import numpy as np

import pandas as pd

from sklearn.discriminant\_analysis import LinearDiscriminantAnalysis

# Create the dataset

X = np.array([[1, 2], [2, 3], [3, 3], [5, 5]]) # Feature values

y = np.array([0, 0, 1, 1]) # Class labels

# Create a DataFrame for clarity

data = pd.DataFrame(X, columns=['Feature 1', 'Feature 2'])

data['Class'] = y

# LDA using Library

lda = LinearDiscriminantAnalysis()

lda.fit(X, y)

print("Using Library:")

print("Coefficients:", lda.coef\_)

print("Intercept:", lda.intercept\_)

# LDA using Matrix Multiplication

# Compute class means

mean\_0 = np.mean(X[y == 0], axis=0) # Mean for class 0

mean\_1 = np.mean(X[y == 1], axis=0) # Mean for class 1

# Compute within-class scatter matrix

S\_w = np.zeros((2, 2))

for i in range(len(X)):

if y[i] == 0:

diff = (X[i] - mean\_0).reshape(2, 1)

else:

diff = (X[i] - mean\_1).reshape(2, 1)

S\_w += diff @ diff.T

# Fix singular matrix by adding regularization

S\_w += np.eye(2) \* 1e-5

# Compute linear discriminant direction

S\_w\_inv = np.linalg.inv(S\_w)

w = S\_w\_inv @ (mean\_1 - mean\_0)

# Compute intercept

intercept\_manual = -0.5 \* (mean\_1 + mean\_0) @ w

print("\nUsing Matrix Multiplication:")

print("Coefficients:", w)

print("Intercept:", intercept\_manual)