Shantanu Pande - 181060057

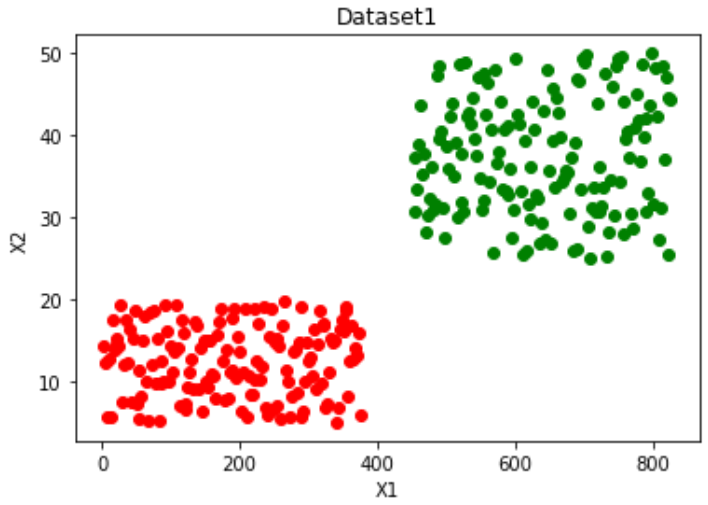
Data Science Lab Assignment 3

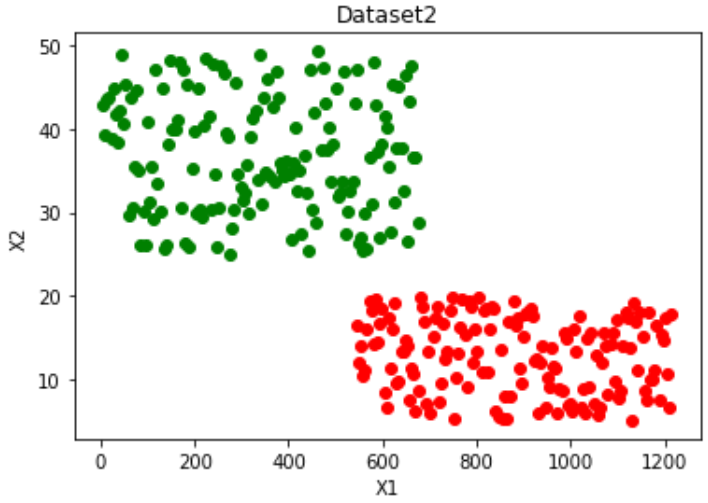
Problem Statement: During our classroom discussions, we have explored the utility of basis functions and Eigen analysis in context of classification. Please find attached the data files and explore the classification possibilities using the techniques studied in the class (classification in the original basis (all cases), and change of basis). The submission will contain the data plots with correct labels in the original space, the script, analytical form of the classifier, different classified results with various techniques, your observations and comments (the most important!).

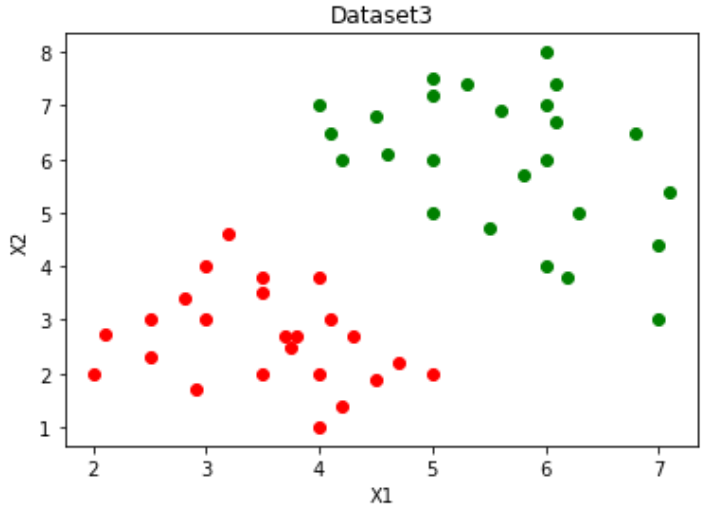
Code for original basis plots:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

def plot_in_org():
    df = pd.read_excel("D1.xlsx")
    df = pd.DataFrame(pd.read_excel("D1.xlsx"))
    df_1 = df.dropna(axis=1,how='all')
    df_1 = df.drop(["Unnamed: 4","Unnamed: 5"],axis=1)
    df_1 = df_1.dropna(axis=0)
    data1 = df_1
    data1
    colors = {'C1':'r', 'C2':'g'}
    fig, ax = plt.subplots()
    for i in range(len(data1['X1'])):
        ax.scatter(data1['X1'][i], data1['X2'][i],color=colors[data1['Class'][i]])
    ax.set_title('Dataset1')
    ax.set_xlabel('X1')
    ax.set_ylabel('X2')

    df_2 = pd.read_excel("D2.xlsx")
    data2 = df_2
    colors = {'C1':'r', 'C2':'g'}
    fig, ax = plt.subplots()
    for i in range(len(data2['X1'])):
        ax.scatter(data2['X1'][i], data2['X2'][i],color=colors[data2['Class'][i]])
    ax.set_title('Dataset2')
    ax.set_xlabel('X1')
    ax.set_ylabel('X2')

    df_3 = pd.read_excel("D3.xlsx")
    data3 = df_3
    colors = {'C1':'r', 'C2':'g'}
    fig, ax = plt.subplots()
    for i in range(len(data3['X1'])):
        ax.scatter(data3['X1'][i], data3['X2'][i],color=colors[data3['Class'][i]])
    ax.set_title('Dataset3')
    ax.set_xlabel('X1')
    ax.set_ylabel('X2')
    return data1, data2, data3
data1, data2, data3 = plot_in_org()
```

Plots in original space:



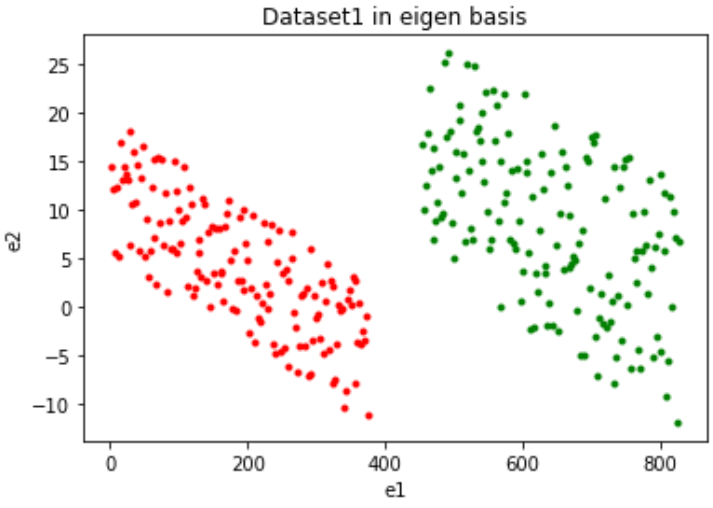


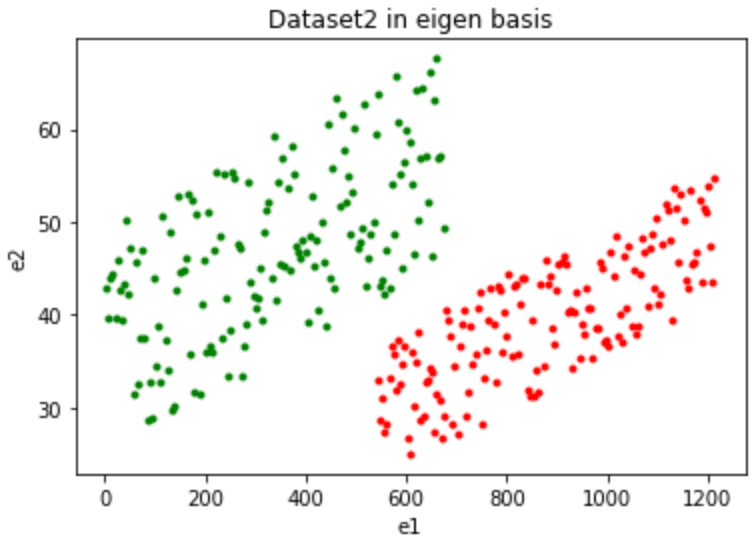


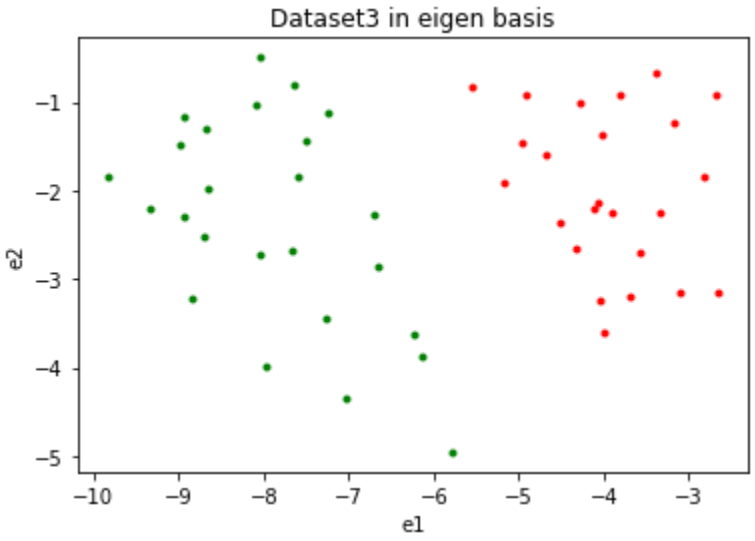
Code for analysis in eigen basis:

```
def calculate_eigen(data):
    dl_array = data.to_numpy()
    X = np.array([[],[]])
    for i in range (len(dl_array)):
        X = np.append(X,[[dl_array[i][1]],[dl_array[i][2]]], axis= 1)
    cov = np.cov(X,ddof=0)
    e,trans = np.linalg.eig(cov)
    sort_index = np.argsort(-1*e)
    e = e[sort_index]
    trans = trans[:,sort_index]
    trans = trans.T
    Y = np.matmul(trans,X)
    Y = np.array(Y)
    print(trans)
    class1 = np.array([[],[]])
    class2 = np.array([[],[]])
    eig_class1 = np.array([[],[]])
    eig_class2 = np.array([[],[]])
    for i in range(len(dl_array)):
        if dl_array[i][3] == "C1":
            class1 = np.append(class1, [[dl_array[i][1]], [dl_array[i][2]]], axis=1)
            eig_class1 = np.append(eig_class1, [[Y[0][i]], [Y[1][i]]], axis=1)
        else:
            class2 = np.append(class2, [[dl_array[i][1]], [dl_array[i][2]]], axis=1)
            eig_class2 = np.append(eig_class2, [[Y[0][i]], [Y[1][i]]], axis=1)
    plt.plot(eig_class1[0],eig_class1[1],'r.')
    plt.plot(eig_class2[0],eig_class2[1],'g.')
    plt.title("Dataset in eigen basis")
    plt.show()
calculate_eigen(data1)
calculate_eigen(data2)
calculate_eigen(data3)
```

Visualization in eigen basis:




Dataset2 in eigen basis
60
50
40
30
e2
0
200
400
600
800
1000
1200
e1


Dataset3 in eigen basis
−1
−2
−3
−4
−5
e2
−10
−9
−8
−7
−6
−5
−4
−3
e1

Code for data classification:

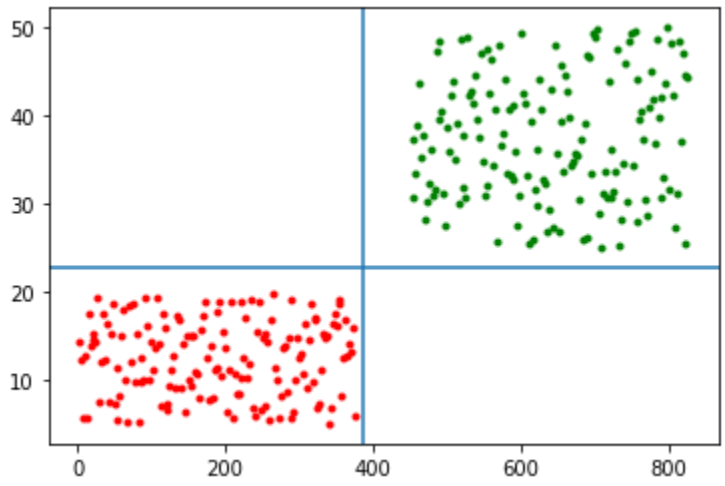
```
flag1 = 0
flag2 = 0
if (np.mean(class1[0])>np.mean(class2[0])):
    if (np.amin(class1[0])>np.amax(class2[0])):
        flag2 = 1
        print("X2")
    else :
        print("Not in orig ")
elif (np.mean(class1[0])==np.mean(class2[0])):
    flag2 = 0
    print("Not in orig ")
else:
    if (np.amin(class1[0])<np.amax(class2[0])):
        flag2 = 1
        print("X2")
    else :
        print("Not in orig ")
if (np.mean(class1[1])>np.mean(class2[1])):
    if (np.amin(class1[1])>np.amax(class2[1])):
        flag1 =1
        print("X1")
    else :
        print("Not in orig ")
        flag1 = 0
elif (np.mean(class1[1])==np.mean(class2[1])):
    flag1 = 0
    print("Not in orig ")
else:
    if (np.amin(class1[1])<np.amax(class2[1])):
        flag1 = 1
        print("X1")
    else :
        print("Not in orig ")
```

```
#eigen

if (np.mean(eig_class1[0])>np.mean(eig_class2[0])):
    if (np.amin(eig_class1[0])>np.amax(eig_class2[0])):
        flag2 = 1
        print("e2")
    else :
        print("Not in eigen ")
elif (np.mean(eig_class1[0])==np.mean(eig_class2[0])):
    flag2 = 0
    print("Not in eigen ")
else:
    if (np.amin(eig_class1[0])<np.amax(eig_class2[0])):
        flag2 = 1
        print("e2")
    else :
        print("Not in eigen ")
if (np.mean(eig_class1[1])>np.mean(eig_class2[1])):
    if (np.amin(eig_class1[1])>np.amax(eig_class2[1])):
        flag1 =1
        print("e1")
    else :
        print("Not in eigen ")
        flag1 = 0
elif (np.mean(eig_class1[1])==np.mean(eig_class2[1])):
    flag1 = 0
    print("Not in eigen ")
else:
    if (np.amin(eig_class1[1])<np.amax(eig_class2[1])):
        flag1 = 1
        print("e1")
    else :
        print("Not in eigen ")
```

```
if(flag1 == 1 and flag2 ==1):
    plt.plot(class1[0],class1[1],'r.')
    plt.plot(class2[0],class2[1],'g.')
    plt.axhline(np.amax(class1[1])+3)
    plt.axvline(np.amax(class1[0])+10)
    print('1')

elif(flag2 == 0 and flag1 ==1):
    plt.plot(class1[0],class1[1],'r.')
    plt.plot(class2[0],class2[1],'g.')
    plt.axhline(np.amax(class1[1])+2)
    print('2')
    #calculate_eigen(data1)
else:
    plt.plot(eig_class1[0],eig_class1[1],'r.')
    plt.plot(eig_class2[0],eig_class2[1],'g.')
    plt.axvline(np.amax(eig_class2[0])+0.05)
```

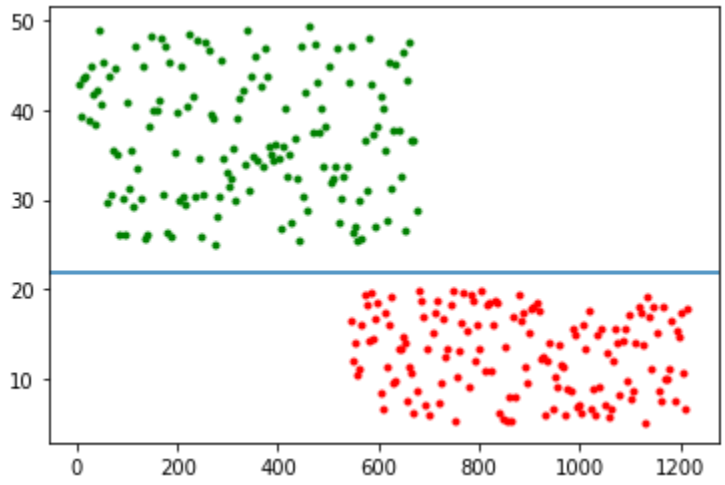
Outputs and observations:

Dataset 1



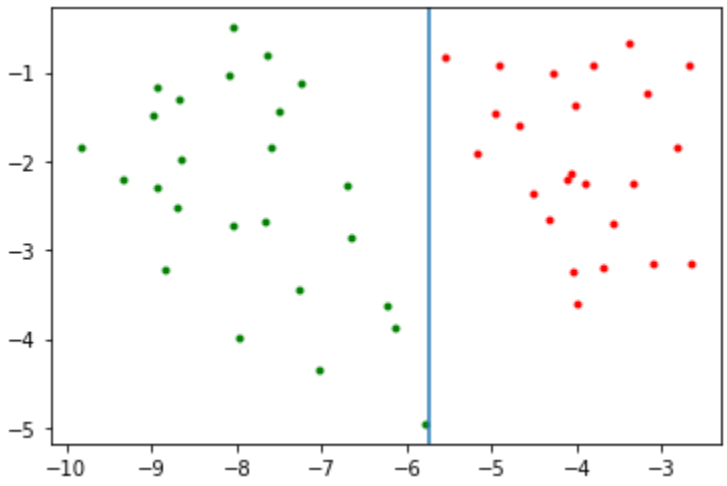
From the above result it is clear that the Dataset1 can be separated into classes C1 and C2 using the original basis of X1 and X2 and both X1 and X2 can act as separators here . Also after representing this data into eigen basis we can conclude that E2 can be used as a separator for the classes C1 and C2 in the eigen basis.

Dataset 2



From the above result it is clear that the Dataset2 can be separated into classes C1 and C2 using the original basis and X1 acts as separators here . Also after representing this data into eigen basis we can conclude that neither E1, nor E2 can be used as a separator for the classes C1 and C2 in the eigen basis.

Dataset 3



The above separation is obtained only after representing the Dataset3 in the eigen basis , where e2 acts as a separator. In the original basis we were not able to separate the classes using X1 or X2 as separators. So, with the help of eigen basis we were able to separate the classes C1 and C2 , which was not possible in the original basis of X1 and X2.

Conclusion:

Different datasets were visualised in the original as well as the eigen basis and using this the individual datasets could be classified and separated into classes C1 and C2 with the help of either the original basis or the eigen basis. And the results were verified analytically and mathematically as depicted above.