Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №4**

По дисциплине «Модели решения задач в интеллектуальных системах»

Тема: «MLP. Задача исключающего или»

**Выполнил:**

Студент 3 курса

Группы ИИ-21

Литвинюк Т. В.

**Проверил:**

Туз И. С.

Брест 2023

**Цель:** Написать нейронную сеть (multilayer perceptron с одним скрытым слоем) решающую задачу исключающее или.

**Ход работы**

import numpy as np

# Генерация датасета XOR

np.random.seed(0)

X = np.array([[0, 0], [0, 1], [1, 0], [1, 1]]) # входные данные

y = np.array([0, 1, 1, 0]) # выходные данные (результат XOR)

# Функция активации - сигмоида

def sigmoid(x):

return 1 / (1 + np.exp(-x))

# Производная функции активации сигмоиды

def sigmoid\_derivative(x):

return x \* (1 - x)

# Настройка параметров сети

input\_neurons = X.shape[1] # количество входных нейронов

hidden\_neurons = 4 # количество нейронов в скрытом слое

output\_neurons = 1 # количество выходных нейронов

learning\_rate = 0.1 # коэффициент обучения

# Инициализация весов сети случайными значениями

weights\_input\_hidden = np.random.uniform(size=(input\_neurons, hidden\_neurons))

weights\_hidden\_output = np.random.uniform(size=(hidden\_neurons, output\_neurons))

# Обучение нейронной сети

epochs = 10000 # количество эпох обучения

for epoch in range(epochs):

# Прямое распространение (forward propagation)

hidden\_input = np.dot(X, weights\_input\_hidden)

hidden\_output = sigmoid(hidden\_input)

output\_input = np.dot(hidden\_output, weights\_hidden\_output)

predicted\_output = sigmoid(output\_input)

# Вычисление ошибки

error = y.reshape(-1, 1) - predicted\_output

# Обратное распространение (backpropagation)

# Вычисление градиентов и коррекция весов

output\_error = error \* sigmoid\_derivative(predicted\_output)

hidden\_layer\_error = output\_error.dot(weights\_hidden\_output.T) \* sigmoid\_derivative(hidden\_output)

weights\_hidden\_output += hidden\_output.T.dot(output\_error) \* learning\_rate

weights\_input\_hidden += X.T.dot(hidden\_layer\_error) \* learning\_rate

# Проверка обученной сети

test\_input = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])

predicted\_output = sigmoid(np.dot(sigmoid(np.dot(test\_input, weights\_input\_hidden)), weights\_hidden\_output))

# Вывод результатов

print("Предсказанные значения для входных данных XOR:")

print(predicted\_output.ravel())

****

**Вывод:** в ходе лабораторной работы я создал ИНС для решения задачи XOR.