Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра интеллектуальных информационных технологий

Лабораторная работа №1

По дисциплине «Модели решения задач в ИС»

Тема: «бинарная классификация»

Выполнил:

Студент 3 курса

Факультета ЭИС

Группы ИИ-26

Сугак В.А.

Проверила:

Андренко К.В.

Брест 2025

Цель работы: изучить принципы бинарной классификации и реализовать однослойную нейронную сеть (персептрон) для решения задачи классификации с использованием пороговой функции активации, а также исследовать процесс обучения модели с применением среднеквадратичной ошибки (MSE).

Задачи лабораторной работы:

1. Реализовать алгоритм обучения однослойной нейронной сети с использованием MSE в качестве функции ошибки.
2. Провести обучение сети с разными значениями шага обучения и построить график зависимости MSE от номера эпохи.
3. Выполнить визуализацию результатов классификации:
   - исходные точки обучающей выборки,
   - разделяющую линию (границу между двумя классами).
4. Реализовать режим функционирования сети:
   - пользователь задаёт произвольный входной вектор,
   - сеть вычисляет выходной класс,
   - соответствующая точка отображается на графике,
   - для корректной визуализации рекомендуется выбирать значения из диапазона вставить свой диапазон, например $-0.5 \leq x1, x2 \leq 1.5$
5. Написать вывод по выполненной работе.

Допускается применение математических и графических библиотек

ML-библиотеки и ML-фреймворки использовать нельзя (например: scikit-learn, TensorFlow, PyTorch - запрещены)

x1, x2 - входные данные сети, e - эталонные значения

Вариант:

x1	x2	e
4	1	1
-4	1	1
4	-1	1
-4	-1	0

Ход работы

Код:

```
import numpy as np

import matplotlib.pyplot as plt

X_RAW = np.array([

    [4.0,  1.0],

    [-4.0,  1.0],

    [4.0, -1.0],

    [-4.0, -1.0],

], dtype=float)

E = np.array([1.0, 1.0, 1.0, 0.0], dtype=float)

SCALE = np.max(np.abs(X_RAW), axis=0)

X = X_RAW / SCALE


def step(v):
```

```
    return 1 if v >= 0.5 else 0

def mse(y, e):
    return np.mean((e - y) ** 2)

class Net:
    def __init__(self, lr):
        rng = np.random.default_rng()
        self.w = rng.uniform(-0.5, 0.5, 2)
        self.b = rng.uniform(-0.5, 0.5)
        self.lr = lr

    def forward(self, x):
        return np.dot(self.w, x) + self.b

    def train_epoch(self):
        for x, e in zip(X, E):
            y = self.forward(x)
            d = e - y
            self.w += self.lr * d * x
            self.b += self.lr * d

        y_all = np.array([self.forward(x) for x in X])
        return mse(y_all, E)

    def predict(self, x):
        return step(self.forward(x))

lrs = [0.01, 0.05, 0.1]
epochs = 100

history = {}
best_net = None
best_mse = 1e9

for learning_rate in lrs:
    net = Net(learning_rate)
    h = []
```

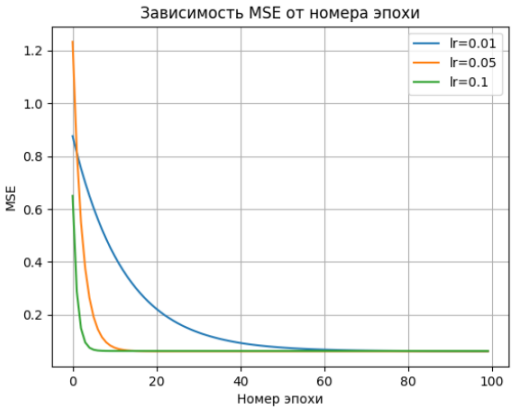
```
    for _ in range(epochs):
        h.append(net.train_epoch())
    history[learning_rate] = h

    if h[-1] < best_mse:
        best_mse = h[-1]
        best_net = net

print("Лучший шаг обучения:", best_net.lr)
print("Веса:", best_net.w)
print("Смещение:", best_net.b)
print("Финальное MSE:", best_mse)

plt.figure()
for lr, h in history.items():
    plt.plot(h, label=f"lr={lr}")
plt.xlabel("Номер эпохи")
plt.ylabel("MSE")
plt.title("Зависимость MSE от номера эпохи")
plt.grid()
plt.legend()
plt.show()

plt.figure()

c0 = X_RAW[E == 0]
c1 = X_RAW[E == 1]

plt.scatter(c0[:, 0], c0[:, 1], s=80, label="Класс 0")
plt.scatter(c1[:, 0], c1[:, 1], s=80, label="Класс 1")

w1 = best_net.w[0] / SCALE[0]
w2 = best_net.w[1] / SCALE[1]
b = best_net.b

xs = np.linspace(-6, 6, 200)
ys = (0.5 - b - w1 * xs) / w2
```

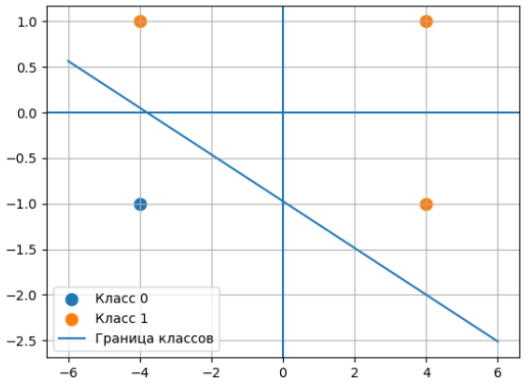
```
plt.plot(xs, ys, label="Граница классов")


plt.axhline(0)
plt.axvline(0)
plt.grid()
plt.legend()
plt.show()


x1, x2 = map(float, input("Введите x1 x2: ").split())


x_user = np.array([x1, x2])
x_user_norm = x_user / SCALE


prediction = best_net.predict(x_user_norm)
print("Класс точки:", prediction)


plt.figure()
plt.scatter(c0[:, 0], c0[:, 1], s=80, label="Класс 0")
plt.scatter(c1[:, 0], c1[:, 1], s=80, label="Класс 1")
plt.plot(xs, ys, label="Граница классов")
plt.scatter(x_user[0], x_user[1], s=120, marker="s", label="Точка пользователя")


plt.axhline(0)
plt.axvline(0)
plt.grid()
plt.legend()
plt.show()
```

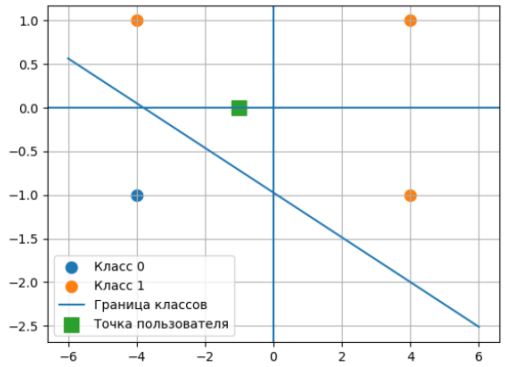
Вывод программы:

```
Лучший шаг обучения: 0.05
Веса: [0.26351351 0.25675676]
Смещение: 0.7499999995402019
Финальное MSE: 0.0627282688056549
Введите x1 x2: -1 0
Класс точки: 1
```

Графики:







Вывод: изучила принципы бинарной классификации и реализовала однослойную нейронную сеть (персептрон) для решения задачи классификации с использованием пороговой функции активации, а также исследовала процесс обучения модели с применением среднеквадратичной ошибки (MSE).