****

**Submitted By:** laiba binte tahir

**Reg No#** FA21-BSE-019

**Submitted To:** Dr. Mubashir Ahmad

# Question:

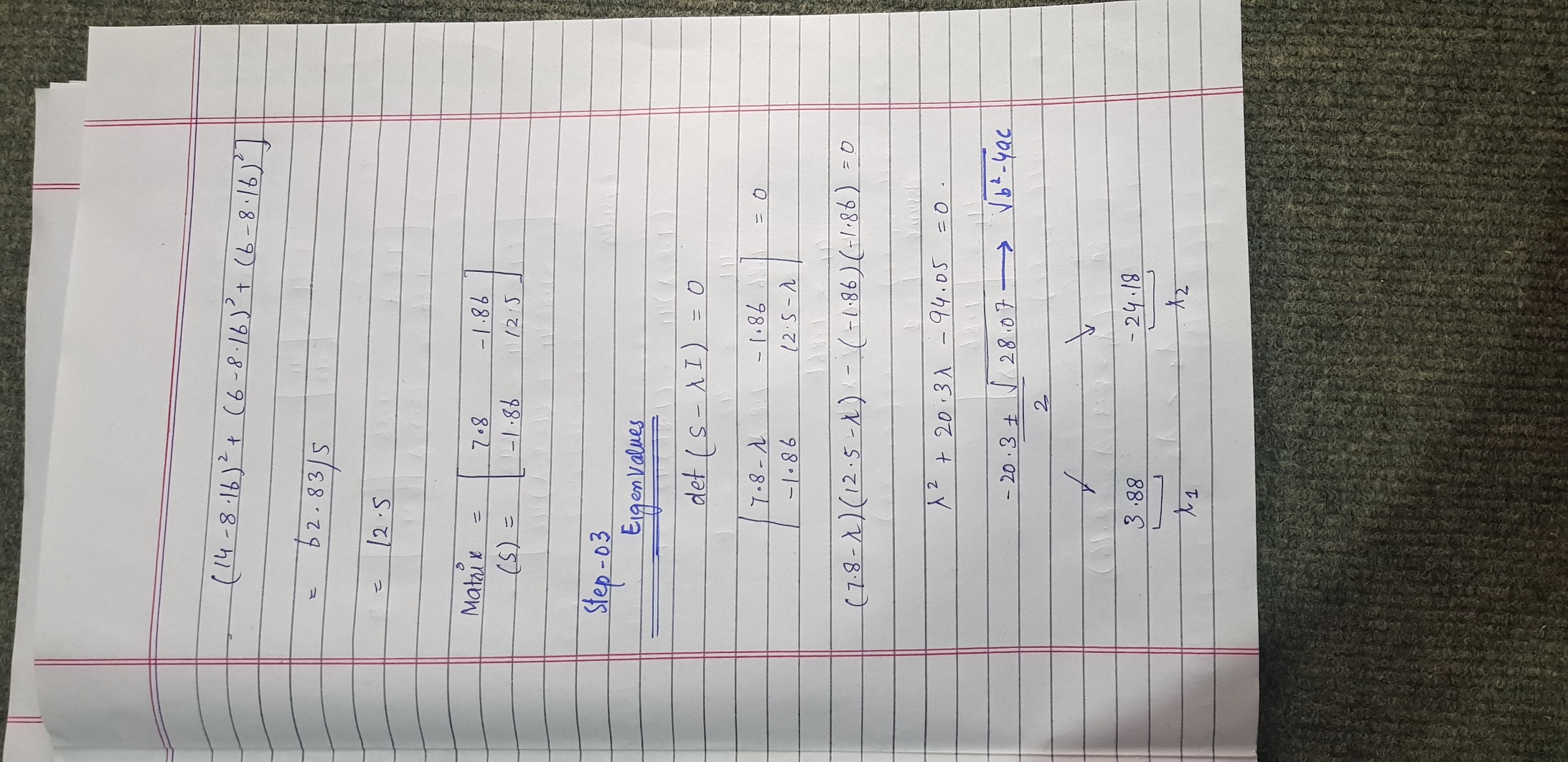
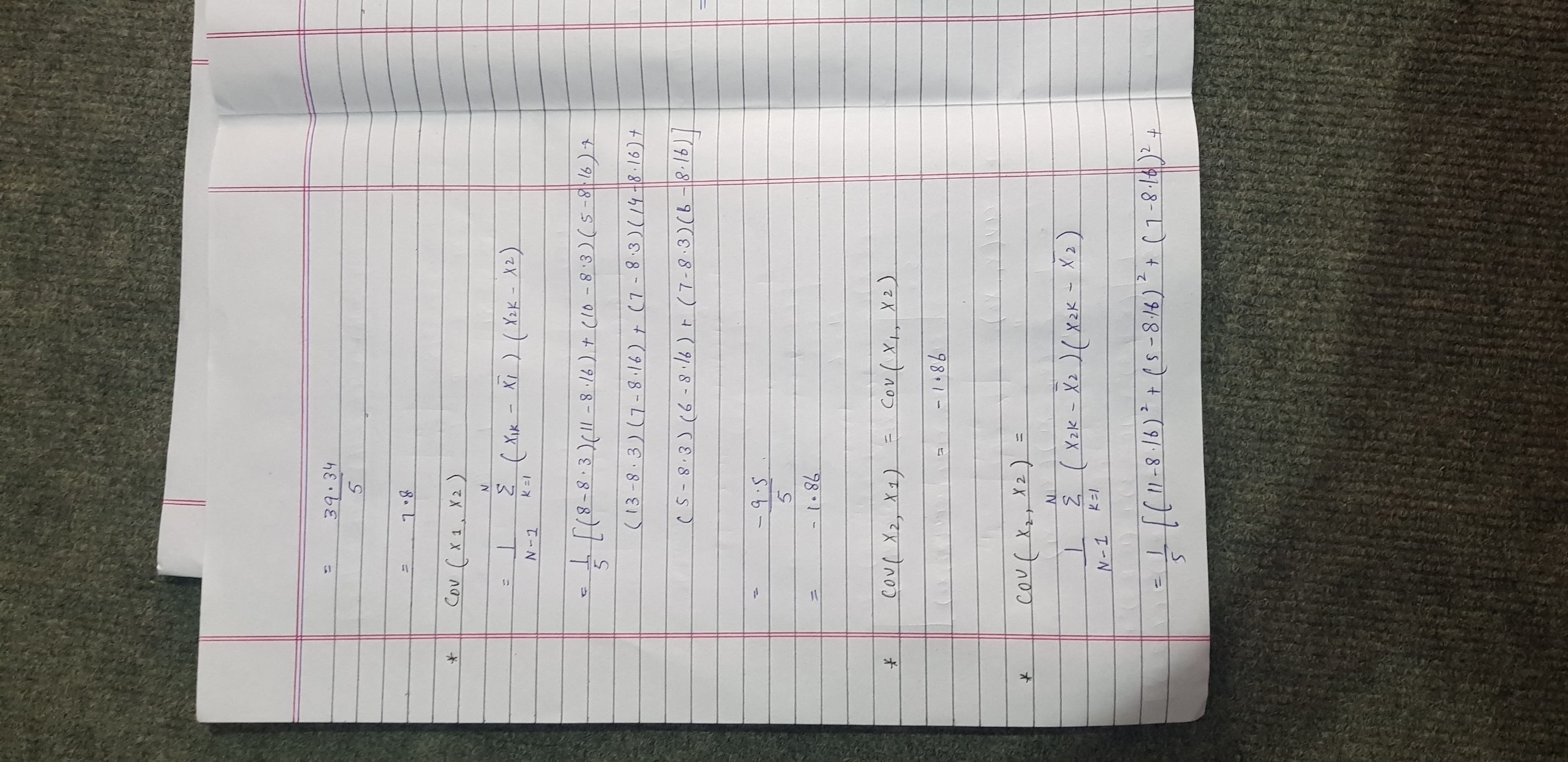
For a given dataset reduce the dimension from 2 to 1 using (PCA).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Features** | **Example 1** | **Example 2** | **Example 3** | **Example 4** | **Example 5** | **Example 6** |
| **X1** | 8 | 10 | 13 | 7 | 5 | 7 |
| **X2** | 11 | 5 | 7 | 14 | 6 | 6 |

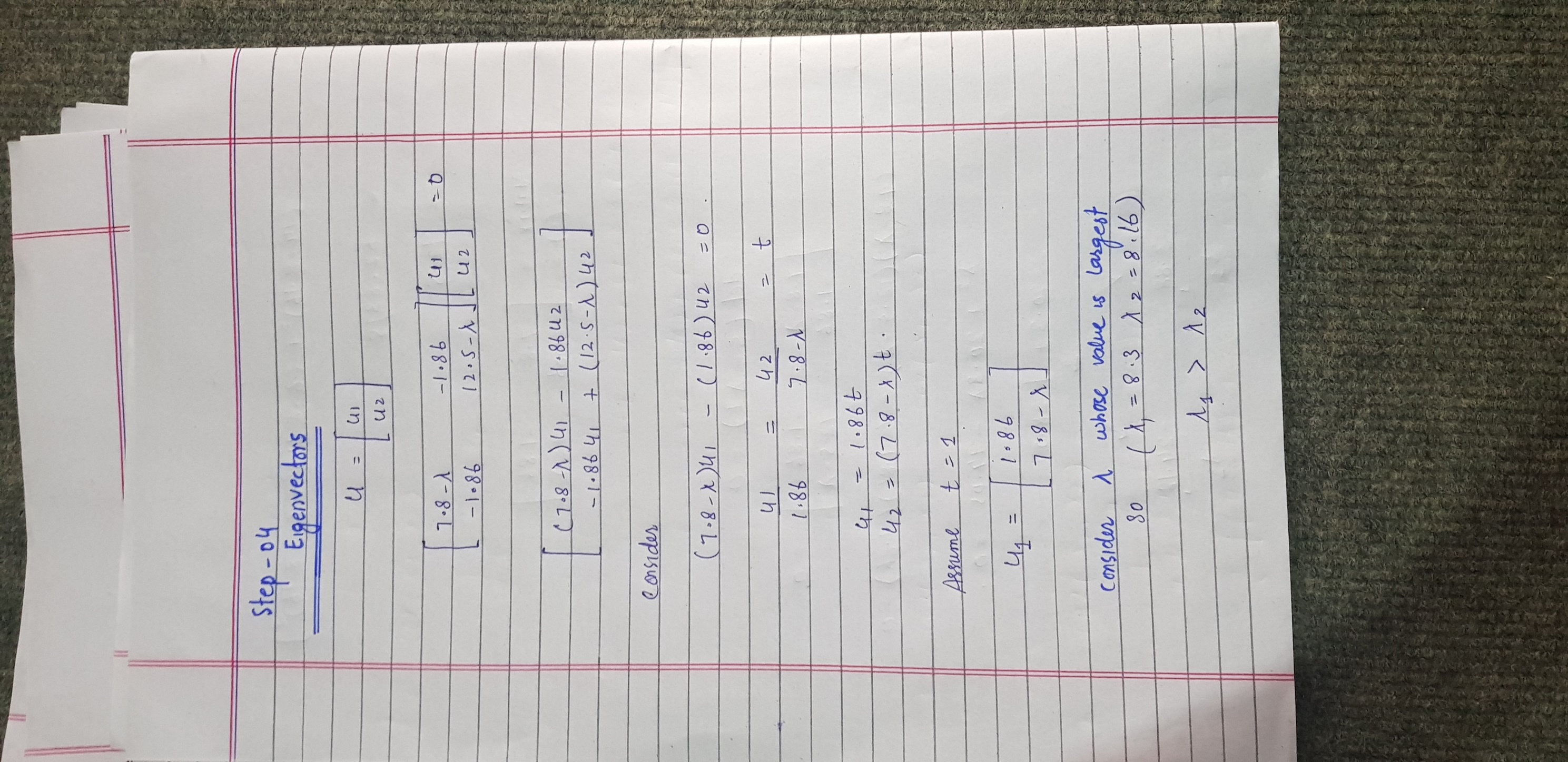
# Handwritten

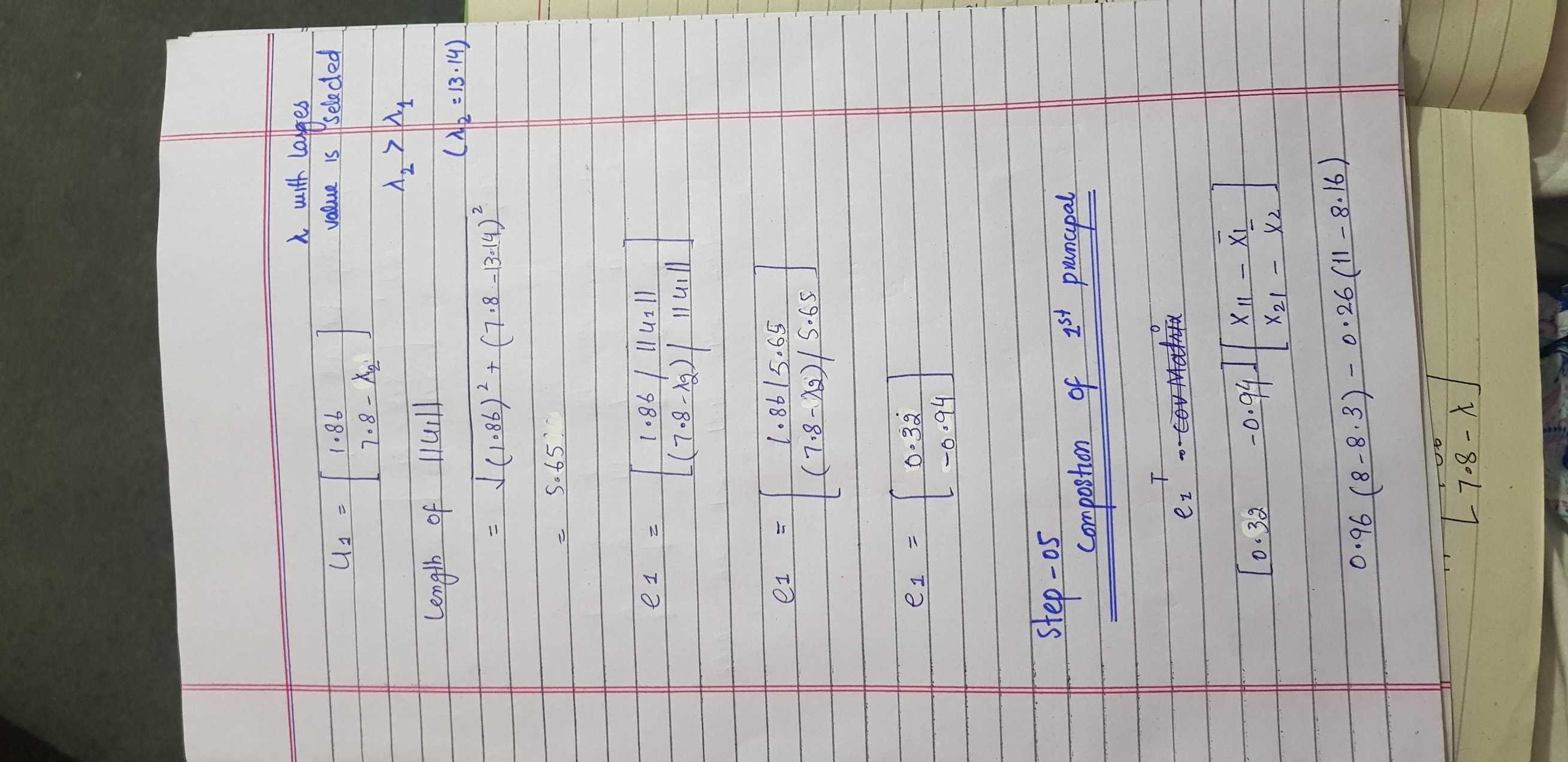
A piece of paper with writing on it

Description automatically generated

A person writing on a white board

Description automatically generated





A graph on a piece of paper

Description automatically generated

# Code In python

"""

Created on Sun Jun 16 12:55:13 2024

@author: Laiba Binta Tahir

"""

import numpy as np

import matplotlib.pyplot as plt

# Define the data

#Data

data = np.array([[8,11], [10,5], [13,7], [7,14], [5,6], [7,6]])

# Step-01 : calculate Mean

mean = np.mean(data,axis=0)

print("-----------------------------")

print("\t Mean : ", np.round\_(mean,2))

print("-----------------------------")

#Step-02 : center data

centered\_data = data - mean

# Step-03 : Covarience Matrix

cov\_matrix = np.cov(centered\_data, rowvar=False)

print("Covariance matrix : \n", np.round\_(cov\_matrix,2))

print("----------------------------------------")

#Step-04 : eigen values

eigenvalues, eigenvectors = np.linalg.eig(cov\_matrix)

print("EigenValues: ", np.round(eigenvalues,2))

print("----------------------------------------")

#Step-05 : eigen vectors

print("EigenVectors : \n", np.round(eigenvectors,2))

print("----------------------------------------")

#Step-06 Transform data

transfoemData = np.dot(centered\_data, eigenvectors[:, ::-1])

print("Transformed Data:\n", np.round(transfoemData,2))

print("----------------------------------------")

# Step 7:Reduce dimentionality

principalComponent\_1 = transfoemData[:,0]

print("Pricipal component: \n", np.round(principalComponent\_1,2))

print("-----------------------------------------")

#plot given data

plt.figure(figsize=(14,7))

plt.subplot(1,2,1)

plt.scatter(data[:,0],data[:,1], color='purple', label='data points')

plt.axhline(y=mean[1],color='orange', linestyle='--')

plt.axvline(x=mean[0],color='orange', linestyle='--')

plt.scatter(mean[0],mean[1], color='orange', label='mean')

plt.title("Original Dataset")

plt.xlabel("x1")

plt.ylabel("x2")

plt.legend()

# plot Principal components

plt.subplot(1, 2, 2)

plt.scatter(data[:,0],data[:,1], color='purple', label='data points')

plt.scatter(mean[0],mean[1], color='orange', label='mean')

for length, vec in zip(eigenvalues,eigenvectors.T):

v=vec\*np.sqrt(length)\*3

plt.plot([mean[0], mean[0] + v[0]], [mean[1], mean[1] + v[1]], color='orange')

plt.plot([mean[0], mean[0] - v[0]], [mean[1], mean[1] - v[1]], color='orange')

plt.title("Principal Component")

plt.xlabel("x1")

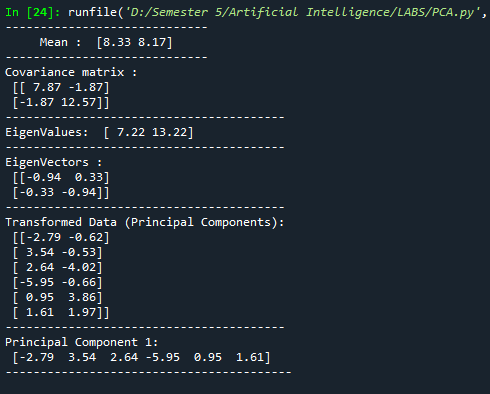
plt.ylabel("x2")

plt.legend()

plt.tight\_layout()

plt.show()

# Output



A graph of a graph with points and lines

Description automatically generated with medium confidence