МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных систем

Факультет автоматики и вычислительной техники

Кафедра систем автоматизации управления

Дата сдачи на проверку:

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2022 г.

Проверено:

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2022 г.

**Классификация методом опорных векторов (SVM)**

Отчет по лабораторной работе №1  
по дисциплине

«Методы искусственного интеллекта»

Разработал студент гр. ИТб-3302-02-00 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Лежнин Р.А./

(Подпись)

Руководитель ст. преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Родионов К.В./

(Подпись)

Работа защищена с оценкой «\_\_\_\_\_\_\_\_\_\_\_» «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2022 г.

Киров 2022

Цель работы: изучение алгоритма классификации данных «Метод опорных векторов».

В ходе работы необходимо выполнить следующее задание:

* исследовать параметры Cи и sigma(σ);
* исследовать линейное и гауссово ядро;
* получить графики исследований.

Ход работы:

Задание 1: загрузить данные из файла dataset1.mat, и отобразить на графике.

Код, необходимый для 1 задания, представлен в листинге 1.

График, необходимый для 1 задания, представлен на рисунке 1.

Листинг 1 – Программа для 1 задания

import scipy.io as sio

import numpy as np

import lib.svm as svm

mat = sio.loadmat("dataset1.mat")

y = np.float64(mat["y"])

x = np.float64(mat["X"])

svm.visualize\_boundary\_linear(x, y, "", "exc 1")

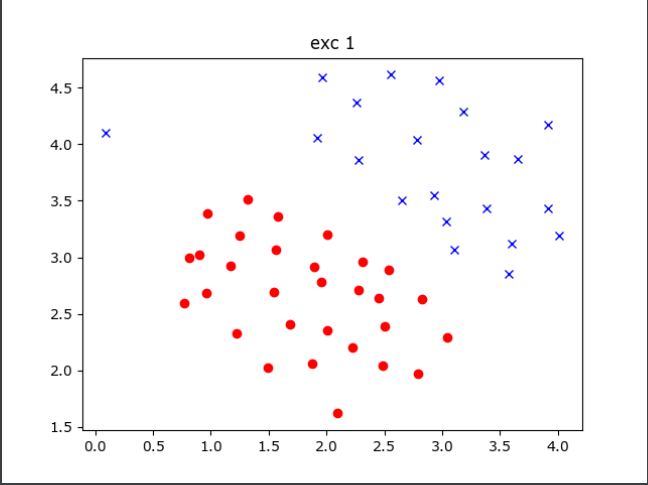


Рисунок 1 – Исходные данные из файла dataset1.mat

Задание 2: обучить классификатор на обучающей выборке и отобразить границу.

Код, необходимый для 2 задания, представлен в листинге 2.

График, необходимый для 2 задания, представлен на рисунке 2.

Листинг 2 – Программа для 2 задания

import scipy.io as sio

import numpy as np

import lib.svm as svm

import scipy.io as sio

mat = sio.loadmat("dataset1.mat")

y = np.float64(mat["y"])

x = np.float64(mat["X"])

C = 1

model = svm.svm\_train(x, y, C, svm.linear\_kernel, 0.001, 20)

svm.visualize\_boundary\_linear(x, y, model, "exc 2")

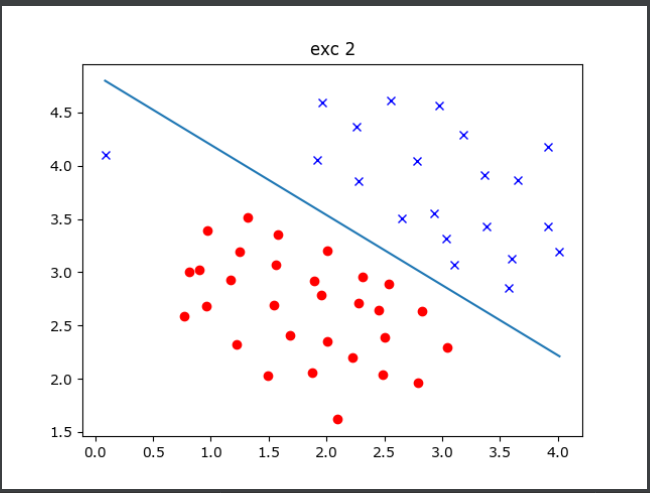


Рисунок 2 – Разделяющая граница при С = 1

Задание 3: выполнить обучение модели с С = 100, привести график.

Код, необходимый для 3 задания, представлен в листинге 3.

График, необходимый для 3 задания, представлен на рисунке 3

Листинг 3 – Программа для 3 задания

import numpy as np

import lib.svm as svm

import scipy.io as sio

mat = sio.loadmat("dataset1.mat")

y = np.float64(mat["y"])

x = np.float64(mat["X"])

C = 100

model = svm.svm\_train(x, y, C, svm.linear\_kernel, 0.001, 20)

svm.visualize\_boundary\_linear(x, y, model, "exc 3")

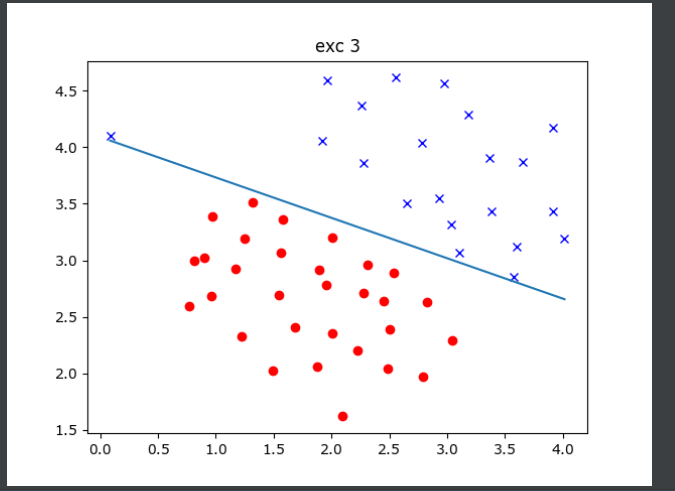


Рисунок 3 – Разделяющая граница при С = 100

Задание 4: реализовать функцию гауссова ядра, построить контурные графики для sigma = 1 и sigma = 3.

Код, необходимый для 4 задания, представлен в листинге 4.

Графики, необходимые для 4 задания, представлены на рисунках 4 и 5.

Листинг 4 – Программа для 4 задания

import lib.svm as svm

svm.contour(1)

svm.contour(3)



Рисунок 4 – Контурный график гауссова ядра при sigma = 1

Изображение выглядит как текст, электроника

Автоматически созданное описание

Рисунок 5 – Контурный график гауссова ядра sigma = 3

Задание 5: загрузить и отобразить данные из файла dataset2.mat.

Код, необходимый для 5 задания, представлен в листинге 5.

График, необходимый для задания 5, представлен на рисунке 6.

Листинг 5 – Программа для 5 задания

import scipy.io as sio

import numpy as np

import lib.svm as svm

mat = sio.loadmat("dataset2.mat")

y = np.float64(mat["y"])

x = np.float64(mat["X"])

svm.visualize\_boundary\_linear(x, y, "", "exc 5")

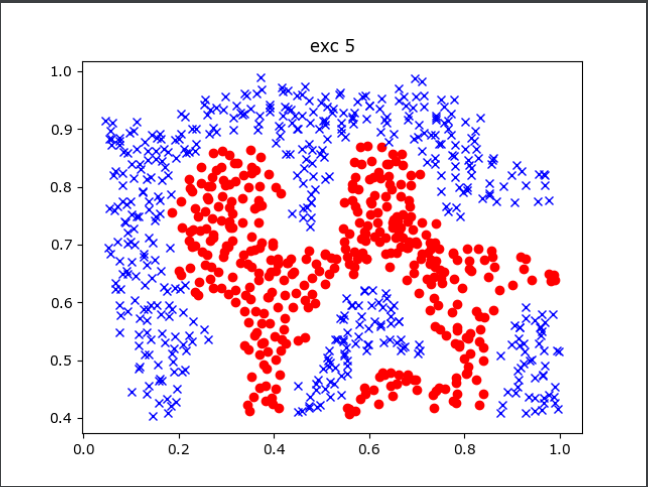


Рисунок 6 – Исходные данные из файла dataset2.mat

Задание 6: запустить обучение модели, отобразить получившуюся границу.

Код, необходимый для 6 задания, представлен в листинге 6.

График, необходимый для задания 7, представлен на рисунке 7.

Листинг 6 – Программа для 6 задания

import lib.svm as svm

import scipy.io as sio

import numpy as np

mat = sio.loadmat("dataset2.mat")

y = np.float64(mat["y"])

x = np.float64(mat["X"])

C = 1.0

sigma = 0.1

gaussian = svm.partial(svm.gaussian\_kernel, sigma=sigma)

gaussian.\_\_name\_\_ = svm.gaussian\_kernel.\_\_name\_\_

model = svm.svm\_train(x,y, C, gaussian)

svm.visualize\_boundary(x, y, model)

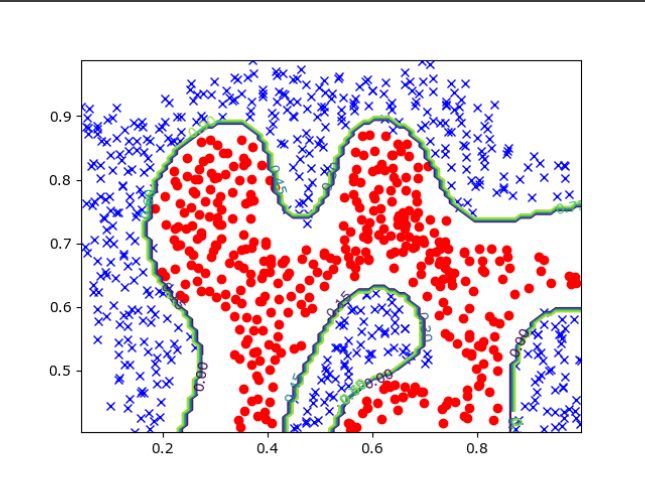


Рисунок 7 – Результирующая граница на основе гауссова ядра

Задание 7: загрузить и отобразить данные из файла dataset3.mat.

Код, необходимый для 7 задания, представлен в листинге 7.

Графики, необходимые для 7 задания, представлены на рисунке 8 и 9.

Листинг 7 – Программа для 7 задания

import scipy.io as sio

import numpy as np

import lib.svm as svm

data1 = sio.loadmat('dataset2.mat')

y1 = np.float64(data1['y'])

X1 = data1['X']

data2 = sio.loadmat('dataset3.mat')

Xval = data2['Xval']

yval = data2['yval']

y2 = np.float64(data2['y'])

X2 = data2['X']

C1 = 1.0

sigma1 = 0.1

gaussian = svm.partial(svm.gaussian\_kernel, sigma=sigma1)

gaussian.\_\_name\_\_ = svm.gaussian\_kernel.\_\_name\_\_

model1 = svm.svm\_train(X1, y1, C1, gaussian)

svm.visualize\_boundary(X1, y1, model1)

svm.visualize\_boundary\_linear(X2, y2, None, title="exc\_7\_1")

svm.visualize\_boundary\_linear(Xval, yval, None, title="exc\_7\_2")

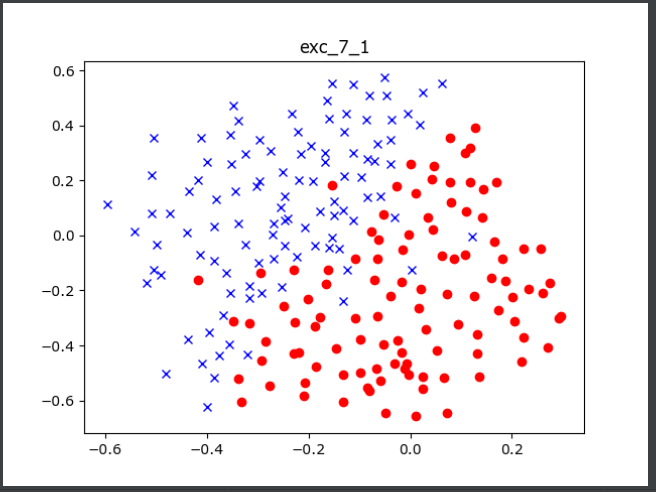


Рисунок 8 – Данные для обучающей выборки из файла dataset3.mat

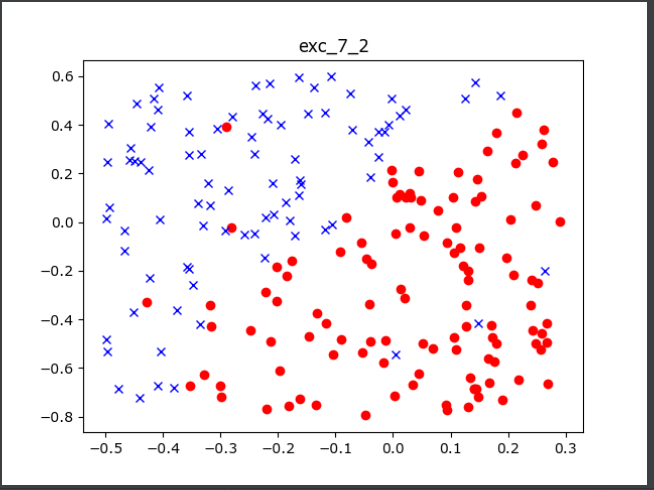


Рисунок 9 – Данные для тестовой выборки из файла dataset3.mat

Задание 8: выполнить обучение модели с неоптимальными параметрами, вывести график.

Код, необходимый для 8 задания, представлен в листинге 8.

График, необходимый для 8 задания, представлен на рисунке 10.

Листинг 8 – Программа для 8 задания

import lib.svm as svm

import scipy.io as sio

import numpy as np

data2 = sio.loadmat('dataset3.mat')

Xval = data2['Xval']

yval = data2['yval']

y2 = np.float64(data2['y'])

X2 = data2['X']

C2 = 1.0

sigma2 = 0.5

gaussian1 = svm.partial(svm.gaussian\_kernel, sigma=sigma2)

gaussian1.\_\_name\_\_ = svm.gaussian\_kernel.\_\_name\_\_

model2 = svm.svm\_train(X2, y2, C2, gaussian1)

svm.visualize\_boundary(X2, y2, model2)

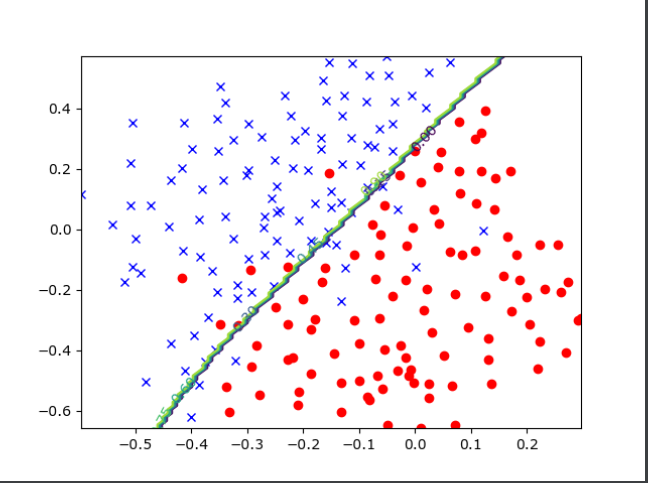


Рисунок 10 – График модели при неоптимальных параметрах

Задание 9: вычисление оптимальных значений Си и sigma.

Код, необходимый для 9 задания, представлен в листинге 9.

График, необходимый для 9 задания, представлен на рисунке 11.

Листинг 9 – Программа для 9 задания

import lib.svm as svm

import scipy.io as sio

import numpy as np

data2 = sio.loadmat('dataset3.mat')

Xval = data2['Xval']

yval = data2['yval']

y2 = np.float64(data2['y'])

X2 = data2['X']

minError = 9999

C4 = 0

sigma4 = 0

for C3 in [0.01, 0.03, 0.1, 0.3, 1, 3, 10, 30]:

for sigma3 in [0.01, 0.03, 0.1, 0.3, 1, 3, 10, 30]:

gaussian2 = svm.partial(svm.gaussian\_kernel, sigma=sigma3)

gaussian2.\_\_name\_\_ = svm.gaussian\_kernel.\_\_name\_\_

model3 = svm.svm\_train(X2, y2, C3, gaussian2)

ypred = svm.svm\_predict(model3, Xval)

error = np.mean(ypred != yval.ravel())

if (error < minError):

minError = error

C4 = C3

sigma4 = sigma3

gaussian4 = svm.partial(svm.gaussian\_kernel, sigma=sigma4)

gaussian4.\_\_name\_\_ = svm.gaussian\_kernel.\_\_name\_\_

model = svm.svm\_train(X2, y2, C4, gaussian4)

svm.visualize\_boundary(X2, y2, model)

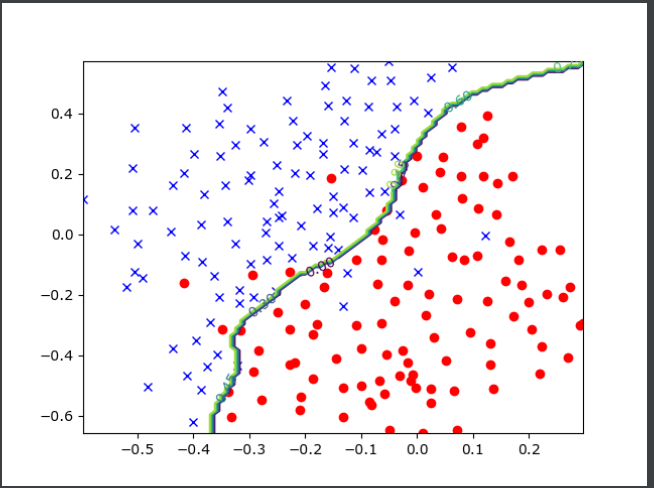


Рисунок 11 – График модели с оптимальными параметрами

Вывод: в ходе работы была изучена классификация метом опорных векторов, также были исследованы параметры Си и sigma, а также линейное и гауссово ядро. Получены графики исследований.