**BỘ GIÁO DỤC & ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐIỆN – ĐIỆN TỬ**

**BỘ MÔN TỰ ĐỘNG ĐIỀU KHIỂN**

**-----------------⸙∆⸙-----------------**



**TRÍ TUỆ NHÂN TẠO**

**NÂNG CAO**

**BÀI TẬP TUẦN 5**

**GVGD: TS. Trần Vũ Hoàng**

**HVTH: Trần Văn Danh**

**MSHV: 2341106**

**Tp. Hồ Chí Minh tháng 07 năm 2024**

**MỤC LỤC**

[DANH SÁCH HÌNH ẢNH 2](#_Toc172964092)

[Chương 1. YÊU CẦU 3](#_Toc172964093)

[Chương 2. THỰC HIỆN 4](#_Toc172964094)

# DANH SÁCH HÌNH ẢNH

[Hình 1. Biểu diễn các dữ liệu trong bộ dữ liệu cho sẵn. 4](#_Toc172965993)

[Hình 2. Biểu diễn các dữ liệu Validation đã được lọc ra. 6](#_Toc172965994)

[Hình 3. Hàm J với Lamda = 1 và Regularization L1. 11](#_Toc172965995)

[Hình 4: Hàm J với Lamda = 10 và Regularization L1. 12](#_Toc172965996)

[Hình 5. Hàm J với Lamda = 0.1 và Regularization L1. 12](#_Toc172965997)

[Hình 6. Hàm J với Lamda = 0.01 và Regularization L1. 12](#_Toc172965998)

[Hình 7. Hàm J với Lamda = 1 và Regularization L2. 13](#_Toc172965999)

[Hình 8. Hàm J với Lamda = 10 và Regularization L2. 13](#_Toc172966000)

[Hình 9. Hàm J với Lamda = 0.1 và Regularization L2. 14](#_Toc172966001)

[Hình 10. Hàm J với Lamda = 0.01 và Regularization L2. 14](#_Toc172966002)

# YÊU CẦU

Các bạn sử dụng tập dữ liệu của bài tập 5, mô hình dữ liệu x theo dạng bậc 6

và làm các công việc sau:

1. Chia dữ liệu ra thành 2 tập: training (70%) và validation (30%). Phải đảm bảo

việc chia dữ liệu là ngẫu nhiên và tỷ lệ positive và negative cân bằng.

2. Viết chương trình cho phép học các tham số của mô hình phân loại phi

tuyến trên có sử dụng regularization L2 và L1.

3. Tính J ở mỗi vòng lặp cho cả hai tập, chọn điểm dừng phù hợp.

4. Thay đổi lambda và tính J cho mỗi lambda tương ứng cho cả hai tập. Vẽ

biểu đồ quan hệ giữa J và lambda từ đó lựa chọn lambda phù hợp.

# THỰC HIỆN

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

df = pd.read\_csv("data\_ex5.txt",header = None, sep=",")

x = df.values[:,:-1]

y = df.values[:,-1]

m = len(y)

x\_pos = x[y==1,:]

x\_neg = x[y==0,:]

plt.scatter(x\_pos[:,0],x\_pos[:,1],c='r');

plt.scatter(x\_neg[:,0],x\_neg[:,1],c='k');

A diagram of red and black dots

Description automatically generated

Hình 1. Biểu diễn các dữ liệu trong bộ dữ liệu cho sẵn.

* Step 1: Chia dữ liệu ra thành 2 tập: training (70%) và validation (30%). Đảm bảo việc chia dữ liệu là ngẫu nhiên và tỷ lệ positive và negative cân bằng.

np.random.seed(1)

np.random.shuffle(x\_pos)

np.random.shuffle(x\_neg)

x\_train\_pos = x\_pos[:int(len(x\_pos)\*0.7)+1,:]

x\_val\_pos = x\_pos[int(len(x\_pos)\*0.7)+1:,:]

x\_train\_neg = x\_neg[:int(len(x\_neg)\*0.7)+1,:]

x\_val\_neg = x\_neg[int(len(x\_neg)\*0.7)+1:,:]

x\_train = np.concatenate((x\_train\_pos,x\_train\_neg),0)

x\_val = np.concatenate((x\_val\_pos,x\_val\_neg),0)

y\_train = np.concatenate((np.ones((len(x\_train\_pos), 1)), np.zeros((len(x\_train\_neg), 1))))

y\_val = np.concatenate((np.ones((len(x\_val\_pos), 1)), np.zeros((len(x\_val\_neg), 1))))

#plt.scatter(x\_train\_pos[:,0],x\_train\_pos[:,1],c='r')

#plt.scatter(x\_train\_neg[:,0],x\_train\_neg[:,1],c='k')

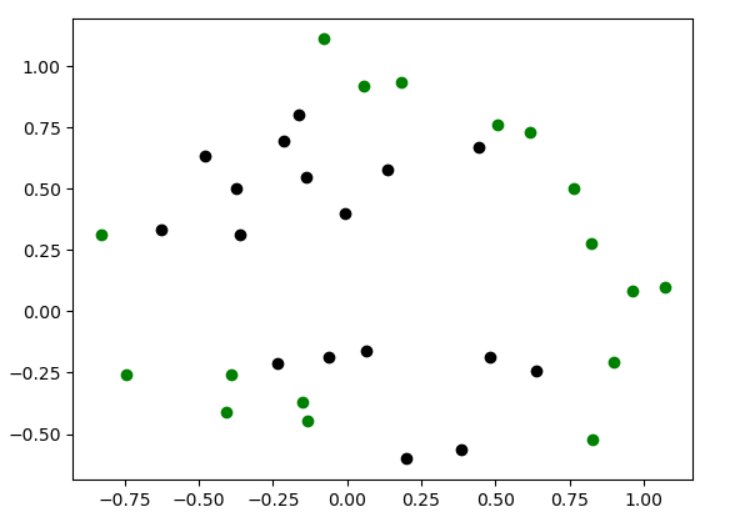
plt.scatter(x\_val\_pos[:,0],x\_val\_pos[:,1],c='black')

plt.scatter(x\_val\_neg[:,0],x\_val\_neg[:,1],c='green')

a=[len(x\_train\_pos),len(x\_train\_neg),len(x\_val\_pos),len(x\_val\_neg)]

a

[41, 43, 17, 17]



Hình 2. Biểu diễn các dữ liệu Validation đã được lọc ra.

* Step 2: Viết chương trình cho phép học các tham số của mô hình phân loại phi tuyến trên có sử dụng regularization L2 và L1.

# map features to polynomial terms up to a specified degree

def mapfeature(x1, x2, m, degree):

    x = np.zeros((m, sum(np.arange(2, degree+2))))

    col = 0

    for i in range(1, degree+1):

        for j in range(i+1):

            x[:, col] = ((x1\*\*(i-j))\*(x2\*\*j)).reshape(-1)

            col += 1

    return x

# normalize the features

def normalization(x):

    mean = np.mean(x, axis=0)

    std = np.std(x, axis=0)

    return (x - mean)/std

degree = 6

m\_train=len(y\_train)

m\_val =len(y\_val)

x\_train\_map = mapfeature(x\_train[:, 0], x\_train[:, 1], m\_train, degree)

x\_train\_norm = normalization(x\_train\_map)

X\_train = np.concatenate((np.ones((m\_train, 1)), x\_train\_norm), 1)

x\_val\_map = mapfeature(x\_val[:, 0], x\_val[:, 1], m\_val, degree)

x\_val\_norm = normalization(x\_val\_map)

X\_val = np.concatenate((np.ones((m\_val, 1)), x\_val\_norm), 1)

print(X\_train.shape)

print(X\_val.shape)

(84, 28)

(34, 28)

* Step 3a: Tính J ở mỗi vòng lặp cho cả hai tập, chọn điểm dừng phù hợp với Regularization L1.

ite = 10000

lr = 0.1

lamda=0.1

J\_train=np.zeros(ite)

y\_train = np.reshape(y\_train,(len(y\_train),1))

J\_val=np.zeros(ite)

y\_val = np.reshape(y\_val,(len(y\_val),1))

np.random.seed(10)

theta = np.random.rand(X\_train.shape[1],1)

z\_train = np.dot(X\_train,theta)

z\_val = np.dot(X\_val,theta)

h\_train = 1.0/(1.0+np.exp(-z\_train))

h\_val = 1.0/(1.0+np.exp(-z\_val))

for i in range(ite):

    theta\_temp=theta.copy()

    theta\_temp[0]=0

    theta = theta - (lr/m\_train)\*(np.dot((X\_train.T),(h\_train-y\_train)))-(lr/m\_train)\*lamda\*np.sign(theta\_temp)

    z\_train = np.dot(X\_train,theta)

    z\_val = np.dot(X\_val,theta)

    h\_train = 1.0/(1.0+np.exp(-z\_train))

    h\_val = 1.0/(1.0+np.exp(-z\_val))

    J\_train[i]=np.mean(-y\_train\*np.log(h\_train)-(1-y\_train)\*np.log(1-h\_train))

    J\_val[i]=np.mean(-y\_val\*np.log(h\_val)-(1-y\_val)\*np.log(1-h\_val))

print(J\_train)

print(J\_val)

[2.85856374 2.72430506 2.59816541 ... 0.28654032 0.28653958 0.28653899]

[4.54456006 4.35562817 4.16999659 ... 0.35967932 0.35968991 0.35968489]

* Step 3b: Tính J ở mỗi vòng lặp cho cả hai tập, chọn điểm dừng phù hợp với Regularization L2.

ite = 10000

lr = 0.1

lamda=0.1

J\_train=np.zeros(ite)

y\_train = np.reshape(y\_train,(len(y\_train),1))

J\_val=np.zeros(ite)

y\_val = np.reshape(y\_val,(len(y\_val),1))

np.random.seed(10)

theta = np.random.rand(X\_train.shape[1],1)

z\_train = np.dot(X\_train,theta)

z\_val = np.dot(X\_val,theta)

h\_train = 1.0/(1.0+np.exp(-z\_train))

h\_val = 1.0/(1.0+np.exp(-z\_val))

for i in range(ite):

    theta\_temp=theta.copy()

    theta\_temp[0]=0

    theta = theta - (lr/m\_train)\*(np.dot((X\_train.T),(h\_train-y\_train)))-(lr/m\_train)\*lamda\*theta\_temp

    z\_train = np.dot(X\_train,theta)

    z\_val = np.dot(X\_val,theta)

    h\_train = 1.0/(1.0+np.exp(-z\_train))

    h\_val = 1.0/(1.0+np.exp(-z\_val))

    J\_train[i]=np.mean(-y\_train\*np.log(h\_train)-(1-y\_train)\*np.log(1-h\_train))

    J\_val[i]=np.mean(-y\_val\*np.log(h\_val)-(1-y\_val)\*np.log(1-h\_val))

print(J\_train)

print(J\_val)

[3.66054264 3.45248321 3.25005108 ... 0.26047062 0.26046997 0.26046932]

[2.48504689 2.34369264 2.20756113 ... 0.57191273 0.57191564 0.57191854]

* Step 4: Thay đổi lambda và tính J cho mỗi lambda tương ứng cho cả hai tập. Vẽ biểu đồ quan hệ giữa J và lambda từ đó lựa chọn lambda phù hợp.

plt.plot(J\_train)

plt.plot(J\_val)

plt.legend(["train", "valid"])

plt.show()

* **Với Regularization L1**.
* Lamda = 1

A screenshot of a computer

Description automatically generated

Hình 3. Hàm J với Lamda = 1 và Regularization L1.

* Lamda = 10

A screenshot of a computer

Description automatically generated

Hình 4: Hàm J với Lamda = 10 và Regularization L1.

* Lamda = 0.1

A screenshot of a computer

Description automatically generated

Hình 5. Hàm J với Lamda = 0.1 và Regularization L1.

* Lamda = 0.01

A screenshot of a computer

Description automatically generated

Hình 6. Hàm J với Lamda = 0.01 và Regularization L1.

* **Với Regularization L2.**
* Lamda = 1

A screenshot of a computer

Description automatically generated

Hình 7. Hàm J với Lamda = 1 và Regularization L2.

* Lamda = 10

A screenshot of a computer

Description automatically generated

Hình 8. Hàm J với Lamda = 10 và Regularization L2.

* Lamda = 0.1

A screenshot of a computer

Description automatically generated

Hình 9. Hàm J với Lamda = 0.1 và Regularization L2.

* Lamda = 0.01

A screenshot of a computer

Description automatically generated

Hình 10. Hàm J với Lamda = 0.01 và Regularization L2.