**Министерство образования Российской Федерации**

**Пензенский государственный университет**

**Кафедра „Вычислительная техника“**

**ОТЧЕТ**

по лабораторной работе №2

по курсу „Основы python обучение нейрона“

Выполнили студенты группы 23ВВВ1 :

Казаков А.Д.

Приняли:

д.т.н. доцент Митрохин М.А.

к.т.н. доцент Юрова О.В.

Пенза 2025

**Задание 1:**

# -\*- coding: utf-8 -\*-

from random import \*

mass = []

for i in range(10):

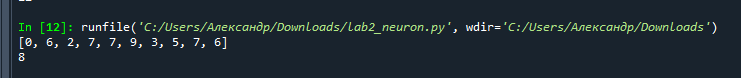
mass.append(randint(0, 10))

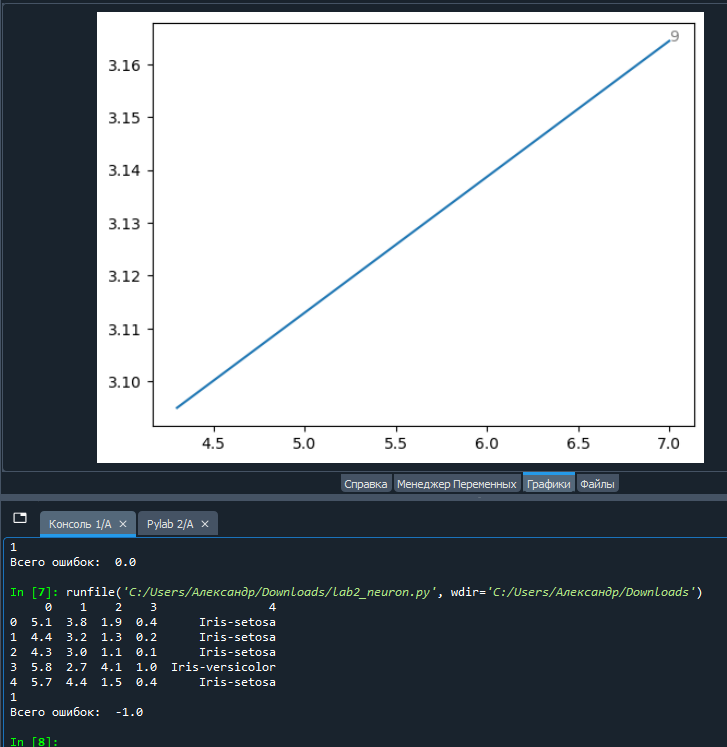
a = 0

for i in mass:

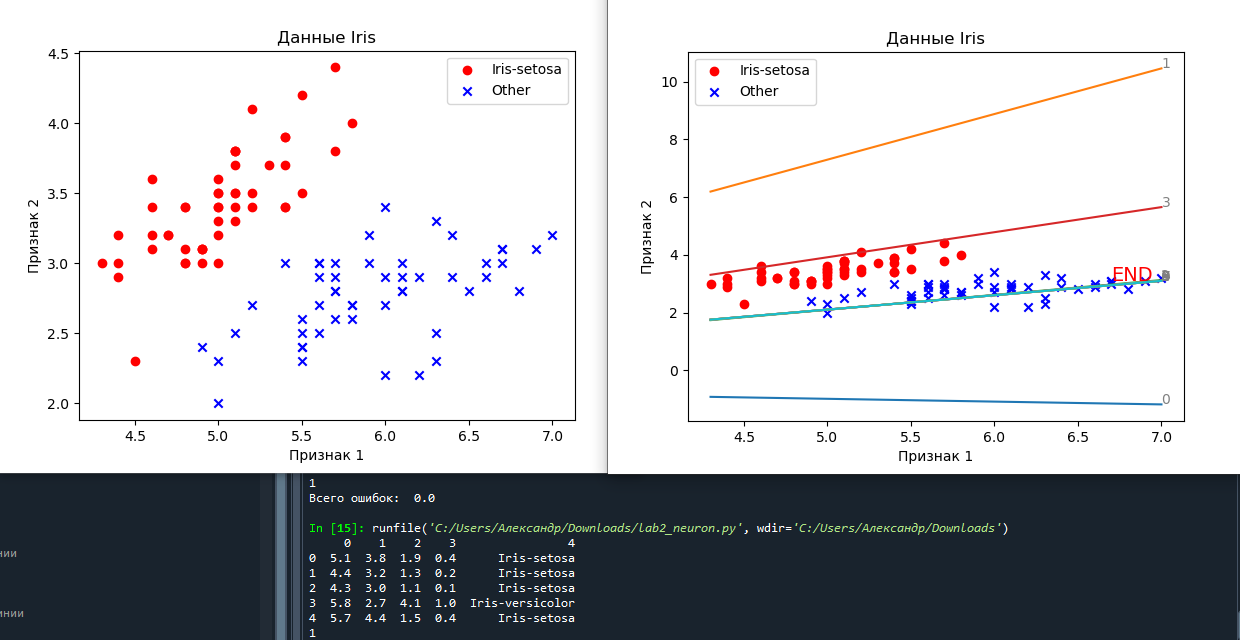
if (mass[i]%2 == 0): a+= mass[i]

print(a)





**Задание 2:**



import pandas as pd # библиотека pandas нужна для работы с данными

import matplotlib.pyplot as plt # matplotlib для построения графиков

import numpy as np # numpy для работы с векторами и матрицами

# Считываем данные

df = pd.read\_csv('data.csv')

# смотрим что в них

print(df.head())

# три столбца - это признаки, четвертый - целевая переменная (то, что мы хотим предсказывать)

# выделим целевую переменную в отдельную переменную

y = df.iloc[:, 4].values

# так как ответы у нас строки - нужно перейти к численным значениям

y = np.where(y == "Iris-setosa", 1, -1)

# возьмем три признака, чтобы нейрон работал с тремя признаками

X = df.iloc[:, [0, 1, 2]].values # Изменено на [0, 1, 2]

# Признаки в X, ответы в y - постмотрим на плоскости как выглядит задача

plt.figure()

plt.scatter(X[y == 1, 0], X[y == 1, 1], color='red', marker='o', label='Iris-setosa')

plt.scatter(X[y == -1, 0], X[y == -1, 1], color='blue', marker='x', label='Other')

plt.xlabel('Признак 1')

plt.ylabel('Признак 2')

plt.legend()

plt.title('Данные Iris')

plt.show()

# переходим к созданию нейрона

# функция нейрона:

# значение = w1\*признак1 + w2\*признак2 + w3\*признак3 + w0

# ответ = 1, если значение > 0

# ответ = -1, если значение < 0

def neuron(w, x):

if (w[1] \* x[0] + w[2] \* x[1] + w[3] \* x[2] + w[0]) >= 0:

predict = 1

else:

predict = -1

return predict

# проверим как это работает (веса зададим пока произвольно)

w = np.array([0, 0.1, 0.4, 0.2]) # Изменено на 4 веса

print(neuron(w, X[1])) # вывод ответа нейрона для примера с номером 1

# теперь создадим процедуру обучения

# корректировка веса производится по выражению:

# w\_new = w\_old + eta\*x\*y

# зададим начальные значения весов

w = np.random.random(4) # Изменено на 4 веса

eta = 0.01 # скорость обучения

w\_iter = [] # пустой список, в него будем добавлять веса, чтобы потом построить график

for xi, target, j in zip(X, y, range(X.shape[0])):

predict = neuron(w, xi)

w[1:] += (eta \* (target - predict)) \* xi # target - predict - это и есть ошибка

w[0] += eta \* (target - predict)

# каждую 10ю итерацию будем сохранять набор весов в специальном списке

if (j % 10 == 0):

w\_iter.append(w.tolist())

# посчитаем ошибки

sum\_err = 0

for xi, target in zip(X, y):

predict = neuron(w, xi)

sum\_err += (target - predict) / 2

print("Всего ошибок: ", sum\_err)

# попробуем визуализировать процесс обучения

xl = np.linspace(min(X[:, 0]), max(X[:, 0])) # диапазон координаты x для построения линии

# построим сначала данные на плоскости

plt.figure()

plt.scatter(X[y == 1, 0], X[y == 1, 1], color='red', marker='o', label='Iris-setosa')

plt.scatter(X[y == -1, 0], X[y == -1, 1], color='blue', marker='x', label='Other')

plt.xlabel('Признак 1')

plt.ylabel('Признак 2')

plt.legend()

plt.title('Данные Iris')

plt.show()

# потом в цикле будем брать набор весов из сохраненного списка и по нему строить линии

for i, w in zip(range(len(w\_iter)), w\_iter):

yl = -(xl \* w[1] + w[0]) / w[2] # уравнение линии

plt.plot(xl, yl, label=f'Iteration {i}') # строим разделяющую границу

plt.text(xl[-1], yl[-1], i, dict(size=10, color='gray')) # подписываем номер линии

plt.pause(1)

plt.text(xl[-1] - 0.3, yl[-1], 'END', dict(size=14, color='red'))

plt.show()