ЛАБОРАТОРНА РОБОТА № 1

Нейронна реалізація логічних функцій AND, OR, XOR

Мета роботи: Дослідити математичну модель нейрона.

Завдання №1:

Реалізувати обчислювальний алгоритм для функції xor(x1, x2) через функції or(x1, x2) і and(x1, x2) в програмному середовищі (C++, Python, та ін.).

Для реалізації обчислювальних алгоритмів рекомендується використання онлайн середовищ тестування (наприклад repl.it, trinket, і.т.д.).

```python
def AND(x1, x2):
    return x1 and x2
def OR(x1, x2):
    return x1 or x2
def NOT(x):
    return not x
# Реалізація функції XOR через OR і AND
def XOR(x1, x2):
    return OR(AND(x1, NOT(x2)), AND(NOT(x1), x2))

# Тестування функції XOR
print(XOR(0, 0))
print(XOR(0, 1))
print(XOR(1, 0))
print(XOR(1, 1))
```

Результат

```
0
1
True
False
```

Завдання №2:

Зобразити двохслойний персептрон для функції xor(x1, x2) та скласти відповідне рівняння розділяючої прямої, використовуючи теоретичний матеріал даної лабораторної роботи.

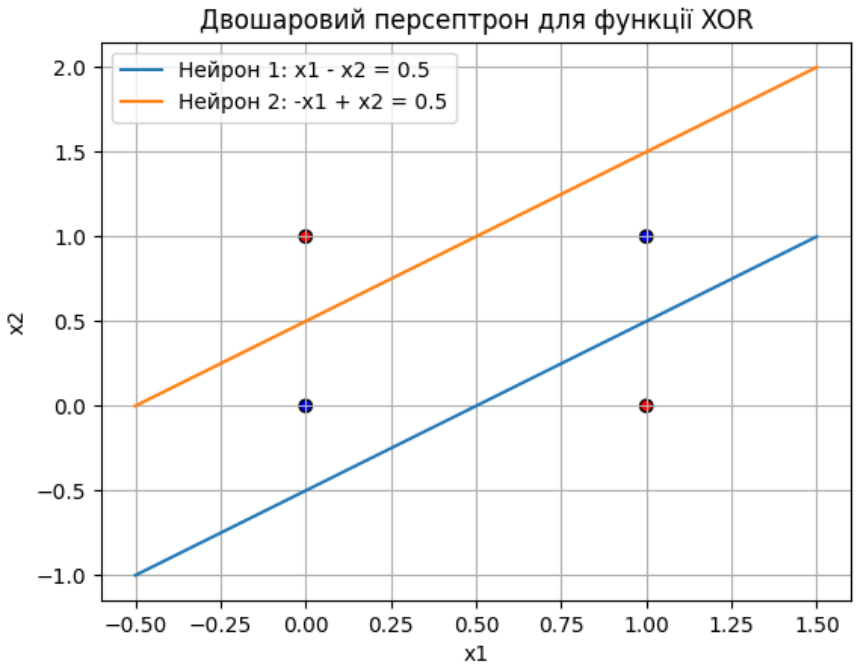
Захист лабораторної роботи передбачає виконання практичних завдань поставлених в роботі, та виконання завдань теоретичного характеру.

					ДУ «Житомирська політехніка».24.123.15.0 – Лр1		
Змн.	Арк.	№ докум.	Підпис	Дата			
Розроб.		Тарнопольський			Звіт з лабораторної роботи	Літ.	Арк. 1 / Аркушів
Перевір.		Маєвський О.В.					
Керівник						ФІКТ Гр. КІ-21-1	
Н. контр.							
Зав. каф.							

```python
import numpy as np

def step_function(x):
    return np.where(x >= 0, 1, 0)

inputs = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])
# Встановлюємо ваги та зміщення для прихованого шару
weights_hidden = np.array([[1, -1], [-1, 1]])
bias_hidden = np.array([-0.5, -0.5])
# Встановлюємо ваги та зміщення для вихідного шару
weights_output = np.array([1, 1])
bias_output = -0.5
# Обчислення значень нейронів у прихованому шарі та вихідному шарі
def forward_pass(x):
    hidden_input = np.dot(x, weights_hidden) + bias_hidden
    hidden_output = step_function(hidden_input)
    final_input = np.dot(hidden_output, weights_output) + bias_output
    final_output = step_function(final_input)
    return final_output
# Візуалізація рівнянь розділяючих прямих
def plot_decision_boundary():
    import matplotlib.pyplot as plt
    x = np.linspace(-0.5, 1.5, 400)
    y1 = (0.5 - 1 * x) / -1
    y2 = (0.5 - (-1) * x) / 1
    plt.plot(x, y1, label='Нейрон 1: x1 - x2 = 0.5')
    plt.plot(x, y2, label='Нейрон 2: -x1 + x2 = 0.5')
    plt.scatter(inputs[:, 0], inputs[:, 1], c=[forward_pass(x) for x in inputs],
cmap='bwr', edgecolor='k')
    plt.xlabel('x1')
    plt.ylabel('x2')
    plt.title('Двошаровий персептрон для функції XOR')
    plt.legend()
    plt.grid(True)
    plt.show()
plot_decision_boundary()
```



Висновки: в ході виконання лабораторної роботи було досліджено та отримано знання, уміння та навики, щодо особливостей дослідження математичної моделі нейрона.