МГТУ им. Н. Э. Баумана
Факультет ФН «Фундаментальные Науки»
Кафедра ФН-12 «Математическое моделирование»

Численные методы решения задач теории управления

Домашнее задание №1

«Бинарная классификация с помощью перцептрона
(однослойная сеть с пороговой функцией активации)»

Вариант №4

Группа: ФН12-61Б

Студент: Дорохов М. А.

Преподаватель: Тверская Е. С.

Москва, 2024

Постановка задачи

1. Для указанных множеств построить бинарную классификацию, с помощью простейшей нейронной сети с пороговой функцией активации (перцептрон).
2. Построить прямую, разделяющую эти множества, и вычислить результирующие весовые коэффициенты.
3. При тестировании использовать нулевые начальные приближения для весов и смещения.
4. Изменить количество точек множества (уменьшить до 10-15). Рассмотреть работу алгоритма при различных начальных весовых коэффициентах и смещениях. Как влияют эти изменения на результат?

Предложенные по варианту множества:

$$X = \{x_i = (x_i^1, x_i^2),\ i = \overline{1,n}\},\ X \subset \mathbb{R}^2;$$
$$Y = \{x_i = (x_i^1, x_i^2) : (5x_i^1 - 0,9) + (-4x_i^2 - 0,1) > 0,5\}.$$

Ход работы

Построение бинарной классификации

Для построения бинарной классификации для множеств X и Y будем использовать нейронную сеть с пороговой функцией активации (перцептрон). Введём обозначения: X - входные данные, W - матрица весовых коэффициентов, b - смещение, f - функция активации, y - выход.

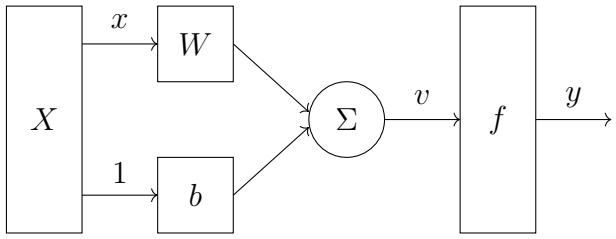


Схема перцептрона

Мы будем использовать пороговую функцию активации:

$$f(x) = \begin{cases} 0,\ x < 0 \\ 1,\ x \geq 0. \end{cases}$$

Алгоритм обучения нейрона

1. Вычисляем $v = Wx_1 + b$, $y_1 = f(v)$;
2. Вычисляем ошибку $e = t_1 - y_1$, где t_1 - предполагаемое значение (таргет), $e \in \{-1, 0, 1\}$;
3. В соответствии в ошибкой меняем весовые коэффициенты и смещение:

$$W_{new} = W_{old} + ex_1,$$
$$b_{new} = b_{old} + e;$$

4. Вычисляем $y_i,\ i = \overline{1,n}$ и проверяем их равенство соответствующим таргетам t_i;
5. В случае несовпадения повторяем алгоритм со следующим индексом.

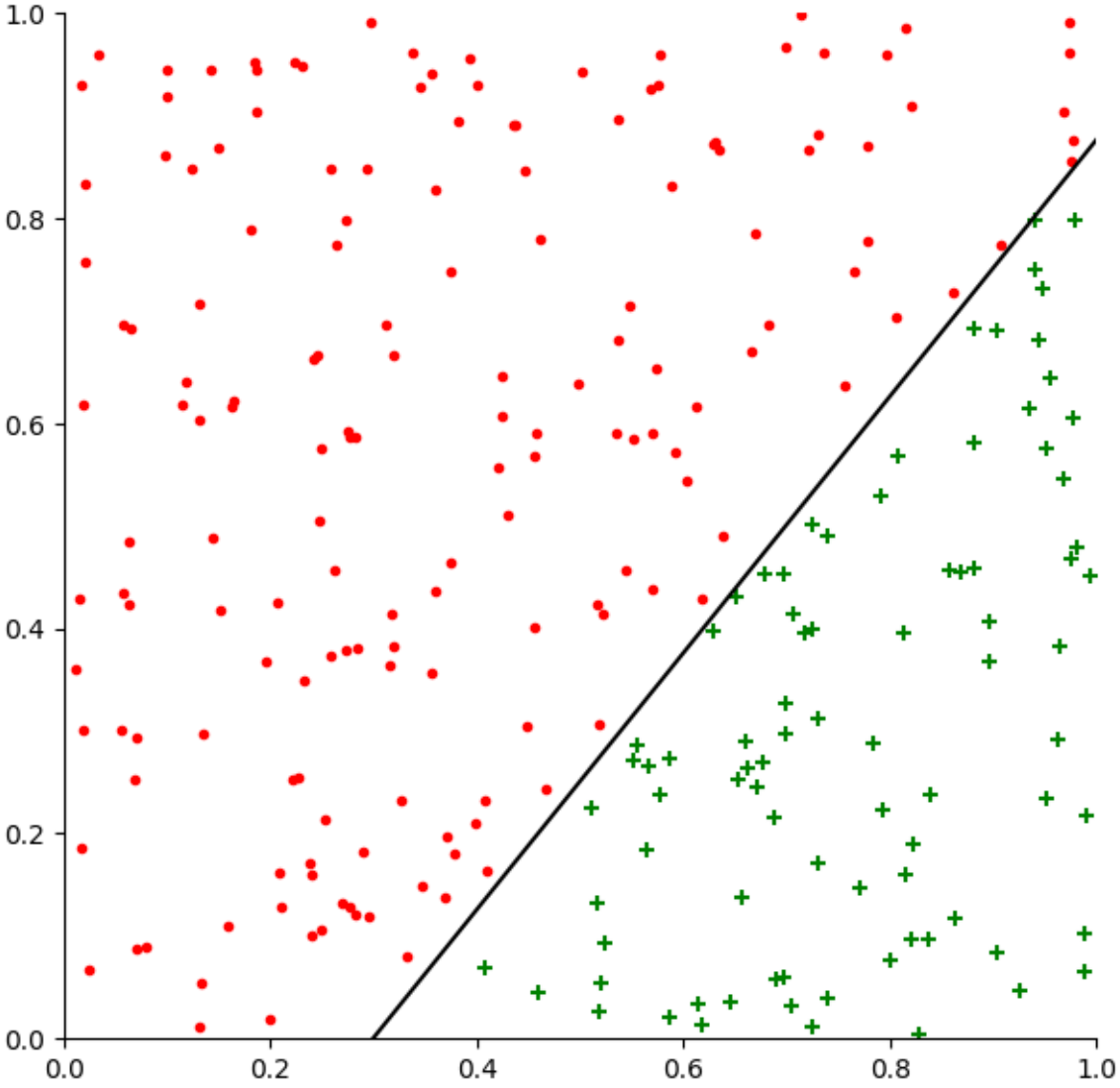
Код программы

```
import numpy as np
import matplotlib.pyplot as plt
np.random.seed(0)
n_samples = 250

X = np.random.rand(n_samples, 2)
Y = (((5*X[:, 0]-0.9))+(-4*X[:, 1]-0.1) > 0.5)

W = np.array([0, 0])
b = 0
j = 0
f = lambda v: 0 if v < 0 else 1
iterations = 0

def check(W, X, b, Y, n_samples):
    global j
    for i in range(j, j + n_samples):
        v = np.dot(W, X[i%n_samples]) + b
        if (Y[i%n_samples] - f(v) != 0):
            j = i
            return False
    return True

while True:
    iterations += 1
    i = j % n_samples
    v = np.dot(W, X[i]) + b
    e = Y[i] - f(v)
    W = W + e*X[i]
    b = b + e
    j += 1
    if check(W, X, b, Y, n_samples):
        break

print(W, b, j)

x = np.arange(-10, 10, 0.1)
fig = plt.figure(figsize=(10, 10))
ax = plt.gca()
for i in range(0, n_samples):
    if (Y[i]):
        plt.scatter(X[i, 0], X[i, 1], color = "g", marker = "+")
    else:
        plt.scatter(X[i, 0], X[i, 1], color = "r", marker = "-")
y = - (W[0] / W[1]) * x - (b / W[1])
plt.plot(x, y, color = "black")
plt.ylim([0, 1])
plt.xlim([0, 1])
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
```

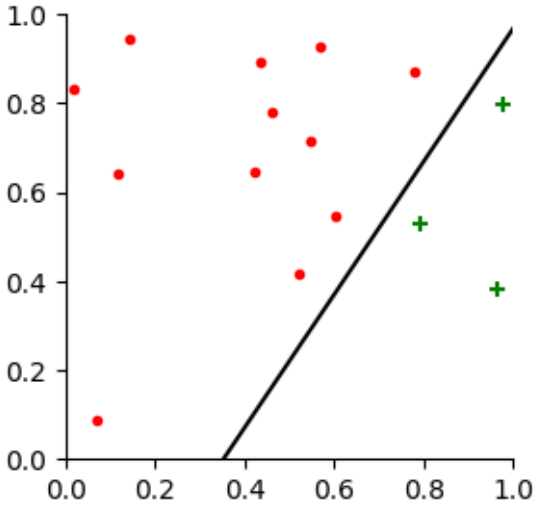
Результаты

j - число итераций, зелеными плюсами обозначены точки множества Y, красными кругами - точки не вошедшие в это множество.

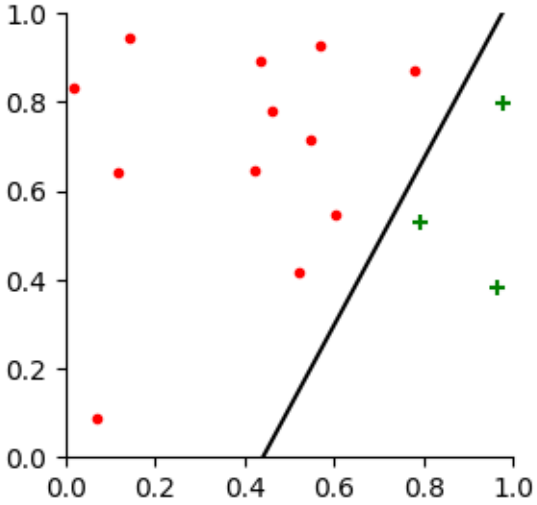
- $W = (0, 0),\ b = 0,\ n = 250,$
  Результат обучения: $W = (13.35718718, -10.67642379),\ b = -4,\ j = 364.$



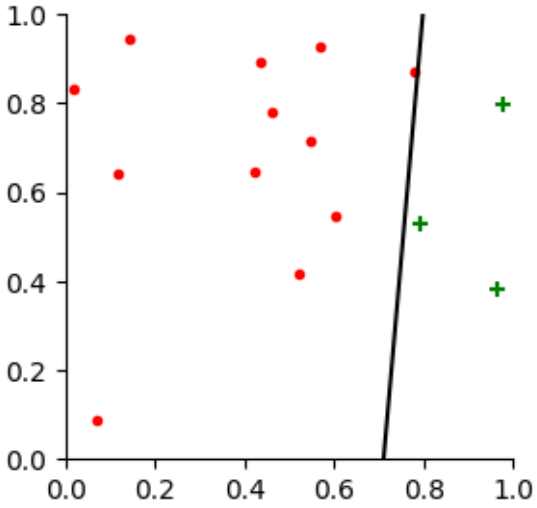
- $W = (1, 0.4),\ b = 2,\ n = 15,$
  Результат обучения: $W = (2.83173156, -1.89261137),\ b = -1,\ j = 20.$



- $W = (3, -1.5),\ b = -1,\ n = 15,$
  Результат обучения: $W = (4.53204706, -2.42228246),\ b = -2,\ j = 11.$



- $W = (10, 10),\ b = -10,\ n = 15,$
  Результат обучения: $W = (14.06718955, -1.23371028),\ b = -10,\ j = 64.$



- $W = (18.01, -5.13),\ b = -11,\ n = 15,$
  Результат обучения: $W = (18.01, -5.13),\ b = -11,\ j = 1.$