021 р.).

Ключові слова: ОЦІНЮВАННЯ РОЗМІРУ, ТЕЛЕГРАМ-БОТ, PYTHON, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ОДНОФАКТОРНА НЕЛІНІЙНА РЕГРЕСІЙНА МОДЕЛЬ

ABSTRACT

Shakula Oleksandr

A regression model for estimating the size of telegrams-bots created in Python and developing the software for its implementation

Qualification work for obtaining a master's degree in Specialty 121 – Software Engineering. Admiral Makarov National University of Shipbuilding. Mykolaiv, 2021.

The work contains: 103 pages, 15 tables, 11 figures, 19 references, 5 appendices.

Relevance of the topic. The size of the software is crucial for many software product evaluations. At present, there is no effective, universal method for estimating the size of software being developed.

Therefore, the improvement of methods and models for estimating the size of software in general, and including telegrams-bots created in Python, is an urgent task of software engineering.

The purpose and objectives of the study. The purpose of the work is to increase the reliability of estimating the size of telegrams-bots created on Python, and defines the following tasks: to perform a comparative analysis of methods and models for estimating the size of software; justify the need to improve the regression model for estimating the size of telegram bots; perform model specification; assess the parameters, level of trust, forecast capabilities and quality of the model.

The object of research is the process of estimating the size of telegram bots created in Python.

The subject of research is a one-factor nonlinear regression model for estimating the size of telegram bots created in Python.

Research methods. The following methods were used in the study: regression analysis, mathematical statistics, probability theory and structural modeling.

Scientific novelty of the obtained results. Improved one-factor regression model for estimating the size of telegrams-bots created in Python, using the normalizing conversion of the decimal logarithm, which increases the reliability of estimating the size of telegrams-bots compared to linear models.

The practical significance of the results obtained. Techniques, algorithms and software developed in the qualification work can be used in software design to speed up calculations and increase accuracy.

Approbation of research results. The main provisions of the work were published at the II All-Ukrainian scientific-practical Internet conference "Information technologies: models, algorithms, systems" (Mykolaiv, October 26-28, 2021).

Keywords: SIZE ESTIMATION, TELEGRAM BOT, PYTHON, SOFTWARE, ONE-FACTOR NONLINEAR REGRESSION MODEL

ЗМІСТ

ВСТУП

Актуальність теми. Розрахунок витрат – одне з найскладніших завдань в управлінні проектом. Процес оцінювання витрат на створення програмного забезпечення включає оцінювання розміру програмного продукту, який розроблятиметься, оцінювання необхідних зусиль і ресурсів, написання попередніх планів робіт і, нарешті, розрахунок повної вартості проекту [1,2].

Одним з основних показників, які впливають на вартість розробки, є розмір.

Існує ряд методів та моделей для визначення розміру програмного забезпечення. Досить часто для прогнозування розміру використовуються регресійні моделі. Відомо, що розмір програми в значній мірі залежить від особливостей мови програмування, методології розробки, платформи розробки та інших факторів. Тому існуючі методи та моделі прогнозування розміру програмного продукту призначені для вузького класу задач. І тому для підвищення точності оцінювання розміру програмного забезпечення певного класу необхідно удосконалювати існуючі та розробляти нові методи та моделі, що робить задачу удосконалення регресійної моделі для оцінювання розміру телеграм-ботів, що створюються на Python, актуальною [3-5].

Мета і завдання дослідження. Мета роботи полягає в підвищенні достовірності оцінювання розміру телеграм-ботів на Python.

Мета роботи визначає наступні завдання:

– Виконати порівняльний аналіз методів та моделей для оцінювання розміру програмного забезпечення;

– Обґрунтувати необхідність удосконалення регресійної моделі для оцінювання розміру телеграм-ботів, що створюються на Python.

– Зібрати емпіричні дані.

– Виконати передобробку емпіричних даних.

– Виконати специфікацію моделі, тобто обрати вид функціональної залежності.

– Оцінити параметри моделі.

– Оцінити рівень довіри до отриманої моделі

– Оцінити прогнозні можливості моделі.

– Виконати порівняння лінійної моделі для вихідних даних у припущенні про нормальність їх розподілу та отриманої нелінійної моделі для вихідних даних по R^2, MMRE та PRED(0,25).

– Розробити ПЗ для оцінювання розміру телеграм-ботів, що створюються на Python.

Об'єктом досліджень є процес оцінювання розміру телеграм-ботів, що створюються на Python.

Предметом досліджень є однофакторна нелінійна регресійна модель для оцінювання розміру телеграм-ботів, що створюються на Python.

Методи дослідження. При дослідженні використовувалися наступні методи: регресійний аналіз, математична статистика, теорії ймовірності і структурне моделювання.

Наукова новизна одержаних результатів. Удосконалено однофакторну регресійну модель для оцінювання розміру телеграм-ботів, що створюються на Python, за рахунок використання нормалізуючого перетворення десятковий логарифм, що дозволяє підвищити достовірність оцінювання розміру телеграм-ботів порівняно з лінійними моделями.

Практичне значення отриманих результатів. Методики, алгоритми і ПЗ, що були розроблені в кваліфікаційній роботі, можуть бути використані при проектуванні програмного забезпечення для пришвидшення розрахунків та підвищення точності.

Особистий внесок здобувача. Усі результати кваліфікаційної роботи, що виносяться на захист, отримані автором особисто. У тезах доповіді [6], що

написані у співавторстві з керівником роботи, здобувачу належить розробка методики удосконалення регресійної моделі для оцінювання розміру телеграм-ботів на Python.

Апробація результатів досліджень. Основні положення роботи були оприлюднені на II Всеукраїнській науково-практичній інтернет конференції «Інформаційні технології: моделі, алгоритми, системи» (м. Миколаїв,26-28 жовтня 2021 р.).

Публікації. По результатам досліджень опубліковано 1 наукову працю – тези конференції.

1 ДОСЛІДЖЕННЯ ІСНУЮЧИХ МЕТОДІВ І МОДЕЛЕЙ ОЦІНЮВАННЯ РОЗМІРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Мова програмування Python: переваги, недоліки та сфери застосування

Особливості, переваги та недоліки мови програмування Python

Python – це мова програмування з досить тривалою історією. Розробник Python – Гвідо ван Россума. Версія Python 1.0 з'явилася 1994 року. Мова постійно розвивається, додаються нові функції, які залучають все більше розробників. Щорічно в мову вносяться значні доповнення, що випускаються із новими версіями. 2020 року з'явилася версія 3.9. Згідно з рейтингом IEEE популярності мов програмування, в 2021 року рейтинг очолила мова Python [7]. Також Python вважається однією із мов, що має найшвидше зростання.

Python – високорівнева мова програмування, що має мінімум синтаксису та максимум корисних функцій. Розробники Python особливо виділяють продуктивність розробника та читання коду. Основні архітектурні риси мови – динамічна типізація, автоматичне управління пам'яттю, механізм обробки виняткових ситуацій, підтримка багатопоточних обчислень та зручні високорівневі структури даних.

В Python підтримується структурний, об'єктно-орієнтований та функціональний стилі програмування. Код Python організовується в функції і класи, які можуть об'єднуватися в модулі. Модулі можуть бути об'єднані в пакети.

Основними перевагами мови Python є

– Низький поріг входження. Початківцям програмістам Python дається легше, ніж інші мови.

– Логічна, лаконічна та зрозуміла. У порівнянні з багатьма іншими мовами Python має синтаксис, що легко читається.

– Кросплатформова. Підходить для різних платформ.

– Інтерпретована мова. Це означає, що до запуску Python не компілюється та є звичайним текстовим файлом.

– Динамічна типізація. Можна передавати функції будь-який тип даних, попередньо його не вказуючи.

– Широке застосування. Має широку сферу застосування.

– Сильна інтернет-спільнота та багато конференцій. У мови величезна підтримка і безліч додатків від сторонніх розробників

– Потужна підтримка компаній-гігантів IT-індустрії. Такі компанії, як Google, Facebook, Dropbox, Netflix на певних етапах розробки використовували Python.

– Висока затребуваність ринку праці. Зростає попит незважаючи на збільшення кількості програмістів, що працюють цією мовою.

– У Python багато якісних бібліотек та фреймворків. Це спрощує створення програм.

– Python висуває жорсткі вимоги до написання коду (вимагає відступів),

– Менше коду. Розробка на Python йде швидше, ніж іншими мовами.

– Гнучкість та масштабованість. Легко розширювати складні програми.

Недоліками мови є:

– Повільність.

– Залежність від системних бібліотек

– Не дуже правильно розпоряджається пам'яттю.

– Інтерпретатор не дозволяє одночасно виконувати кілька потоків.

Сфери застосування мови програмування Python

Python сьогодні використовується у різних сферах. Свої програми на Python розробляють компанії різного рівня.

Основними сферами застосування Python є:

– Наукові дослідження;

– Машинне навчання, аналітика;

– Аналіз великих даних;

– Розробка веб-додатків,

– Розробка мобільних додатків;

– Розробка вбудованих систем;

– Розробка ігор;

– Створення скриптів;

– Парсинг даних;

– Системне адміністрування.

Розглянемо деякі приклади застосування Python:

– Відомі торгові мережі Amazon та Spotify використовують Python для аналізу даних та створення рекомендацій;

– Walt Disney використовує Python для створення анімацій;

– Instagram та Youtube написані на Python;

– NASA застосовує мову для наукових обчислень;

– Агенство національної безпеки США застосовує Python для шифрування та аналізу інформації.

– iRobot використовує мову для розробки роботизованих пристроїв.

І таких прикладів можна навести дуже багато. Наприклад, на Python розробляються Telegram боти. Головне завдання бота – автоматична відповідь після введеної користувачем команди. Написані для платформи Telegram, боти призначені для виконання різних функцій: від отримання новин до пошуку інформації.

Таким чином, мова програмування Python розвивається, зростає її популярність та розширюються сфери застосування.

1.2 Дослідження метрик кількісного обчислення розміру програмного забезпечення

Оцінювання результату програмних проектів – дуже складне завдання. Найчастіше якісь показники потрібні ще до початку робіт, щоб спланувати майбутні витрати, термін розробки, визначити розмір винагороди за працю.

Наука не стоїть на місці і пропонує нові підходи до вирішення цього завдання, проте досі немає надійної універсальної методики, що дає гарантований результат. Тому в більшості випадків доводиться випробовувати різноманітні методи.

Обчислення багатьох показників програмного проекту базується на такій метриці, як розмір програмного забезпечення [8-12].

Основними одиницями розміру є строки коду [8] та функціональні точки [11].

Метрика кількості строк коду визначається як загальна кількість строк вихідного коду, включаючи коментарі та порожні строки. У цієї метрики є багато похідних метрик:

– кількість порожніх строк;

– кількість строк, які містять коментарі;

– кількість строк, що містять вихідний код та коментарі;

– кількість строк, які містять декларативний вихідний код;

– відсоток коментарів;

– середня кількість строк для функцій;

– середня кількість строк, що містять вихідний код для функцій;

– середня кількість строк для модулів;

– середня кількість строк для класів та інші.

Перевагою використання строк коду в якості одиниці розміру є простота аналізу метрики, частота використання в проектах; простота підрахунку у випадку завершених проектів, недоліком – залежність від мов програмування та стандартів кодування, не врахування якості коду, складність в оцінці продуктивності.

Функціональна точка – це одиниця виміру функціональності.

Під функціональною точкою розуміється будь-який з наступних елементів програмного забезпечення, що розробляється:

– вхідний елемент програми (вхідний документ або екранна форма);

– вихідний елемент програми (звіт, документ, екранна форма);

– запит (пара «запитання/відповідь»);

– внутрішні файли (дані, що використовуються всередині програми);

– інтерфейс програми (дані, що передаються іншому додатку або одержуються від нього).

Застосування функціональних точок базується на вивченні вимог до проекту.

Від функціональних точок можна перейти до розміру коду програмного продукту у рядках. Для вирішення подібних завдань створено таблиці відповідностей для різних мов програмування з огляду на їх особливості. Цей показник використовується у формулах під час знаходження оцінки проекту.

1.3 Методи та моделі оцінювання розміру програмного забезпечення

Існує що найменш п’ять типів методів та моделей для оцінювання кількості строк, коду а саме: на основі експертних оцінок, аналогові, регресійні, які базуються на функціональних точках, параметричні.

– Експертне оцінювання. Даний метод ґрунтується на зібранні оціночних думок інженерів-розробників програмного забезпечення, Потім оцінки колективно обговорюються, а результати узагальнюються. Надалі пропозиції та оцінки протоколюють та обговорюють на загальних зборах, а результати оцінювання інтегрують у єдину систему. Весь процес основується на дельфійському методі, який орієнтується на досягнення єдиної думки при прийнятті рішення.

– Оцінювання за аналогією. Метод оцінювання за аналогією оснований на використанні емпіричних даних про проекти, що раніше виконувались. На відміну від алгоритмічних методів, де ці дані використовуються для налаштування моделі, метод оцінювання за аналогією дозволяє визначити схожі проекти-аналоги і складається з трьох кроків:

1) збір даних із проекту;

2) пошук та аналіз проектів-аналогів;

3) експертне оцінювання нового проекту. Цей етап є останнім, за основу оцінювання беруться значення характеристик знайдених проектів-аналогів та безпосередньо за ними проводиться оцінювання нового проекту.

– Регресійні методи, в основі яких лежить формула регресії з параметрами, які отримуються із попередньо виконаних проектів.

Регресійні методи відносяться до алгоритмічних методів. В основі будь-якого алгоритмічного методу оцінювання програмного забезпечення лежить одна або кілька функцій, що описує залежність характеристик проекту від тих чи інших незалежних параметрів [12].

В якості незалежних параметрів для оцінювання розміру програмного забезпечення часто використовують наступні метрики діаграми класів: загальна кількість класів, середня кількість методів, середня кількість атрибутів, загальна кількість методів, загальна кількість атрибутів, загальна кількість зв'язків, середня кількість атрибутів [3-5].

– Метод функціональних точок

Розглянемо методику функціональних точок, яку вперше запропонував співробітник IBM Алан Альбрехт у 1979 році. Суть методу розрахунку функціональних точок полягає в тому, що обсяг проекту розробки ПЗ оцінюється на основі логічної моделі кількості функціоналу, який є необхідний замовнику і який надається розробником. Отримана оцінка не залежить від мови програмування та середовища розробки. Недоліки методики функціональних точок полягають в тому, що результати базуються на експертних оцінках, використовуються непрямі виміри, досить трудомістка процедура підрахунку.

Існує ряд модифікацій методу функціональних точок:

– Метод точок властивостей (1988, Кейперс Джонс) – близький до методу функціональних точок, з тією лише відмінністю, що реалізовує коригування отриманої оцінки, враховуючи алгоритмічну складність.

– Метод Mark II (1988, Чарльз Саймонс) – надає можливість отримати один і той же результат, оцінюючи вся систему, чи підсумовуючи отримані оцінки всіх підсистем.

– Метод тривимірних функціональних точок (1991, софтверний підрозділ корпорації Boeing). Метод базується на ідеї, що складність задачі в програмному середовищі можна представити у трьох вимірах – дані (кількість вводів/виводів), функції (складність обчислень) та контроль (керуюча логіка).

– Метод об'єктних точок. Так як в базовій версії методу функціональних точок не передбачається застосування об'єктно-орієнтованого підходу, у сучасних проектах використовується його адаптований під об'єктно-орієнтовану методологію варіант.

1.4 Основні показники якості регресійних моделей

Розглянемо наступні показники якості регресійних моделей

– Коефіцієнт детермінації R^2 [13, 14]:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \qquad (1.1)$$

де y_i – емпірична величина y;

$\hat{y}_i$ – розраховане значення y згідно рівняння регресії;

$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ – середнє значення y_i ;

n – кількість значень.

Коефіцієнт детермінації для моделі з константою приймає значення від 0 до 1. Чим ближче значення коефіцієнта до 1, тим сильніша залежність. При оцінюванні регресійних моделей це інтерпретується як відповідність моделі

даним. Для прийнятних моделей передбачається, що коефіцієнт детермінації повинен бути хоча б не менше 50%. Моделі з коефіцієнтом детермінації вище 80% вважаються досить добрими. Значення коефіцієнта детермінації 1 означає функціональну залежність між змінними [13].

– Сума квадратів відхилень S_y:

$$S_y = \sum_{i=1}^{n} [y_i - \hat{y}_i]^2,$$

де y_i – емпіричне значення y;

$\hat{y}_i$ – розраховане значення y згідно рівняння регресії.

Сума квадратів відхилень S_y показує суму квадратів різниць між емпіричним значенням y_i та розрахованим значенням $\hat{y}_i$ згідно рівняння регресії.

– Величина відносної похибки MRE_i

Оцінками точності регресійних моделей, які частіше використовуються, є середня величина відносної похибки $MMRE$ і рівень прогнозування $PRED(l)$, які обчислюються через величини відносної похибки MRE_i [13, 14].

$$MRE_i = \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \qquad (1.2)$$

де y_i – емпіричне значення y;

$\hat{y}_i$ – розраховане значення y згідно рівняння регресії.

– Середня величина відносної похибки $MMRE$:

$$MMRE = \frac{1}{N} \sum_{i=1}^{N} MRE_i \qquad (1.3)$$

Вважається $MMRE \leq 0{,}25$ є прийнятним рівнем для моделі прогнозування.

– Рівень прогнозування $PRED(l)$:

$$PRED(l) = \frac{k}{N}, \tag{1.4}$$

де k – кількість значень з $MRE \leq l$.

Зазвичай використовується $PRED(0{,}25)$ але деякі дослідження також розглядають $PRED(0{,}3)$ з невеликою різницею в результатах. Зазвичай $PRED(0{,}25) \geq 0{,}75$ вважається прийнятною точністю моделі. $PRED(l)$ визначається як середнє значення відхилення відносної похибки MRE_i не більше, ніж на l.

1.5 Обґрунтування необхідності проведення досліджень

Із результатів аналізу існуючих методів та моделей для визначення розміру програмного забезпечення, в тому числі і телеграм-ботів на Python, випливає, що незважаючи на різноманітність методів і моделей оцінювання, досі не існує надійної універсальної методики, що дає гарантований результат з необхідною достовірністю.

Досить часто для прогнозування розміру використовуються регресійні моделі. Відомо, що розмір програми в значній мірі залежить від особливостей мови програмування, методології розробки, платформи розробки та інших факторів. Тому існуючі методи та моделі прогнозування розміру програмного продукту призначені для вузького класу задач. І тому для підвищення достовірності прогнозування розміру телеграм-ботів на Python необхідно удосконалити регресійну модель оцінювання розміру з урахуванням мови програмування, типу програмного забезпечення та технології розробки.

Зазвичай вихідні дані не мають нормального розподілу та нелінійно пов’язані. Тому для отримання якісного передбачення необхідно провести

нормалізацію даних, вилучити викиди, побудувати лінійну регресійну модель для нормалізованих даних без викидів, для неї у звичайний спосіб побудувати довірчі інтервали та інтервали передбачення. Нелінійну регресійну модель та відповідні довірчі інтервали та інтервали передбачення отримаємо через зворотне перетворення. Такий підхід дає змогу застосувати лінійний регресійний аналіз до нелінійних негаусівських даних та підвищити достовірність оцінювання розміру телеграм-ботів на Python. В той же час відбудеться калібровка параметрів моделі з врахуванням мови програмування, типу програмного забезпечення та технології розробки.

2 УДОСКОНАЛЕННЯ РЕГРЕСІЙНОЇ МОДЕЛІ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ТЕЛЕГРАМ-БОТІВ НА PYTHON ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ ПРОГРАМИ ДЛЯ ЇЇ РЕАЛІЗАЦІЇ

2.1 Основні етапи побудови нелінійної регресійної моделі оцінювання розміру телеграм-ботів на Python з використанням методу нормалізуючи перетворень

Розглянемо основні кроки побудови нелінійної регресійної моделі оцінювання розміру телеграм-ботів на Python з використанням методу нормалізуючи перетворень:

– Визначити залежні та незалежні величини.

– Зібрати емпіричні дані.

– Виконати передобробку емпіричних даних з нормалізацією та вилученням викидів

– Виконати специфікацію моделі, тобто обрати вид функціональної залежності.

– Виконати оцінювання параметрів моделі.

– Виконати оцінювання рівня довіри до отриманої моделі

– Виконати оцінювання прогнозних можливостей моделі.

– Виконати порівняння лінійної моделі для вихідних даних у припущенні про нормальність їх розподілу та отриманої нелінійної моделі для вихідних даних по R^2, MMRE та PRED(0,25).

Із огляду останніх досліджень та публікацій випливає, що застосування нелінійної регресійної теорії являється актуальним в задачах підвищення якості оцінювання розміру програмного забезпечення, в тому числі і телеграм-ботів на Python [15,17]. В якості незалежної змінної в дослідженнях

буде використовуватися кількість класів, в якості залежної – кількість строк коду.

2.2 Побудова удосконаленої регресійної моделі оцінювання розміру телеграм-ботів на Python

– Визначити незалежні та залежні змінні

В якості неалежної змінної для побудови регресійної моделі оцінювання розміру телеграм-ботів на Python обрано кількість класів *X*, в якості залежної змінної – кількість строк коду *Y*.

– Зібрати емпіричні дані

На ресурсі GitHub було зібрано інформацію по Telegram ботам на Python, а саме: кількість класів та кількість строк коду, для 40 проектів.

– Виконати передобробку емпіричних даних

За допомогою критерію згоди Пірсона χ^2 визначено, що емпіричні дані не являються нормально розподіленими. Тому виконано нормалізацію даних з використанням нормалізуючого перетворенням десятковий логарифм [16]:

$$Z_X = \log_{10} X, \tag{2.1}$$

$$Z_Y = \log_{10} Y, \tag{2.2}$$

де X, Y – емпіричні значення, а Z_X, Z_Y – нормалізовані значення.

З нормалізованих даних вилучено викиди.

При аналізі даних на викиди використовували квадрат відстані Махаланобіса, який обчислюється за формулою (2.1).

$$d_i^2 = (Z_i - \overline{Z})^T S_N^{-1} (Z_i - \overline{Z}), \tag{2.3}$$

де Z_i – i - ий елемент (Z_{x_i}, Z_{y_i}),

N – кількість даних без викидів;

$\overline{Z}$ – вектор з середніх значень по нормалізованим Z_Y, Zx;

S_N – коваріаційна матриця:

$$S_N = \frac{1}{N}\sum_{i=1}^{N}(Z - \overline{Z})\,(Z - \overline{Z})^T.$$

Для знаходженя викидів після обчислення d_i^2 можемо використовувати критерій Пірсона χ^2 чи критерій Фішера F. В роботі знаходилися викиди з допомогою критерію Фішера. Для використання критерію розрахували попередньо статистики

$$T_{S_i} = (N - k)Nd_i^2/(N^2 - 1)k,$$

де N – кількість значень без викидів,

k – кількість параметрів.

Ті елементи (Z_{x_i}, Z_{y_i}), для яких T_{S_i}перевищує критичне значення Фішера $F_{k,N-k,\alpha}$ для обраного рівня значущості α (0,05), являються викидами. Їх необхідно вилучати. Після усунення викидів для отриманого нового набору даних знову знаходиться квадрат відстані Махаланобіса. Процедура знаходження викидів повинна повторюватися до того моменту, поки не виконається нерівність $T_{S_i} \le F_{k,N-k,\alpha}$ для всіх i. Для нормалізованих даних було знайдено та вилучено 2 викиди.

Вихідні дані без викидів для побудови регресійної моделі наведені в таблиці 2.1.

Таблиця 2.1 – Вихідні дані для побудови регресійної моделі

№ пр.	X (класи)	Y (строки)
1	2	3
1	1	41
2	1	40
3	2	64
4	2	110
5	2	159
6	2	155
7	2	143

Продовження табл. 2.1

1	2	3
8	2	145
9	2	156
10	2	160
11	2	155
12	2	91
13	3	174
14	3	100
15	3	92
16	3	90
17	3	212
18	3	172
19	3	237
20	3	170
21	3	100
22	3	180
23	3	214
24	3	199
25	3	203
26	4	254
27	4	328
28	4	300
29	4	325
30	4	470
31	4	330
32	4	322
33	4	336
34	5	510
35	5	500
36	6	502
37	7	794
38	9	980

Нормалізовані величини Z_Y та Zx без викидів представлені в таблиці 3.2.

Таблиця 2.2 – Нормалізовані вихідні дані

№ пр.	Zx	Z_Y
1	2	3
1	0,0000	1,6128
2	0,0000	1,6021
3	0,3010	1,8062
4	0,3010	2,0414
5	0,3010	2,2014

Продовження табл. 2.2

1	2	3
6	0,3010	2,1903
7	0,3010	2,1553
8	0,3010	2,1614
9	0,3010	2,1931
10	0,3010	2,2041
11	0,3010	2,1903
12	0,3010	1,9590
13	0,4771	2,2405
14	0,4771	2,0000
15	0,4771	1,9638
16	0,4771	1,9542
17	0,4771	2,3263
18	0,4771	2,2355
19	0,4771	2,3747
20	0,4771	2,2304
21	0,4771	2,0000
22	0,4771	2,2553
23	0,4771	2,3304
24	0,4771	2,2989
25	0,4771	2,3075
26	0,6021	2,4048
27	0,6021	2,5159
28	0,6021	2,4771
29	0,6021	2,5119
30	0,6021	2,6721
31	0,6021	2,5185
32	0,6021	2,5079
33	0,6021	2,5263
34	0,6990	2,7076
35	0,6990	2,6990
36	0,7782	2,7007
37	0,8451	2,8998
38	0,9542	2,9912

Розподіл емпіричних даних та розподіл нормалізованих даних для вибірки X представлено на рис. 2.1.а та рис. 2.1.б відповідно.

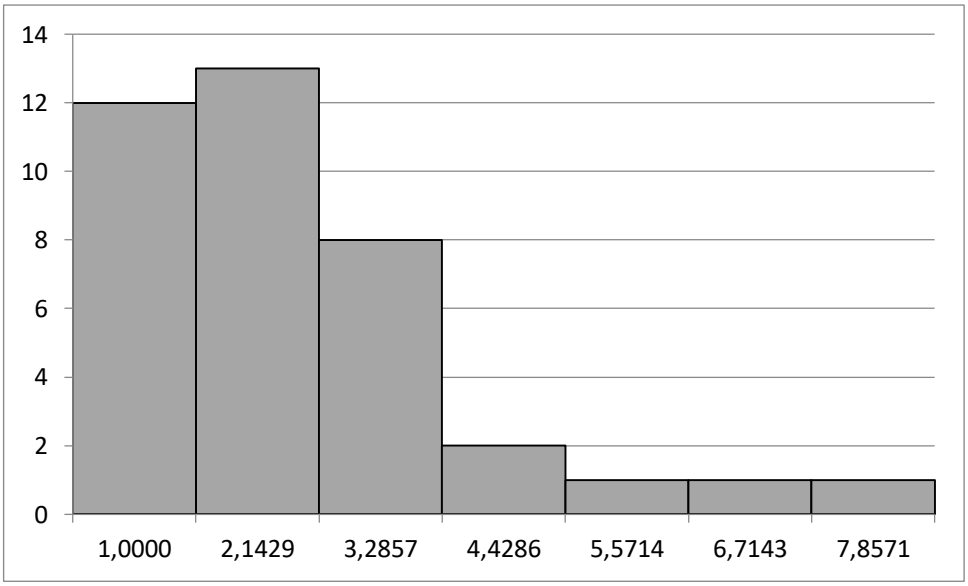

Рисунок 2.1.а – Розподіл емпіричних даних для вибірки X

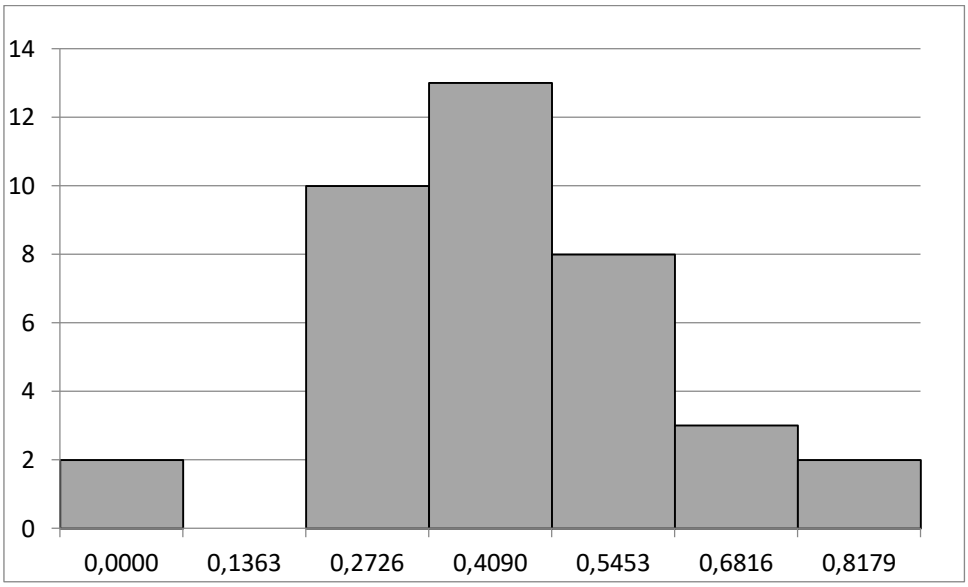

Рисунок 2.1.б – Розподіл нормалізованих даних для вибірки X

Розподіл емпіричних даних та розподіл нормалізованих даних для вибірки Y представлено на рис. 2.2.а та рис. 2.2.б. відповідно.

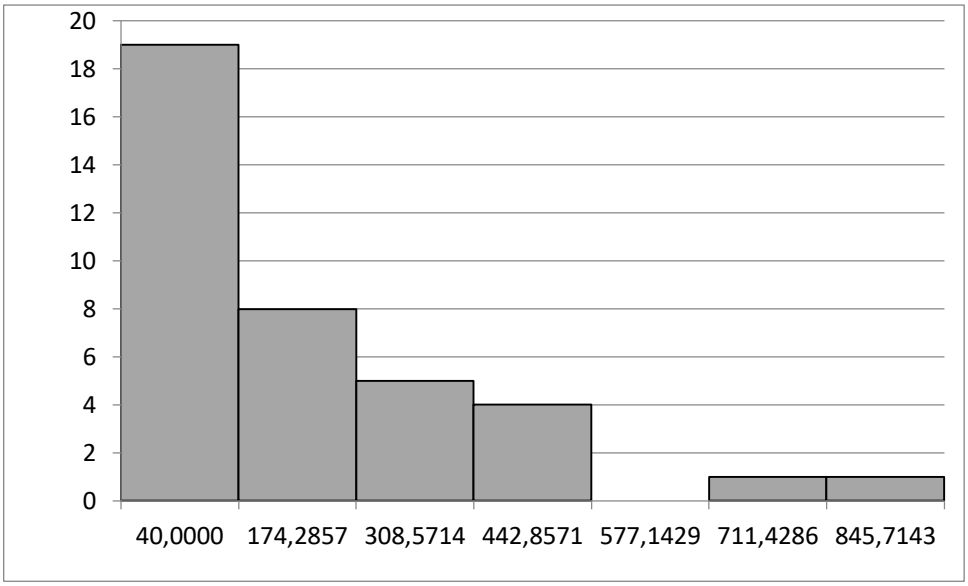

Рисунок 2.2.а – Розподіл емпіричних даних для вибірки Y

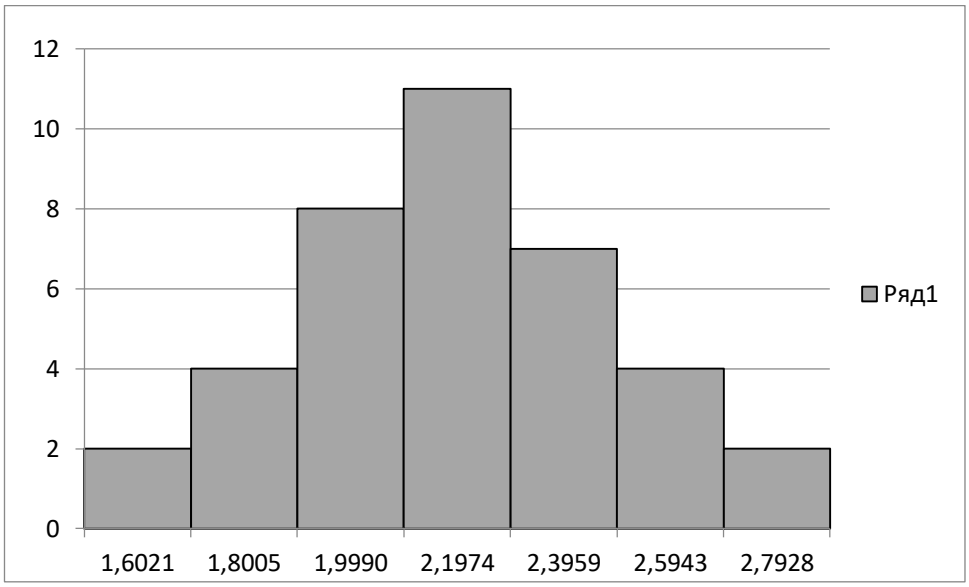

Рисунок 2.2.б – Розподіл нормалізованих даних для вибірки Y

– Виконати специфікацію моделі, тобто обрати вид функціональної залежності

Для побудови нелінійної регресійної моделі використовуємо метод нормалізуючих перетворень.

В якості функціональної залежності обрано однофакторну лінійну регресійну модель для очищених нормалізованих даних та нелінійну

однофакторну регресійну модель для вихідних даних, яку будемо отримувати через зворотне перетворення.

Загальний вигляд лінійної регресійної моделі з двома параметрами представлений наступною формулою [18]:

$$\hat{Z}_y = \mathrm{b}_0 + \mathrm{b}_1 \cdot Z_x + \varepsilon\,, \tag{2.4}$$

де $\mathrm{b}_0, \mathrm{b}_1$ – коефіцієнти лінійної регресії;

ε – випадкова помилка.

Загальний вигляд нелінійної регресійної моделі представлений формулою:

$$\hat{Y} = 10^{b_0+\varepsilon} \cdot X^{b_1}\quad. \tag{2.5}$$

– Виконати оцінювання параметрів моделі.

Коефіцієнти лінійної регресії $\mathrm{b}_0, \mathrm{b}_1$ визначили методом найменших квадратів [18]. Отримали: $\mathrm{b}_0 = 1{,}6116;\ \ \mathrm{b}_1 = 1{,}4288.$

Тоді, згідно (2.4), лінійна регресійна модель для нормалізованих даних має вигляд:

$$\hat{Z}_y = 1{,}6116 + 1{,}4288 \cdot Z_x + \varepsilon\,, \tag{2.6}$$

а нелінійна регресійна модель для оцінювання розміру телеграм-ботів на Python згідно (2.5) наступна:

$$\hat{Y} = 10^{1{,}6116+\varepsilon} \cdot X^{1{,}4288}\,. \tag{2.7}$$

– Виконати оцінювання рівня довіри до отриманої моделі

Довірчий інтервал – це границі прогнозу (верхня і нижня), в рамки яких із заданою ймовірністю потраплять фактичні значення.

Ліва та права границі довірчого інтервалу для лінійної регресії обчислюються за наступною формулою [18]:

$$Z_y = \hat{Z}_y \pm t_{\alpha/2,\vartheta}\, S_{Z_y} \sqrt{\frac{1}{n} + \frac{(x-\overline{x})^2}{\sum_n^1 (x_i - \overline{x})^2}}, \quad (2.8)$$

де $S_{Z_y} = \sqrt{\frac{1}{n-2} \sum_1^n (y_i - \hat{y}_i)^2}$;

α – рівень статистичної значущості;

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

n – кількість елементів.

Під час дослідження згідно (2.8) побудовано довірчі інтервали для лінійної та, через зворотне перетворення, для нелінійної моделей.

– Виконати оцінювання прогнозних можливостей моделі

Інтервал прогнозування – це кількісна оцінка невизначеності в прогнозі. Інтервал прогнозування зазвичай більше довірчого інтервалу, оскільки він повинен враховувати довірчий інтервал і прогнозовану дисперсію вихідної змінної.

Співвідношення для обчислення лівої та правої границь інтервалу прогнозування для лінійної регресії обчислюються має наступний вигляд [18]:

$$Z_y = \hat{Z}_y \pm t_{\alpha/2,\vartheta}\, S_{Z_y} \sqrt{1 + \frac{1}{n} + \frac{(x-\overline{x})^2}{\sum_n^1 (x_i - \overline{x})^2}}, \quad (2.9)$$

де $S_{Z_y} = \sqrt{\frac{1}{n-2} \sum_1^n (y_i - \hat{y}_i)^2}$;

α – рівень статистичної значущості;

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

n – кількість елементів.

В роботі згідно (2.9) обчислено ліву та праву границі інтервалу прогнозування для лінійної та, через зворотне перетворення, для нелінійної моделей.

В таблиці 2.3 знаходяться довірчі інтервали та інтервали прогнозування для нелінійної регресійної моделі, яка отримана з використанням нормалізуючого перетворення у вигляді десяткового логарифму.

Таблиця 2.3 – Довірчі інтервали та інтервали прогнозування для нелінійної регресійної моделі

№ п / п	Нижня границя довірчого інтервалу	Верхня границя довірчого інтервалу	Довжина довірчого інтервалу	Нижня границя інтервалу прогнозування	Верхня границя інтервалу прогнозування	Довжина інтервалу прогнозування
1	2	3	4	5	6	7
1	30,9795	53,9701	22,9906	20,1573	82,9461	62,7888
2	30,9795	53,9701	22,9906	20,1573	82,9461	62,7888
3	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
4	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
5	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
6	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
7	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
8	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
9	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
10	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
11	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
12	95,6053	126,7696	31,1643	56,5768	214,2194	157,6426
13	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
14	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
15	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
16	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
17	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
18	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
19	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
20	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
21	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
22	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
23	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
24	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
25	176,8154	218,3749	41,5595	101,6536	379,8394	278,1858
26	261,2203	336,3258	75,1055	152,7776	575,0523	422,2747
27	261,2203	336,3258	75,1055	152,7776	575,0523	422,2747
28	261,2203	336,3258	75,1055	152,7776	575,0523	422,2747

Продовження табл. 2.3

1	2	3	4	5	6	7
29	261,2203	336,3258	75,1055	152,7776	575,0523	422,2747
30	261,2203	336,3258	75,1055	152,7776	575,0523	422,2747
31	261,2203	336,3258	75,1055	152,7776	575,0523	422,2747
32	261,2203	336,3258	75,1055	152,7776	575,0523	422,2747
33	261,2203	336,3258	75,1055	152,7776	575,0523	422,2747
34	346,9779	479,0825	132,1046	208,5724	796,9942	588,4218
35	346,9779	479,0825	132,1046	208,5724	796,9942	588,4218
36	434,9846	643,4558	208,4712	268,1866	1043,6515	775,4649
37	525,4110	827,5818	302,1708	331,0415	1313,4927	982,4512
38	713,0615	1250,5216	537,4601	464,8983	1918,0513	1453,1530

– Побудувати лінійну регресійну модель для вихідних даних у припущенні про лінійність зв’язків та нормальність закону розподілу.

Згідно (2.6) та методу найменших квадратів лінійна регресійна модель для вихідних даних має вигляд:

$$Y = 143{,}513 + 119{,}7314\mathrm{X} + \varepsilon.$$

– Виконати оцінювання рівня довіри та прогнозних можливостей лінійної регресійної моделі для вихідних даних у припущенні про лінійність зв’язків та нормальність закону розподілу.

Довірчі інтервали та інтервали прогнозування для вихідних даних у припущенні про лінійність зв’язків та нормальність закону розподілу обчислені згідно (2.8), (2.9) та представлені в таблиці 2.4.

Таблиця 2.4 – Довірчі інтервали та інтервали прогнозування для вихідних даних у припущенні про лінійність зв’язків та нормальність закону розподілу

№ п / п	Нижня границя довірчого інтервалу	Верхня границя довірчого інтервалу	Довжина довірчого інтервалу	Нижня границя інтервалу прогнозув.	Верхня границя інтервалу прогнозув.	Довжина інтервалу прогнозування
1	-61,0893	13,5295	74,6188	-158,2476	110,6878	268,9354
2	-61,0893	13,5295	74,6188	-158,2476	110,6878	268,9354
3	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
4	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
5	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
6	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
7	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
8	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
9	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
10	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
11	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
12	68,7222	123,1810	54,4588	-36,0750	227,9781	264,0531
13	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
14	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
15	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
16	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
17	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
18	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
19	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
20	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
21	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
22	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
23	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
24	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
25	194,3656	237,0004	42,6348	84,7479	346,6181	261,8702
26	312,3717	358,4572	46,0854	204,1874	466,6415	262,4541
27	312,3717	358,4572	46,0854	204,1874	466,6415	262,4541
28	312,3717	358,4572	46,0854	204,1874	466,6415	262,4541
29	312,3717	358,4572	46,0854	204,1874	466,6415	262,4541
30	312,3717	358,4572	46,0854	204,1874	466,6415	262,4541
31	312,3717	358,4572	46,0854	204,1874	466,6415	262,4541
32	312,3717	358,4572	46,0854	204,1874	466,6415	262,4541
33	312,3717	358,4572	46,0854	204,1874	466,6415	262,4541
34	423,9843	486,3075	62,3232	322,2526	588,0391	265,7865
35	423,9843	486,3075	62,3232	322,2526	588,0391	265,7865
36	532,7505	617,0042	84,2537	438,9941	710,7605	271,7664
37	640,3700	748,8475	108,4775	554,4966	834,7209	280,2243
38	854,2793	1013,8640	159,5846	782,2284	1085,9149	303,6866

Виконавши порівняння довжин довірчих інтервалів для нелінійної (табл. 2.3) та лінійної (табл. 2.4) регресійних моделей оцінювання розміру телеграм-ботів на Python, можна зробити висновок про те, що довжини більшості довірчих інтервалів для нелінійної моделі менші, ніж для лінійної. Також для 2-ох довірчих інтервалів лінійної моделі нижні границі – від'ємні числа.

Порівнюючи довжини інтервалів передбачення для нелінійної (табл. 2.3) та лінійної (табл. 2.4) регресійних моделей оцінювання розміру телеграм-ботів на Python бачимо, що довжини довірчих більшості інтервалів передбачення для нелінійної моделі менші, ніж для лінійної. Також для 12-и інтервалів передбачення лінійної моделі нижні границі – від'ємні числа.

– Порівняти нелінійну та лінійну моделі по R^2, $MMRE$ та $PRED(l)$.

По співвідношенням, які наведено в (1.1), (1.3) та (1.4) обчислили R^2 $MMRE$ $PRED(l)$ для нелінійної та лінійної моделей:

– нелінійна модель: R^2= 0,9062; $MMRE$ =0,2562; $PRED(0,25)$ =0,7105;

– лінійна модель: R^2= 0,8994; $MMRE$ =0,3710; $PRED(0,25)$ =0,5263.

Таким чином, нелінійна модель має більше значення коефіцієнта детермінації, менше значення середньої величини відносної похибки та більший рівень прогнозування. Це свідчить про кращу якість нелінійної моделі.

2.3 Постановка задачі на розробку програмного забезпечення для оцінювання розміру телеграм-ботів на Python

Виконавши аналіз існуючих методів та моделей для прогнозування розміру програмного забезпечення та побудувавши нелінійну регресійну модель для оцінювання розміру телеграм-ботів на Python, сформулюємо

постановку задачі: розробити проект програмного забезпечення для оцінювання розміру телеграм-ботів на Python.

Дане програмне забезпечення має реалізовувати наступні функції:

1) введення інформації про Telegram бот на Python (кількість строк, кількість класів);

2) нормалізація введеної інформації десятковим логарифмом;

3) прогнозування розміру Telegram боту на Python;

4) визначення границь інтервалу довіри;

5) визначення границь інтервалу передбачення;

6) корегування даних.

Більш детально постановка задачі викладена в технічному завданні, яке наведені у додатку А.

3 ПРОЕКТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ТЕЛЕГРАМ-БОТІВ, ЩО СТВОРЮЮТЬСЯ НА PYTHON

Проаналізувавши предметну галузь і взявши до уваги вимоги, що ставляться до програмного забезпечення, перейдемо до проектування і реалізації поставлених завдань.

У ході розробки проекту програмного забезпечення було виконане ескізне, технічне та робоче проектування. Під час розрбки проекту програмного забезпечення використовується об’єктно-орієнтована методологія

3.1 Ескізний проект програмного забезпечення для оцінювання розміру телеграм-ботів на Python

Ескізний проект містить результати аналізу вимог до програми. У його рамках була визначена загальна схема функціонування системи, структури вхідних і вихідних даних. Також були визначені стани програми і переходи між ними, виконувані програмою по командах користувача в процесі роботи, та розроблений інтерфейс.

3.1.1 Розробка контекстної діаграми ПЗ для оцінювання розміру телеграм-ботів на Python

Для детального визначення зовнішніх характеристик програмного забезпечення необхідно провести його зовнішнє проектування. Зовнішнє проектування складається з розробки структури вхідних та вихідних потоків даних та інтерфейсів їх вводу і виводу.

Важливу роль в вирішенні цієї задачі відіграє контекстна діаграма, що моделює систему найбільш загальним чином. Контекстна діаграма задає межі

даної моделі, визначаючи її оточення (зовнішні входи та виходи) й основні процеси, що розглядаються. Дуги у діаграмі відповідають потокам даних; вони помічені назвами відповідних даних. У центрі діаграми знаходиться головний процес [19].

Тепер перейдемо до програми, що проектуємо для оцінювання розміру телеграм-ботів на Python.

Контекстна діаграма (DFD нульового рівня) програмного забезпечення для оцінювання розміру телеграм-ботів на Python представлена на рис.3.1.

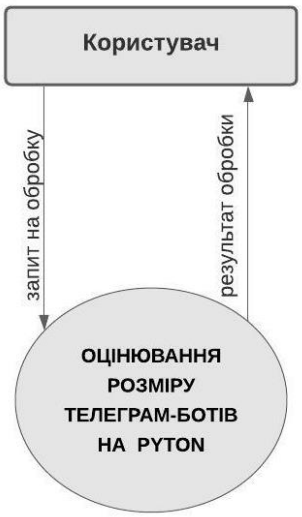

Рисунок 3.1 – DFD нульового рівня ПЗ для оцінювання розміру телеграм-ботів на Python

«Користувач» являється зовнішньою сутністю по відношенню до системи. Вхідним потоком у систему є запит на обробку телеграм-ботів на Python, вихідним потоком із системи є результат обробки телеграм-ботів на Python.

Словник даних побудованої DFD нульового рівня представлений у таблиці 3.1.

Таблиця 3.1 – Словник даних побудованої DFD нульового рівня

Назва потоку	Складові потоку
запит на обробку	– запит на введення: назва проекту, кількість класів – запит на корегування: назва проекту, кількість класів – запит на прогнозування(запит на довірчий інтервал, запит на інтервал прогнозування, запит на оцінювання);
результат обробки	– результат введення (підтвердження) – результат корегування (підтвердження) – результат прогнозування (ліва та права межі інтервалу довіри, ліва та права межі прогнозного інтервалу, результат оцінювання)

Як бачимо, DFD нульового рівня показує загальну картину і представляє систему як єдиний процес «Оцінювання розміру телеграм-ботів на Python», який має зв'язки із зовнішньою сутністю «Користувач». DFD нульового рівня зрозумілі широкій аудиторії, включаючи бізнес-аналітиків, розробників та замовників.

3.1.2 Концептуальна модель даних ПЗ для оцінювання розміру телеграм-ботів на Python

Концептуальна модель даних – це загальна інформаційна модель предметної галузі, що охоплює питання класифікації, структуризації та семантичної цілісності (достовірності та узгодженості даних).

Концептуальна модель даних розробляється незалежно від моделей даних, незалежно від СКБД.

Перш ніж почати проектувати БД, необхідно розібратися, як здійснюється функціонування предметної галузі, для відображення якої створюється БД, тобто створюється її попередній опис. Для опису

концептуальної моделі може бути використана природна мова, але такий опис буде громіздким і неоднозначним, тому для опису концептуальної моделі найчастіше використовується формалізована мова.

Розглянемо основні етапи розробки концептуальної моделі даних:

– визначення сутностей предметної галузі;

– визначення атрибутів сутностей;.

– визначення зв’язків між сутностями та їх типів.

На першому етапі в рамках побудови концептуальної моделі даних були визначені наступні сутності:

– програмні проекти (телеграм-боти на Python);

– дані для прогнозування;

– результати прогнозування.

На другому етапі побудови концептуальної моделі були визначені атрибути сутностей:

– програмні проекти: назва телеграм-боту на Python;

– дані для прогнозування: кількість класів телеграм-боту на Python;

– результати прогнозування: прогнозована кількість строк коду на Python, ліва межа інтервалу довіри, права межа інтервалу довіри, ліва межа прогнозного інтервалу, права межа прогнозного інтервалу.

На третьому етапі побудови концептуальної моделі були визначені зв’язки між сутностями та їх типи (табл.3.2).

Таблиця 3.2 – Зв’язки між сутностями та їх типи

Зв’язки між сутностями	Опис зв’язків між сутностями	Типи зв’язків
1	2	3
Програмні проекти – дані для прогнозування	Одному програмному проекту відповідають одні дані для прогнозування, і одним даним для прогнозування відповідає один програмний проект.	1:1

Продовження табл.. 3.2

1	2	3
Програмні проекти – результати прогнозування	Одному програмному проекту відповідають одні дані результати прогнозування, і одним результатам прогнозування відповідає один програмний проект.	1:1

Використовуючи визначені сутності, їх атрибути, а також виявлені зв'язки побудовано концептуальну модель даних ПЗ для оцінювання розміру телеграм ботів на Python (рис. 3.2).

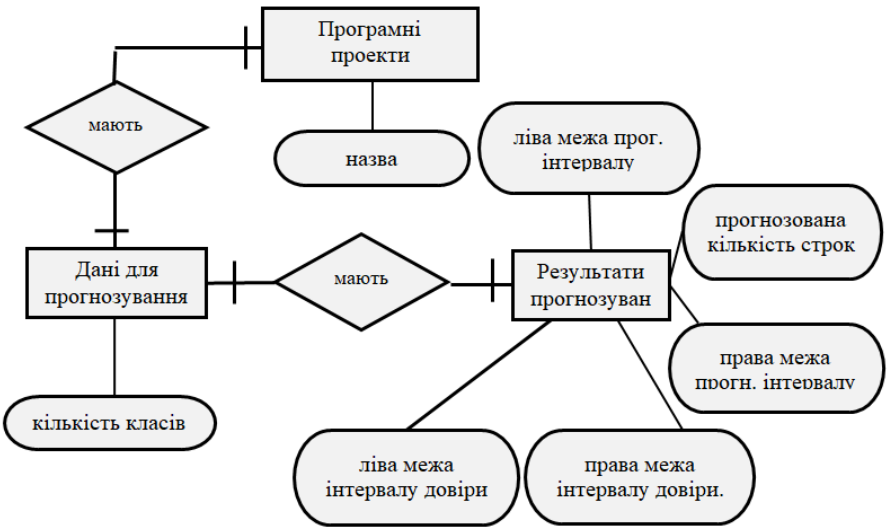

Рисунок 3.2 – Концептуальна модель даних ПЗ для оцінювання розміру телеграм-ботів на Python

3.1.3 Діаграма переходів станів ПЗ для оцінювання розміру телеграм-ботів на Python

Діаграма переходів станів є графічною формою представлення кінцевого автомата – математичної абстракції, що використовується для моделювання детермінованої поведінки технічних об'єктів чи реального світу. Іншими словами, діаграма переходів станів відображає специфікацію управління, тобто моделювання та документування аспектів системи, що залежать від часу чи реакції на подію. Специфікації управління дають можливість декомпозувати процеси і описати співвідношення вхідних і вихідних управляючих потоків на процесі.

Для побудови STD необхідно, відповідно до теорії кінцевих автоматів, визначити: основні стани, керуючі впливи (або умови переходу), дії, що виконуються, і можливі варіанти переходів з одного стану в інший [19]. На рисунку 3.3 представлена STD програмного забезпечення прогнозування розміру телеграм-ботів на Python.

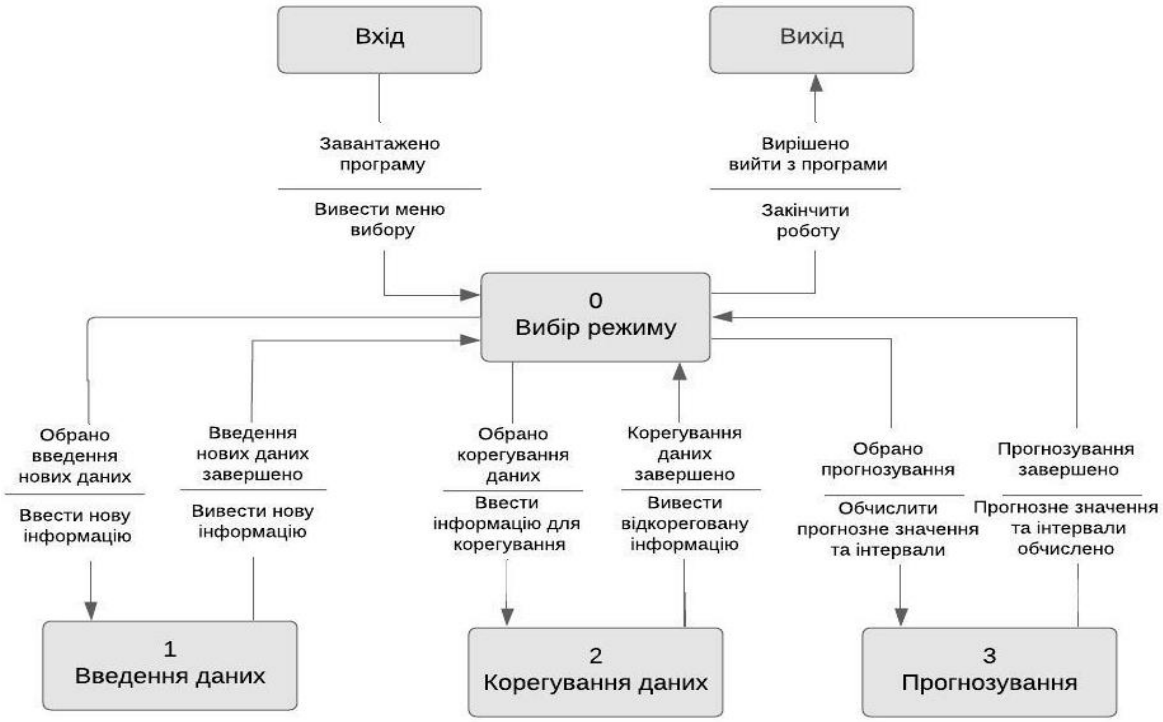

Рисунок 3.3 – STD програмного забезпечення прогнозування розміру телеграм-ботів на Python

На етапі аналізу вимог і визначення специфікацій STD відображає реакцію розроблюваного програмного забезпечення на отримані управляючі впливи, що приходять ззовні. Такими впливами можуть бути як команди користувачів, так і сигнал датчика. Реакцією на такі впливи є виконання певних операцій та/або перехід в інший стан.

3.1.4 Розробка графічного інтерфейсу ПЗ для оцінювання розміру телеграм-ботів на Python

Під графічним інтерфейсом користувача (Graphical User Interface - GUI) мається на увазі тип екранного уявлення, у якому користувач може вибирати команди, запускати завдання та переглядати списки файлів, вказуючи на піктограми чи пункти у списках меню, показаних на екрані. Дії можуть, як правило, виконуватися за допомогою миші або натисканням клавіш на клавіатурі [19].

У розробці програмного забезпечення однією з найважливіших частин виробводного процесу є проектування. Саме проектування ПЗ як процес створення проекту програмного забезпечення можна розбити (дуже умовно) на 2 великі частини: проектування функціоналу та проектування інтерфейсу. Для проектування функціоналу використовуються такі засоби, як UML та IDEF0, які вже стали промисленими стандартами розробки ПЗ. У проектуванні графічного інтерфейсу користувача немає усталених стандартів, є окремі рекомендації, прийоми, закони дизайну, традиції, умови експлуатації ПЗ тощо.

Графічний інтерфейс користувача (Graphic user interface) полягає у використанні представлення інформації за допомогою графічних елементів: вікон, піктограм (значків), меню тощо.

Розглянемо проекти екранних форм для програмного забезпечення оцінювання розміру телеграм-ботів на Python. На рисунку 3.4 представлений

графічний інтерфейс головного вікна ПЗ для оцінювання розміру телеграм-ботів на Python.

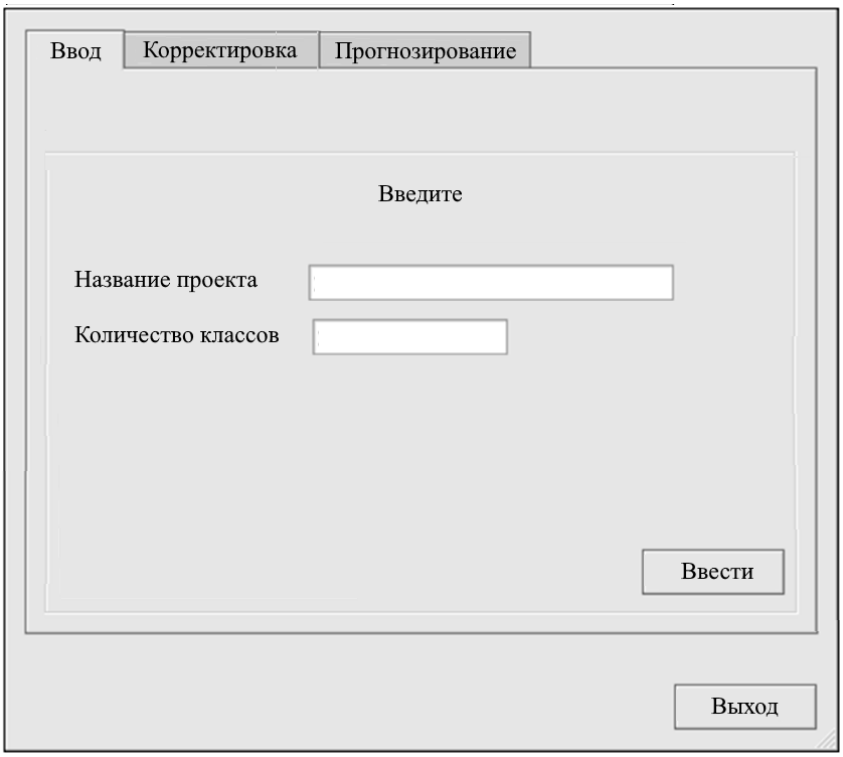

Рисунок 3.4 – Графічний інтерфейс головного вікна ПЗ для оцінювання розміру телеграм-ботів на Python

Як видно з рисунку 3.4, в програмному забезпеченні реалізовані функції введення даних для прогнозування, корегування введених даних та виконання прогнозування, яке включає в себе розрахунок прогнозного значення розміру програмного забезпечення, розрахунок довірчого інтервалу та інтервалу прогнозування. При натисненні на клавішу «Ввести» інформація зберігається. При натисненні на клавішу «Выход» програма завершуэ роботу.

3.2 Технічний проект програмного забезпечення для оцінювання розміру телеграм-ботів на Python

Технічний проект містить набір процесів та структур даних, які будуть використовуватись для реалізації функцій; логічну модель даних, яка відповідає обраній реляційній моделі та відображає структури подання даних в програмному забезпеченні для оцінювання розміру телеграм-ботів на Python.

3.2.1 Декомпозиція контекстного процесу ПЗ для оцінювання розміру телеграм-ботів на Python

Побудова DFD-моделі виходить з принципу декомпозиції. DFD-модель включає три документи, які посилаються один на одного:

– графічні діаграми;

– міні специфікації;

– словник даних.

Контекстний процес на DFD нульового рівня підлягає декомпозиції. Отримані підпроцеси також необхідно деталізувати чи окремою діаграмою FD чи окремої діаграми або міні специфікацією.

При декомпозиції мають виконуватись такі правила:

– правило балансування – при деталізації процесу дочірня діаграма у якості зовнішніх джерел/приймачів даних може мати ті компоненти (підсистеми, процеси, зовнішні сутності, накопичувачі даних), з якими має інформаційний зв'язок відповідний процес на батьківській діаграмі;

– правило нумерації – при деталізації процесів має підтримуватися їхня ієрархічна нумерація;

– правило семи – для того, щоб діаграма легко читалася, кількість функцій на діаграмі не має бути більшою за сім.

Процеси деталізуються до рівня специфікацій програмних модулів.

Контекстний процес «Оцінити розмір телеграм-ботів на Python» декомпозуємо на три підпроцеси: «1 Введення даних для оцінювання», «2.Корегування даних для оцінювання», «3 Прогнозування». Потоки даних також підлягають декомпозиції та уточненню. DFD першого рівня зображена на рисунку 3.5.

Таблиця відповідності потоків даних на DFD нульового та першого рівнів наводиться в таблиці 3.3.

Таблиця 3.3 – Таблиця відповідності потоків даних на DFD нульового та першого рівнів

Назви потоків на DFD нульового рівня	Назви потоків на DFD першого рівня
запит на обробку	запит на введення даних
	запит на корегування даних
	запит на прогнозування
результат обробки	результат введення даних
	результат корегування даних
	результат прогнозування

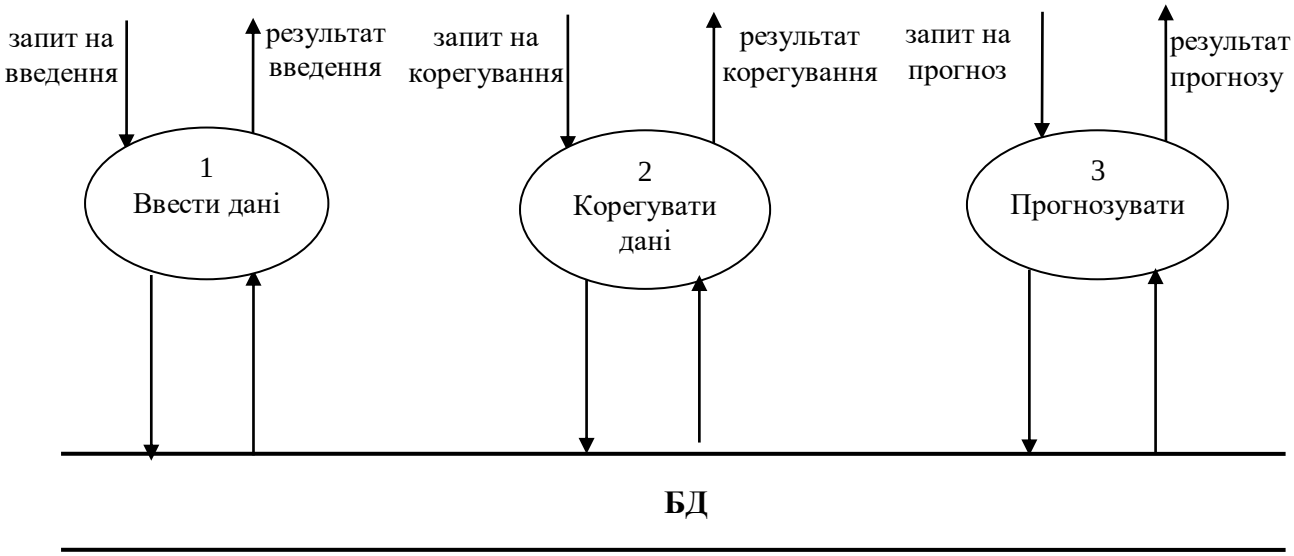

Рисунок 3.5 – DFD першого рівня ПЗ оцінювання розміру телеграм-ботів на Python

Процес 3 «Прогнозувати» на DFD першого рівня необхідно деталізувати, так як він являється складним.

3.2.2 Декомпозиція процесів до DFD другого рівня для ПЗ оцінювання розміру оцінювання телеграм-ботів на Python

Процес 3 «Прогнозувати», який представлений на DFD першого рівня, декомпозуємо до DFD другого рівня підпроцесами: 3.1 «Оцінювання рівня довіри», 3.2 «Оцінювання прогнозних можливостей», 3.3 «Оцінити розмір» (рис. 3.6):

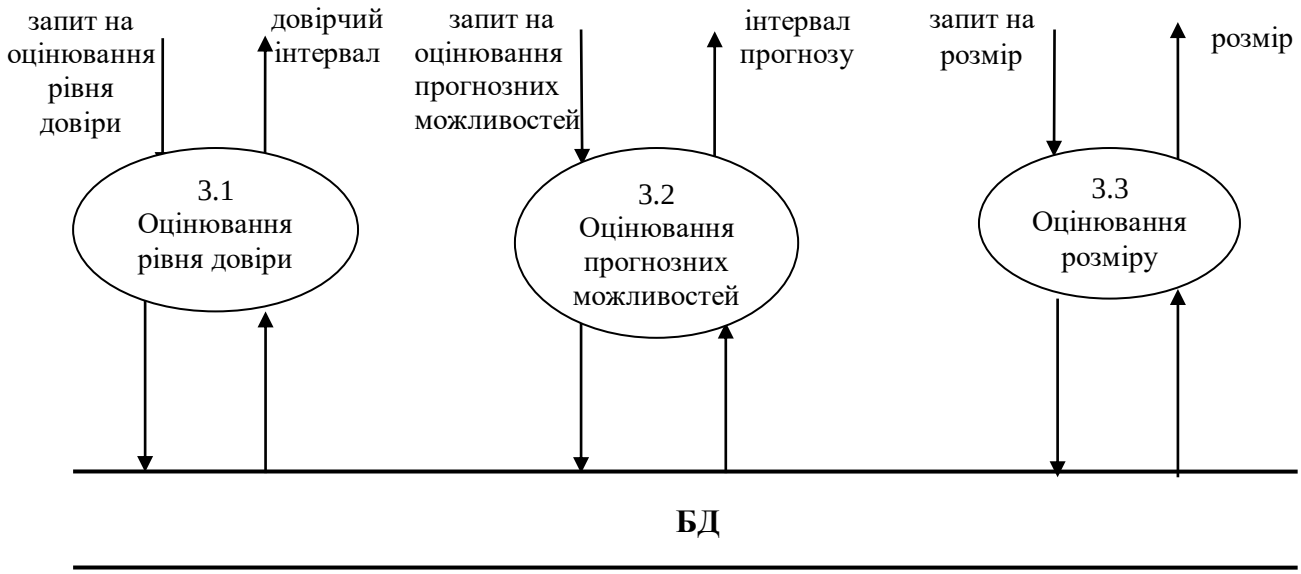

Рисунок 3.6 – Деталізація процесу 3 «Прогнозувати» ПЗ оцінювання розміру телеграм-ботів на Python

Деталізація процесів на DFD доведена рівня, який не потребує подальшої декомпозиції. Для листків отриманої ієрархії необхідно розробити мініспецифікації.

Таблиця відповідності потоків на DFD першого та другого рівнів наведена в таблиці 3.4.

Таблиця 3.4 – Таблиця відповідності потоків на DFD першого та другого рівнів

Назви потоків на DFD першого рівня	Назви потоків на DFD другого рівня
запит на прогнозування	запит на оцінювання рівня довіри
	запит на оцінювання прогнозних можливостей
	запит на розмір
Результат прогнозування	довірчий інтервал
	інтервал прогнозу
	розмір

.

Побудуємо міні-специфікації програмних модулів ПЗ оцінювання розміру телеграм-ботів на Python

Мініспеціфікація – документ, що детально описує логіку процесу. Вона містить номер процесу, списки вхідних та вихідних даних, тіло процесу – докладний алгоритм функції, що перетворює вхідні потоки даних у вихідні.

Мініспеціфікація є кінцевою вершиною ієрархії моделі DFD. Рішення про завершення деталізації процесу та використання мініспеціфікації приймається аналітиком виходячи з наступних критеріїв:

– у процесу невелика кількість вхідних та вихідних потоків даних (2-3 потоки);

– процес можна описати як послідовний алгоритм;

– процес виконує єдину логічну функцію перетворення вхідної інформації на вихідну;

– описати логіку процесу можна як мініспеціфікації невеликого обсягу (трохи більше 20-30 рядків).

Розглянемо мініспецифікації ПЗ оцінювання розміру.

Мініспецифікація підпроцесу № 1 «Ввести дані»

Опис: Внесення інформації про телеграм-бот на Python до бази даних для подальшого оцінювання розміру.

Вхідні дані: Назва проекту, кількість класів.

Вихідні дані: Результат вводу.

Умови виконання: Внесення інформації правильного формату й натискання кнопки «Ввести»

Процедура обробки: В результаті внесення у форму даних, які мають правильний формат, в таблиці «Програмні проекти» та «Дані для прогнозування» заноситься введена інформація.

Мініспецифікація процесу №2 «Корегувати дані»

Опис: Корегування введених даних.

Вхідні дані: Нові дані.

Вихідні дані: Результати корегування.

Умови виконання: Внесення інформації правильного формату й натискання кнопки «Ввести».

Процедура обробки: Вивести існуючі дані з таблиць БД в форму. Виконати корегування даних. Натиснути на кнопку «Ввести». В таблиці «Програмні проекти» та «Дані для прогнозування» заноситься оновлена інформація.

Мініспецифікація підпроцесу № 3.1 «Оцінювання рівня довіри»

Опис: Визначення лівої та правої меж інтервалу довіри.

Вхідні дані: Запит на оцінювання рівня довіри до моделі

Вихідні дані: Ліва та права межі інтервалу довіри.

Умови виконання: Коректність інформації про кількість класів в телеграм-боті на Python.

Процедура обробки: Розраховуються ліва та права межі довірчого інтервалу. Отримана інформація заноситься до таблиці «Результат» бази даних.

Мініспецифікація підпроцесу № 3.2 «Оцінювання прогнозних можливостей»

Опис: Визначення лівої та правої меж інтервалу прогнозу.

Вхідні дані: Запит на оцінювання прогнозних можливостей моделі.

Вихідні дані: Ліва та права межі інтервалу прогнозу.

Умови виконання: Коректність інформації про кількість класів в телеграм-боті на Python.

Процедура обробки: Розраховуються ліва та права межі інтервалу прогнозу. Ортимана інформація заноситься до таблиці «Результат» бази даних.

Мініспецифікація підпроцесу № 3.3 «Оцінювання розміру»

Опис: Оцінювання розміру телеграм-боту на Python.

Вхідні дані: Кількість класів в телеграм-боті на Python.

Вихідні дані: Прогнозна кількість строк коду.

Умови виконання: Коректність інформації про кількість класів в телеграм-боті на Python.

Процедура обробки: Оцінюється розмір телеграм-боту на Python згідно побудованої нелінійної регресійної моделі.

3.2.3 Логічна модель даних ПЗ оцінювання розміру телеграм-ботів на Python

Мета логічного проектування – розвинути концептуальне уявлення БД з урахуванням прийнятої моделі БД (ієрархічної, мережевої, реляційної і т. д.

При побудові логічної моделі реляційної бази даних особливе значення для вирішення завдання формування відношень бази даних має поняття функціональної залежності (ФЗ).

Встановлення ФЗ і отримання найкращої, з точки зору мінімальності представлення, множини ФЗ дозволять побудувати найбільш оптимальний варіант бази даних, що забезпечує надійність зберігання і обробки даних на основі методів еквівалентних перетворень схем відношень реляційної бази даних.

Процес вирішення такого завдання називається нормалізацією відношень інформаційної моделі предметної області [19].

Приймемо в якості моделі реляційну БД в третій нормальній формі (набір нормалізованих відношень з кратністю зв'язків 1: N, 1: 1).

Для побудови такої моделі необхідно:

– видалити зв'язки N: M;

– перевірити модель за допомогою правил нормалізації (база повинна бути приведена до третьої нормальної форми);

– визначити типи атрибутів (числовий, текстовий, дата / час, логічний);

– визначити ключі (первинні (PK), зовнішні (FK), альтернативні (AK));

– визначити обов'язковість даних (NULL, NOT NULL);

– при розбіжності імен сутностей, атрибутів логічної і концептуальної моделей побудувати таблицю відповідності.

Логічна модель даних ПЗ оцінювання розміру телеграм-ботів на Python представлена на рисунку 3.7.

Рисунок 3.7 – Логічна модель даних ПЗ оцінювання розміру телеграм-ботів на Python

Логічна модель враховує модель даних програмного забезпечення.

3.3 Робочий проект програмного забезпечення для оцінювання розміру телеграм-ботів на Python

Стадія розробки програмного забезпечення «Робочий проект» складається з наступних етапів: вибір мови програмування, розробка та налагодження програми, розробка програмної документації, тестування та випробування програмного забезпечення [19].

3.3.1 Вибір мови програмування для оцінювання розміру телеграм-ботів на Python

В якості мови програмування була обрана Java. Java – це мова програмування загального призначення, яка слідує парадигмі об'єктно-орієнтованого програмування. Java – це не тільки мова програмування, але й екосистема інструментів, що охоплює те, що може знадобитися під час роботи з мовою. Вона включає:

– JDK – набір розробника Java. З його допомогою, використовуючи блокнот, можна писати та запускати/ компілювати програму.

– JRE – виконуюча система Java. Механізм розповсюдження програмного забезпечення складається з автономної віртуальної машини,

• IDE – інтегроване середовище розробки. Інструменти, які допомагають запускати, редагувати та компілювати програму. Найпопулярніші з них – IntelliJ IDEA, Eclipse та NetBeans.

Розглянемо плюси програмування на Java:

– Об'єктно-орієнтоване програмування.

– Простий синтаксис та плавне навчання.

– Стандарт у корпоративних інформаційних системах.

– Безпека.

– Незалежність від платформи.

– Мова розподіленого програмування та комфортної віддаленої спільної роботи

– Автоматичне управління пам'яттю

– Багатопоточність.

– Стабільність та спільнота

Мінуси програмування на Java:

– Платне комерційне використання.

– Низька продуктивність.

– Багатослівне та складне кодування.

Сфери використовують мови:

– Програми для Android.

– Прикладні програмні продукти.

– Фінансові програми.

– Програми касових терміналів.

– Торговельні системи.

– Програми до роботи з великими даними.

В якості СКБД була обрана MySQL, що являється однією з найбільш популярних та поширених СКБД.

Важливим фактором є її безплатність. MySQL поширюється за умов загальної ліцензії GNU.

Переваги MySQL:

– одна з найбільш швидких баз;

– високопродуктивна та відносно проста;

– безкоштовна для домашнього використання;

– розуміє команди SQL, підтримує інтерфейс ODBC, протокол інтерфейсу з базами даних, розроблений фірмою Microsoft.

– може підключитися велика кількість користувачів одночасно;

– призначена для роботи в мережі, захищена від сторонніх втручань;

– працює і під UNIX і під Windows;

– доступний вихідний код.

Є сенс вказати і деякі недоліки СУБД MySQL. Це, в першу чергу, наявні в інших системах, але відсутні в MySQL можливості використання вкладених вибірок, транзакцій, цілісності посилань, тригерів, процедур і курсорів, що зберігаються. Але, з одного боку, для багатьох програм такі можливості повною мірою і не потрібні, а з іншого – розробка системи продовжується, і деякі з перерахованих недоліків поступово усуваються.

3.3.2 Фізична модель даних програмного забезпечення для оцінювання розміру програмних застосунків

На рисунку 3.8 представлена фізична модель даних ПЗ оцінювання розміру телеграм-ботів на Python

Для атрибутів таблиць вказані назви полів, типи даних, довжини та ключові поля (де необхідно). При позначеннях назв полів та їх типів враховувалась обрана СКБД.

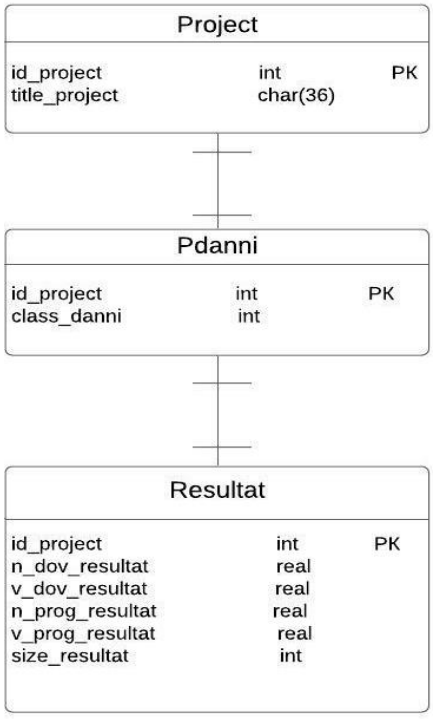

Рисунок 3.8 – Фізична модель даних ПЗ оцінювання розміру телеграм-ботів на Python.

3.3.3 Кодування та тестування ПЗ для оцінювання розміру телеграм-ботів на Python

Код програми поданий у додатку Д.

Розглянемо основні групи методів тестування [19].

Перша група методів – це методи структурного тестування (білої скриньки).

При структурному тестуванні необхідне знання тексту програми. Необхідно побудувати стільки тестів, щоб забезпечити покриття всіх можливих гілок програми. Дані для тестів обираються, базуючись на внутрішньому устрою програми. До методів структурного тестування відносяться методи покриття операторів, покриття умов, комбінованого покриття умов та інші.

Друга група методів – це методи функціонального тестування (чорної скриньки).

Методи функціонального тестування вимагають знання специфікацій програми. Основними методами функціонального тестування є метод розбиття на класи еквівалентності, метод граничних значень та інші.

В методі розбиття на класи еквівалентності виділяють правильні та неправильні класи еквівалентності. Клас еквівалентності – це такий набір даних, при виборі довільного представника якого програма працює по одному сценарію, тобто виконуються одні й ті ж оператори.

Тестування програмного забезпечення виконувалося одним із методів функціонального тестування, а саме розбиття на класи еквівалентності. Розглянемо тестування функції «Корегувати дані».

В таблиці 3.5 наведені правильні та неправильні класи еквівалентності функції «Корегувати дані».

Таблиця 3.5 – Правильні та неправильні класи еквівалентності функції «Корегувати дані».

Вхідні дані	Правильні	Неправильні
Назва проекту	1.Набір символів	2.Дані не введено
Число класів	3. Числова величина >0	4. Числова величина <= 0 5. Не числова величина

Таблиця 3.6 містить тести для правильних класів еквівалентності.

Таблиця 3.6 – Тестування правильних класів функції «Корегувати дані»

№ п/п	№ класу	Введені дані	Результат
1	1	PR 5	Програма записала введені дані в базу. Результат правильний.
	3	15	

Таблиця 3.7 містить тести для неправильних класів еквівалентності.

Таблиця 3.7 – Тестування неправильних класів функції «Корегувати дані»

№ п/п	№ класу	Введені дані	Результат (видача повідомлення)
1	2	Дані не введені	«Введіть назву проекту»
2	4	-20	«Неправильна кількість класів»
3	5	про	«неправильний формат даних»

3.3.4 Випробування ПЗ для оцінювання розміру телеграм-ботів на Python

На етапі випробування якості готове програмне забезпечення зазнає суворої системної перевірки з боку групи осіб, які зазвичай не є розробниками.

Це робиться для гарантії, що готове програмне забезпечення задовольняє всім вимогам користувача та специфікацій, може бути використане в середовищі користувача, вільне від будь-яких дефектів і містить необхідну документацію, яка точно і повно описує програмний продукт.

Фаза випробовування починається, щойно всі компоненти (модулі) налагоджено і зібрано разом у одне ціле, тобто після повного налагодження готового програмного продукту.

Вона закінчується після отримання підтвердження, що програмне забезпечення пройшло всі випробування та готове до експлуатації.

Мета випробувань - визначення ступеня відповідності створеного програмного забезпечення технічному завданню.

Проведемо випробування ПЗ для оцінювання розміру телеграм-ботів на Python згідно Програми та методики випробування, що наведена в Додатку Б.

– Випробування функції «Корегування даних»

Після обрання пункту «Корегування» отримали можливість змінити назву програмного проекту та кількість класів. Занесли нові дані та натиснули на кнопку «Ввести». В таблиці «Програмні проекти» та «Дані для прогнозування» занеслася оновлена інформація. Функція відпрацювала правильно.

– Випробування функції «Прогнозувати розмір»

Обрали пункт «Прогнозування» (рис. 3.9).

. На екран вивелися результати прогнозування для введених даних, а саме: прогнозне значення, інтервали довіри та інтервали передбачення.

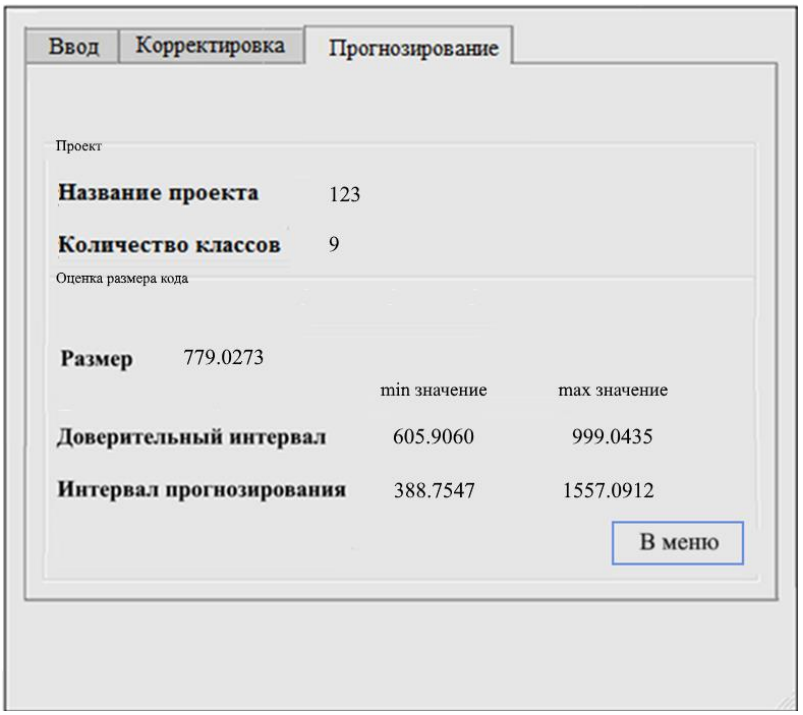

Рисунок 3.9 – Графічний інтерфейс вікна «Прогнозування» ПЗ для оцінювання розміру телеграм-ботів на Python

Після перегляду натиснули кнопку «В меню» та перейшли до вибору дії. Функція відпрацювала правильно.

– Випробування функції «Введення даних»

Після обрання пункту «Введення» отримали можливість ввести назву програмного проекту та кількість класів. Занесли нові дані та натиснули на кнопку «Ввести». В таблиці «Програмні проекти» та «Дані для прогнозування» занеслася нова інформація. Функція відпрацювала правильно.

4 РЕЗУЛЬТАТИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ТЕЛЕГРАМ-БОТІВ НА PYTHON

Мета кваліфікаційної роботи полягала в підвищенні достовірності оцінювання розміру телеграм-ботів на Python.

Для досягнення мети були вирішені наступні задачі:

– Виконано порівняльний аналіз методів та моделей для оцінювання розміру програмного забезпечення.

– Обґрунтувано необхідність удосконалення регресійної моделі для оцінювання розміру телеграм-ботів на Python.

– Зібрано емпіричні дані.

На ресурсі GitHub було зібрано інформацію по Telegram ботам на Python, а саме: кількість класів та кількість строк коду для 40 проектів.

– Виконано передобробку емпіричних даних.

За допомогою критерію згоди Пірсона χ^2 визначено, що емпіричні дані не являються нормально розподіленими. Тому виконано нормалізацію даних з використанням нормалізуючого перетворенням десятковий логарифм.

З нормалізованих даних вилучено дублікати та викиди. При аналізі даних на викиди використано квадрат відстані Махаланобіса та допоміжну статистику, яку порівняно з критичним значенням Фішера $F_{k,N-k,\alpha}$ (N – кількість елементів у вибірці, k – кількість параметрів, α – рівень значущості). Ті елементи вибірки, для яких статистика перевищувала критичне значення Фішера, являлися викидами. Їх прибирали з вибірки і заново повторювали розрахунок. Було знайдено 2 викиди.

– Виконано специфікацію моделі, тобто обрано вид функціональної залежності.

В якості функціональної залежності обрано лінійну регресійну модель для очищених нормалізованих даних та нелінійну регресійну модель для вихідних даних.

– Виконано оцінювання параметрів моделі.

Побудовано лінійну регресійну модель для очищених нормалізованих даних та через зворотне перетворення отримано нелінійну регресійну модель для вихідних даних. Параметри лінійної регресії було визначено методом найменших квадратів.

– Виконано оцінювання рівня довіри до отриманої моделі.

Побудовано довірчі інтервали для лінійної та, через зворотне перетворення, для нелінійної моделей.

– Виконано оцінювання прогнозних можливостей моделі.

Побудовано інтервали прогнозування для лінійної та нелінійної моделей. Обчислено коефіцієнт детермінації R^2, середню величину відносної похибки MMRE та рівень прогнозування PRED(0,25).

– Виконано порівняння лінійної моделі для вихідних даних у припущенні про нормальність їх розподілу та отриманої нелінійної моделі для вихідних даних по R^2, MMRE та PRED(0,25).

По всім показникам побудована нелінійна модель є кращою, ніж лінійна, та значення R^2, MMRE та PRED(0,25) знаходяться у допустимих межах (відповідно >0,7; <0,3 та >0,7). Таким чином, нелінійна модель більше підходить для прогнозування розміру Telegram ботів на Python.

– Розроблено ПЗ для оцінювання розміру телеграм-ботів на Python.

Таким чином мета роботи була повністю досягнута.

Також розроблено наступну документацію:

– технічне завдання на розробку ПЗ для оцінювання розміру телеграм-ботів на Python (додаток А);

– програма і методика випробувань ПЗ для оцінювання розміру телеграм-ботів на Python (додаток Б);

– опис програми ПЗ для оцінювання розміру телеграм-ботів на Python (додаток В);

– інструкція користувача ПЗ для оцінювання розміру телеграм-ботів на Python (див. додаток Г).

5 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВІД РОЗРОБКИ ТА ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1 Вступ

Дана робота присвячена розробці програми для оцінювання розміру телеграм-ботів на Python.

Ефективність програмних систем визначається ступенем їх позитивного впливу на підприємство, що управляється, або процес в результаті використання засобів обчислювальної техніки, програмного забезпечення, зміни інформаційних потоків та організаційної структури управління.

Створення програмного забезпечення вимагає одноразових витрат на розробку, придбання необхідних технічних засобів і поточних витрат на функціонування продукту. Економія від функціонування ПЗ визначається з урахуванням витрат на його експлуатацію. Відношення цієї економії до витрат на створення програмного забезпечення характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються за діючими на момент розрахунку оптовими цінами, тарифами і ставками заробітної плати.

5.2 Розрахунок витрат на створення й експлуатацію програми для оцінювання розміру телеграм-ботів на Python

Витрати на розробку системи складаються з витрат:

- на заробітну платню розробника;
- на амортизацію ЕОМ, на якій виконується розробка;
- на експлуатацію цієї ЕОМ;
- на засоби розробки;
- на матеріали і комплектуючі.

Розробка програмного забезпечення виконується програмістом, місячна заробітна плата якого складає 10000 грн.

Вартість сучасного ПК становить 25000 грн.

При вартості кіловат-години електроенергії 1.68 грн, розраховується вартість розробки програми.

Витрати на допоміжні матеріали наведені в табл. 5.1.

Таблиця 5.1 – Витрати на допоміжні матеріали

№	Пункти витрат	Сума, грн
1	Флеш-накопичувач даних	320
2	Папір для принтера	90
3	Картридж та тонер для принтера	240
	Разом	650

Вартість ПЗ знаходиться за формулою:

$$C_{пр} = (З_{зп} + З_{сз} + З_{зг} + З_{е}) * Т + З_{м} \quad (5.1)$$

де Т – тривалість розробки, міс.

$З_{зп}$ – основна і додаткова заробітна плата обслуговуючого персоналу, грн.

$З_{сз}$ – відрахування на соціальні заходи (15% від основної і додаткової заробітної плати), грн.

$З_{зг}$ – загальногосподарські витрати (10% від основної заробітної плати), грн.

$З_{е}$ – витрати на електроенергію, грн.

$З_{м}$ – витрати на основні і допоміжні матеріали, грн.

$З_{е}$ при споживанні потужності 0.5 кВт, тривалості роботи на місяць, рівної 22*8 = 176 годин, вартості кіловат-години електроенергії 1.68 грн становить:

$$З_е = 176 * 1.68 * 0.5 = 113.52 \text{ грн}$$

Витрати на розробку ПЗ наведені в табл. 5.2.

Таблиця 5.2 – Витрати на розробку програмного забезпечення

№	Найменування витрат	Одиниця	Кількість
1	Тривалість розробки (Т)	Міс.	3
2	Основна і додаткова заробітна плата ($З_{зп}$)	Грн.	10000.00
3	Відрахування на соціальні заходи ($З_{сз}$)	Грн.	1200.00
4	Загальногосподарські витрати ($З_{зг}$)	Грн.	900.00
5	Витрати на допоміжні матеріали ($З_м$)	Грн.	660.00
6	Витрати на електроенергію ($З_е$)	Грн.	113.52

За формулою 5.1 розрахуємо вартість програми:

$$С_{пр} = (10000 + 1200 + 900 + 113.52) * 3 + 660 = 37300.56 \text{ грн}$$

Амортизаційні відрахування складають 25% балансової вартості на рік:

$$А_{об} = 15000 * 0.25 = 3750 \text{ грн}$$

В масштабах підприємства річні витрати на основні і допоміжні матеріали визначаються в розмірі 5% вартості основного устаткування:

$$В_м = 15000 * 0.05 = 750 \text{ грн}$$

Річний обсяг робіт ПК у годинах визначається в такий спосіб:

$$Ф_м = 264.5 * ТЗ \quad (5.2)$$

Тз – це середнє місячне навантаження устаткування (близько 6 годин), 264.5 – середня кількість робочих днів на рік.

Таким чином, річний обсяг роботи ПК складає:

$$Ф_м = 264.5 * 6 = 1587 \text{ годин}$$

Витрати на електроенергію $З_е$ при 1587 годинах роботи устаткування на рік становлять:

$$З_е = 1587 * 1.29 * 0.5 = 1023.62 \text{ грн}$$

Експлуатаційні витрати на ПК на рік складуть:

$$З_{зр} = 3750 + 750 + 1023.62 = 5523.62 \text{ грн}$$

Отже, у перший рік витрати на створення й експлуатацію ПЗ складуть:

$$З_р = 37300.56 + 5523.62 = 42824.18 \text{ грн}$$

5.3 Економічна ефективність розробки і впровадження програми для оцінювання розміру телеграм-ботів на Python

Основним показником економічної ефективності програмного забезпечення є зменшення трудових витрат та часу на оцінювання розміру програмного забезпечення, а також ефективне розподілення трудових ресурсів на основі отриманих результатів. Ефективний розподіл ресурсів дає змогу зменшити час оцінювання, і, відповідно, грошові витрати на роботу спеціалістів.

Визначимо пряму економічну ефективність, ґрунтуючись на тому, що впровадження програмного продукту вивільняє 1.5 працівника (за експертною оцінкою фахівців підприємства).

Заробітна плата 1.5 працівника в рік складає:

$$З = 10000 * 12 * 1.5 = 180000 \text{ грн}$$

Річний економічний ефект розраховується за формулою:

$$E_{рік} = \Delta C_п – E_п * k \quad (5.3)$$

де $\Delta C_п$ – вивільнені кошти після впровадження продукту (180000 грн) мінус експлуатаційні витрати (42824.18 грн) = 137175.82 грн;

$E_п$ – коефіцієнт ефективності (дорівнює коефіцієнту амортизації 0.25);

k – одноразові витрати на впровадження продукту (37300.56 грн).

$E_{рік} = 137175.82– 0.25 * 37300.56 = 127850.68$ грн

Термін окупності ПЗ розраховується за формулою:

$$T = k / \Delta C_п = 37300.56 / 137175.82 = 0.27 \text{ року}$$

Отже, термін окупності ПЗ становить приблизно 4 місяці.

5.4 Висновки

У цьому розділі були здійснені розрахунки витрат, економічної ефективності та терміну окупності ПЗ для оцінювання розміру телеграм-ботів на Python.

Виходячи з розрахунків, впровадження даного програмного забезпечення окупить себе протягом 4 місяців. Таким чином, його розробка є економічно обґрунтованою.

6 ОХОРОНА ПРАЦІ

6.1Вступ

Охорона праці – це така система, що піклується про збереження життя та здоров’я працівників у ході їх трудової діяльності через правові, соціально-економічні, організаційно-технічні, санітарно-гігієнічні, лікувально-профілактичні, реабілітаційні та інші заходи.

Законодавство про працю містить норми і вимоги з техніки безпеки і виробничої санітарії, норми, що регулюють робочий час і час відпочинку, звільнення та переведення на іншу роботу, норми праці щодо жінок, молоді, гігієнічні норми і правила тощо.

Загальний нагляд за додержанням норм охорони праці покладено на прокуратуру, спеціальний – на професійні спілки. Контроль за безпекою праці здійснюють також державні й відомчі спеціалізовані інспекції.

Комп’ютерна техніка дуже широко застосовується у житті людини: вдома, на роботі та відпочинку, у різноманітних організаціях та виробництвах – інакше кажучи, комп'ютери міцно увійшли до повсякденного життя людей та масштаби цієї інтеграції з часом тільки збільшуються.

Недотримання вимог безпеки та експлуатації приводить до того, що з часом оператори комп’ютерної техніки починають відчувати перевтому та загальне погіршення стану здоров’я, що проявляється у погіршенні зору, головних болях, проблемах з хребтом, шиєю та кистями рук.

6.2 Аналіз шкідливих та небезпечних факторів на робочому місці програміста

Недостатн’є освітлення

Освітлення – використання світлової енергії Сонця та штучних джерел світла для забезпечення можливості безперешкодного зорового сприйняття оточуючого середовища.

Світло є необхідною умовою житт’єдіяльності людини, воно потрібно для збереження здоров’я та високої продуктивності праці, що виявляється у якості проведених робіт та виробленої продукції.

Виступаючи подразником не тільки для органів зору, а і для всього організму, достатнє освітлення діє тонізуючи, покращує діяльність нервової системи та емоційний стан, настрій, впливає на формування добового ритму фізіологічних процесів організму людини.

Недостатнє освітлення приміщень та робочих поверхонь, навпаки, може потягнути за собою цілу купу негативних наслідків, котрі у результаті негативно позначаться на здоров’ї працівників та результатах їх праці. До впливів на здоров’я слід віднести такі прояви, як: втрата зору, головні болі, психологічні розлади тощо.

Згідно СНіП II-4-79, в приміщенні зі встановленими комп’ютерами, необхідно застосовувати систему комбінованого освітлення. У таких приміщеннях висуваються наступні вимоги щодо освітленості: при виконанні зорових робіт високої точності загальна освітленість повинна складати 300лк, а комбінована –750лк; аналогічні вимоги при виконанні робіт середньої точності - 200 і 300лк відповідно. Крім того усе поле зору повинне бути освітлено максимально рівномірно – це основна гігієнічна вимога. Іншими словами, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими, тому що яскраве світло у районі периферійного зору значно збільшує напруженість очей і, як наслідок,

призводить до швидкого стомлення очей, що є однією з головних причин погіршення зору.

Параметри мікроклімату

Параметри мікроклімату можуть змінюватись у широких межах, в той час як необхідною умовою життєдіяльності людини є підтримка сталості температури тіла завдяки терморегуляції, тобто здатності організму регулювати власну температуру через віддачу тепла в навколишнє середовище. Принцип нормування мікроклімату – створення оптимальних умов для теплообміну тіла людини з оточуючим середовищем, що зроблять знаходження людей у приміщені комфортним та приємним.

Обчислювальна техніка є джерелом істотних тепловиділень, так, температура центрального процесора під час роботи може перевищувати 60°С, графічного ядра – більше ніж 50°С, тому для охолодження вузлів комп'ютера використовуються повітряні або водяні системи охолодження, що відводять тепло від елементів системи, розсіюючи його у навколишнє середовище. Такий вплив на мікроклімат приміщення може призвести до підвищення температури і зниження відносної вологості в приміщенні. У санітарних нормах СН-245-71 встановлені величини параметрів мікроклімату, що створюють комфортні умови для більшості працівників. Ці норми встановлюються залежно від пори року, характеру трудового процесу і характеру виробничого приміщення (дивись таблицю 6.1).

Таблиця 6.1 – Параметри мікроклімату для приміщень, де встановлені комп'ютер.

Період року	Параметр мікроклімату	Величина
Холодний	Температура повітря в приміщенні Відносна вологість Швидкість руху повітря	22…24°С 40…60% до 0,1м/с
Теплий	Температура повітря в приміщенні Відносна вологість Швидкість руху повітря	23…25°С 40…60% 0,1…0,2м/с

На кожну людину, що працює у приміщені, об'єм цього приміщення повинен складати не менше ніж 19,5 $м^3$ на особу з урахуванням

максимального числа одночасно працюючих у зміну. Норми подачі свіжого повітря в приміщення, де розташовані комп'ютери, приведені в таблиці 6.2.

Таблиця 6.2 – Норми подачі свіжого повітря в приміщення, де розташовані комп'ютери.

Характеристика приміщення	Об'ємна витрата свіжого повітря, що подається в приміщенні м3 /на одну людину в годину
Обсяг до 20м^3 на людину	Не менше 30
20…40м^3 на людину	Не менше 20
Більше 40м^3 на людину	Природна вентиляція

Шум і випромінювання

Шум – це сукупність звуків різної частоти та інтенсивності, що виникають у результаті коливального руху частинок у пружних, тобто шумом можна назвати звуки котрі не не суть ніякого корисного інформаційного навантаження та своїм існуванням заважать працювати та концентруватися на певних задачах.

Шум погіршує умови праці, роблячи шкідливий вплив на організм людини: працюючи в умовах тривалого шумового впливу люди відчувають дратівливість, головний біль, запаморочення, нервову втомленість, крім того, шум значно зменшує об'єм засвоєної інформації та підвищує кількість зроблених помилок під час виконання роботи порушуючи концентрацію.

Тривала дія інтенсивного шуму, рівень якого перевищую 80 дБА, на слух людини приводить до суттєвої втрати працеспроможності.

Рівень шуму на робочому місці не повинен перевищувати 50 дБА. Для зниження рівня шуму стіни і стеля приміщень, де встановлені комп'ютери, можуть бути облицьовані звукопоглинальними матеріалами, а самі робочі машини слід або ізолювати від працівників у інше приміщення, або використовувати якісні корпуса, що самі собі гарно ізолюють працюючі елементи системи від середовища, виступають бар'єром, та значно зменшують негативні впливи на людей.

Вважається, що як короткочасний, так і тривалий вплив усіх видів випромінювання від екрану монітора або системного блоку безпечно для людини. Проте вичерпних даних щодо небезпеки дії випромінювання на працюючих з комп’ютерами не існує.

Максимальний рівень рентгенівського випромінювання на робочому місці оператора комп'ютера звичайно не перевищує 10мкбер/год, а інтенсивність ультрафіолетового і інфрачервоного випромінювань від екрану монітора лежить в межах 10…100мВт/м^2. Для зниження дії цих видів випромінювання рекомендується застосовувати якісні монітори та іншу периферійну техніку, а також дотримуватися регламенту періодів праці та відпочинку. Слід обмежити тривалість робочої діяльності біля екрана, не розміщувати дисплеї концентровано у робочій зоні, вимикати монітори, якщо за ним не працюють.

Ергономічні вимоги до робочого місця

Проектування робочих місць належить до важливих проблем ергономічного проектування у області обчислювальної техніки.

Робоче місце і взаємне розташування всіх його елементів повинне відповідати антропометричним, фізичним і психологічним вимогам. Велике значення має також характер роботи. Зокрема, при організації робочого місця співробітника повинні бути дотримані наступні основні умови: оптимальне розміщення обладнання, достатній робочий простір, що дозволяє здійснювати всі необхідні рухи і переміщення.

Ергономічними аспектами проектування робочих місць, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому, характеристики робочого крісла, вимоги до поверхні робочого столу, можливість регулювання елементів робочого місця.

Головними елементами робочого місця співробітника є робочий стіл і крісло. Основним робочим положенням є положення сидячи.

Робоча поза сидячи викликає мінімальне стомлення співробітника(але водночас може привести до сколіозу через постійну, ледве помітну

перевтому, що може виникати при порушені режиму праці та відпочинку, та часто ігнорується співробітниками). Раціональна планування робочого місця передбачає чіткий порядок і сталість розміщення предметів, засобів праці і документації. Те, що потрібно для виконання робіт частіше, має бути розташоване у зоні легкої досяжності робочого простору.

Для комфортної роботи оператора стіл повинен задовольняти наступним вимогам:

– така висота столу, щоб можна було зручно та вільно сидіти, спиратися на підлокітник якщо потрібно;

– вільне місце для ніг знизу;

– антиблікове покриття робочої поверхні столу;

– не менше трьох висувних ящиків.

Висота від підлоги до робочої поверхні повинна бути близько 680-760 мм. Висота поверхні, на яку встановлюється клавіатура – 650 мм.

Велике значення надається характеристикам робочого крісла. Так, висота сидіння над рівнем підлоги перебуває в межах 420-550 мм. Поверхня сидіння м'яка, передній край закруглений, а кут нахилу спинки – не фіксований, з можливістю регулювання.

Положення екрану визначається:

– відстанню від оператора до матриці (0,6 ... 0,7 м);

– кутом зчитування, напрямком погляду на 20° нижче горизонталі екрану.

Повинна також передбачатися можливість регулювання екрану:

– за висотою +/- 3 см;

– за нахилом від -10° до +20° по вертикалі;

Особливо важливою є поза користувача. При незручній робочій позі можуть з'явитися болі у м'язах, суглобах і сухожиллях, виникнути хронічні проблеми з поставою. Вимоги до робочої пози користувача наступні:

– голова не повинна бути нахилена більш ніж на 20°,

– плечі повинні бути розслаблені,

– лікті – під кутом 80° ... 100°,

– передпліччя і кисті рук – у горизонтальному положенні.

Причина неправильної пози користувачів обумовлена наступними чинниками: недостатній простір для ніг, не правильно відрегульоване крісло, монітор стоїть на неправильній відстані, стіл завеликий або навпаки, робоча поверхня розташовано занадто низько.

Вагоме значення для продуктивної і якісної роботи на комп'ютері мають розміри знаків, густина їх розміщення, контраст і співвідношення яскравості символів і фону екрану. Якщо відстань від очей до екрана дисплея становить 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 дисплея 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, викликаючи обезводнення очей. До того ж, якщо екран встановлений високо, а очі широко відкриті, порушується функція моргання: очі не закриваються повністю, чи не омиваються слізної рідиною, не отримують 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, так і для підвищення його привабливості, якісно впливає на продуктивність.

6.3 Техніка безпеки при роботі на комп’ютері

Загальні положення:

1) При виконанні робіт на персональних комп’ютерах необхідно дотримуватись вимог загальної та даної інструкцій з охорони праці.

2) До самостійної роботи на комп’ютерах допускаються особи, які пройшли медичний огляд, навчання з професії, ввідний інструктаж з охорони праці і первинний інструктаж з охорони праці на робочому місці. У подальшому вони проходять повторні інструктажі з охорони праці на робочому місці один раз на півріччя.

3) Основним обладнанням робочого місця користувача ПК є: монітор, системний блок, клавіатура, друкуючий пристрій (принтер), зовнішні пристрої пам’яті, сканер, "миша", блок безперервного живлення, робочий стіл, стілець та інше; допоміжним – шафи, полиці, сейф та ін.

Робочі місця повинні бути розташовані на відстані не менше 1,5 м від стіни з вікнами, від інших стін – на відстані 1м, між собою – на відстані не менше 1,5 м. Відносно вікон робочі місця доцільно розташувати таким чином, щоб природне світло падало на нього збоку, переважно зліва.

5) Монітори слід розташовувати таким чином, щоб запобігти потраплянню в очі прямого світла. Джерела освітлення слід розташовувати з обох боків екрану, паралельно напрямку зору.

Для запобігання світлових відблисків від екрана, клавіатури в напрямку очей користувача від освітлювачів загального призначення або сонячних променів необхідно застосовувати спеціальні фільтри, захисні козирки, на вікнах регульовані жалюзі.

Фільтри із металевої або нейлонової сітки використовувати не рекомендується тому, що сітка спотворює зображення внаслідок інтерференції світла. Найкраща якість зображення забезпечується скляними поляризаційними фільтрами – вони усувають практично усі відблиски і забезпечують чітке та контрастне зображення.

6) При роботі з текстовою інформацією (у режимі введення даних, редагування тексту і читання з екрану) найбільш фізіологічним є зображення чорних знаків на світлому (білому) фоні.

7) Розташовувати монітор на робочому місці слід так, щоб поверхня екрану знаходилась у центрі поля зору на відстані 400-700 мм від очей користувача. Пропонується розташовувати елементи робочого місця таким чином, щоб підтримувалась однакова відстань від очей користувача до екрана, клавіатури, пюпітра.

8) Робота комп'ютера супроводжується електромагнітним випромінюванням різних частот і рівнів, інтенсивність котрого зменшується пропорційно квадрату відстані до екрану. Оптимальною для працюючого за ПК є відстань 50 см від екрану монітору.

9) Раціональною робочою позою працюючого за ПК вважається таке розташування тіла, при якому ступні працівника розташовані горизонтально на підлозі або на підставці для ніг, стегна зорієнтовані в горизонтальній площині, верхні кінцівки рук – вертикально, кут ліктьового суглоба коливається в межах 70-90 град., зап'ястя зігнуті під кутом, не перевищує 20 град., нахил голови – у межах 15-20 град.

10) Для нейтралізації зарядів статичної електрики в приміщенні, де виконуються роботи на комп'ютерах, в тому числі на лазерних та струменевих принтерах, пропонується збільшувати вологість повітря за допомогою кімнатних зволожувачів. Не бажано носити одяг з синтетичних матеріалів.

Вимоги безпеки перед початком роботи

1) Перевірити надійність встановлення апаратури на робочому столі. Монітор не повинен стояти на краю столу. Повернути монітор треба так, щоб було зручно дивитись на екран – під прямим кутом (а не збоку) і трохи згори вниз, а екран повинен бути нахиленим під невеликим кутом (нижній край ближче до оператора).

2) Перевірити загальний стан апаратури, справність проводки електромережі, з'єднувальних шнурів, штепсельних вилок та розеток, заземлення захисного екрана.

4) Відрегулювати освітленість робочого місця.

5) Відрегулювати і зафіксувати висоту стільця (крісла.

6) Під'єднати до системного блока необхідну апаратуру (принтер, сканер і т.п.). Всі кабелі, що з'єднують системний блок з іншими пристроями слід вставляти і виймати тільки при вимкнутому електричному живленні.

7) Увімкнути апаратуру ПК вмикачами на корпусах в такій послідовності : стабілізатор (або блок безперервного живлення), монітор, системний блок, принтер (якщо передбачається друкування).

8) Відрегулювати яскравість світіння екрану, фокусировку, контрастність. Не слід робити яскравість екрану занадто великою, тому що це втомлює очі.

Рекомендується:

- яскравість світіння екрану – не менше 100 кд/ кв.м (кандел на кв.м);
- відношення яскравості екрану до яскравості оточуючих його поверхонь в робочій зоні – не більше 3:1;
- мінімальний розмір точки світіння – не більше 0,4 мм для монохромного екрану і не більше 0,6 мм для кольорового;
- контрастність зображення знака – не менше 0,8.

9) При появі несправностей комп'ютерної техніки повідомити керівника.

Вимоги безпеки при закінченні роботи на ПК

1) Закінчити і зберегти в пам'яті ПК файли, які знаходились у роботі. Виконати всі дії для коректного завершення роботи в оперативній системі.

2) Вимкнути принтер та інші периферійні пристрої, вимкнути монітор і системний блок. Вимкнути стабілізатор (або блок безперервного живлення

при його наявності). Штепсельні вилки витягнути з розеток. Накрити клавіатуру кришкою для попередження потрапляння в неї пилу.

3) Навести порядок на робочому місці.

4) Вимкнути освітлення та електроприлади.

Вимоги безпеки при аварійних ситуаціях

1) При раптовому припиненні подачі електричної енергії вимкнути всі пристрої ПК в такій послідовності: периферійні пристрої, монітор, системний блок, стабілізатор (або блок безперервного живлення). Витягнути вилки з розеток.

2) При наявності ознак горіння (дим, запах горілого) необхідно вимкнути всі пристрої ПК, згідно з попереднім пунктом. Знайти місце загоряння і виконати всі можливі заходи для його ліквідації, попередивши терміново про це керівництво.

3) У випадку виникнення пожежі негайно попередити про це пожежну частину та керівництво, виконати усі можливі заходи щодо евакуації людей з приміщення і розпочати гасіння пожежі первинними засобами пожежогасіння.

7 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

7.1 Вступ

Комплекс екологоорієнтованих засобів щодо захисту навколишнього середовища охоплює заходи, спрямовані на охорону і раціональне використання природних ресурсів, і заходи, які забезпечують нормативні санітарно-гігієнічні параметри середовища міських і сільських поселень. Соціально необхідні охоронні заходи поділяються на організаційні, економічні і містобудівні.

Організаційні заходи забезпечують на законодавчому рівні використання територій, форми власності, правовий захист територій, створення системи адміністративно-господарського управління територіями та спеціальної екологічної служби їх охорони.

Економічні заходи забезпечують впровадження ресурсозберігаючих технологій, введення штрафних санкцій за порушення норм природокористування, визначення платежів і податків за використання територій, надання пільгових кредитів виробникам екологічно чистої продукції тощо.

Містобудівні заходи забезпечують охорону природного середовища за рахунок раціонального функціонального зонування території, створення санітарно-захисних зон, визначення територій природно-заповідного фонду, забезпечення екологічного балансу природно-ландшафтних та урбанізованих територій. Основні принципи екологічного захисту навколишнього середовища такі:

– збереження та раціональне використання цінних природних ресурсів;

– дотримання нормативів гранично допустимих рівнів екологічного навантаження на природне середовище та санітарних нормативів в місцях забудови;

– виділення природно-заповідних, ландшафтних, курортно-рекреаційних, історико-культурних зон з відповідним режимом їх охорони;

– встановлення санітарно-захисних зон для охорони водойм, джерел водопостачання і мінеральних вод, покладів лікувальних грязей, морських пляжів тощо.

Для захисту найбільш цінних елементів території навколишнього середовища вживаються заходи, спрямовані на заборону в їх межах, не властивої для них, містобудівної діяльності (крім будівництва об'єктів, що пов'язані з функціональною експлуатацією цих територій]. Це стосується природних заповідників, заказників, природних національних парків, водоохоронних зон, зелених зон міст, зон санітарної охорони курортів.

Не допускається містобудівна діяльність на площах залягання корисних копалин (до погодження з органами державного гірничого нагляду], в районах розміщення породних відвалів вугільних шахт (ближче 200-500 м залежно від характеристик терикону], на земельних ділянках, забруднених органічними і радіоактивними відходами, у небезпечних зонах зсувів, селевих потоків і снігових лавин, у зонах можливого затоплення, у сейсмічних районах тощо.

Для охорони навколишнього середовища міських і сільських поселень у межах приміських зон на землях лісового фонду формуються «зелені зони» у складі лісопаркової та лісогосподарської частин, місць відпочинку, заповідних об'єктів.

Навколо міських і сільських поселень, які розташовані у безлісих районах, створюються вітрозахисні і берегоукріплювальні лісові смуги завширшки 500 м (для найзначніших і значних міст], 100 м (для великих і середніх міст] і 50 м (для малих міст і сільських поселень].

Історичне середовище з пам'ятками історії та культури зберігається і захищається на засадах створення спеціальних зон, які охоплюють місця концентрації пам'яток, зони регулювання забудови, які прилягають до охоронних зон, зони ландшафту, що охороняється, заповідні зони.

Конкретні заходи щодо захисту навколишнього середовища вживаються відповідно до специфіки окремих джерел забруднення.

7.2 Забруднення навколишнього середовища офісом комп'ютерної фірми

Ніхто з учених так і не придумав можливості сконструювати повністю безпечну для навколишнього середовища техніку. Стара техніка нерідко стає причиною забруднення навколишнього середовища. Відправлена на звалище вийшла з ладу електротехніка цілком здатна дійсно завдати нашій планеті великої шкоди. Електротехніка в наш час не купується на довгі роки, тому що на зміну техніці майже відразу приходять нові моделі техніки. Рідко буває, що обладнання в компаніях в наші дні взагалі ремонтується - зазвичай замість негайно купується нове обладнання. Не секрет, що якщо зламане оснащення міститься в неправильних умовах, то воно загрожує завдати шкоди чистоті довкілля. Без сумніву, правильне поводження з непотрібною технікою – це гарантія того, що екології електронне сміття не завдасть шкоди. Необхідно усвідомлювати, як захистити навколишнє середовище.

Всі, хто застосовують лампи денного світла для освітлення офісів, магазинів, кафе та ін. Приміщень, зазвичай не підозрюють про можливі проблеми, які виникнуть під час перевірок. Люмінесцентні лампи містять небезпечні для здоров'я людини хімічні речовини. У кожної люмінесцентної лампи знаходиться від 20 до 500 мг ртуті (в залежності від типу і технології), тому вони відносяться до першого, тобто найвищого класу небезпеки. Утилізація люмінесцентних ламп, їх зберігання, повинні проводитися відповідно до вимог законодавства, розглянутими нижче.

7.3 Заходи для зменшення забруднення

За допомогою повністю безпечних способів зберегти природу від шкоди людям дає шанс процедура утилізації. Кілька етапів утилізації за правилами повинна пройти утилізованих техніка. Той факт, що необхідність кожного з етапів утилізації величезна, професіонали цієї сфери обов'язково аргументують. Доставка відправленого на утилізацію оснащення – це дуже складне завдання, яку доручають професіоналам. Наступний етап – розбирання привезеного оснащення настає відразу після доставки на підприємство. Зрозуміло, сортувати обладнання теж не просто, цією проблемою в свою чергу займаються професіонали. Ці етапи покликані за всіма правилами відокремити одні елементи техніки від інших. У той час, коли співробітники в процесі сортування знаходять електронний компонент, вони відправляють цей компонент на проведення такої процедури, як афінаж. Під час цієї процедури дорогоцінні метали відділяються від домішок. Треба, до того ж, пам'ятати, що закон передбачає обов'язкове проведення процесу афінажу. Фонд дорогоцінних металів і дорогоцінного каміння – ось куди відправляться оброблені дорогоцінні метали.

Способи утилізації ламп:

1) термічна демеркуризація

Процес по розбору лампи відбувається на спеціальній установці – демеркуризаторі. Лампи подаються по одній в установку, де після подрібнення з склометалевого брухту проходить сублімація парів ртуті. Далі під дією сорбенту пари ртуті уловлюються і осідають в конденсаторі.

2) термовакуумний метод

Відходи, у яких знаходиться ртуть, потрапляють у вакуумну пастку, відбувається конденсація небезпечного пара, після чого пари ртуті піддаються виморожуванню рідким азотом. Потім розморожена зібрана ртуть стікає в приймач.

3) реагентний метод

Спосіб оснований на обробці роздріблених виробів хімічними демеркуризаторами з метою переведення ртуті в важкорозчинні сполуки.

Всі ці процеси небезпечні, оскільки биті лампи – відкрите джерело отруйних парів ртуті. Відомо, що утилізація однієї лампи КЛЛ вагою до 150 грам в процесі переробки може дати до 50 грамів скла (відправляється на повторне використання для виробництва тих же ламп або абразивів) і близько 5 мг ртуті на виготовлення нових ламп, а також до 3 грам люмінофора, що підлягає.

ВИСНОВКИ

Метою даної кваліфікаційної роботи є підвищення достовірності оцінювання розміру телеграм-ботів на Python. Мета роботи була повністю досягнута.

В кваліфікаційній роботі було удосконалено регресійну модель для оцінювання розміру телеграм-ботів на Python та розроблено програмне забезпечення для її реалізації.

Також в роботі розроблена документація:

– технічне завдання (додаток А);
– програма і методика випробувань програмного забезпечення (додаток Б);
– опис програми (додаток В);
– інструкція користувача (див. додаток Г).

Всі задачі, які ставилися в кваліфікаційній роботі, були виконані в повному обсязі.

Виконано тестування та випробування розробленого програмного забезпечення, які показали, що програмне забезпечення повністю відповідає задачам дослідження та вимогам, які висувалися в технічному завданні.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кантор М. Управление программными проектами. Практическое руководство по разработке успешного программного обеспечения.: Пер. с. англ. [Текст] / М. Кантор. – М.: Издательский дом "Вильямс", 2002. – 176 с.

2. Malathi, S. et al. Analysis of size metrics and effort performance criterion in software cost estimation [Текст] / S. Malathi et al. // Indian Journal of Computer Science and Engineering. –2012. – Volume 2, Issue 1. – P. 24-31.

3. Приходько, С. Б. Оцінювання розміру PHP-застосунків з відкритим кодом за нелінійними регресійними моделями з різними факторами / С. Б. Приходько, М. В. Ворона // Зб. наук. пр. НУК. – Миколаїв : НУК, 2021. – № 1 (484). – С. 92-98.

4. Приходько, Н. В. Нелінійна регресійна модель для оцінювання розміру програмного забезпечення інформаційних систем на базі VB [Текст] / Н. В. Приходько, С. Б. Приходько // Інформаційні технології та компютерна інженерія. –2018. – № 3. – С. 37-42.

5. Макарова, Л. М. Побудова нелінійної регресійної моделі для оцінювання розміру веб-додатків, реалізованих мовою Java [Текст] / Л. М. Макарова, Н. В. Приходько, О. О. Кудін // Вестник Херсонского национального университета. –2019. – №2(69), ч.1. – С. 145-154.

6. Шакула О.В. Нелінійна регресійна модель оцінювання трудомісткості підготовки файлів з описами товарів інтернет-магазинів / О.В. Шакула, Л.О. Латанська // Інформаційні технології: моделі, алгоритми, системи (ITMAS – 2021 р.): Матеріали II Всеукраїнської науково-практичної інтернет конференції (26-28 жовтня 2021 р.).– Миколаїв: НУК імені адмірала Макарова, 2021. – с.30-31.

7. Top Programming Languages 2021 [Електронний ресурс]. – Режим доступу: https://spectrum.ieee.org/top-programming-languages/ – Назва з екрану

8. Боэм Б.У. Инженерное проектирование программного обеспечения: Пер. с англ. [Текст] / Б.У. Боэм. – М.: Радио и связь, 1985. – 512 с.

9. Новичков, А. Метрики кода и их практическая реализация в Subversion и ClearCase. Часть 1 – метрики / А. Новичков, А. Шамрай, А. Черников [Электронный ресурс]. – Режим доступу: http://cmcons.com/articles/ CC_CQ /dev_metrics/mertics_part_1/ – Назва з екрану.

10. Ледовских, И. Метрики сложности кода [Электронный ресурс]. – Режим доступа: http://www.ispras

11. Albrecht A.J., Gaffney J.E. Software function, source lines of codes, and development effort prediction: a software science validation. – IEEE Transaction on Software Engineering, Vol. 6, 1983, pp. 639-647

12. Применение алгоритмических сетей для оценки ресурсов в программных проектах [Електронний ресурс]. – Режим доступу: https://www.dissercat.com/content/primenenie-algoritmicheskikh-setei-dlya-otsenki-resursov-v-programmnykh-proektakh – Назва з екрану

13. Кобзарь, А.И. Прикладная математическая статистика. Для инженеров и научных работников [Текст] / А.И. Кобзарь. – М.: ФИЗМАТЛИТ, 2006. – 816с.

14. Магнус, Я.Р. Эконометрика. Начальный курс: Учеб. – 6-е изд., перераб. и доп. [Текст] / Я.Р. Магнус, П.К. Катышев, А.А. Пересецкий. – М.: Дело, 2004. – 576 с.

15. Нелінійна регресія. вибір і порівняння нелінійних регресійних моделей [Електронний ресурс]. – Режим доступу: http://dn.khnu.km.ua/dn/ k_default.aspx?M=k1314&T=02&lng=1&st=0 – Назва з екрану

16. Регресійний аналіз [Електронний ресурс]. – Режим доступу: https://pidruchniki.com/17280924/ekonomika/regresiyniy_analiz – Назва з екрану

17. Chatterjee, Samprit. Handbook of Regression Analysis [Text] / Samprit Chatterjee, Jeffrey S. Somonoff. Wiley, 2012. – 240 p.

18. Приходько, С.Б. Методичні вказівки до виконання лабораторних робіт з дисципліни "Обробка експериментальних даних на ЕОМ" [Текст] / С.Б.Приходько – Миколаїв: НУК , 2005. – 52 с.

19. Вендров А.М., Проектирование программного обеспечения экономических информационных систем. – М.: Финансы и статистика, 2006. – 544 с.

ДОДАТОК А

Технічне завдання на розробку програмного забезпечення «ПЗ для оцінювання розміру телеграм-ботів на Python»

Вступ

Назва програми – ПЗ оцінювання розміру телеграм-ботів на Python. Програмне забезпечення являється програмою, яка буде надавати змогу користувачу провести оцінювання розміру.

1. Підстави для розробки

Документом, на підставі якого ведеться розробка програмного продукту, є завдання на кваліфікаційну роботу «Регресійна модель для оцінювання розміру телеграм-ботів, що створюються на Python, та розробка програмного забезпечення для її реалізації».

2. Призначення розробки

2.1. Функціональне призначення

Даний програмний продукт дозволить автоматизувати та скоротити час відповідних розрахунків.

2.2. Експлуатаційне призначення

Програмний продукт буде використовуватись для оцінювання розміру телеграм-ботів на Python та оптимізації коду і побудови інтервалу прогнозування та довірчого інтервалу на основі десяткового логарифму.

3. Вимоги до програми

3.1. Вимоги до функціональних характеристик

3.1.1. Вимоги до складу функцій, що виконуються

Програма повинна забезпечувати можливість виконання наведених нижче функцій:

1) введення інформації про telegram-бот на Python (кількість строк, кількість класів);

2) нормалізація введеної інформації десятковим логарифмом;

3) прогнозування розміру telegram-боту на Python;

4) визначення границь інтервалу довіри;

5) визначення границь інтервалу передбачення;

6) корегування даних.

3.1.2 Вимоги до організації вхідних та вихідних даних

Введення вхідних даних здійснюється користувачем у відповідні поля діалогового вікна програмного забезпечення.

Вихідні дані (результат оцінювання розміру telegram-боту на Python, довірчий інтервал, інтервал передбачення) виводиться у діалоговому вікні програмного забезпечення.

3.2 Вимоги до надійності

3.2.1 Вимоги до забезпечення надійності функціонування програми

Надійне (стійке) функціонування програми має бути забезпечено надійним (стійким) функціонуванням операційної системи.

3.2.2 Відмова виконання програмного продукту через некоректні дії користувача

Відмова програми можлива внаслідок некоректних дій користувача при взаємодії з операційною системою. Щоб уникнути виникнення відмов

програми за вказаною вище причини необхідно забезпечити стабільне функціонування операційної системи.

3.4 Умови експлуатації

Кліматичні умови експлуатації програмного продукту відповідають умовам експлуатації апаратної частини персонального комп'ютера.

3.5 Вимоги до інформаційної та програмної сумісності

Системні програмні засоби (не вільно поширювані), використовувані програмою, повинні бути представлені ліцензійною версією ОС Windows.

3.6 Вимоги до захисту інформації та програм

Вимоги до захисту інформації та програм відсутні.

3.7 Вимоги до маркування та упаковки

3.7.1 Вимоги до маркування

Вимоги до маркування відсутні.

3.7.2 Вимоги до упаковки

Вимоги до упаковки відсутні.

3.8 Вимоги до транспортування та зберігання

Вимоги до транспортування програми відсутні.

4 Вимоги до програмної документації

Склад програмної документації:

- технічне завдання;
- текст програми;

– опис програми;
– програма і методика випробувань;
– інструкція користувача.

5 Вимоги до складу та параметрів технічних засобів

– ПК з процесором не нижче : Celeron N2840;
– Обсяг оперативної пам‘яті - не менше 2 Гб;
– Необхідний обсяг на жорсткому диску - не менше 300 Мб.

5 Техніко-економічні показники

Економічна ефективність впровадження програмного забезпечення розрахована у розділі 6. Згідно з отриманими результатами впровадження даного ПЗ є доцільним, а термін його окупності становить приблизно 4 місяці.

6 Стадії та етапи розробки програмного забезпечення

Стадії та етапи розробки програми оцінювання розміру телеграм-ботів на Python, зазначені в таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки програмного забезпечення

Стадії розробки	Етапи робіт	Контрольні точки
Технічне завдання	Обумовлення необхідності розробки	З 02.09.2021 до 08.09.2021
	Розробка та затвердження технічного завдання	З 09.09.2021 до 13.09.2021
Ескізний проект	Розробка ескізного проекту	З 14.09.2021 до 29.09.2021
	Затвердження ескізного проекту	З 30.09.2021 до 15.10.2021
Технічний проект	Розробка технічного проекту	З 16.10.2021 до 26.10.2021
	Затвердження технічного проекту	З 27.10.2021 до 03.11.2021
Робочий проект	Розробка ПЗ	З 04.11.2021 до 12.11.2021
	Розробка програмної документації	З 13.11.2021 до 23.11.2021
	Випробування програми	З 23.11.2021 до 30.11.2021

7 Порядок контролю та приймання

Контроль здійснюється замовником на кожній стадії розробки ПЗ. Приймання здійснюється замовником за програмою та методикою випробувань. Випробування проводиться в зазначений термін.

Хід проведення приймально-здавальних випробувань документується за допомогою протоколу проведення випробувань. На підставі протоколу проведення випробувань виконувач сумісно з замовником підписують акт приймання-здачі програми в експлуатацію.

ДОДАТОК Б

Програма і методика випробувань ПЗ для оцінювання розміру телеграм-ботів на Python

1 Об’єкт випробувань

Об’єктом випробувань є ПЗ для оцінювання розміру телеграм-ботів на Python.

2 Мета випробувань

Перевірка функціональних можливостей програмного забезпечення на відповідність вимогам технічного завдання.

3 Вимоги до програми

В ході тестування програмного продукту необхідно перевірити, що розроблене ПЗ реалізує наступні функції:

1) введення інформації про telegram-бот на Python (кількість строк, кількість класів);

2) нормалізація введеної інформації десятковим логарифмом;

3) прогнозування розміру telegram-боту на Python;

4) визначення границь інтервалу довіри;

5) визначення границь інтервалу передбачення;

6) корегування даних.

4 Вимоги до програмної документації

Склад програмної документації, що надається на випробування:

- технічне завдання;
- текст програми;
- опис програми;
- програма і методика випробувань;

– інструкція користувача.

5 Вимоги до складу та параметрів технічних засобів

– ПК з процесором не нижче : Celeron N2840;

– Обсяг оперативної пам‘яті - не менше 2 Гб;

– Необхідний обсяг на жорсткому диску - не менше 300 Мб.

6 Засоби і порядок випробувань

Порядок випробувань:

1) перевірити можливість введення інформації про telegram-бот на Python (кількість строк, кількість класів);

2) перевірити можливість прогнозування розміру telegram-боту на Python;

4) перевірити можливість визначення границь інтервалу довіри;

5) перевірити можливість визначення границь інтервалу передбачення;

6) перевірити можливість корегування даних.

7 Методи випробувань

Випробування будуть проводитися у наступному порядку:

– Випробування функції «Корегування даних»

Обрати пункт «Корегування». Ввести нові дані та натиснути на кнопку «Ввести». Очікуваний результат: в таблиці «Програмні проекти» та «Дані для прогнозування» повинна занестися оновлена інформація.

– Випробування функції «Прогнозувати розмір»

Обрали пункт «Прогнозування». Очікуваний результат: на екран повинні вивестися результати прогнозування для введених даних. Після перегляду необхідно натиснути кнопку «В меню» та перейти до вибору дії.

– Випробування функції «Введення даних»

Обрати пункт «Введення». Ввести нові дані та натиснути на кнопку «Ввести». Очікуваний результат: в таблиці «Програмні проекти» та «Дані для прогнозування» повинна занестися оновлена інформація.

ДОДАТОК В

Опис програмного забезпечення для оцінювання розміру телеграм-ботів на Python

1 Загальні відомості

Програмне забезпечення для оцінювання розміру телеграм-ботів на Python, яке дозволяє автоматизувати та знизити трудомісткість розрахункових робіт.

Програмне забезпечення реалізовано з використанням мови програмування Java на базі jre 1.8. Для функціонування програмного забезпечення необхідний встановлений і налаштований SQL сервер баз даних.

2 Функціональне призначення

Програмне забезпечення призначене для оцінювання розміру телеграм-ботів на Python.

3 Опис логічної структури

За своєю логічною структурою програмне забезпечення складається з наступних модулів.

Модуль аналітика – забезпечує користувача такими функціями, як введення початкових даних, редагування даних, а також дозволяє переглянути результати оцінювання розміру телеграм-ботів на Python.

Модуль розрахунку – дозволяє виконати функції нормалізації початкових даних, оцінювання розміру телеграм-ботів на Python, розрахунку довірчого інтервалу та інтервалу прогнозування.

4 Вимоги до складу та параметрів технічних засобів

- ПК з процесором не нижче : Celeron N2840;
- Обсяг оперативної пам‘яті - не менше 2 Гб;
- Необхідний обсяг на жорсткому диску - не менше 300 Мб.

5 Вхідні дані

Вхідними даними є найменування проекту, кількість класів.

6 Вихідні дані

Вихідні дані (результат оцінювання розміру, довірчий інтервал, інтервал передбачення).

ДОДАТОК Г

Керівництво користувача програмного забезпечення для оцінювання розміру телеграм-ботів на Python

1 Вступ

Це керівництво призначене для ознайомлення користувача із технічними і функціональними можливостями програмного забезпечення для оцінювання розміру телеграм-ботів на Python.

.

Програмне забезпечення призначене для оцінювання розміру телеграм-ботів на Python.

Програмне забезпечення має графічний інтерфейс, який дозволяє введення даних, переглянути результати оцінювання розміру телеграм-ботів на Python.

2 Загальні відомості про програму

2.1 Позначення і найменування програми

Найменування програми – «Оцінювання розміру телеграм-ботів на Python».

2.2 Мови програмування, на яких написана програма

Програма написана на мові програмування Java.

2.3 Призначення програми

Програмне забезпечення призначене для оцінювання розміру телеграм-ботів на Python.

2.4 Можливості програми

Програмне забезпечення виконує наступні функції:

- введення початкових даних;
- редагування даних;
- нормалізація даних;
- оцінювання розміру коду;
- розрахунок довірчого інтервалу;
- розрахунок інтервалу прогнозування.

3 Умови застосування програми

3.1 Умови, необхідні для виконання програми

Спеціальні умови для виконання програми відсутні.

3.2 Вимоги до технічних засобів, необхідних для функціонування програми

Рекомендовані вимоги:

- ПК з процесором не нижче : Celeron N2840;
- Обсяг оперативної пам‘яті – не менше 2 Гб;
- Необхідний обсяг на жорсткому диску – не менше 300 Мб.

4 Загальний опис роботи програми

Робота користувача починається з вибору відповідного пункту меню.

В меню Файл користувач може виконати наступні дії:

- Ввести дані;
- Відкорегувати дані;
- Виконати прогнозування.

На рисунку Г.1 представлений графічний інтерфейс головного вікна ПЗ для оцінювання розміру телеграм-ботів на Python.

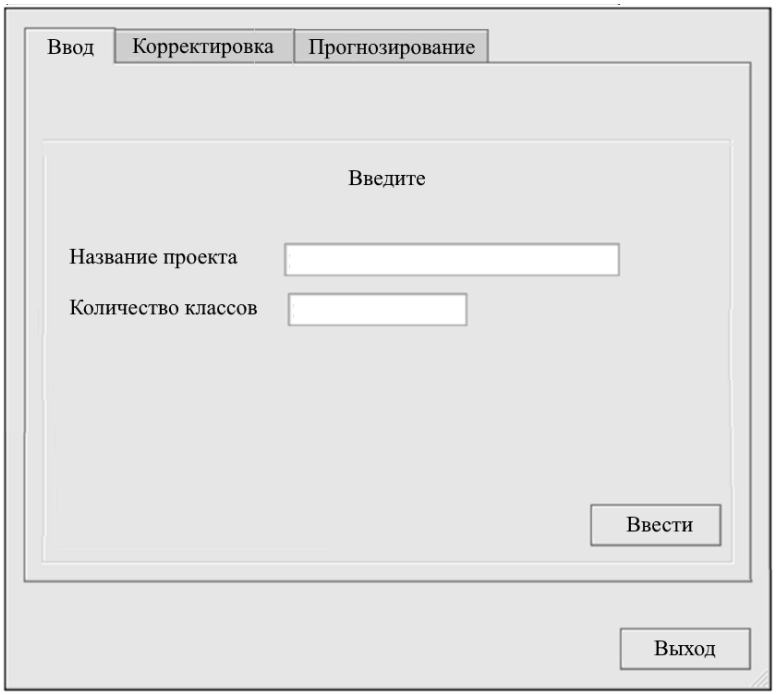

Рисунок Г.1 – Графічний інтерфейс головного вікна ПЗ для оцінювання розміру телеграм-ботів на Python

Як видно з рисунку Г.1, в програмному забезпеченні реалізовані функції введення даних для прогнозування, корегування даних та виконання прогнозування, яке включає в себе розрахунок прогнозного значення розміру програмного забезпечення, розрахунок довірчого інтервалу та інтервалу прогнозування. При натисненні на клавішу «Ввести» інформація зберігається. При натисненні на клавішу «Вихід» програма завершує роботу.

На рисунку Г.2 представлений графічний інтерфейс вікна, в якому виводяться результати прогнозування розміру ПЗ, а також довірчий інтервал та інтервал прогнозування.

При натисненні на клавішу «В меню» повертаємося до вибору дії.

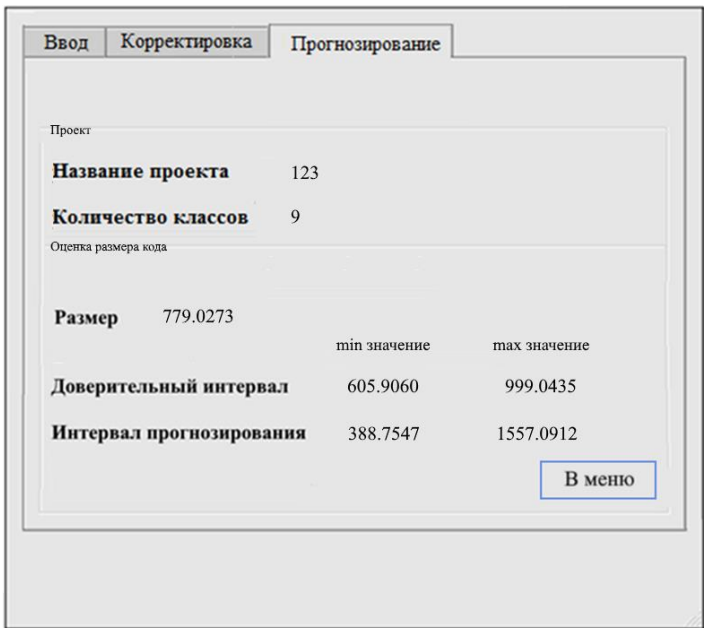

Рисунок Г.2 – Графічний інтерфейс вікна «Прогнозування» ПЗ для оцінювання розміру телеграм-ботів на Python

ДОДАТОК Д

Текст програмного забезпечення

```
jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder("Оценка розмера кода"));

jLabel8.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
jLabel8.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
jLabel8.setText("Размер:");

jLabel9.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

jLabel10.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
jLabel10.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
jLabel10.setText("min значение");

jLabel11.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
jLabel11.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
jLabel11.setText("max значение");

jLabel12.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
jLabel12.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
jLabel12.setText("Доверительный интервал");

jLabel13.setFont(new java.awt.Font("Tahoma", 1, 12)); // NOI18N
jLabel13.setText("Интервал прогнозирования

jLabel14.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

jLabel15.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

jLabel16.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
```

```
        jLabel17.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jSeparator2)
                .addContainerGap())
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addComponent(jLabel10)
                        .addGap(36, 36, 36))
                    .addGroup(jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
                            .addGroup(jPanel2Layout.createSequentialGroup()
                                .addGap(121, 121, 121)
                                .addComponent(jLabel8)
                                .addGap(18, 18, 18)
                                .addComponent(jLabel9))
                            .addGroup(jPanel2Layout.createSequentialGroup()
```

```
                    .addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jLabel12)
                        .addComponent(jLabel13))
                    .addGap(32, 32, 32)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jLabel16)
                        .addComponent(jLabel14))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jLabel15)
                .addComponent(jLabel11)
                .addComponent(jLabel17))
            .addGap(38, 38, 38))
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jLabel8)
                .addComponent(jLabel9))
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jSeparator2,
javax.swing.GroupLayout.PREFERRED_SIZE,                    10,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.BASELINE)
                .addComponent(jLabel10)
                .addComponent(jLabel11))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.BASELINE)
                .addComponent(jLabel12)
                .addComponent(jLabel14)
                .addComponent(jLabel15))
            .addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.BASELINE)
                .addComponent(jLabel13)
                .addComponent(jLabel16)
                .addComponent(jLabel17))
            .addContainerGap(19, Short.MAX_VALUE))
        );

        jButton1.setText("В меню");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
```

```
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAI
LING)
                    .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jSeparator1)
                    .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
            .addGroup(layout.createSequentialGroup()
                .addGap(169, 169, 169)
                .addComponent(jButton1)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jSeparator1,
javax.swing.GroupLayout.PREFERRED_SIZE,                    12,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jButton1)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        pack();
    }
```