CS 111. Homework 8

Xiao Zhou

1. load the images

```
Face = plt.imread( "faces/face000.bmp", format="bmp" )
HEIGHT = WIDTH = 64
imageToVector = lambda Bitmap: Bitmap.reshape( -1 ).astype( np.float64 )
vectorToImage = lambda Vector: Vector.clip( 0, 255 ).reshape( HEIGHT, WIDTH ).astype( np.uint8 )
renderVector = lambda Vector: plt.imshow( vectorToImage( Vector ), cmap="gray" )
scaleImageIntensities = lambda A: np.round( (A - np.min( A )) / (np.max( A )-np.min( A )) * 255 ).astype( np.uint8 )
```

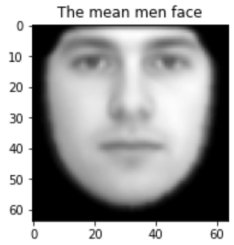
```
import face_descriptions
M = []
F = []
for fileName in sorted( face_descriptions.face_features ):
    if fileName not in face_descriptions.image_to_omit:
        bitMap = plt.imread( "faces/" + fileName )
        if (face_descriptions.face_features[fileName][0] == 'Male'):
            M.append(imageToVector(bitMap))
        else:
            F.append(imageToVector(bitMap))
print(len(M))
print(len(F))
```

```
73
96
```

2. Compute and render the mean male face, and the mean female face

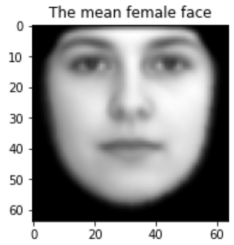
```
M = np.array(M)
mm = np.mean(M,axis = 0)
plt.figure( figsize=(3,3) )
plt.title( r"The mean male face" )
renderVector( mm )
```

```
<matplotlib.image.AxesImage at 0x7ffa0f7e3a00>
```




```
F = np.array(F)
mf = np.mean(F,axis = 0)
plt.figure( figsize=(3,3) )
plt.title( r"The mean female face" )
renderVector( mf )
```

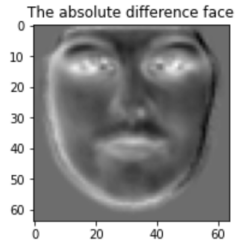
```
<matplotlib.image.AxesImage at 0x7ff09e6b9310>
```



3. the absolute difference between the mean male face and the mean female face.

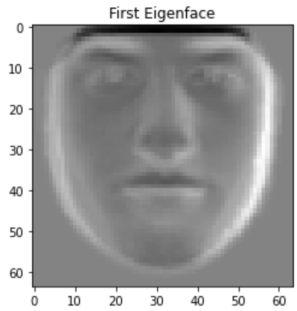
```
v = scaleImageIntensities(mm-mf)
plt.figure( figsize=(3,3) )
plt.title( r"The absolute difference face" )
renderVector( v )
```

```
<matplotlib.image.AxesImage at 0x7fa79e6e5e50>
```



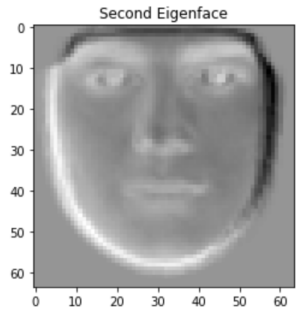
4. Compute the Eigenfaces for males, V_{male}, and females, V_{female}.

```
X_1 = M - mm
C = np.cov(X_1.T)
U, sigma, Vt = spla.svd(C)
Eigenfaces1 = Vt.T
plt.title( "First Eigenface" )
renderVector( scaleImageIntensities( Eigenfaces1[:,0] ) )
plt.show()
```



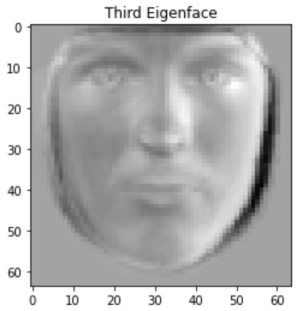

```
plt.title( "Second Eigenface" )
renderVector( scaleImageIntensities( Eigenfaces1[:,1] ) )
```

```
<matplotlib.image.AxesImage at 0x7ff09f4d97c0>
```

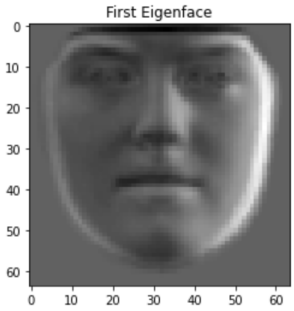


```
plt.title( "Third Eigenface" )
renderVector( scaleImageIntensities( Eigenfaces1[:,2] ) )
```

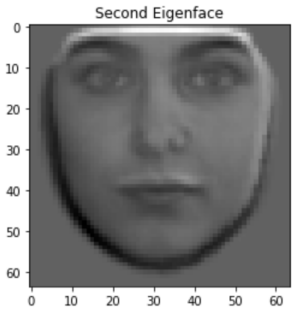
```
<matplotlib.image.AxesImage at 0x7ff0288c98e0>
```



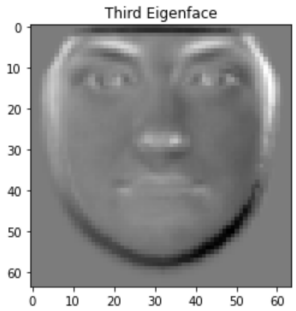

```
X_2 = F - mf
C = np.cov(X_2.T)
U, sigma, Vt = spla.svd(C)
Eigenfaces2 = Vt.T
plt.title( "First Eigenface" )
renderVector( scaleImageIntensities( Eigenfaces2[:,0] ) )
plt.show()
```




```
plt.title( "Second Eigenface" )
renderVector( scaleImageIntensities( Eigenfaces2[:,1] ) )
plt.show()
```



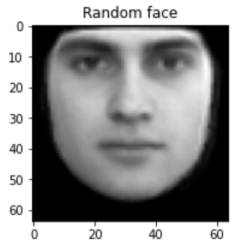

```
plt.title( "Third Eigenface" )
renderVector( scaleImageIntensities( Eigenfaces2[:,2] ) )
plt.show()
```



5. generate and render a random male and a random female face.

```
k = 20
coe = X_1@Eigenfaces1[:,:k]
mu = np.mean(coe, axis = 0)
std = np.std(coe, ddof = 1, axis = 0)
newFaceCoeffs = std * np.random.randn( k ) + mu
newFace = mm + Eigenfaces1[:,:k] @ newFaceCoeffs
plt.figure( figsize=(3,3) )
plt.title( "Random face" )
renderVector( newFace )
```

```
<matplotlib.image.AxesImage at 0x7ff0532ba4c0>
```




```
k = 20
coe = X_2@Eigenfaces2[:,:k]
mu = np.mean(coe, axis = 0)
std = np.std(coe, ddof = 1, axis = 0)
newFaceCoeffs = std * np.random.randn( k ) + mu
newFace = mf + Eigenfaces2[:,:k] @ newFaceCoeffs
plt.figure( figsize=(3,3) )
plt.title( "Random face" )
renderVector( newFace )
```

```
<matplotlib.image.AxesImage at 0x7ff0285e3ee0>
```