Звягін микита, КМ-73

Тема: «Розробка програмного забезпечення для реалізації двошарового

персептрону з сигмоїдальною функцією активації»

# Теоретична частина

Нейронна мережа - це модель яку використовують для класифікації даних у системах із багатьма вимірами. Модель вчиться ставити у відповідність вхідним (у модель) масивам даних правильні вихідні масиви.

Помилка

Помилка нейронної мережі показує наскільки результат роботи моделі відрізняється від значення яке модель мала б повертати. Чим менша помилка моделі, тим правильніший результат моделі на цих вхідних даних.

Навчання

Навчання нейронних мереж відбувається шляхом корекції параметрів які входять у функцію що повертає результат. Ці параметри називаються вагами. Ваги змінюються шляхом мінімізації функції помилки моделі методом градієнтного спуску.

Тренувальні дані

Тренувальні дані (навчальні приклади) для моделей навчання із вчителем виглядають як «вхідні дані» - «правильний результат». Набір із багатьох навчальних прикладів називають датасетом.

Тренування

Під час тренування, усі приклади з датасету проходять через нейронну мережу, для кожного прикладу обчислюється помилка моделі, відбувається коригування вагів. Для отримання потрібної точності, датасет подають моделі не один раз. Кожне використання того ж датасету називають епохою.

# Частина 1

Завдання: розробити програмне забезпечення для реалізації класичного

нейрону. Передбачити режим навчання на одному навчальному прикладі та

режим розпізнавання.

Сама нейронна мережа реалізована у вигляді классу, в якому передбачено:

-Ініціалізацію початкових вагових коефіцієнтів.

-Реалізацію сигмоїдальної функції та її похідної.

-Функцію train(), яка відповідає за навчання нашої нейронної мережі.

-Функцію активації activate().

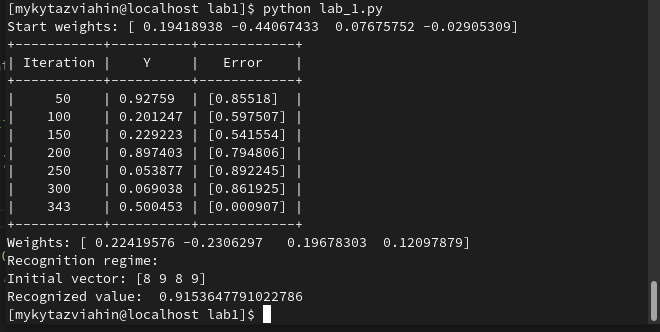
Найбільш інтересною для нас є функція train():



Вона пропускає через функцію активації вхідні тренувальні дані, після чого обчислює помилку, та додає вихідне значення та помилку до масиву для їх збереження, після чого якщо помилка недостатньо мала, починається зворотній хід, та за допомогою методу градієнтного спуску корегуються ваги.

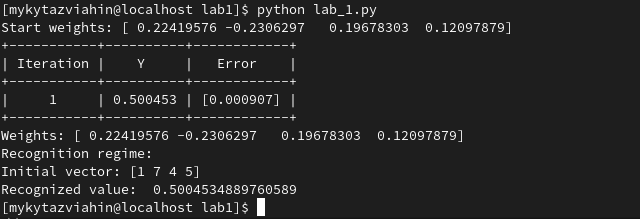
Далі дані додаються в таблицю для виведення на екран, та починається наступна ітерація тренування.

Результат роботи першої частини:



Зазначу, що до цього вже було зроблено 1000 ітерацій, але для компактності я залишив другу спробу навчання.

Також приклад з використання навчального вектора, як тестового:



Частина 2.

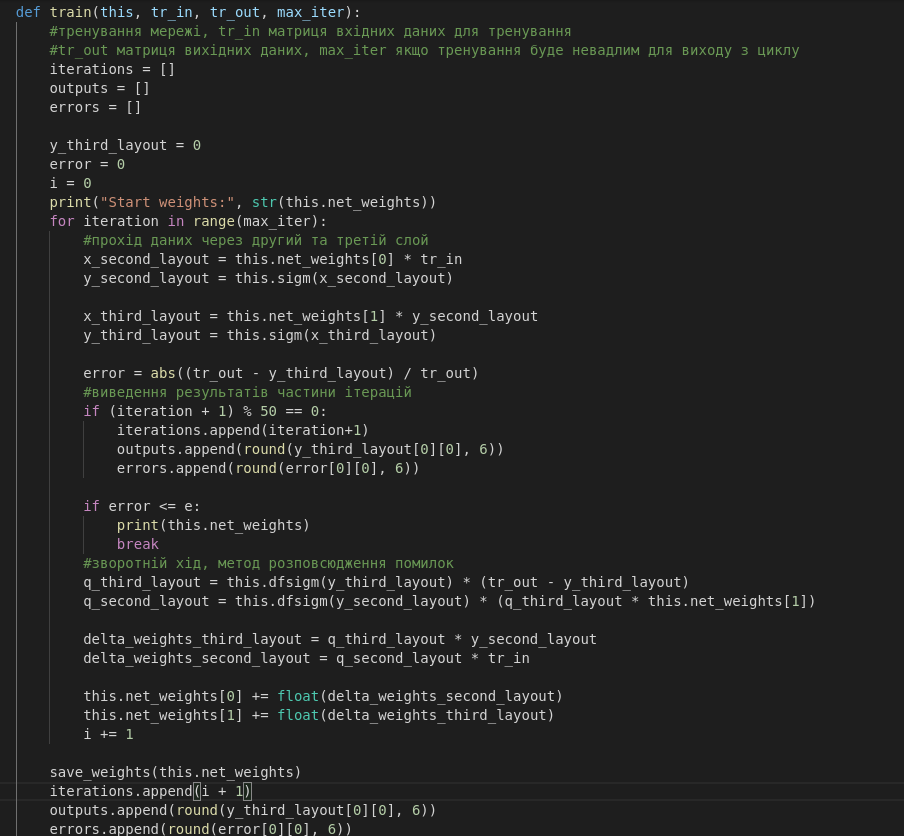
Завдання: розробити програмне забезпечення для реалізації елементарного

двошарового персептрону із структурою 1-1-1. Передбачити режим навчання

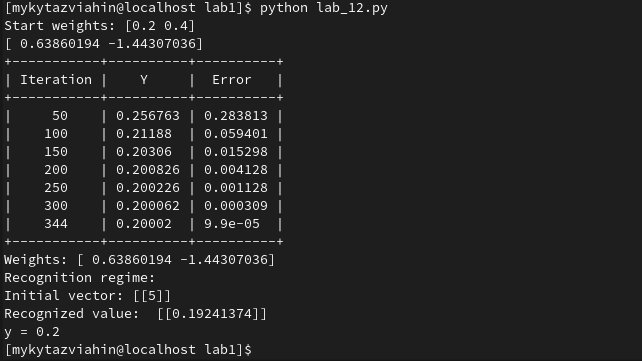
на одному навчальному прикладі та режим розпізнавання.

Структура програми схожа на першу частину, для функції активації також використовується сигмоїдальна функція. На вхід подається вектор довжиною 1.

Після цього в функції train() реалізовано три слоя нейронної мережі. Коли дані проходять через всі слої обчислюється похибка, та частина значень буде виводитись на екран. Вихідні значення проходять через функцію активації, Після цього корегуються ваги.



Результат роботи програми:



Як бачимо, розпізнаний образ дорівнює 0.192, що досить близко до 0.2.

Третя частина.

Завдання: розробити програмне забезпечення для реалізації двошарового

персептрону із структурою 2-3-1. Передбачити режим навчання «ON-LINE»

та режим розпізнавання.

Піддослідна функція х1+х2=у

Реалізуємо двошаровий персептрон, використаємо функції з попередніх частин

роботи, такі як:

-sigm

-dfsigm

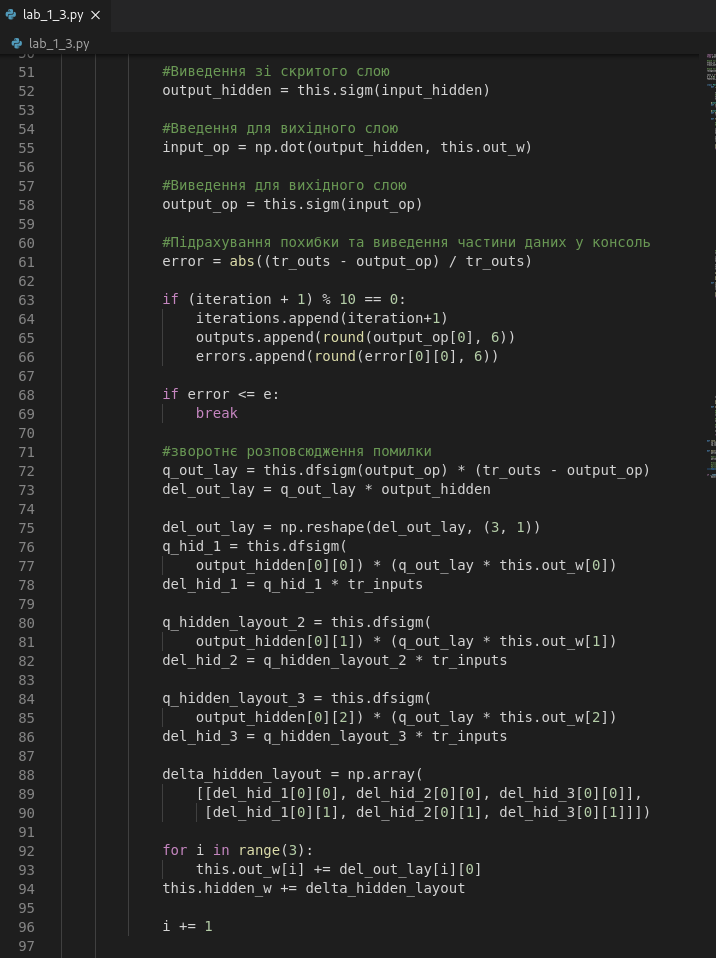
-train()

-activate

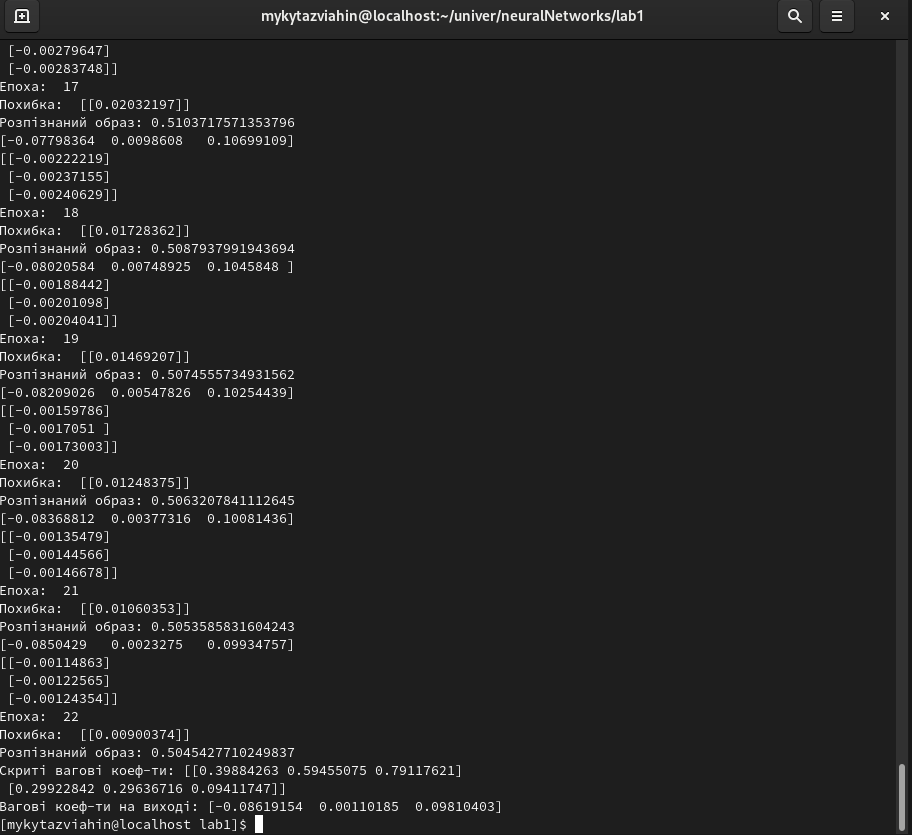
-save\_result

В функції train() буде реалізовано персептрон структури 2-3-1:

Прохід даних через скриті шари нейронної мережі в наступному коді це q\_hidden\_layout\_1, q\_hidden\_layout\_2, q\_hidden\_layout\_3.



Тренування та результат роботи програми:



Висновок:

Було реалізовано нейронні мережі наступних видів:

Класичний нейрон структури 4-1.

Бажаний результат: 0.9

Отриманий результат: 0.915

Персептрон структури 1-1-1.

Бажаний результат: 0.2

Отриманий результат: 0.192

Персептрон структури 2-3-1.

Бажаний результат: 0.5

Отриманий результат: 0.505

Результат було отримано за 22 епохи. (тренування на одному тренувальному

наборі даних).