**Міністерство освіти та науки України**

**Державний вищий навчальний заклад “Київський національний економічний університет ім. Вадима Гетьмана”**

**Кафедра інформаційних систем в економіці**

**Дисципліна “** **Нейромережі та нейроконтроллери”**

**ЗВІТ**

**з лабораторної роботи №6(8)**

**«Градієнт. Градієнтний спуск»**

**Підготував:**

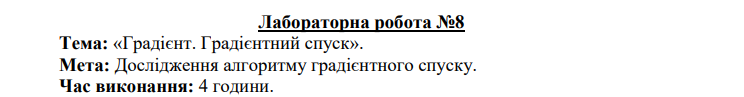
студент 5 курсу гр. ІШІ-501

спеціальності «8і09»

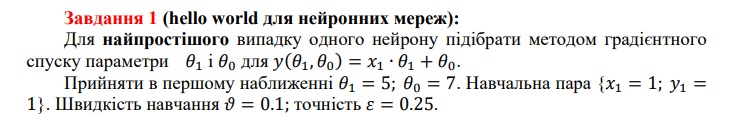
Летич А.А.

**Прийняв:**

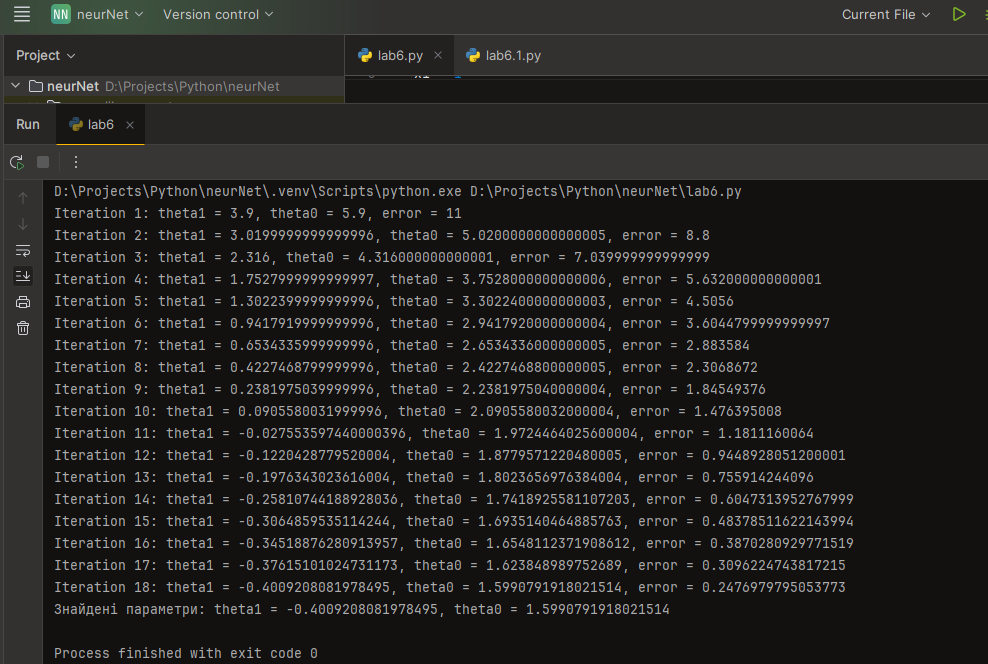
Маєвський О.В.



Посилання на github: <https://github.com/SubAtom131/Neural-networks>

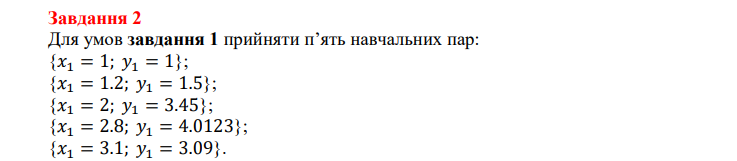


Результат роботи програми:

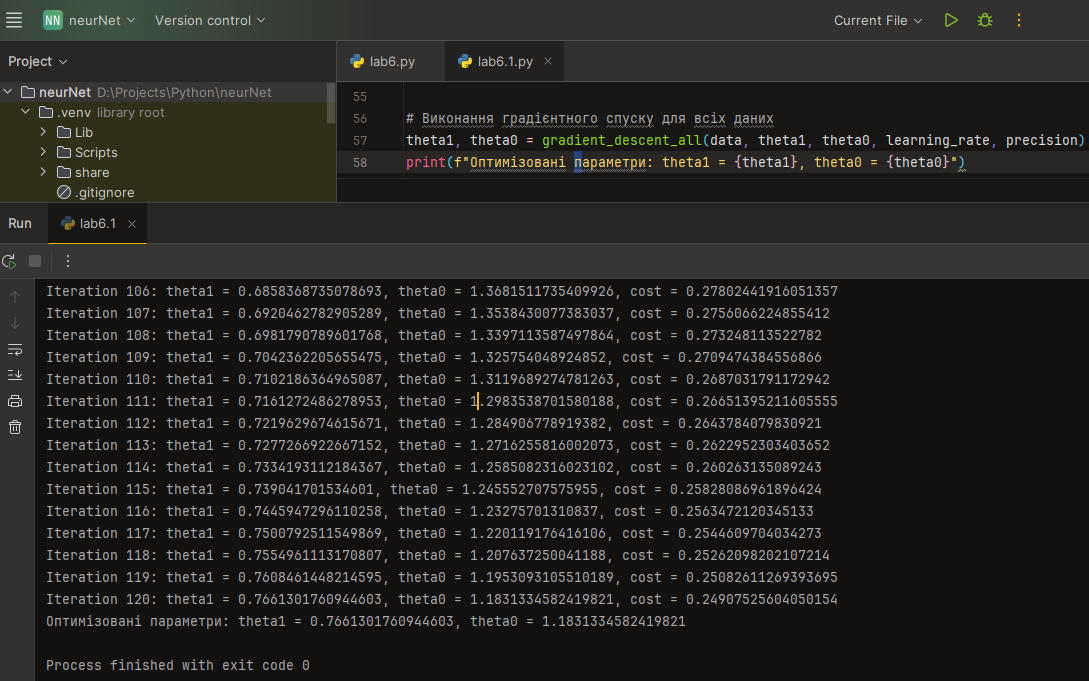


Текст програми:

# Ініціалізація параметрів  
theta1 = 5  
theta0 = 7  
  
# Навчальна пара  
x1 = 1  
y1 = 1  
  
# Швидкість навчання  
alpha = 0.1  
  
# Точність  
epsilon = 0.25  
  
  
# Функція для передбачення значення y  
def predict(x, theta1, theta0):  
 return x \* theta1 + theta0  
  
  
# Функція для обчислення похибки  
def compute\_error(y\_pred, y\_true):  
 return y\_pred - y\_true  
  
  
# Функція для оновлення параметрів theta1 і theta0  
def update\_parameters(theta1, theta0, x, y\_true, alpha):  
 y\_pred = predict(x, theta1, theta0)  
 error = compute\_error(y\_pred, y\_true)  
 theta1 -= alpha \* error \* x  
 theta0 -= alpha \* error  
 return theta1, theta0, error  
  
  
# Головний цикл градієнтного спуску  
iterations = 0  
while True:  
 iterations += 1  
 theta1, theta0, error = update\_parameters(theta1, theta0, x1, y1, alpha)  
 print(f'Iteration {iterations}: theta1 = {theta1}, theta0 = {theta0}, error = {error}')  
  
 # Умови завершення  
 if abs(error) <= epsilon:  
 break  
  
print(f'Знайдені параметри: theta1 = {theta1}, theta0 = {theta0}')



Результат роботи програми:



Текст програми:

import numpy as np  
  
# Навчальні дані  
data = np.array([  
 [1, 1],  
 [1.2, 1.5],  
 [2, 3.45],  
 [2.8, 4.0123],  
 [3.1, 3.09]  
])  
  
  
# Функція гіпотези  
def hypothesis(x, theta1, theta0):  
 return x \* theta1 + theta0  
  
  
# Функція вартості для всіх даних  
def cost\_function\_all(data, theta1, theta0):  
 total\_cost = 0  
 for i in range(len(data)):  
 x = data[i, 0]  
 y = data[i, 1]  
 total\_cost += (hypothesis(x, theta1, theta0) - y) \*\* 2  
 return total\_cost / (2 \* len(data))  
  
  
# Градієнтний спуск для всіх даних  
def gradient\_descent\_all(data, theta1, theta0, learning\_rate, precision):  
 cost = cost\_function\_all(data, theta1, theta0)  
 iterations = 0  
  
 while cost > precision:  
 sum1 = 0  
 sum0 = 0  
 for i in range(len(data)):  
 x = data[i, 0]  
 y = data[i, 1]  
 sum1 += (hypothesis(x, theta1, theta0) - y) \* x  
 sum0 += (hypothesis(x, theta1, theta0) - y)  
 theta1 -= learning\_rate \* sum1 / len(data)  
 theta0 -= learning\_rate \* sum0 / len(data)  
 cost = cost\_function\_all(data, theta1, theta0)  
 iterations += 1  
 print(f"Iteration {iterations}: theta1 = {theta1}, theta0 = {theta0}, cost = {cost}")  
  
 return theta1, theta0  
  
  
# Початкові параметри  
theta1 = 5  
theta0 = 7  
learning\_rate = 0.1  
precision = 0.25  
  
# Виконання градієнтного спуску для всіх даних  
theta1, theta0 = gradient\_descent\_all(data, theta1, theta0, learning\_rate, precision)  
print(f"Оптимізовані параметри: theta1 = {theta1}, theta0 = {theta0}")