CSCI3230 / ESTR3108 2023-24 First Term Assignment 2

I declare that the assignment here submitted is original except for source material explicitly acknowledged, and that the same or closely related material has not been previously submitted for another course. I also acknowledge that I am aware of University policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the following websites. University Guideline on Academic Honesty: http://www.cuhk.edu.hk/policy/academichonesty/ Faculty of Engineering Guidelines to Academic Honesty: <http://www.erg.cuhk.edu.hk/erg-intra/upload/documents/ENGG_Discipline.pdf>

Student Name: Fong Long Wai

Student ID :1155177220

Q2:

The first model is using Polynomial kernel function with a degree of 4, c is 10. The output accuracy is 0.86. (graph 1 and graph 5)

一張含有 文字, 行, 圖表, 繪圖 的圖片

自動產生的描述

The second model is using Radial basis function kernel with a degree of 1, c is 0.1. The output accuracy is 0.9. (graph 4 and graph 8)

一張含有 文字, 圖表, 字型, 行 的圖片

自動產生的描述

Q2 code:

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import make\_blobs

from sklearn import svm

def getTest():

    test\_csv\_file='./test.csv';

    testtemp = np.genfromtxt(test\_csv\_file, delimiter=',', skip\_header=1)

    testy=testtemp[:,2];

    testX=testtemp[:,:2];

    return testy,testX;

def plot\_svc\_decision\_function(model, ax=None, plot\_support=True):

    """Plot the decision function for a 2D SVC"""

    if ax is None:

        ax = plt.gca()

    xlim = ax.get\_xlim()

    ylim = ax.get\_ylim()

    # create grid to evaluate model

    x = np.linspace(xlim[0], xlim[1], 30)

    y = np.linspace(ylim[0], ylim[1], 30)

    Y, X = np.meshgrid(y, x)

    xy = np.vstack([X.ravel(), Y.ravel()]).T

    P = model.decision\_function(xy).reshape(X.shape)

    # plot decision boundary and margins

    ax.contour(X, Y, P, colors='k',

               levels=[-1, 0, 1], alpha=0.5,

               linestyles=['--', '-', '--'])

    """

    # plot support vectors

    if plot\_support:

        ax.scatter(model.support\_vectors\_[:, 0],

                    model.support\_vectors\_[:, 1],

                    s=100, linewidth=1, facecolors='none',edgecolors='k')

    """

    ax.set\_xlim(xlim)

    ax.set\_ylim(ylim)

"""X, y = make\_blobs(n\_samples=30, centers=2,

                  random\_state=0, cluster\_std=1.2)

"""

csv\_file='./training.csv';

temp = np.genfromtxt(csv\_file, delimiter=',', skip\_header=1)

y=temp[:,2];

X=temp[:,:2];

plt.scatter(X[:, 0], X[:, 1], c=y, s=50,cmap=plt.cm.Paired)

# %%

superparameter=[10,5,1,0.1];

fig, ax = plt.subplots(1,len(superparameter)\*2, figsize=(16, 6))

fig.subplots\_adjust(left=0.0625, right=0.95, wspace=0.1)

model=[]

for C in superparameter:

    model.append(svm.SVC(kernel="poly",degree=4, C=C).fit(X, y))

for axi, m,C in zip(ax, model,superparameter):

    axi.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap=plt.cm.Paired)

    plot\_svc\_decision\_function(m, axi)

    axi.scatter(m.support\_vectors\_[:, 0],

                m.support\_vectors\_[:, 1],

                s=300, lw=1, facecolors='none');

    axi.set\_title('C = {0:.1f}'.format(C), size=14)

testy,testX=getTest();

for axi, m,C in zip(ax[len(superparameter):], model,superparameter):

    axi.scatter(testX[:, 0], testX[:, 1], c=testy, s=50, cmap=plt.cm.Paired)

    plot\_svc\_decision\_function(m, axi)

    axi.set\_title('C = {0:.1f}'.format(C), size=14)

for m in model:

    result=m.predict(testX);

    accuracy=(testy==result);

    print(accuracy.sum()/len(accuracy));