import numpy as np

import matplotlib.pyplot as plt

from numpy.linalg import inv

from sklearn.preprocessing import PolynomialFeatures

import random

import pandas as pd

#將網頁上的資料下載到桌面，存儲到同一個目錄下，打開命名

def train():

    data = np.genfromtxt("hw3\_train.dat.txt")

    X = data[:, :-1]

    y = data[:, -1]

    #产生数据

    #r, d = data.shape

    #獲取幾行幾列

    return X, y

def test():

    data = np.genfromtxt("hw3\_test.dat.txt")

    X = data[:, :-1]

    y = data[:, -1]

    #产生数据

    #r, d = data.shape

    #獲取幾行幾列

    return X, y

#Problem12，Problem13和Problem14

def multi\_feature1(x,n):

  c = np.empty((x.shape[0],0))

  for i in range(n+1):

    for j in range(x.shape[1]):

      h=(x[:,j]\*\*i)

      c=np.c\_[c,h]

  return c[:,x.shape[1]-1:]

X, y = train()

poly = PolynomialFeatures(degree=2)#Problem14

X\_poly=poly.fit\_transform(X)

#X\_poly = multi\_feature1(X,2)#problem12時置為2，problem13時置為8

#初始化

Eout = np.array([])

Ein = np.array([])

w\_poly = inv(X\_poly.T.dot(X\_poly)).dot(X\_poly.T).dot(y)

ein = np.mean(np.sign(X\_poly.dot(w\_poly) \* y) < 0)

Ein = np.append(Ein, ein)

#導入test的資料

X\_test, y\_test = test()

#X\_test\_poly = multi\_feature1(X\_test,2)#problem12時置為2，problem13時置為8

X\_test\_poly = poly.fit\_transform(X\_test)#Problem14

eout = np.mean(np.sign(X\_test\_poly.dot(w\_poly) \* y\_test) < 0)

Eout  = np.append(Eout, eout)

print(Ein-Eout)

#Problem15和Problem16

#設式中x0均為1，不影響結果

#Problem15

def multi\_feature2(x,n,k):

  c = np.empty((x.shape[0],0))

  for i in range(n):

    for j in range(x.shape[1]):

      h=(x[:,j]\*\*i)

      c=np.c\_[c,h]

  return c[:,x.shape[1]-1:x.shape[1]\*2-k]

#導入train和test的資料

X, y = train()

X\_test, y\_test = test()

#初始化

V = []#用來記錄ein-eout的值

Eout = np.array([])

Ein = np.array([])

#本例中的train集和test集的維數相等X.shape[1]=X\_test.shape[1]

for i in range(X.shape[1]):

  X\_poly=multi\_feature2(X,2,i)#0是下標最大，減少至下標最小需要變成x.shape[1]-1

  w\_poly = inv(X\_poly.T.dot(X\_poly)).dot(X\_poly.T).dot(y)

  ein = np.mean(np.sign(X\_poly.dot(w\_poly) \* y) < 0)

  #Ein = np.append(Ein, ein)

  X\_test\_poly=multi\_feature2(X\_test,2,i)

  eout = np.mean(np.sign(X\_test\_poly.dot(w\_poly) \* y\_test) < 0)

  #Eout  = np.append(Eout, eout)

  V=np.append(V,ein-eout)

  #print(ein-eout)

print(V.max())

print(np.unravel\_index(np.argmax(V),V.shape))#可見下標為7，即對應為fai等於3時

#Problem16

#初始化

Eout = np.array([])

Ein = np.array([])

v = np.array([])

for n in range(200):

    X, y = train()

    list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

    slice = random.sample(list, 5)

    x1=X[:,slice]

    X\_poly=np.c\_[np.ones(X.shape[0]),x1]

    w\_poly = inv(X\_poly.T.dot(X\_poly)).dot(X\_poly.T).dot(y)

    ein = np.mean(np.sign(X\_poly.dot(w\_poly) \* y) < 0)

    Ein = np.append(Ein, ein)

    #導入test的資料

    X\_test, y\_test = test()

    x2=X\_test[:,slice]

    X\_test\_poly=np.c\_[np.ones(X\_test.shape[0]),x2]

    eout = np.mean(np.sign(X\_test\_poly.dot(w\_poly) \* y\_test) < 0)

    Eout  = np.append(Eout, eout)

    if n==199:

        print(np.mean(Ein-Eout))