PRACTICAL 5

Name:	Harsh Shah	**Semester:**	VII	**Division:**	6
Roll No.:	21BCP359	**Date:**	20-08-24	**Batch:**	G11
Aim:	Understanding Linear Discriminant projection in Datasets.				

Compute the Linear Discriminant projection for the following two dimensional dataset.

- **Samples for class ω1: X1= (x1, x2) = {(6, 4), (4, 5), (3, 4), (5, 7), (6, 6)}**
- **Sample for class ω2: X2= (x1, x2) = {(11, 12), (7, 9), (10, 7), (10, 9), (12, 10)}**

Linear Discriminant Projection

Linear Discriminant Projection (LDP) refers to the process of projecting data onto a lower-dimensional space in a way that maximizes the separation between different classes. It is a key part of **Linear Discriminant Analysis (LDA)**, a method used in statistics, pattern recognition, and machine learning for dimensionality reduction and classification.

Steps:

1. Define the samples for each class
2. Compute the mean vectors
3. Compute the within-class scatter matrix SW for both classes
4. Compute the between-class scatter matrix SB
5. Compute the eigenvalues and eigenvectors of SW^{-1} * SB
    a. First, compute the inverse of SW
    b. Then, compute the matrix SW^{-1} * SB
    c. Compute the eigenvalues and eigenvectors
    d. Find the eigenvector corresponding to the largest eigenvalue

Code

```
import numpy as np

X1 = np.array([[6, 4], [4, 5], [3, 4], [5, 7], [6, 6]])  # Class ω1
X2 = np.array([[11, 12], [7, 9], [10, 7], [10, 9], [12, 10]])  # Class ω2

# Step 1: Compute the mean vectors
mu1 = np.mean(X1, axis=0)
mu2 = np.mean(X2, axis=0)

# Step 2: Compute the within-class scatter matrices
S_W1 = np.dot((X1 - mu1).T, (X1 - mu1)) / (len(X1) - 1)
S_W2 = np.dot((X2 - mu2).T, (X2 - mu2)) / (len(X2) - 1)
S_W = S_W1 + S_W2

# Step 3: Compute the between-class scatter matrix
mu_diff = (mu2 - mu1).reshape(2, 1)
S_B = np.dot(mu_diff, mu_diff.T)
```

```python
# Step 4: Compute the projection vector (eigenvector)
eigvals, eigvecs = np.linalg.eig(np.linalg.inv(S_W).dot(S_B))

# Sort eigenvectors by eigenvalues in descending order
eigvecs = eigvecs[:, np.argsort(-eigvals)]
w = eigvecs[:, 0]  # Projection vector (corresponding to the largest eigenvalue)

# Output the results
print("Mean vector of class ω1:", mu1)
print("Mean vector of class ω2:", mu2)
print("Within-class scatter matrix S_W:\n", S_W)
print("Between-class scatter matrix S_B:\n", S_B)
print("Projection vector w:", w)

# Project the samples onto the new axis
Y1 = np.dot(X1, w)
Y2 = np.dot(X2, w)

print("Projected samples for class ω1:", Y1)
print("Projected samples for class ω2:", Y2)
```

Output

```
Mean vector of class ω1: [4.8 5.2]
Mean vector of class ω2: [10.   9.4]
Within-class scatter matrix S_W:
 [[5.2 1.8]
 [1.8 5. ]]
Between-class scatter matrix S_B:
 [[27.04 21.84]
 [21.84 17.64]]
Projection vector w: [0.82816079 0.5604906 ]
```

```
Projected samples for class ω1: [7.21092712 6.11509614 4.72644475 8.06423812 8.33190831]
Projected samples for class ω2: [15.83565584 10.84154089 12.20504206 13.32602325 15.54283543]
```