Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №1
По дисциплине «Криптографические методы защиты информации»
Тема: «Бинарная классификация»

Выполнила:
Студент 3 курса
Группы ИИ-26 (2)
Черноиван А.Н.
Проверила:
Андренко К.В.

Брест 2026

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 $-0.5 \leq x_1, x_2 \leq 1.5$
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

10.

x_1	x_2	e
6	6	0
-6	6	0
6	-6	1
-6	-6	0

Код программы:

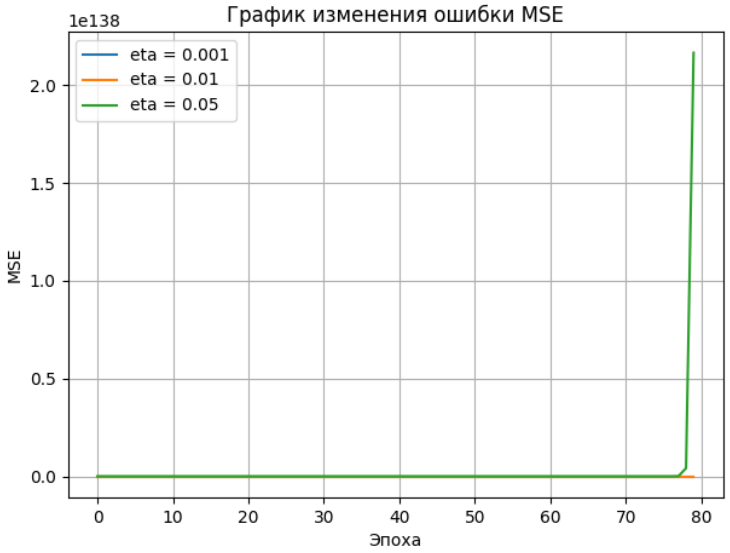
```
import numpy as np
import matplotlib.pyplot as plt

X = np.array([
    [6, 6],
    [-6, 6],
    [6, -6],
    [-6, -6]
], dtype=float)

E = np.array([1 if x[0] > 0 else 0 for x in X], dtype=float)

etas = [0.001, 0.01, 0.05]
epochs = 80
```

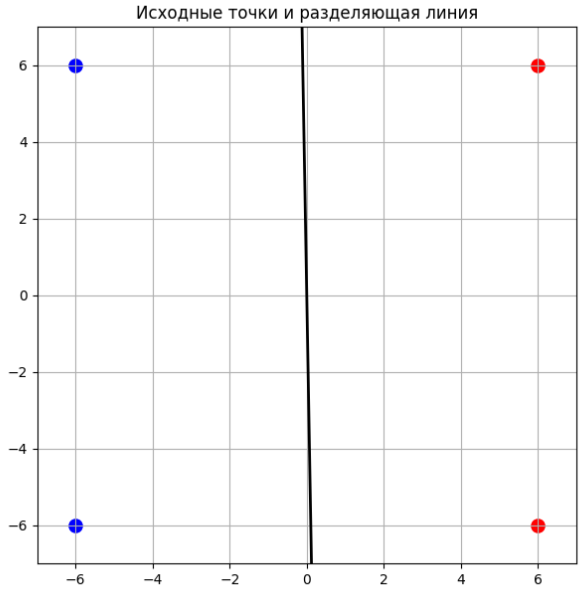
```
all_mse_curves = {}

def train_perceptron(X, E, eta, epochs):
    w = np.random.randn(2)
    T = np.random.randn()

    mse_list = []

    for epoch in range(epochs):
        mse = 0
        for x, e in zip(X, E):

            S = np.dot(w, x) - T
            y_lin = S
            error = e - y_lin
            mse += error**2

            w += eta * error * x
            T -= eta * error

        mse_list.append(mse / len(X))

    return w, T, mse_list


for eta in etas:
    w, T, mse_curve = train_perceptron(X, E, eta, epochs)
    all_mse_curves[eta] = mse_curve

w_vis = w
T_vis = T

plt.figure(figsize=(7, 5))
for eta, curve in all_mse_curves.items():
    plt.plot(curve, label=f"eta = {eta}")
plt.xlabel("Эпоха")
plt.ylabel("MSE")
plt.title("График изменения ошибки MSE")
plt.grid()
plt.legend()
plt.show()
plt.figure(figsize=(7,7))

# обучающие точки
for i in range(len(X)):
    color = "red" if E[i] == 1 else "blue"
    plt.scatter(X[i,0], X[i,1], color=color, s=90)

# разделяющая линия
x_line = np.linspace(-7, 7, 300)
y_line = (T_vis - w_vis[0] * x_line) / w_vis[1]

plt.plot(x_line, y_line, 'k', linewidth=2)

plt.xlim(-7, 7)
```

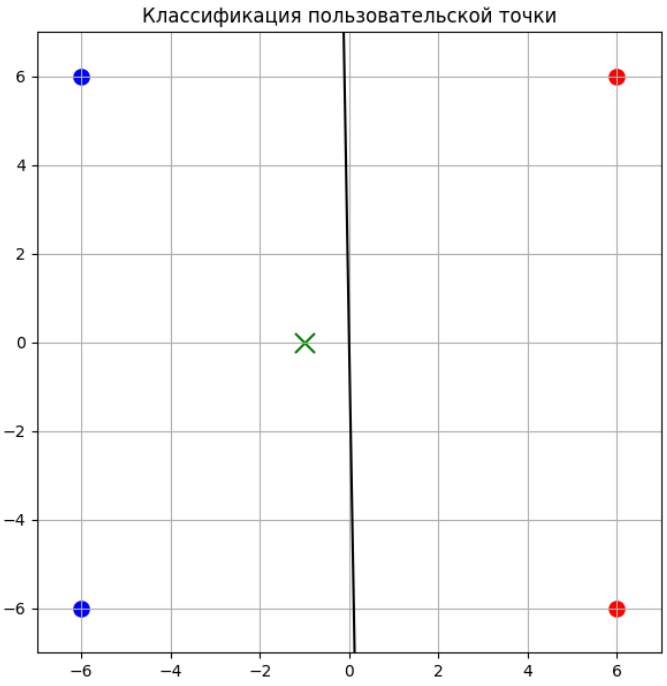
```
plt.ylim(-7, 7)
plt.grid()
plt.title("Исходные точки и разделяющая линия")
plt.show()

def classify_and_plot(x1, x2):
    S = w_vis[0]*x1 + w_vis[1]*x2 - T_vis
    y = 1 if S > 0 else 0

    print("\n========== РЕЗУЛЬТАТ КЛАССИФИКАЦИИ ==========")
    print(f"Введённый вектор: ({x1}, {x2})")
    print(f"Взвешенная сумма S = {S:}")
    print(f"Класс сети: {y}")
    print("============================================\n")

    plt.figure(figsize=(7,7))

    # обучающие точки
    for i in range(len(X)):
        color = "red" if E[i] == 1 else "blue"
        plt.scatter(X[i,0], X[i,1], color=color, s=90)

    # разделяющая линия
    plt.plot(x_line, y_line, 'k')

    # пользовательская точка
    plt.scatter(x1, x2, color="green", s=150, marker="x")

    plt.xlim(-7, 7)
    plt.ylim(-7, 7)
    plt.grid()
    plt.title("Классификация пользовательской точки")
    plt.show()

while True:
    print("\nВведите координаты точки, которую нужно классифицировать.")
    print("Чтобы выйти, введите 'exit'\n")

    x1 = input("x1 = ")
    if x1 == "exit":
        break
    x2 = input("x2 = ")
    if x2 == "exit":
        break

    try:
        x1 = float(x1)
        x2 = float(x2)
        classify_and_plot(x1, x2)
    except:
        print("Ошибка: введите число.")
```

Результат программы:

```
Чтобы выйти, введите 'exit'

x1 = -1
x2 = 0

========== РЕЗУЛЬТАТ КЛАССИФИКАЦИИ ==========
Введённый вектор: (-1.0, 0.0)
Взвешенная сумма S = -1.098575211710201e+69
Класс сети: 0
==========================================


Введите координаты точки, которую нужно классифицировать.
Чтобы выйти, введите 'exit'
```







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).