LAB 4

Question 1

LDA binary classification

Observation:

Here I have performed binary classification using LDA.For , binary class , only 0 and 1 are used for classification from the mnist dataset.As, shapes of 0 and 1 are very different , so the accuracy will be high to classify correct class.Here,covariance used for both classes is the average of the covariancce matrix of both 0 and 1.Pseudo inverse of covariance is taken due to covariance matrix is singular.Accuracy is printed as the output which turn out to be 99.38.

In [ ]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.linalg as la
import math
from scipy.fftpack import fft,fftfreq
from scipy.linalg import toeplitz

from sklearn.model_selection import train_test_split
data1 = pd.read_csv('mnist_train.csv')
data2 = pd.read_csv('mnist_test.csv')
Y_train = np.asfarray(data1.iloc[:,0])
X_train = np.asfarray(data1.iloc[:,1:])
Y_test = np.asfarray(data2.iloc[:,0])
X_test = np.asfarray(data2.iloc[:,1:])
Y_test=Y_test.reshape((len(Y_test),))
Y_train=Y_train.reshape((len(Y_train),))
X0=X_train[np.where(Y_train==0)]
X1=X_train[np.where(Y_train==1)]
p0=len(X0)/(len(X0)+len(X1))
m0=X0.mean(0)
m1=X1.mean(0)
co0=np.cov(X0.T)
co1=np.cov(X1.T)
co=(co0+co1)/2
coin=np.linalg.pinv(co)
yprec=[]


Y_test1=Y_test[(np.where((Y_test==1) | (Y_test==0)))]
X_test1=X_test[(np.where((Y_test==1) | (Y_test==0)))]
for i in range(len(Y_test1)):
    temp0=X_test1[i]-m0
    temp1=X_test1[i]-m1
    temp01 = (-1/2)*np.dot(temp0,np.dot(coin,(temp0.T)))
    temp11 = (-1/2)*np.dot(temp1,np.dot(coin,(temp1.T)))
    temp02=temp01+np.log(p0)
    temp12=temp11+np.log(1-p0)
    if(temp02>temp12):
        yprec.append(0)
    else:
        yprec.append(1)
yprec=np.array(yprec)
ac = (yprec[:]==Y_test1[:]).mean()
print((ac)*100)
```

```
99.38534278959811
```

LDA multiclass classification

Observation:

Here I have performed multiclass classification using LDA on mnist dataset.Here, covariance matrix is taken as average of covariance matrices of all classes.Pseudo inverse of covariance is taken due to covariance matrix is singular.Accuracy is printed as the output which turn out to be 87.228.

In [ ]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.linalg as la
import math
from scipy.fftpack import fft,fftfreq
from scipy.linalg import toeplitz
from itertools import count
from sklearn.model_selection import train_test_split
data1 = pd.read_csv('mnist_train.csv')
data2 = pd.read_csv('mnist_test.csv')
Y_train = np.asfarray(data1.iloc[:,0])
X_train = np.asfarray(data1.iloc[:,1:])
Y_test = np.asfarray(data2.iloc[:,0])
X_test = np.asfarray(data2.iloc[:,1:])
Y_test=Y_test.reshape((len(Y_test),))
Y_train=Y_train.reshape((len(Y_train),))

cov_all = []
mean_all = []
po=np.zeros((10))
for k in range(10):
    tmp = X_train[[i for i, j in zip(count(), Y_train) if j =
= k],:]
    cov_all.append(np.cov(tmp.T))
    mean_all.append(np.mean(tmp,axis = 0))
    po[k] = np.log(len(tmp)/Y_train.shape[0])

mean_all=np.array(mean_all)
cov_all=np.array(cov_all)
cov=cov_all[0]
for k in range(1,10):
    cov=cov+(cov_all[k])
cov=cov/10
coin=np.linalg.pinv(cov)
ypred=[]
for i in range(len(Y_test)):
    max=[]
    for k in range(10):
        temp =X_test[i]-mean_all[k]
        temp1 = (-1/2)*np.dot(temp,np.dot(coin,(temp.T)))
        max.append((temp1 + po[k]))
    ypred.append(np.argmax(max))
ypred=np.array(ypred)
ac = (ypred[:]==Y_test[:]).mean()
print((ac)*100)
```

```
87.22872287228722
```

Question 2

QDA multiclass classification:

Observation:

Here I have performed multiclass classification using QDA on mnist dataset.Here classification is done using covariance for the corresponding class.Pseudo inverse of covariance is taken due to all covariance matrices of classes were singular.Here, determinant will be 0 so, we can't use log on sqrt(det).Singularity is caused due to many zero columns.So,another possible solution can be adding noise, but due to this accuracy has shown drastic fall. So,I have not taken the extra -1/2*log(sqrt(det)) term into consideration for comparison.The singularity can be also removed by preprocessing.the Accuracy is printed as the output which turn out to be 85.718.Here ,QDA is showing slightly less accuracy than LDA due the reason of singularity.If,covariance matrices were non singular than QDA will give more accuracy than LDA.

In [ ]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import scipy.linalg as la
import math
import random
from scipy.fftpack import fft,fftfreq
from scipy.linalg import toeplitz
from itertools import count
from sklearn.model_selection import train_test_split
data1 = pd.read_csv('mnist_train.csv')
data2 = pd.read_csv('mnist_test.csv')
Y_train = np.asfarray(data1.iloc[:,0])
X_train = np.asfarray(data1.iloc[:,1:])
Y_test = np.asfarray(data2.iloc[:,0])
X_test = np.asfarray(data2.iloc[:,1:])
Y_test=Y_test.reshape((len(Y_test),))
Y_train=Y_train.reshape((len(Y_train),))

cov_all = []
mean_all = []
po=np.zeros((10))
for k in range(10):
    tmp = X_train[[i for i, j in zip(count(), Y_train) if j =
= k],:]
    cov_all.append(np.cov(tmp.T))
    mean_all.append(np.mean(tmp,axis = 0))
    po[k] = np.log(len(tmp)/Y_train.shape[0])
coin=np.zeros((10,784,784))
mean_all=np.array(mean_all)
cov_all=np.asfarray(cov_all)
for k in range(10):
    coin[k]=np.linalg.pinv(cov_all[k])
ypred=[]
for i in range(len(Y_test)):
    max=[]
    for k in range(10):
        temp =X_test[i]-mean_all[k]
        temp1 = (-1/2)*np.dot(temp,np.dot(coin[k],(temp.T)))
        max.append(temp1 + po[k])
    ypred.append(np.argmax(max))
ypred=np.array(ypred)
ac = (ypred[:]==Y_test[:]).mean()
print((ac)*100)
```

```
85.71857185718572
```