import numpy as np

from sklearn import svm

from libsvm.svmutil import \*

import scipy

import matplotlib.pyplot as plt #導入繪圖包，使得結果更加直觀

#數據預處理

#train

y\_train,Xtrain=svm\_read\_problem('satimage.scale.txt',return\_scipy=True)

#test

y\_test,Xtest=svm\_read\_problem('satimage.scale.t',return\_scipy=True)

X\_train=Xtrain.toarray()

X\_test=Xtest.toarray()

#Problem11

#訓練模型

y\_train\_5 = (y\_train==5).astype("int")

C = [1]

W = []

for i in C:

    c = 10 \*\* i

    clf = svm.SVC(kernel="linear", C=c)

    clf.fit(X\_train, y\_train\_5)

    w = clf.coef\_

    W.append(np.linalg.norm(w))

W

#Problem 12 and Problem 13

#將標籤修改為-1, 1

y\_trian\_2 = 2 \* (y\_train==2).astype("int") – 1

##y\_trian\_2 = 2 \* (y\_train==3).astype("int") – 1

#實際為y\_trian\_3

##y\_trian\_2 = 2 \* (y\_train==4).astype("int") – 1

#實際為y\_trian\_4

##y\_trian\_2 = 2 \* (y\_train==5).astype("int") – 1

#實際為y\_trian\_5

##y\_trian\_2 = 2 \* (y\_train==6).astype("int") – 1

#實際為y\_trian\_6

C = [1]

Ein = []

alpha = []

for i in C:

    c = 10 \*\* i

    clf = svm.SVC(kernel='poly', degree=3, coef0=1, gamma=1, C=c)

    clf.fit(X\_train, y\_trian\_2)

    e = np.mean(clf.predict(X\_train) != y\_trian\_2)

    #支持向量的索引

    support = clf.support\_

    #计算系数

    coef = np.sum(clf.dual\_coef\_[0] \* y\_trian\_2[support])

    alpha.append(coef)

    Ein.append(e)

Ein

#回答Problem 12

support

#觀察support維度來回答Problem 13

#Problem 14

C = [-2, -1, 0, 1, 2]

Distance = []

y\_train\_1 = (y\_train==1).astype("int")

#將標籤修改為-1, 1

y = 2 \* y\_train\_1 - 1

for i in C:

    c = 10\*\*i

    clf = svm.SVC(kernel='rbf', gamma=10, C=c)

    clf.fit(X\_train, y)

    X = X\_train[clf.support\_]

    #距離矩陣

    d1 = np.sum(X \*\* 2, axis=1).reshape(-1, 1)

    d2 = np.sum(X \*\* 2, axis=1).reshape(1, -1)

    dist = d1 + d2 - 2 \* X.dot(X.T)

    #Kernel矩陣

    K = np.exp(- c \* dist)

    #計算anyn

    y1 = clf.dual\_coef\_[0] \* y[clf.support\_]

    w2 = y1.dot(K).dot(y1.T)

    #計算距離

    distance = 1 / np.sqrt(w2)

    Distance.append(distance)

plt.plot(C, Distance)

plt.title("$\log\_{10}C$ VS distance") #以10為底的橫坐標

plt.show()

#觀察圖像最低點

#Problem 15

y\_test\_1 = (y\_test == 1)

Gamma = [-1,0,1,2,3]

Eout = []

for i in Gamma:

    gamma = 10\*\*i

    clf = svm.SVC(kernel='rbf', gamma=gamma, C=0.1)

    clf.fit(X\_train, y\_train\_1)

    e = np.mean(clf.predict(X\_test) != y\_test\_1)

    Eout.append(e)

plt.plot(Gamma, Eout)

plt.title("$\log\_{10}\gamma$ VS $E\_{out}$")

plt.show()#圖形更加直觀

#Problem 16

from sklearn.model\_selection import train\_test\_split

#合併資料，以便呼叫train\_test\_split函式

Data = np.c\_[X\_train, y\_train]

N = 1000

#記錄最小的Eval對應gamma的索引次數

Cnt = np.zeros(5) #初始化

Gamma = [-1,0,1,2,3]

for \_ in range(N):

    #劃分資料

train\_set, val\_set = train\_test\_split(Data, test\_size=0.05)

#200筆sample

    #取feature

    X\_train = train\_set[:, :36]

    #取label

    y\_train = train\_set[:, 36]

    X\_val = val\_set[:, :36]

    y\_val = val\_set[:, 36]

Eval = []

    for i in Gamma:

        gamma = 10\*\*i

        clf = svm.SVC(kernel='rbf', gamma=gamma, C=0.1)

        clf.fit(X\_train, y\_train.astype('int'))

        e = np.mean(clf.predict(X\_val) != y\_val)

        Eval.append(e)

    #找到最小Eval對應的索引

    index = np.argmin(Eval)

    #索引次數加1

    Cnt[index] += 1

#觀察圖像得到最高頻次gamma

plt.bar(Gamma, Cnt)

plt.show()