**Федеральное агентство по образованию**Государственное образовательное учреждение высшего профессионального   
образования **«Тихоокеанский Государственный университет»**

Факультет компьютерных и фундаментальных наук

Кафедра ПОВТАС

**Лабораторная работа №1**

по дисциплине: «Методы машинного обучения»

на тему: «Расчет коэффициентов разделяющей линии и вычисление отступа (margin) для объектов разных классов»  
Вариант №4

Выполнил: студент группы ПИИ(м)-21

Забавин А.С.

Проверил: преподаватель кафедры ПОВТАС

Тормозов В.С.

# Постановка задачи

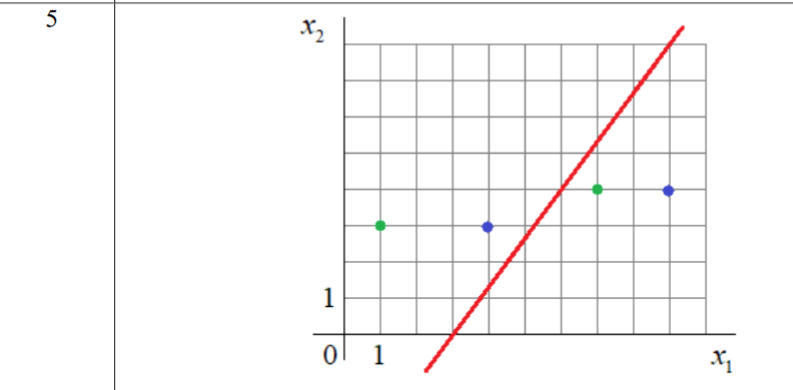
**Цель работы**: научиться вычислять коэффициенты разделяющей линии и  
величину отступа (margin) при бинарной классификации объектов.

**Задания на лабораторную работу (5 вариант)**

1. Используя рисунок своего варианта, необходимо вычислить коэффициенты разделяющей линии, которая определяется выражением:
2. Вычислить отступы (margin) для зеленых точек (с меткой класса -1) и синих точек (с меткой класса +1). Напомню, что отступ вычисляется по формуле:

где - метка класса образа (точки) ; - скалярное произведение векторов.

Вектор должен быть подобран так, чтобы для наиболее удаленных точек от разделяющей линии отступ был положительным, а для ближних - отрицательным.

Рис. 1. График для 5 варианта

# Краткая теория

Прогноз линейной регрессионной модели определяется уравнением:

где – спрогнозированное значение, – количество признаков, – значение -того признака, – -тый параметр модели (включая член смещения и веса признаков ).

Обучение модели означает установку её параметров так, чтобы модель была наилучшим образом подогнана к обучающему набору. Для этой цели первым делом нужна мера того, насколько хорошо (или плохо) модель подогнана к обучающим данным. Самой распространенной мерой производительности регрессионной модели является квадратный корень из среднеквадратической ошибки (RMSE). Однако в данной лабораторной работе не стоит задача определения наилучших параметров, поэтому мы не будем вычислять RMSE.

Подбор гиперпараметров будет осуществляться с использованием ступенчатой функции Хевисайда, хорошо подходящей для задач бинарной классификации:

На Рис. 2 для модели хорошо видно, что радиус-векторы точек для класса образуют острый угол с вектором . Т.к. скалярное произведение – это, фактически, вычисление косинуса угла между векторами, то для острых углов будут получены положительные значения. Точки же класса образуют тупые углы с вектором . Следовательно, для будут получены отрицательные значения.

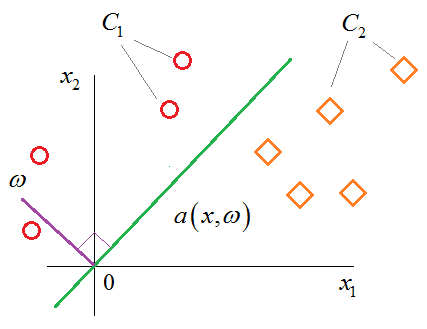


Рис. 2. Демонстрация принципа классификации функцией Хевисайда

В итоге, алгоритм классификации образов с помощью модели можно записать в виде:

Здесь – знаковая функция, которая возвращает +1 для положительных чисел и -1 – для отрицательных:

Математически расчет оптимальной линии разделение осуществляется по минимизации функционала качества Q:



(1)

, где M – отступ точки от линии разделения

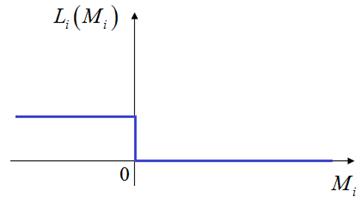
(2)

В качестве функции потерь используется кусочно-непрерывная функция L

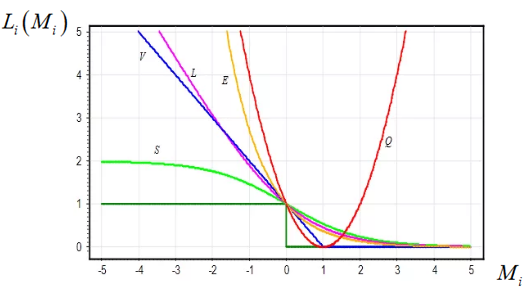
(3)

Здесь квадратные скобки – это индикатор ошибки. В соответствии с нотацией Айверсона они возвращают 1 для True и 0 – для False.

Если изобразить эту функцию потерь, зависящую от отступа, то получим ступенчатый график:

Рисунок 3. Кусочная функция ошибки

Недостаток этой функции в том что она не имеет производных в точках перегиба, а значит аналитическое решение минимизации значения найти невозможно, только перебором. Однако в математике существует много похожих по свойствам (точка экстремума в 0, 1) дифференцируемых функций. Вот некоторые, наиболее известные:

Рисунок 4. Виды функций ошибок

Для нашей задачи подойдет типовая квадратичная функция (FLD) .

(4)

После подстановки в (1) получим функционал качества:



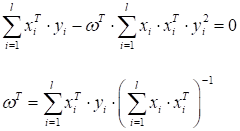
(5)

Тогда оптимальный вектор ω можем найти как производную функционала качества.



(6)

Откуда



# Результаты работы

Работа была выполнена на языке программирования Python 3. Код программы представлена на Листинге 1.

import numpy as np

import matplotlib.pyplot as plt

#===============================================================================

# Обучающая выборка

#===============================================================================

# вариант 5

x\_train = [[1, 3], [7, 4], [4, 3], [9, 4]]

y\_train = [-1, -1, 1, 1] # -1 - зеленые, 1 - синие

#===============================================================================

# Добавим 1 для симметрии признаков и w

x\_train = [list(x) + [1] for x in x\_train]

x\_train = np.array(x\_train)

y\_train = np.array(y\_train)

print(*'Введите по какому алгоритму работать (selection - подбор, RMSE): '*)

txt = *''*

while not txt:

try:

line = input()

except EOFError:

break

txt = line.strip()

if not txt or txt.isspace():

txt = *'RMSE'*

w = [0, 0]

if txt == *'RMSE'*:

# Аналитическое решение с помощью квадратичной функции ошибки

pt = np.sum([x \* y for x, y in zip(x\_train, y\_train)], axis=0) # Сумма вектора x \* y

xxt = np.sum([np.outer(x, x) for x in x\_train], axis=0) # Сумма x \* xT

invmat = np.linalg.inv(xxt) # xxt^-1 или обратная матрица

w = np.dot(pt, invmat)

line\_x = list(range(max(x\_train[:, 0]))) # формирование графика разделяющей линии

line\_y = [-x \* w[0] / w[1] - w[2] / w[1] for x in line\_x]

else:

# Параметры алгоритма (ступенчатая функция Хевисайда)===========================

n\_train = len(x\_train) # размер обучающей выборки

w = [0, -1, 0] # начальное значение вектора w

def **a**(x): return np.sign(x[0] \* w[0] + x[1] \* w[1]) # решающее правило (модель)

N = 50 # максимальное число итераций

nt = 0.1 # (эта) - шаг изменения веса

e = 0.1 # небольшая добавка для w0 чтобы был зазор между разделяющей линией и граничным образом

# ==============================================================================

last\_error\_index = -1 # индекс последнего ошибочного наблюдения

for n in range(N):

for i in range(n\_train): # перебор по наблюдениям

if y\_train[i] \* a(x\_train[i]) < 0: # если ошибка классификации (отступ M = y\_train[i]\*a(x\_train[i])),

w[0] = w[0] + nt \* y\_train[i] # то корректировка веса w0

last\_error\_index = i

Q = sum([1 for i in range(n\_train) if y\_train[i] \* a(x\_train[i]) < 0]) # Функционал качества (list-comprehension) это нотация Айзерсона

if Q == 0: # показатель качества классификации (число ошибок), в общем случае → 0 для дифференцируемых функций. Но так как у нас кусочно-непрерывная

break # останов, если все верно классифицируем

if last\_error\_index > -1:

w[0] = w[0] + e \* y\_train[last\_error\_index]

line\_x = list(range(max(x\_train[:, 0]))) # формирование графика разделяющей линии

line\_y = [w[0] \* x for x in line\_x]

def **margin**(x, y):

*"""*

*Отступы от линии разделения*

*"""*

return (x[0] \* w[0] + x[1] \* w[1] + w[2]) \* y

margins = [margin(x, y) for x, y in zip(x\_train, y\_train)]

def **classify**(x):

*"""*

*Классификатор, -1, +1*

*"""*

return np.sign(x[0] \* w[0] + x[1] \* w[1] + w[2])

print(*f'Весовые коэффициенты (вектор настраиваемых параметров): {list(w)}'*)

for i in range(len(x\_train)):

print(*f'Точка {list(x\_train[i][:-1])}, класс {y\_train[i]}, отступ {margins[i]}'*)

x\_0 = x\_train[y\_train == 1] # формирование точек для 1-го, numpy-python3 сахар, предикат в \_\_getitem\_\_()

x\_1 = x\_train[y\_train == -1] # и 2-го классов

line\_sign = *'-'* if (-1 \* (w[2] / w[1])) < 0 else *'+'*

abs\_w2 = abs(w[2] / w[1])

plt.suptitle(*f'Линейная бинарная классификация\n (вариант №5)\n ω = [{w[2]:.3f}, {w[1]:.3f}, {w[0]:.3f}]ᵀ'*, fontsize=12)

plt.scatter(x\_0[:, 0], x\_0[:, 1], color=*'green'*, label=*f"C1=-1"*) # [:, 0] - питон 3 магия \_\_getitem\_\_ которую юзает numpy, эта запись значит "взять срез, тоесть копировать" и взять 0-й столбец из матрицы (первый)

plt.scatter(x\_1[:, 0], x\_1[:, 1], color=*'blue'*, label=*f"C2=+1"*)

plt.plot(line\_x, line\_y, color=*'red'*, label=*f'Разделяющая линия, x2 = {0-w[0]/w[1]:.3f}\*x1 {line\_sign} {abs\_w2:.3f}'*)

plt.xlim([0, 45])

plt.ylim([0, 75])

plt.ylabel(*"x2"*)

plt.xlabel(*"x1"*)

plt.tick\_params(labelcolor=*'indigo'*)

plt.legend()

plt.grid(True)

plt.show()

Листинг 1. Код программы main.py

Результат работы программы приведен на Рисунках 5, 6.

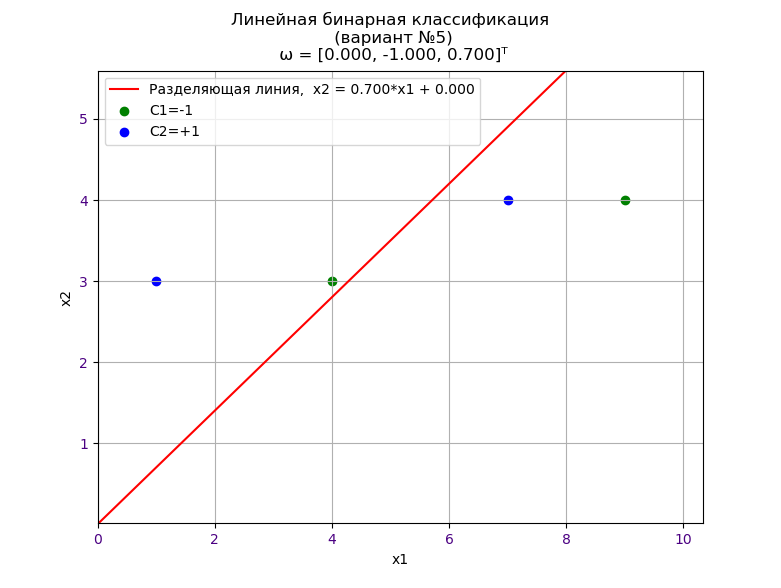


Рисунок 5. Результаты работы программы (перебор)

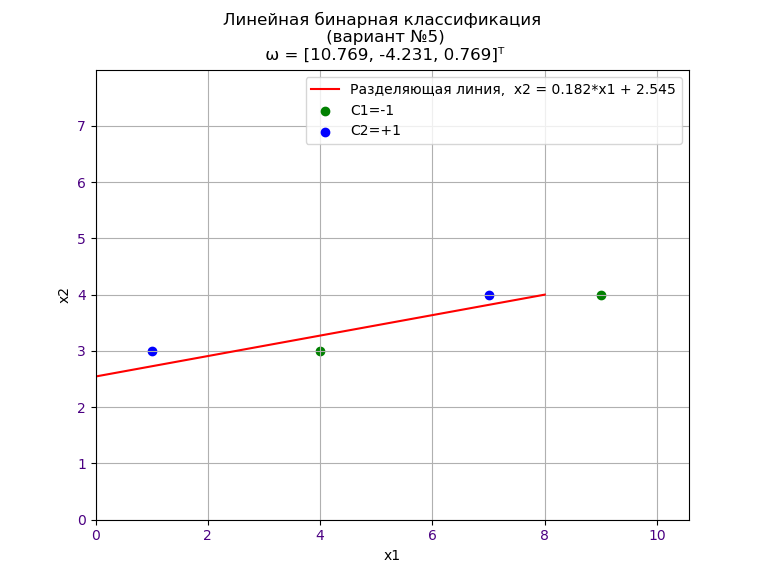
Весовые коэффициенты (вектор настраиваемых параметров): [0.7, -1, 0]

Точка [1, 3], класс -1, отступ 2.3

Точка [7, 4], класс -1, отступ -0.8999999999999995

Точка [4, 3], класс 1, отступ -0.20000000000000018

Точка [9, 4], класс 1, отступ 2.3

Рисунок 6. Результаты работы программы (аналитически RMSE)

Весовые коэффициенты (вектор ω настраиваемых параметров): [0.7692307692307726, -4.230769230769259, 10.769230769230852]

Точка [1, 3], класс -1, отступ 1.153846153846155

Точка [7, 4], класс -1, отступ 0.7692307692307772

Точка [4, 3], класс 1, отступ 1.153846153846164

Точка [9, 4], класс 1, отступ 0.7692307692307683

# Вывод

В ходе лабораторной работы я определил коэффициенты разделяющей линии и отступы для зеленых и синих точек.