**3 ОПИС ПРОГРАМНОГО ПРОДУКТУ**

## **3. 1 Опис функціональної моделі**

Програмний продукт (ПП) містить наступні складові:

1. **робота з індивідуальною задачею (ІЗ):**
   1. зчитування даних ІЗ з файлу;
   2. розв’язання ІЗ усіма розробленими модифікаціями (з виведенням на екран відповідних розв’язків та графіків);
   3. задання параметрів експериментів:

* коефіцієнт швидкості навчання;
* мінімальний допустимий приріст коефіцієнтів;
* кількість ітерацій;
* шукана функція;
* функціонал помилки;
* метод регуляризації;
* коефіцієнт регуляризації (з’являється при виборі методу регуляризації);
* функція масштабування;
* степінь шуканої функції (з’являється при вибору поліноміальної шуканої функції).

1. **експериментальне дослідження розроблених алгоритмів:**
2. розв’язання задачі регресії усіма розробленими модифікаціями (з виведенням на екран відповідних розв’язків та графіків);
3. розв’язання задачі класифікації усіма розробленими модифікаціями (з виведенням на екран відповідних розв’язків та графіків);
4. задання параметрів експериментів:

* коефіцієнт швидкості навчання;
* мінімальний допустимий приріст коефіцієнтів;
* кількість ітерацій;
* шукана функція;
* функціонал помилки;
* метод регуляризації;
* коефіцієнт регуляризації (з’являється при виборі методу регуляризації);
* функція масштабування.

1. задання параметрів для генерації даних задачі регресії:

* кількість точок;
* кількість вхідних змінних;
* кількість інформативних змінних, які використовуються для побудови лінійної залежності;
* параметр для побудови нелінійної залежності;
* стандартне відхилення гауссового шуму, застосовуваного до виходу.

1. задання параметрів для генерації даних задачі класифікації:

* кількість точок;
* кількість вхідних змінних;
* кількість інформативних змінних;
* кількість надлишкових змінних (ці функції породжуються як випадкові лінійні комбінації інформативних ознак);
* кількість кластерів на клас.

**3) історія користувача:**

1. вивід параметрів та результатів виконання алгоритму з бази даних:

* назва алгоритму
* параметри експерименту (шукана функція, коефіцієнт швидкості навчання і тд.);
* дата виконання;
* час виконання;
* коефіцієнти шуканої функції;

1. очищення бази даних.

**4) порівняння розроблених алгоритмів:**

1. вивід графіків, які ілюструють роботу модифікацій алгоритму:

* графік залежності значення функціоналу помилки від номеру ітерації;
* графік залежності часу роботи алгоритму від кількості ітерації;
* графіки порівняння роботи модифікацій (передбачення) у залежності від отриманих коефіцієнтів шуканої функції на кожній ітерацій.

## **3. 2 Рішення з інформаційного забезпечення**

### 3.2.1 Вхідні дані

Дані зчитуються з файлу або генеруються згідно до параметрів, які вказав користувач.

На вхід подаються такі дані, що стосуються індивідуальної задачі:

* матриця точок X;
* вектор розв’язків y;
* відповідні параметри експериментів.

На вхід подаються такі дані, що стосуються генерації задачі регресії:

* кількість точок;
* кількість вхідних змінних;
* кількість інформативних змінних;
* параметр для побудови нелінійної залежності;
* стандартне відхилення гауссового шуму;
* відповідні параметри експериментів.

На вхід подаються такі дані, що стосуються генерації задачі класифікації:

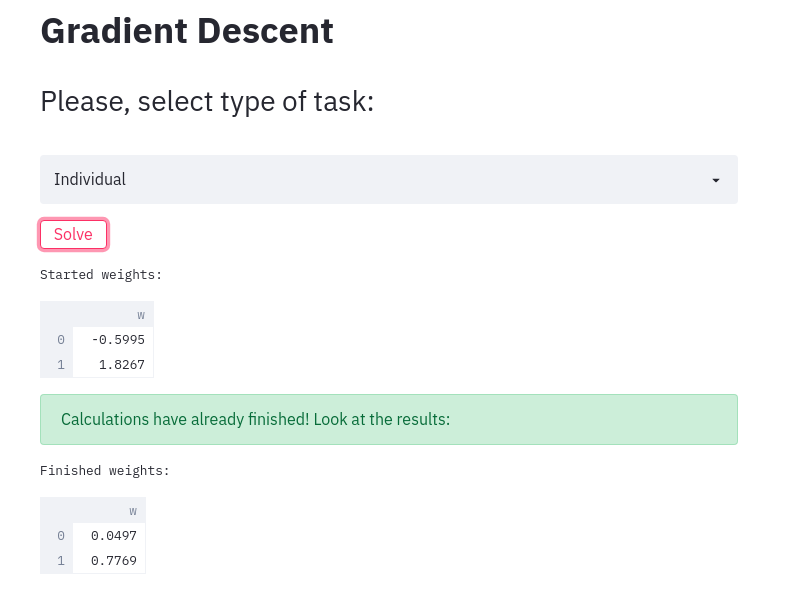
* кількість точок;
* кількість вхідних змінних;
* кількість інформативних змінних;
* кількість надлишкових змінних;
* кількість кластерів на клас;
* відповідні параметри експериментів.

### 3.2.2 Вихідні дані

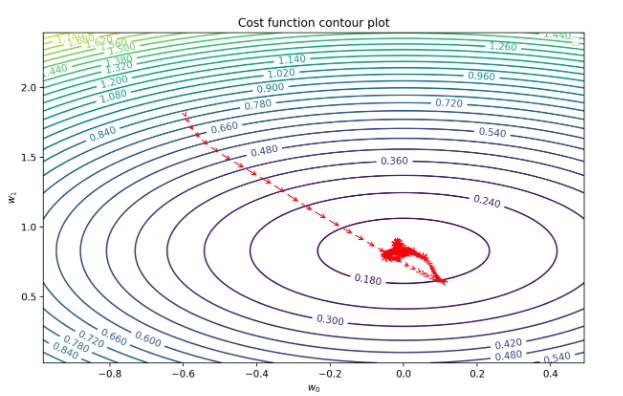
На вихід подаються наступні дані:

* початкові та кінцеві коефіцієнти шуканої функції (рисунок 3.1);
* графік зміни функціоналу помилки у двовимірному просторі (рисунок 3.2);
* графік зміни функціоналу помилки у тривимірному просторі (рисунок 3.3);
* графік значення функціоналу помилки на кожній ітерації (рисунок 3.4);
* графік розсіювання даних та шукана функція (рисунок 3.5)

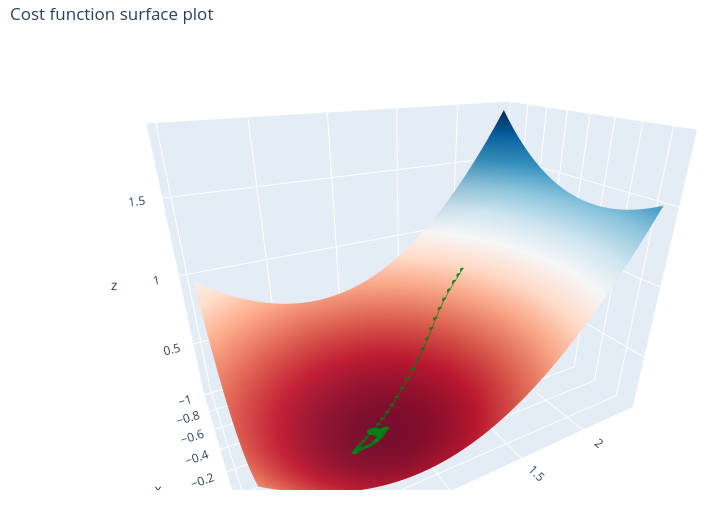
Нижче наведені скріншоти виконання програми:



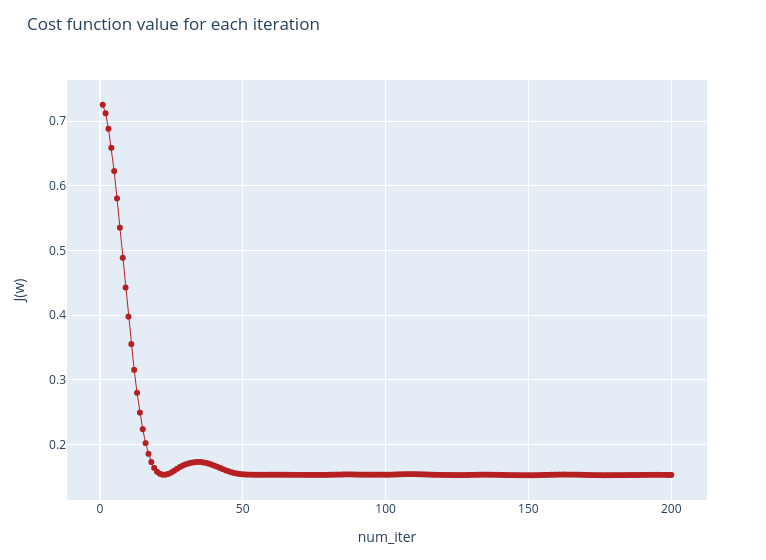
***Рисунок 3. 1 – Початкові та кінцеві коефіцієнти шуканої функції***

****

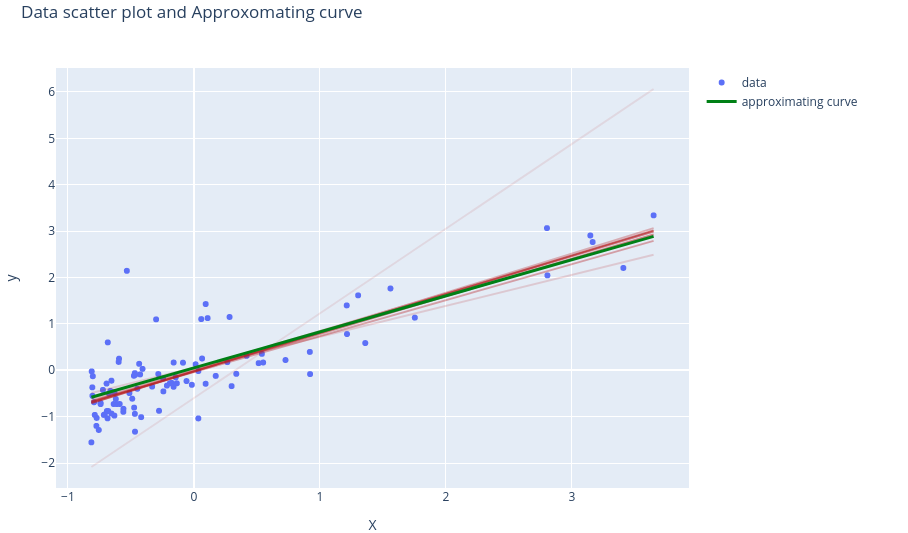
***Рисунок 3. 2 – Графік зміни функціоналу помилки у двовимірному просторі***

****

***Рисунок 3. 3 – Графік зміни функціоналу помилки у тривимірному просторі***

****

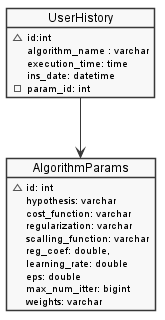
***Рисунок 3. 4 – Графік значення функціоналу помилки на кожній ітерації***

****

***Рисунок 3. 5 – Графік розсіювання даних та шукана функція***

### 3.2.3 Опис інформаційного забезпечення

Дані з процесу розв’язання зберігаються у базі даних. База даних має наступний вигляд:



Дані для порівняння алгоритмів містяться у файлах loss\_history.npy, execution\_time.npy.

Дані індивідуальної задачі містяться у файлі restaurant\_revenue.txt.

У таблиці 3.1 наведений опис файлів.

**Таблиця 3. 1 – Опис файлів**

|  |  |  |
| --- | --- | --- |
| ***Назва файлу*** | ***Тип*** | ***Опис*** |
| loss\_history | .npy  (зберігає numpy array ) | Збереження значення функціоналу помилок на кожній ітерації для всіх модифікацій |
| execution\_time | .npy  (зберігає numpy array ) | Зберігає час виконання модифікацій алгоритму від кількості ітерацій |
| restaurant\_revenue | .txt | Зберігає умову індивідуальної задачі |

## **3.3 Рішення з програмного забезпечення**

### 3.3.1 Опис обраних засобів розробки

### Дане програмне забезпечення було реалізовано на мові програмування Python з використанням веб-фреймворку Streamlit. Для зберігання даних використовуємо базу даних SQLite. Використані бібліотеки: NumPy, Sklearn, Pandas, Matplotlib, Plotly.

Python — [інтерпретована](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) високого рівня зі [строгою](https://uk.wikipedia.org/w/index.php?title=%D0%A1%D1%82%D1%80%D0%BE%D0%B3%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F&action=edit&redlink=1) [динамічною типізацією](https://uk.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D1%96%D1%87%D0%BD%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F). Розроблена в [1990](https://uk.wikipedia.org/wiki/1990) році [Гвідо ван Россумом](https://uk.wikipedia.org/wiki/%D0%93%D0%B2%D1%96%D0%B4%D0%BE_%D0%B2%D0%B0%D0%BD_%D0%A0%D0%BE%D1%81%D1%81%D1%83%D0%BC). [Структури даних](https://uk.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її ефективною для швидкої розробки програм, а також як засіб поєднання наявних компонентів. Python підтримує [модулі](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій, так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька [парадигм програмування](https://uk.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), зокрема: [об'єктно-орієнтована](https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), [процедурна](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F), [функціональна](https://uk.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) та [аспектно-орієнтована](https://uk.wikipedia.org/wiki/%D0%90%D1%81%D0%BF%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F). [1]

Також, математичні модулі, написані для даної мови програмування, є високопродуктивними та мають зручний інтерфейс. Саме тому ми обрали Python для написання коду алгоритмів та функціоналу нашого проекту.

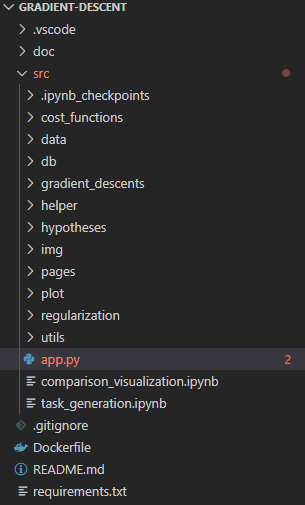
Streamlit - це фреймворк з відкритим кодом, спеціально розроблений для інженерів машинного навчання, які працюють з Python. Він дозволяє створювати додатки без знання HTML. [2]

SQLite - це вбудована кросплатформна база даних, яка підтримує широкий набір команд SQL. Ми обрали саме її, оскільки в Python є встановлені за замовчуванням бібліотеки для роботи з нею та наш проект не потребує використання складних запитів. [3]

***3.3.2 Архітектура програмного забезпечення***

***3.3.2.1 Схема архітектури ПП***

На рисунку 3.6 наведена схема архітектури розробленого програмного забезпечення.

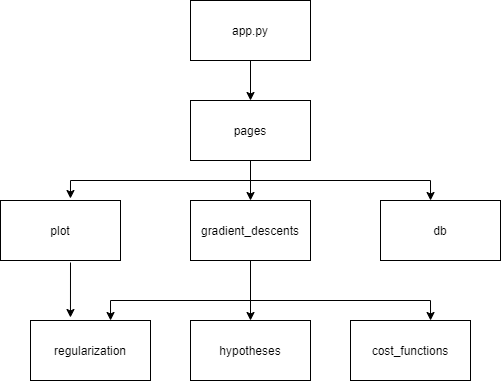
**

***Рисунок 3. 6 – Схема архітектури розробленого програмного забезпечення***

Програмне забезпечення складається з модулів:

* pages - інтерфейс користувача;
* gradient\_descents - реалізація алгоритму та його модифікацій;
* regularization - реалізація методів регуляризації;
* plot - реалізація побудови графіків;
* hypotheses - реалізація шуканих функцій;
* helper - допоміжний функціонал для виводу тексту;
* db - робота з базою даних;
* cost\_functions - реалізація функціоналів помилок;

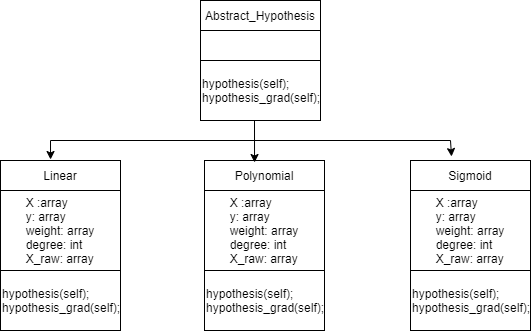
Точка входу знаходиться у файлі app.py, який має посилання на модуль pages. У свою чергу модуль pages має посилання на модулі gradient\_descents, plot, helper, db. Модуль gradient\_descents має посилання на regularization, hypotheses та cost\_functions. Модуль plot також має посилання на regularization.



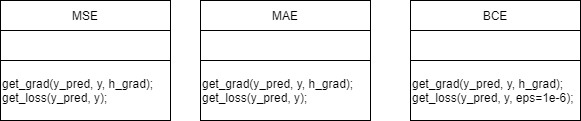
***Рисунок 3. 7 – Схема зв’язку між розробленими модулями програмного забезпечення***

#### **3.3.2.2 Діаграма класів (об'єктів)**

На рисунках 3.8-3.9 наведені діаграми розроблених класів.



***Рисунок 3.8 – Розроблені класи модулю hypotheses***



***Рисунок 3.9 – Розроблені класи модулю cost\_functions***

У таблиці 3.2 наведений опис розроблених класів.

**Таблиця 3. 2 – Опис розроблений класів**

|  |  |
| --- | --- |
| Назва класу | Опис класу |
| Abstract\_Hypothesis | Абстрактний клас шуканої функції, від якого наслідуються базові класи шуканих функцій . |
| Linear | Повертає значення функції та градієнту лінійної гіпотези. |
| Polynomial | Повертає значення функції та градієнту поліноміальної гіпотези. |
| Sigmoid | Повертає значення функції та градієнту сигмоїдної гіпотези. |
| MSE | Повертає значення функції та градієнту середньоквадратичного відхилення. |
| MAE | Повертає значення та градієнт середнього абсолютного функціоналу помилки. |
| BCE | Повертає значення та градієнт бінарного крос-ентропійного функціоналу помилки. |

#### **3.3.2.3 Специфікація функцій**

У таблиці 3.3 наведений табличний опис основних функцій системи.

**Таблиця 3. 3 – Опис основних функцій системи**

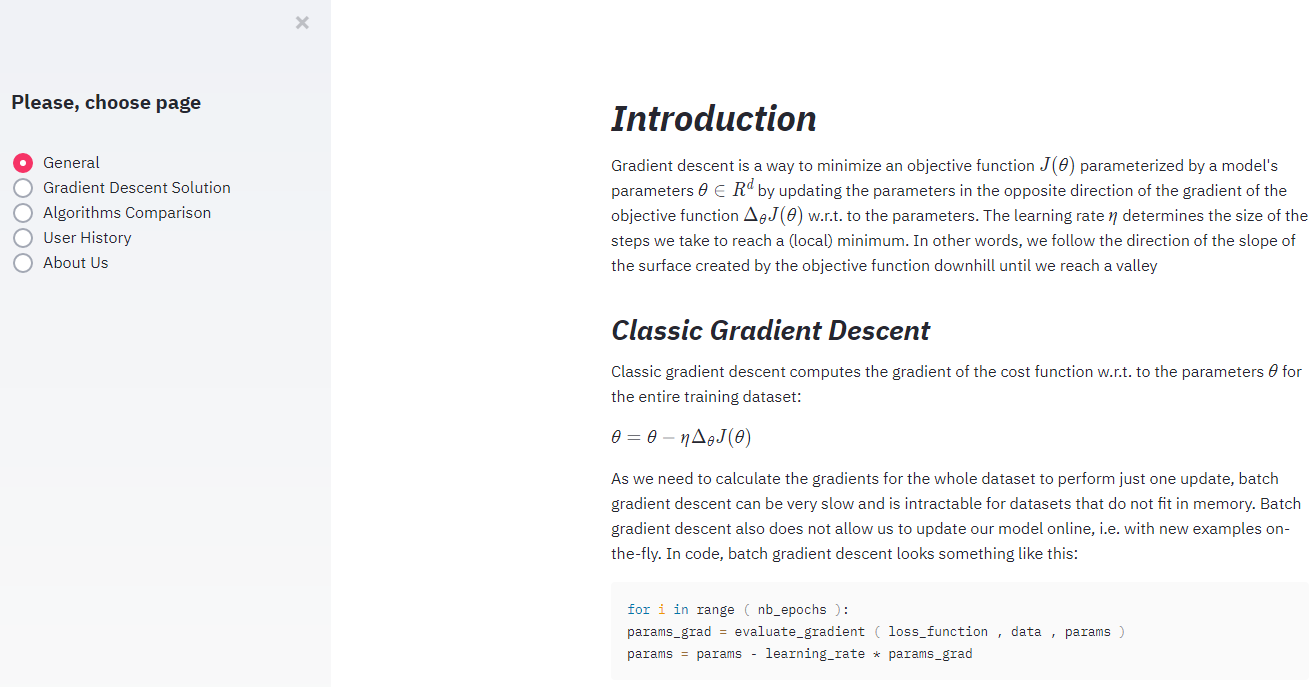
|  |  |  |  |
| --- | --- | --- | --- |
| **Ім’я функції** | **Призначення функції** | **Параметри** | **Семантика параметрів** |
| main | Створення бази даних. Точка входу в програму | Параметри відсутні | – |
| about\_us\_page | Повертає дані зі сторінки “Про нас” | Параметри відсутні | - |
| comparison\_page | Повертає графіки з порівнянням роботи алгоритмів | Параметри відсутні | - |
| gd\_solution\_page | Повертає сторінку розв’язання задачі | Параметри відсутні | - |
| general\_page | Повертає сторінку загальної інформації про алгоритм градієнтного спуску та його модифікацій | Параметри відсутні | - |
| user\_history\_page | Повертає таблицю історії вибору користувача з розв’язком та часом роботи алгоритму | Параметри відсутні | - |
| show\_side\_bar | Повертає дані вибору з бокового меню | Параметри відсутні | - |
| select\_task\_type | Повертає дані з вибору типу завдання | Параметри відсутні | - |
| params\_for\_generate\_regression | Повертає параметри для задачі регресії | Параметри відсутні | - |
| params\_for\_generate\_classification | Повертає параметри для задачі класифікації | Параметри відсутні | - |
| generate\_regression\_task | Генерація даних для задачі регресії | h\_type, degree, scaler, data\_degree, \*\*kwargs | тип шуканої функції, степінь шуканої функції, функція масштабування, параметр для побудови нелінійної залежності, параметри для задачі регресії |
| generate\_clasiffication\_task | Генерація даних для задачі класифікації | h\_type, degree, scaler, \*\*kwargs | тип шуканої функції, степінь шуканої функції, функція масштабування, параметр для задачі класифікації |
| individual\_task | Зчитування даних індивідуальної задачі та повернення об’єкту гіпотези | h\_type, degree, scaler | тип шуканої функції, степунь шуканої функції, функція масштабування |
| solve\_btn | Запуск алгоритму, зберігання даних вибору користувача та результату алгоритму, виклик функцій відмальовування графіків | h, properties, choice, task\_type | об’єкт шуканої функції, параметри задачі, вибір користувача, тип завдання |
| hypothesis | Повертає значення шуканої функції | self, X=None, w=None | Об’єкт класу шуканої функції, множина точок X, коефіцієнти шуканої функції |
| hypothesis\_grad | Повертає значення градієнту шуканої функції | self | Об’єкт класу шуканої функції |
| get\_grad | Повертає значення градієнту функціоналу помилки | y\_pred, y, h\_grad | Передбачені значення, отримані значення, градієнт шуканої функції |
| get\_loss | Повертає значення функціоналу помилки | y\_pred, y | Передбачені значення, отримані значення |
| db\_creation | Створення бази даних | Параметри відсутні | - |
| db\_clean | Очищення бази даних | Параметри відсутні | - |
| db\_insert | Вставка у базу даних | h, properties, excecution\_time, choice | Об’єкт шуканої функції, вибір параметрів алгоритму, вибір користувача |
| db\_select | Вибірка з бази даних | Параметри відсутні | - |
| markdown\_to\_string | Переведення даних з файлу .md у строку | file\_path | Шлях до файлу |
| classic\_grad\_descent | Реалізація алгоритму класичного градієнтного спуску | hypothes, max\_num\_itter, cost\_function, regularization=None, C=1, alpha=0.01, eps=0.01, mini\_batch\_size=32 | об’єкт шуканої функції, кількість ітерацій, об’єкт функціоналу помилки, регуляризація, коефіцієнт регуляризації, коефіцієнт швидкості навчання, мінімальний допустимий приріст коефіцієнтів, розмір даних для обробки алгоритму |
| stochastic\_grad\_descent | Реалізація алгоритму стохастичного градієнтного спуску | hypothes, max\_num\_itter, cost\_function, regularization=None, C=1, alpha=0.01, eps=0.01, mini\_batch\_size=1 | об’єкт шуканої функції, кількість ітерацій, об’єкт функціоналу помилки, регуляризація, коефіцієнт регуляризації, коефіцієнт швидкості навчання, мінімальний допустимий приріст коефіцієнтів, розмір даних для обробки алгоритму |
| momentum\_grad\_descent | Реалізація алгоритму стохастичного градієнтного спуску з імпульсом | hypothes, max\_num\_itter, cost\_function, regularization=None, C=1, alpha=0.01, eps=0.01, beta=0.9, mini\_batch\_size=32 | об’єкт шуканої функції, кількість ітерацій, об’єкт функціоналу помилки, регуляризація, коефіцієнт регуляризації, коефіцієнт швидкості навчання, мінімальний допустимий приріст коефіцієнтів, значення імпульсу, розмір даних для обробки алгоритму |
| rmsprop\_grad\_descent | Реалізація модифікації RMSprop | hypothes, max\_num\_itter, cost\_function, regularization=None, C=1, alpha=0.01, eps=0.01, beta=0.9, epsilon=1e-6, mini\_batch\_size=32 | об’єкт шуканої функції, кількість ітерацій, об’єкт функціоналу помилки, регуляризація, коефіцієнт регуляризації, коефіцієнт швидкості навчання, мінімальний допустимий приріст коефіцієнтів, значення імпульсу, дуже мале число, розмір даних для обробки алгоритму |
| adam\_grad\_descent | Реалізація модифікації Adam | hypothes, max\_num\_itter, cost\_function, regularization=None, C=1, alpha=0.01, eps=0.01, beta1=0.9, beta2=0.999, epsilon=1e-6, mini\_batch\_size=32 | об’єкт шуканої функції, кількість ітерацій, об’єкт функціоналу помилки, регуляризація, коефіцієнт регуляризації, коефіцієнт швидкості навчання, мінімальний допустимий приріст коефіцієнтів, значення імпульсу, параметр експоненціально ковзаючого середнього, дуже мале число, розмір даних для обробки алгоритму |

## **3.4 Керівництво користувача**

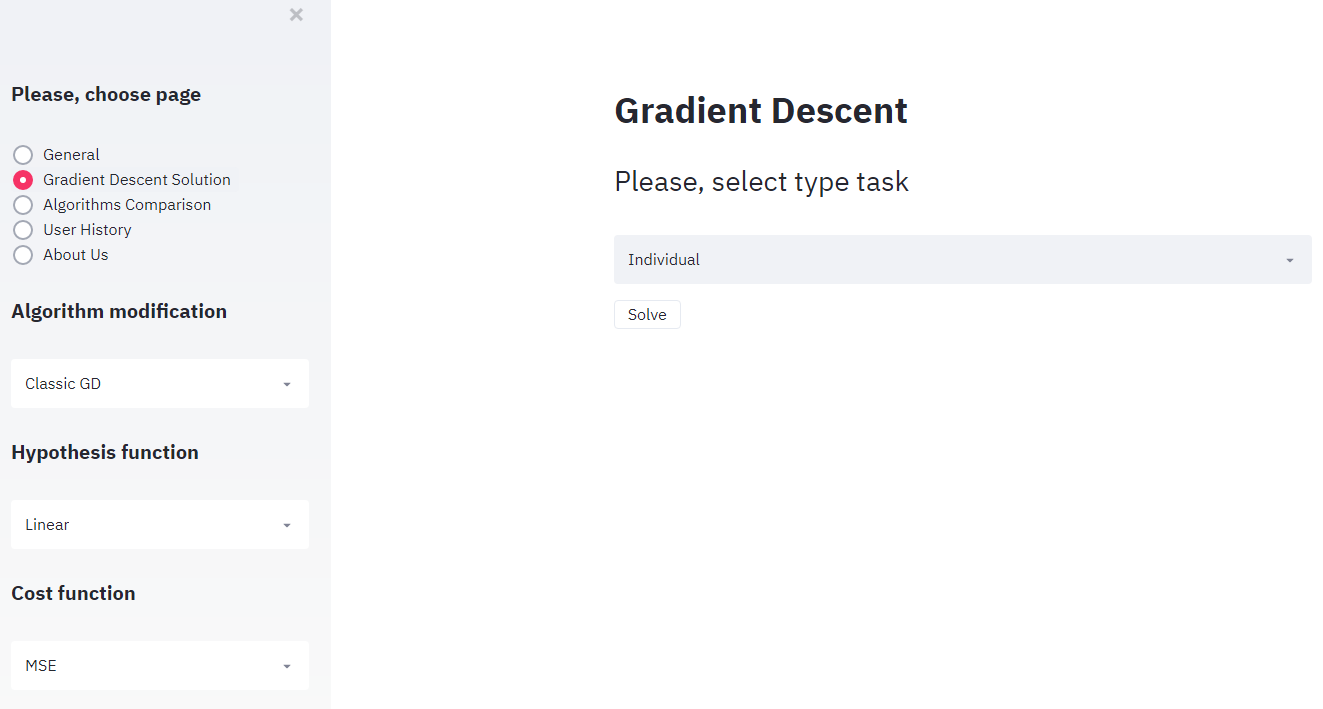
***3.4.1 Інструкція користувача***

***3.4.1.1 Короткий опис системи***

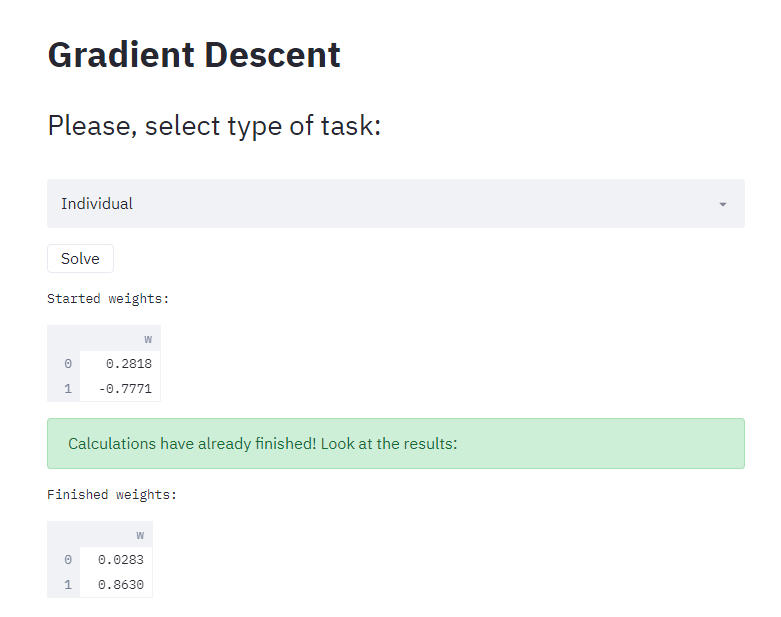
На рисунку 3.10 показаний вигляд стартової сторінки Програмного продукту.

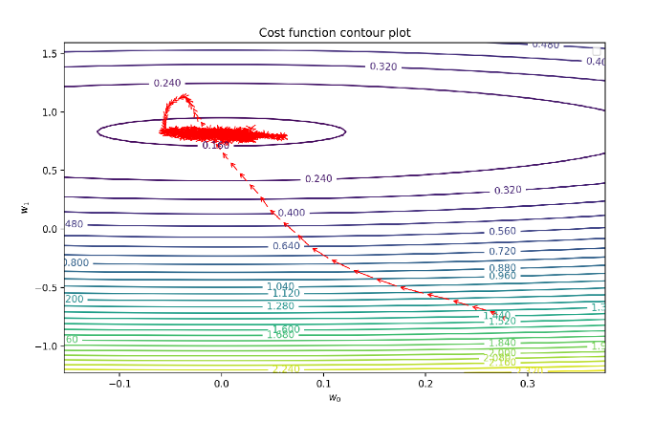
***Рисунок 3.10 – Стартова сторінка***

У боковому меню користувач має змогу обрати сторінку, яку б він хотів переглянути. На рисунку 3.11 показаний вигляд сторінки для вирішення задачі градієнтного спуску.

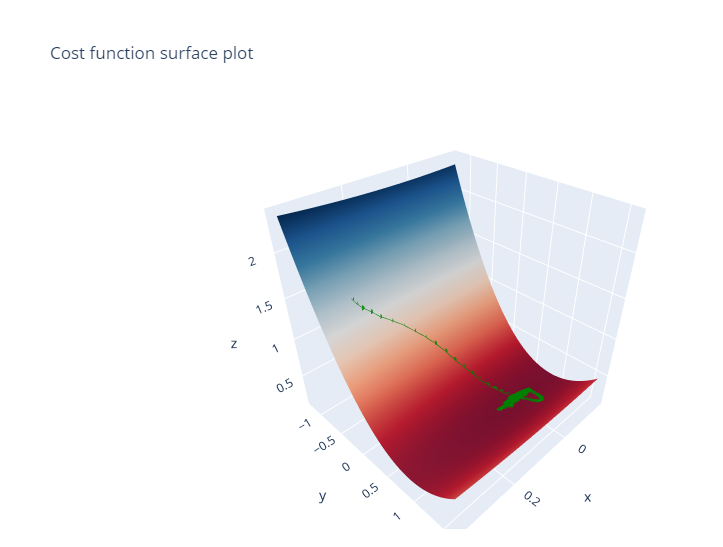


***Рисунок 3.11 – Сторінка вирішення задачі градієнтного спуску***

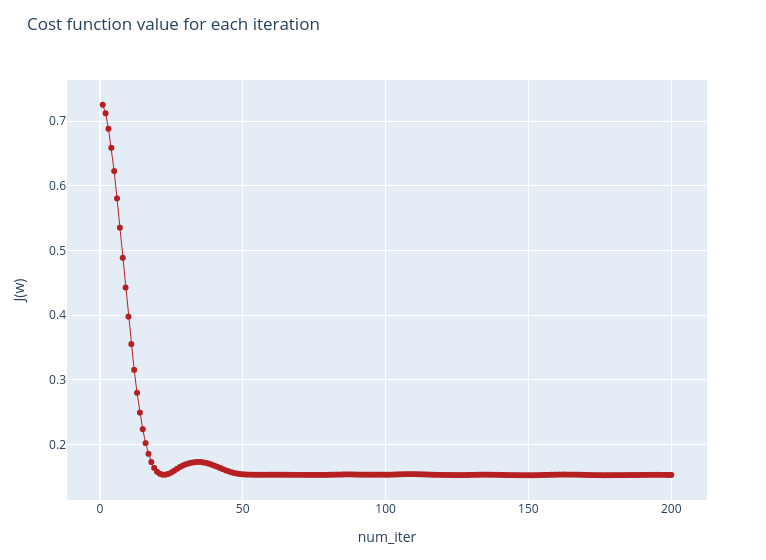
На боковому меню з’являються параметри алгоритму. На сторінці користувач має змогу обрати тип задачі. Типи задач: ІЗ, розв’язок задачі регресії, розв’язок задачі класифікації. Налаштуємо параметри алгоритму та розв’яжемо індивідуальну задачу. На рисунках 3.12 – 3.16 показаний результат роботи алгоритму.

***Рисунок 3.12 – Початкові та кінцеві значення коефіцієнтів гіпотези***

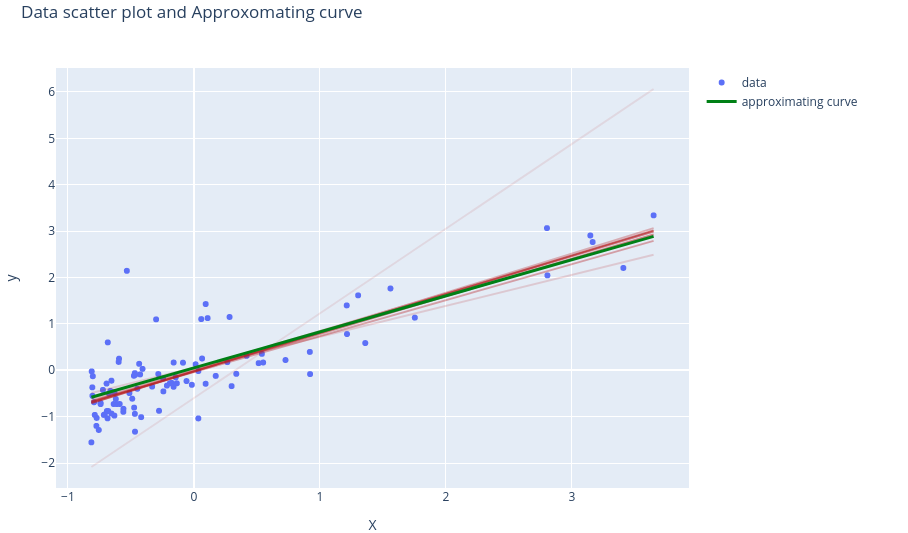
***Рисунок 3. 13 – Графік зміни функціоналу помилки у двовимірному просторі***

******

***Рисунок 3. 14 – Графік зміни функціоналу помилки у тривимірному просторі***

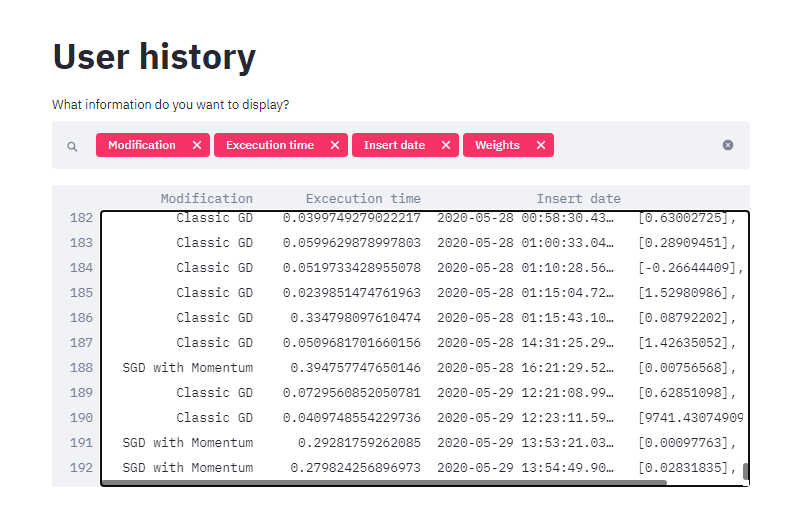
****

***Рисунок 3. 15 – Графік значення функціоналу помилки на кожній ітерації***

****

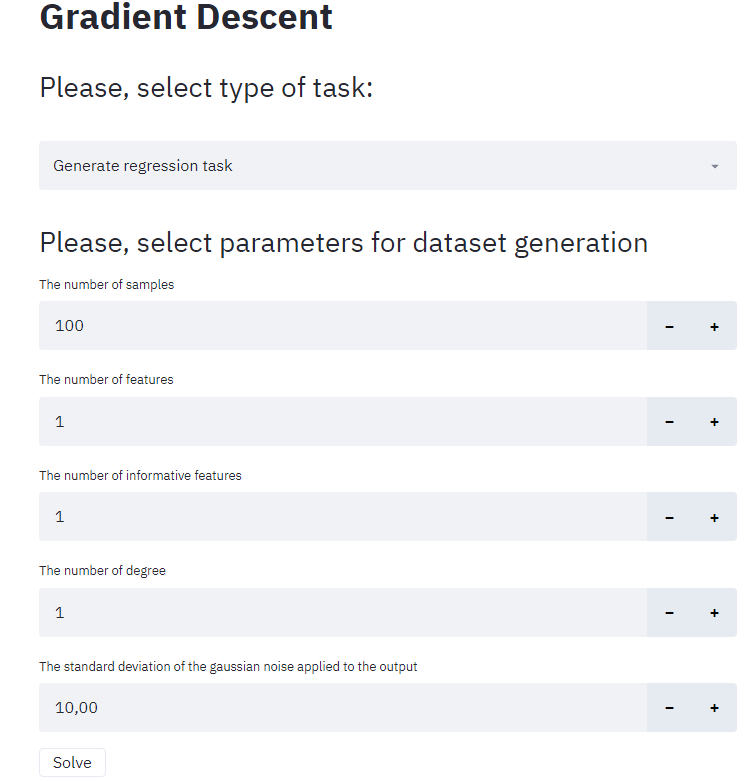
***Рисунок 3. 16 – Графік розсіювання даних та шукана функція***

Переглянувши та проаналізувавши результати, користувач має змогу перейти на сторінку історії та переглянути параметри, які він обирав, час виконання та результат роботи алгоритму. На рисунку 3.17 ми можемо переглянути поточну історію користувача.



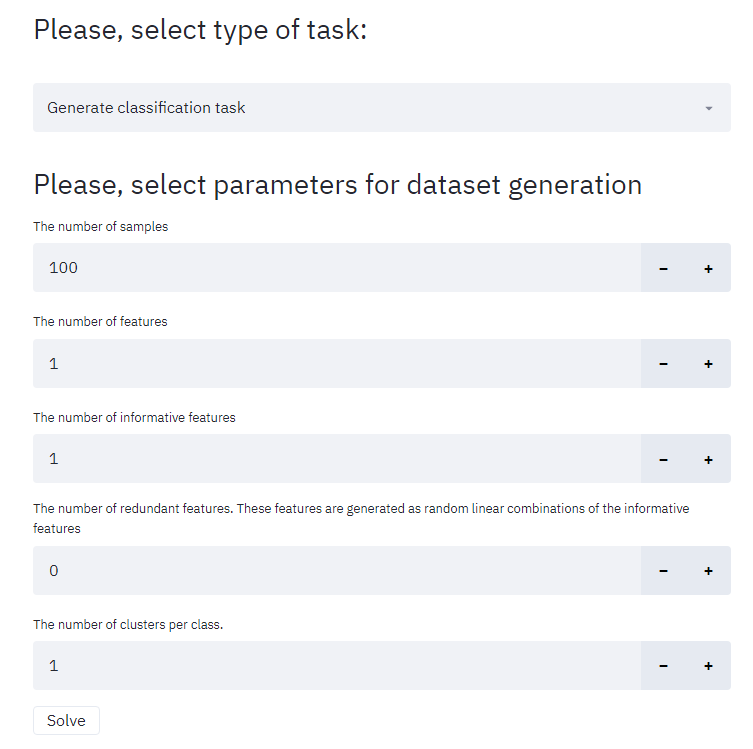
***Рисунок 3. 17 – Історія користувача***

Якщо обрати розв’язок задачі регресії, користувач зможе налаштувати додаткові параметри для генерації даних. На рисунку 3.18 показано сторінку з параметрами для задачі регресії.



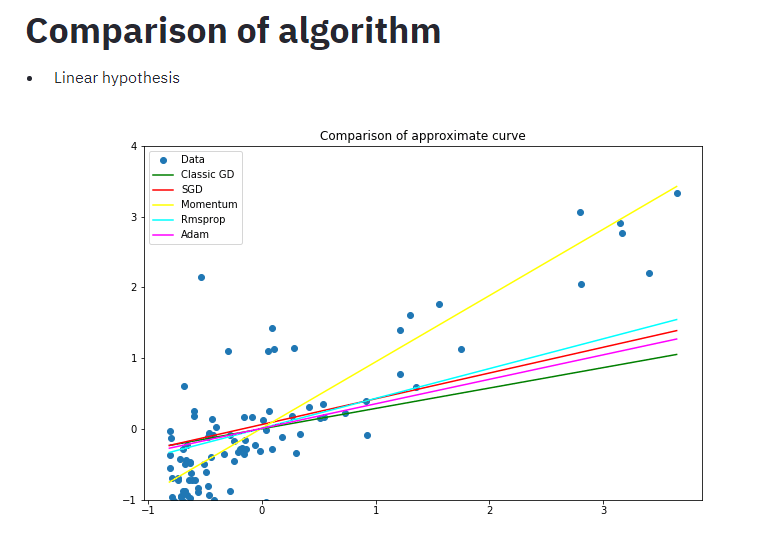
***Рисунок 3. 18 – Параметри генерації даних для задачі регресії***

Якщо обрати розв’язок задачі класифікації, користувач матиме змогу налаштувати додаткові параметри для генерації даних. На рисунку 3.19 показано сторінку з параметрами для задачі класифікації.

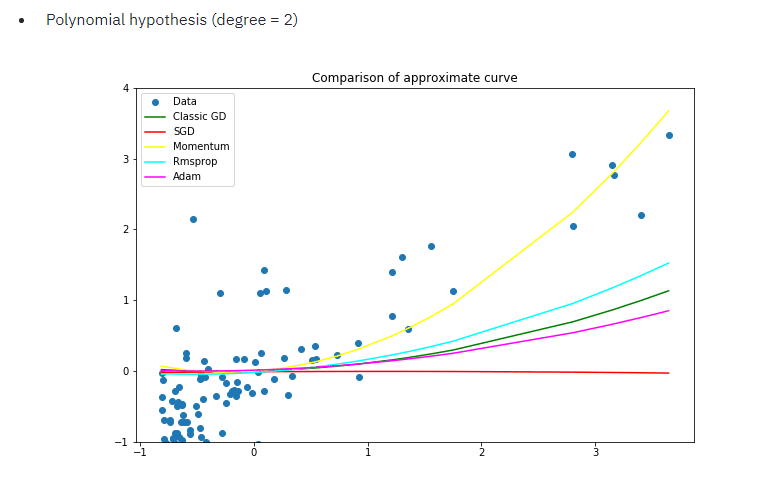


***Рисунок 3.19 – Параметри генерації даних для задачі регресії***

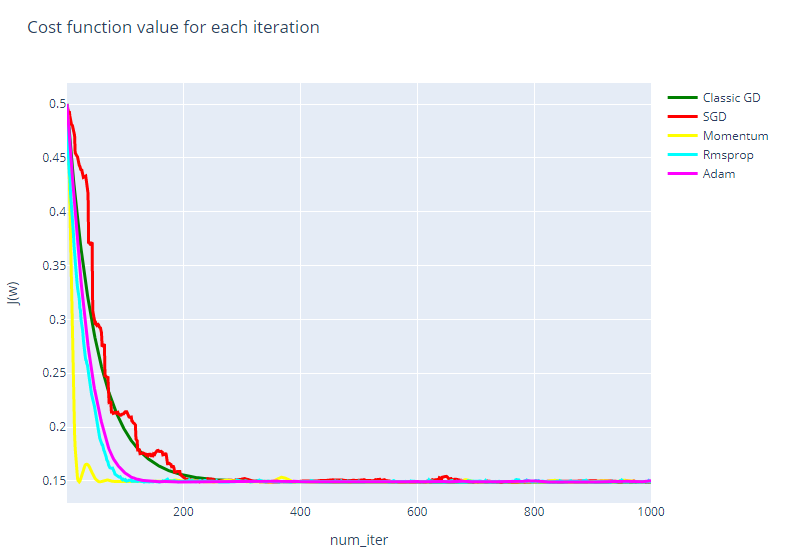
На сторінці порівняння розроблених модифікацій, користувач зможе переглянути графіки, що ілюструють роботу кожного алгоритму та зробити певні висновки стосовно їх якості та ефективності для набору даних з ІЗ. На рисунках 3.20 – 3.23 ми можемо переглянути дані графіки.



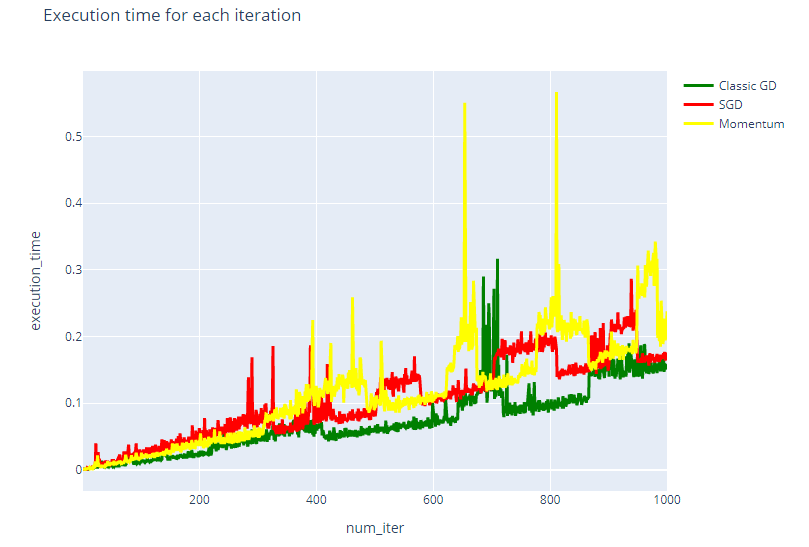
***Рисунок 3.20 – Графік (формат .gif) порівняння роботи модифікацій (передбачення) у залежності від отриманих коефіцієнтів шуканої функції на кожній ітерацій (шукана функція є лінійною)***

******

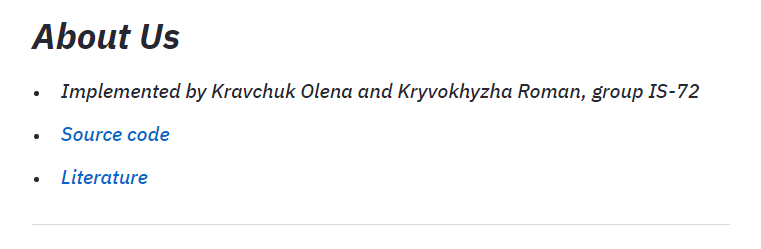
***Рисунок 3.21 – Графік (формат .gif) порівняння роботи модифікацій (передбачення) у залежності від отриманих коефіцієнтів шуканої функції на кожній ітерацій (шукана функція є квадратичною)***



***Рисунок 3.22 – Графік залежності значення функціоналу помилки від номеру ітерації***



***Рисунок 3.23 – Графік залежності часу роботи алгоритму від кількості ітерації***

На сторінці “Про нас” користувач може перейти по посиланню на код програми та переглянути літературу на тему градієнтного спуску. На рисунку 3.24 ми можемо бачити зміст даної сторінки.

***Рисунок 3.24 – Сторінка “Про нас”***

***3.4.1.2 Вимоги до встановлення***

Для розгортання системи потрібно мати встановлений Docker [4] або Pyhton 3.6+ з будь-яким пакетним менеджером.

***3.4.1.3 Встановлення системи***

Використовуючи Docker, розгортання системи на ОС Linux та Windows не будуть відрізнятись.

Встановлення системи з використанням Docker:

1. Перейти за наступним посиланням:

<https://github.com/kryvokhyzha/gradient-descent>

1. Завантажити даний репозиторій як ZIP архів або склонувати його, виконавши нижче наведену команду:

git clone <https://github.com/kryvokhyzha/gradient-descent>

1. Тепер потрібно відкрити консоль у папці зі завантаженим проектом
2. Щоб створити контейнер з проектом, потрібно виконати наступну команду:

docker build -t gradient-descent

Встановлення системи з використанням пакетного менеджера:

1. Перейти за наступним посиланням:

<https://github.com/kryvokhyzha/gradient-descent>

1. Завантажити даний репозиторій як ZIP архів або склонувати його, виконавши нижче наведену команду:

git clone <https://github.com/kryvokhyzha/gradient-descent>

1. Тепер потрібно відкрити консоль у папці зі завантаженим проектом
2. Щоб встановити необхідні модулі, потрібно виконати:

pip install -r requirements.txt

***3.4.1.4 Запуск системи***

Після коректного встановлення системи, Ви можете здійснити її запуск, обравши один з наведених способів нижче.

Запуск системи з використанням Docker:

1. Перейдіть до папки, в якій зберігається проект
2. Щоб запустити контейнер, потрібно виконати:

docker run -p 8501:8501 --rm -it gradient-descent

1. Перевірте чи в консолі з’явилось повідомлення про успішний запуск та відкрити відображений URL.

Запуск системи з використанням пакетного менеджера:

1. Перейдіть до папки, в якій зберігається проект
2. Перейдіть до папки src
3. Якщо Ви використовуєте пакетний менеджер PIP, тоді для запуску проекту виконайте наступну команду:

python -m streamlit.cli run app.py

1. Якщо Ви використовуєте пакетний менеджер Anaconda, тоді для запуску проекту виконайте наступну команду:

streamlit run app.py

1. Перевірте чи в консолі з’явилось повідомлення про успішний запуск та відкрити відображений URL.

***3.4.2 Методика випробування***

Для перевірки працездатності розробленого програмного забезпечення застосовувались функціональне тестування, що виконувалось в межах методу сірого ящику. Використовувалась техніка тестування граничних умов.

У таблиці 3.4 наведені тестові сценарії, що застосовувалися до даного програмного забезпечення.

**Таблиця 3. 4 – Тестові сценарії**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Тема** | **Кроки** | **Вхідні дані** | **Очікуваний результат** | **Фактичний результат** |
| Введення валідного значення степеня для шуканої функції | 1. Ввести степінь шуканої функції у діапазоні від 1 до 4. 2. Забрати курсор з поля вводу. | Число в діапазоні від 1 до 4 | Виконання обраного пункту меню | Виконання обраного пункту меню |
| Введення невалідного значення для степеня шуканої функції | 1. Ввести будь-який набір символів, що не є числами з діапазону від 1 до 4. 2. Забрати курсор з поля вводу. | -1а\*. | Виведення повідомлення про помилку введених даних | Виведення повідомлення про помилку введених даних |
| Введення валідного значення для кількості ітерацій | 1. Ввести значення кількості ітерацій у діапазоні від 1 до 10000. 2. Натиснути клавішу «Enter». | від 1 до 10000 | Виконання обраного пункту меню | Виконання обраного пункту меню |
| Введення невалідного значення для кількості ітерацій | 1. Ввести будь-який набір символів, що не є числами з діапазону від 1 до 10000. 2. Натиснути клавішу «Enter». | -1а\*. | Виведення повідомлення про помилку введених даних | Виведення повідомлення про помилку введених даних |

**3.5 Висновок**

У даному розділі ми описали архітектуру продукту, вхідні та вихідні параметри, навели його функціональний опис. Також, була надана інструкція користувача, використовуючи яку можна розгорнути систему на власному пристрої та ознайомитись з основними функціями та особливостями проекту.

**3.6 Список використаної літератури**

[1] Python – Вікіпедія [Електронний ресурс].. – Режим доступу до ресурсу: <https://uk.wikipedia.org/wiki/Python> .

[2] Streamlit – the fastest way to build apps [Електронний ресурс] – Режим доступу до ресурсу: <https://www.streamlit.io/>.

[3] What Is SQLite? [Електронний ресурс] – Режим доступу до ресурсу: <https://www.sqlite.org/index.html>.

[4] Get started Docker [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.docker.com/get-started/>.