# PCA

Name: Debatreya Das Roll No. 12212070 CS A4 ML Lab 13

# Step 1: Import necessary libraries  
import numpy as np  
import pandas as pd

## Define the data

# Define the data  
data = np.array([[4, 8, 13, 7],  
 [11, 4, 5, 14]])  
df = pd.DataFrame(data.T, columns=['X1', 'X2'])  
df

X1 X2  
0 4 11  
1 8 4  
2 13 5  
3 7 14

## Standardize the data

Standardize the data by subtracting the mean and dividing by the standard deviation for each feature.

# Calculate mean and standard deviation for each feature  
mean = df.mean()  
std\_dev = df.std()  
  
# Standardize the data  
standardized\_data = (df - mean) / std\_dev  
print("Standardized Data:\n", standardized\_data)  
standardized\_data

Standardized Data:  
 X1 X2  
0 -1.069045 0.521286  
1 0.000000 -0.938315  
2 1.336306 -0.729800  
3 -0.267261 1.146829

X1 X2  
0 -1.069045 0.521286  
1 0.000000 -0.938315  
2 1.336306 -0.729800  
3 -0.267261 1.146829

## Calculate the covariance matrix

The covariance matrix shows how much the features vary together.

# Calculate the covariance matrix of the standardized data  
cov\_matrix = np.cov(standardized\_data.T)  
print("\nCovariance Matrix:\n", cov\_matrix)  
cov\_matrix

Covariance Matrix:  
 [[ 1. -0.61300603]  
 [-0.61300603 1. ]]

array([[ 1. , -0.61300603],  
 [-0.61300603, 1. ]])

## Compute the eigenvalues and eigenvectors

Calculate the eigenvalues and eigenvectors of the covariance matrix.

# Compute the eigenvalues and eigenvectors  
eigenvalues, eigenvectors = np.linalg.eig(cov\_matrix)  
print("\nEigenvalues:\n", eigenvalues)  
print("\nEigenvectors:\n", eigenvectors)  
eigenvalues, eigenvectors

Eigenvalues:  
 [1.61300603 0.38699397]  
  
Eigenvectors:  
 [[ 0.70710678 0.70710678]  
 [-0.70710678 0.70710678]]

(array([1.61300603, 0.38699397]),  
 array([[ 0.70710678, 0.70710678],  
 [-0.70710678, 0.70710678]]))

## Sort the eigenvalues and eigenvectors

Sort them in descending order to prioritize the principal components with the most variance.

# Sort the eigenvalues and eigenvectors in descending order  
sorted\_indices = np.argsort(eigenvalues)[::-1]  
sorted\_eigenvalues = eigenvalues[sorted\_indices]  
sorted\_eigenvectors = eigenvectors[:, sorted\_indices]  
  
print("\nSorted Eigenvalues:\n", sorted\_eigenvalues)  
print("\nSorted Eigenvectors:\n", sorted\_eigenvectors)  
sorted\_eigenvalues, sorted\_eigenvectors

Sorted Eigenvalues:  
 [1.61300603 0.38699397]  
  
Sorted Eigenvectors:  
 [[ 0.70710678 0.70710678]  
 [-0.70710678 0.70710678]]

(array([1.61300603, 0.38699397]),  
 array([[ 0.70710678, 0.70710678],  
 [-0.70710678, 0.70710678]]))

## Choose the top k eigenvectors (principal components)

Here, we select the top principal component(s) based on the sorted eigenvalues.

# Choose the top k eigenvectors  
k = 1  
selected\_eigenvectors = sorted\_eigenvectors[:, :k]  
selected\_eigenvectors

array([[ 0.70710678],  
 [-0.70710678]])

## Project the standardized data onto the selected principal components

This will reduce the dimensionality of the data.

# Project the standardized data onto the principal components  
projected\_data = standardized\_data.dot(selected\_eigenvectors)  
print("\nProjected Data:\n", projected\_data)  
projected\_data

Projected Data:  
 0  
0 -1.124534  
1 0.663489  
2 1.460958  
3 -0.999913

0  
0 -1.124534  
1 0.663489  
2 1.460958  
3 -0.999913