**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования

САМАРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт автоматики и информационных технологий

Кафедра «Информатика и Вычислительная техника»

**ОТЧЕТ**

**о выполнении лабораторной работы №5**

по дисциплине Компьютерные средства искусственного интеллекта

на тему Нейросеть своими руками

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Преподаватель** |  |  |  | А.А. Тюгашев |
|  | (должность) | (подпись) | (дата) | (инициалы, фамилия)  К.В. Портнов |
|  | (должность) | (подпись) | (дата) | (инициалы, фамилия) |
| **Студенты** | 4-ИАИТ-119 |  |  | Е.А. Щаев |
|  | (группа) | (подпись) | (дата) | (инициалы, фамилия) |

Самара 2024 г.

**Цель работы –** Разработать нейросеть не использую библиотеки.

**Программный код на языке Python:**

import numpy as np

# Сигмоида и её производная

def sigmoid(x):

    return 1 / (1 + np.exp(-x))

def sigmoid\_derivative(x):

    return x \* (1 - x)

# Входные данные: [ночная температура, уровень влажности]

X = np.array([[10, 0.8],   # Ночью 10°C, влажность 80%

              [15, 0.6],   # Ночью 15°C, влажность 60%

              [20, 0.4],   # Ночью 20°C, влажность 40%

              [25, 0.2]])  # Ночью 25°C, влажность 20%

# Выходные данные: дневная температура

y = np.array([[15],   # 15°C

              [20],   # 20°C

              [25],   # 25°C

              [30]])  # 30°C

# Нормализация данных (от 0 до 1)

X = X / np.max(X, axis=0)

y = y / 30

# Размеры слоев

input\_layer\_size = 2

hidden\_layer\_size = 4

output\_layer\_size = 1

# Скорость обучения и количество эпох

learning\_rate = 0.1

epochs = 10000

# Инициализация весов

np.random.seed(42)

weights\_input\_hidden = np.random.rand(input\_layer\_size, hidden\_layer\_size)

weights\_hidden\_output = np.random.rand(hidden\_layer\_size, output\_layer\_size)

# Обучение сети

for epoch in range(epochs):

    # Прямой проход

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

    hidden\_layer\_output = sigmoid(hidden\_layer\_input)

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

    predicted\_output = sigmoid(output\_layer\_input)

    # Ошибка

    error = y - predicted\_output

    # Обратное распространение

    d\_predicted\_output = error \* sigmoid\_derivative(predicted\_output)

    weights\_hidden\_output += np.dot(hidden\_layer\_output.T, d\_predicted\_output) \* learning\_rate

    hidden\_layer\_error = np.dot(d\_predicted\_output, weights\_hidden\_output.T)

    d\_hidden\_layer\_output = hidden\_layer\_error \* sigmoid\_derivative(hidden\_layer\_output)

    weights\_input\_hidden += np.dot(X.T, d\_hidden\_layer\_output) \* learning\_rate

# Де-нормализация предсказаний

predicted\_output = predicted\_output \* 30

# Результаты

print("Обучение завершено.")

print("Прогнозируемые дневные температуры:")

for i, pred in enumerate(predicted\_output):

    print(f"Input (ночь={X[i][0] \* 30}°C, влажность={X[i][1] \* 100}%): "

          f"Predicted дневная температура = {pred[0]:.2f}°C")

# Функция тестирования сети

def test\_temperature\_model(test\_input):

    test\_input = test\_input / np.max(X, axis=0)  # Нормализация

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

    hidden\_layer\_output = sigmoid(hidden\_layer\_input)

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

    predicted\_output = sigmoid(output\_layer\_input)

    return predicted\_output \* 30  # Де-нормализация

print(f"Weights input hidden = {weights\_input\_hidden}\nWeight hidden output = {weights\_hidden\_output}")

# Тестирование

new\_data = np.array([[18, 0.5], [22, 0.3]])  # Новые данные

predictions = test\_temperature\_model(new\_data)

for i, pred in enumerate(predictions):

    print(f"Test Input (ночь={new\_data[i][0]}°C, влажность={new\_data[i][1] \* 100}%): "

          f"Predicted дневная температура = {pred[0]:.2f}°C")

**Пример работы программы:**

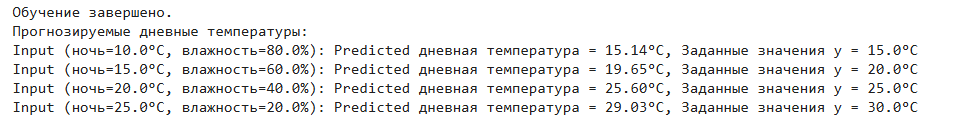
****

Рисунок 1 – Проверка нейросети на входных данных

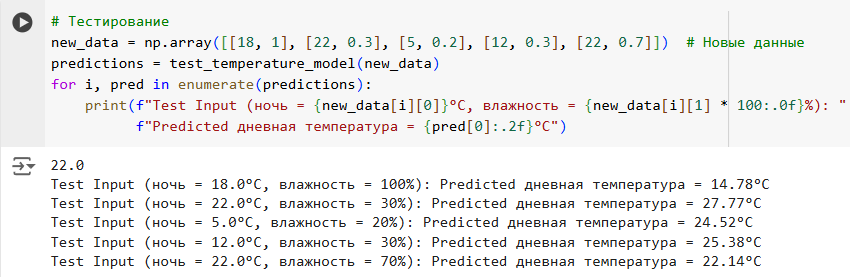


Рисунок 2 – Тестирование

**Вывод:** нейросеть принимает на вход определенные значения, обрабатывает их через скрытые слои с помощью весов и смещений, применяет функции активации и затем выдает выходной результат.