Guided Project : Face Feature Extraction

Name	Lakshmi Thirunavukkarasu
Course	AI and ML (Batch 5)
Problem Statement	Build a classification model for face recognition purpose after dimensional reduction using PCA

Prerequisites

Software Requirements:

1. Anaconda

2. Python 3.8

3. Python Packages

- scikit-learn

- numpy

- matplotlib

Dataset Used	LFW_peoples
Method for Dimensionality reduction	PCA
Algorithm used Classifier	Scikit learn MLPClassifier

Step 1: Load the Dataset

```
In [2]: lfw_people = fetch_lfw_people(min_faces_per_person=70, resize=0.4)
```

```
In [3]: # the label to predict is the id of the person
y = lfw_people.target
print(np.unique(y, return_counts = True))
target_names = lfw_people.target_names
n_classes = target_names.shape[0]

print("n_classes: %d" % n_classes)
print(target_names)

print("Image shape")
n_samples, h, w = lfw_people.images.shape
print(n_samples, h, w)

X = lfw_people.data
print(X.shape)
n_features = X.shape[1]
print(n_features)
```

Step 2: Identify number of components for Dimensionality Reduction

In [92]:
```
from sklearn.decomposition import PCA
p = PCA(n_components=500)
p.fit(X_train)
print("Transformed Shape:" , p.transform(X_train).shape)
```

```
Transformed Shape: (1159, 500)
```

In [93]:
```
cmp = p.components_
print(cmp.shape)
```

```
(500, 1850)
```

In [94]:
```
var_sum = np.sum(p.explained_variance_)
print("Sum of explained Variance", var_sum)
sorted_index = np.argsort(var)[::-1]
```

```
Sum of explained Variance 2575006.8
```

Number of components that explanins 98% variance

In [99]:
```
temp_sum = 0
principal_vec = []
principal_val = []
i = 0
while (temp_sum < 0.98*var_sum):
    principal_vec.append(cmp[sorted_index[i],:])
    principal_val.append(var[sorted_index[i]])
    temp_sum += var[sorted_index[i]]
    i += 1
print("Number of components is {}", format(i))
```

```
Number of components is {} 229
```

Step 3: Plot Eigen Faces

Plot Eigen Faces

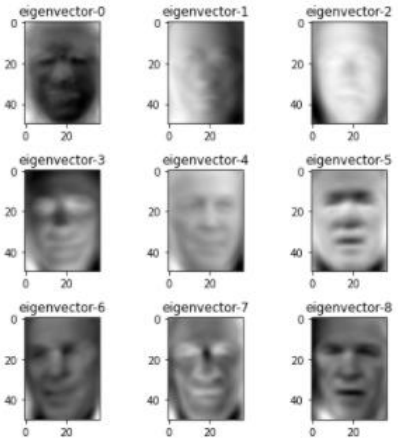
In [101]:
```
n_components = 228
mean_imgs = []
for i in range(n_components):
    v = principal_vec[i,:]
    img = v.reshape(h,w)
    mean_imgs.append(img)
mean_imgs = np.array(mean_imgs)
print(mean_imgs.shape)
```

```
(228, 50, 37)
```

In [102]:
```
titles =[ f"eigenvector-{i}" for i in range(n_components)]
len(titles)
```

Out[102]:
```
228
```

In [103]:
```
plot_grid(mean_imgs, titles,h,w)
```



Step 4: Build a classification Model

In [112]:
```
from sklearn.neural_network import MLPClassifier
classifier = MLPClassifier(hidden_layer_sizes=(256,),batch_size=128,verbose=True,early_stopping=True)
classifier.fit(X_train_trans,y_train)
```

```
C:\Users\venka\miniconda3\envs\MLEnv\lib\site-packages\sklearn\utils\validation.py:590: FutureWarning: np.matrix usage is deprecated in 1.0 and will raise a TypeError in 1.2. Please convert to a numpy array with np.asarray. For more information see: https://numpy.org/doc/stable/reference/generated/numpy.matrix.html
  FutureWarning,

Iteration 1, loss = 10.77097746
Validation score: 0.301724
Iteration 2, loss = 8.75281673
Validation score: 0.474138
Iteration 3, loss = 5.98967501
Validation score: 0.465517
Iteration 4, loss = 4.60511570
Validation score: 0.465517
```

Step 5: Build the accuracy metrics

Prediction Metrics

In [113]:
```
from sklearn.metrics import classification_report
y_pred = classifier.predict(X_test_trans)
print(classification_report(y_test,y_pred,target_names=target_names))
```

```
                   precision    recall  f1-score   support

     Ariel Sharon       0.50      0.57      0.53         7
     Colin Powell       0.86      0.81      0.83        31
  Donald Rumsfeld       0.54      0.70      0.61        10
    George W Bush       0.91      0.92      0.92        53
Gerhard Schroeder       0.64      0.64      0.64        11
      Hugo Chavez       0.75      0.50      0.60         6
       Tony Blair       0.50      0.45      0.48        11

         accuracy                           0.78       129
        macro avg       0.67      0.66      0.66       129
     weighted avg       0.78      0.78      0.78       129
```