**First Examination Source Code Problem 1**

|  |
| --- |
| Student: Do Thanh Trung  ID: 113998401  Teacher: Leeh-Ter Yao  Course: Artificial Intelligent and Machine Learning |

**Problem 1**

# Import libraries

import pandas as pd

import numpy as np

'''

Data properties: 8 classes (1,2,3,4,5,6,7,8), 8 features

X\_train shape: (10800, 8)

y\_train shape: (10800,)

X\_test shape:  (1200, 8)

'''

train\_data = pd.read\_csv('./data/trn1.csv')

test\_data = pd.read\_csv('./data/tst1.csv', *header*=None)

# Prepare train and test data

X\_train = train\_data.iloc[:, :-1].values

y\_train = train\_data.iloc[:, -1].values

X\_test = test\_data.values

C = len(np.unique(y\_train)) # number of classes = 8

N = X\_train.shape[0] # number of samples = 10800

d = X\_train.shape[1] #number of features = 8

class\_probs = np.array([np.mean(y\_train == i) for i in range(1, C + 1)]) # (8,)

# print(class\_probs.shape)

class\_means = np.array([np.mean(X\_train[y\_train == i], *axis*=0) for i in range(1, C + 1)]) #(8, 8)

# print(class\_means.shape)

class\_cov = np.array([np.cov(X\_train[y\_train == i].T) for i in range(1, C + 1)]) #(8, 8, 8)

# print(class\_cov.shape)

*def* discriminant\_func(*x*, *class\_idx*):

    x\_diff = x - class\_means[class\_idx - 1]

    cov\_inv = np.linalg.inv(class\_cov[class\_idx - 1])

    return -0.5 \* np.log(np.linalg.det(class\_cov[class\_idx - 1])) - \

           0.5 \* np.sum(x\_diff @ cov\_inv \* x\_diff, *axis*=1) + np.log(class\_probs[class\_idx - 1])

determinants = np.array([np.linalg.det(cov\_matrix) for cov\_matrix in class\_cov])

discriminant\_functions = np.array([discriminant\_func(X\_train, i) for i in range(1, C + 1)]).T

train\_prediction = np.argmax(discriminant\_functions, *axis*=1) + 1

accuracy = np.sum(train\_prediction==y\_train, *axis*=0)/y\_train.shape[0]

print(*f*"Training Accuracy = {accuracy}")

discriminant\_functions = np.array([discriminant\_func(X\_test, i) for i in range(1, C + 1)]).T

test\_prediction = np.argmax(discriminant\_functions, *axis*=1) + 1

# Add the predicted class to the testing data

result\_data = np.column\_stack((test\_data, test\_prediction))

# Save results to class1\_result.csv

result\_df = pd.DataFrame(result\_data)

result\_df.to\_csv('./results/result1.csv', *index*=False, *header*=False)

Github Link: