PROJECT REPORT

Assignment Project Report: Face Feature Extraction
NAME: RISHI RAJ SINGH
AI AND ML B-4

Problem Statement:
Using PCA creates a face recognition system that gives access to only certain people. To implement this, you can use the LFW_peoples dataset provided in the sci-kit-learn library. Given this dataset, use only those classes that have a minimum (use min_faces_per_person = 70, resize = 0.4) 70 images (should give you only 11 classes). Given this subset of images, apply PA to obtain the corresponding eigen face for each class. You can additionally train a classifier for recognition purposes.

Prerequisites:
- Software: Python 3
Tools:

- Pandas
- Numpy
- Matplotlib
- Seaborn
- Scipy

Dataset:

The data source used for this project is been generated using the sk-learn library it's a famous dataset that consists of famous personality images.

Implementation:
PCA IS USED FOR DETECTION

In [1]:
```
from sklearn.datasets import fetch_lfw_people
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.neural_network import MLPClassifier
```

Loading Dataset:

In [2]:
```
dataset=fetch_lfw_people(min_faces_per_person=70,resize=0.4) #resize value will reduce the target feature
x=dataset.data
y=dataset.target
target_names=dataset.target_names
images=dataset.images
```

Splitting data for training and Computing PCA:

In [9]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1)
```

In [10]:
```
x_train.shape
```

Out[10]: (1159, 1850)

In [11]:
```
p1=PCA(n_components=500)
p1.fit(x_train)
```

Out[11]: PCA(n_components=500)

Classifier :

In [15]:
```
clf=MLPClassifier(hidden_layer_sizes=(128,),batch_size=128,verbose=True,early_stopping=True)
clf.fit(x_train_trans_1,y_train)
```

```
Iteration 1, loss = 11.62155228
Validation score: 0.284483
Iteration 2, loss = 7.26393055
Validation score: 0.491379
Iteration 3, loss = 4.42709035
Validation score: 0.612069
Iteration 4, loss = 2.21960940
Validation score: 0.663793
Iteration 5, loss = 1.07162274
Validation score: 0.672414
Iteration 6, loss = 0.44493115
Validation score: 0.681034
Iteration 7, loss = 0.14600975
Validation score: 0.715517
Iteration 8, loss = 0.04876872
Validation score: 0.706897
Iteration 9, loss = 0.01728741
Validation score: 0.698276
Iteration 10, loss = 0.00025804
```

Evaluating The Model:

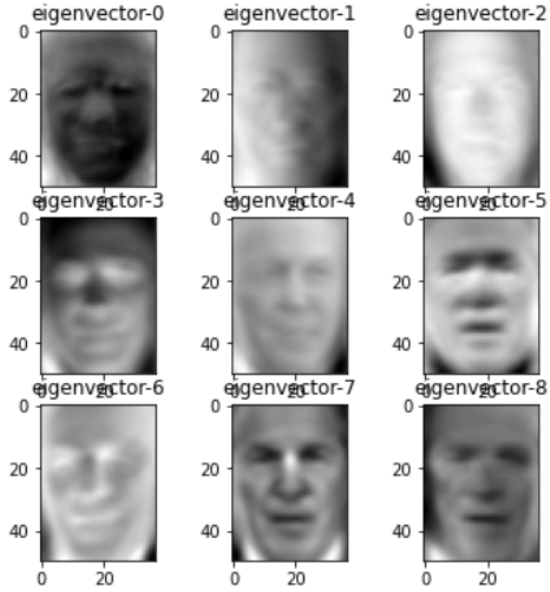
In [16]:
```
y_pred=clf.predict(x_test_trans_1)
print(classification_report(y_test,y_pred,target_names=target_names))
```

```
                   precision    recall  f1-score   support

     Ariel Sharon       0.67      0.60      0.63        10
     Colin Powell       0.78      0.64      0.70        22
  Donald Rumsfeld       0.64      0.50      0.56        18
    George W Bush       0.74      0.80      0.77        50
Gerhard Schroeder       0.50      0.50      0.50        10
      Hugo Chavez       0.56      0.56      0.56         9
       Tony Blair       0.53      0.80      0.64        10

         accuracy                           0.67       129
        macro avg       0.63      0.63      0.62       129
     weighted avg       0.68      0.67      0.67       129
```

Plotting The Most Significant Eigen face

In [38]:
```
def plot_grid(images,titles,h,w,rows=3,cols=3):
    plt.figure(figsize=(2*cols,2*rows))
    for i in range(rows*cols):
        plt.subplot(rows,cols,i+1)
        plt.imshow(images[i].reshape(h,w),cmap='gray')
        plt.title(titles[i])
```

Final Output :


eigenvector-0
eigenvector-1
eigenvector-2
eigenvector-3
eigenvector-4
eigenvector-5
eigenvector-6
eigenvector-7
eigenvector-8
0
20
40