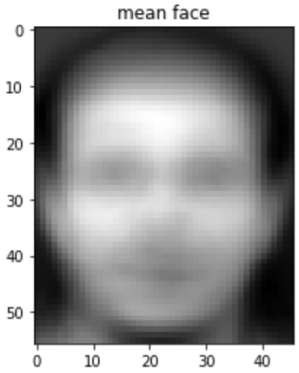
DLCV HW0　　R09942091 謝承軒

使用 python

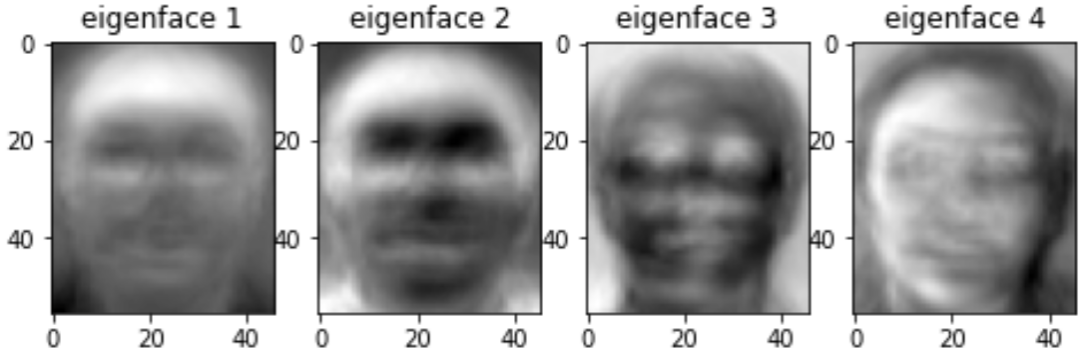
1. 將每張臉轉成一維-> 1*2576

   train_X.shape: 360*2576

   mean: train_X.mean(axis=0)



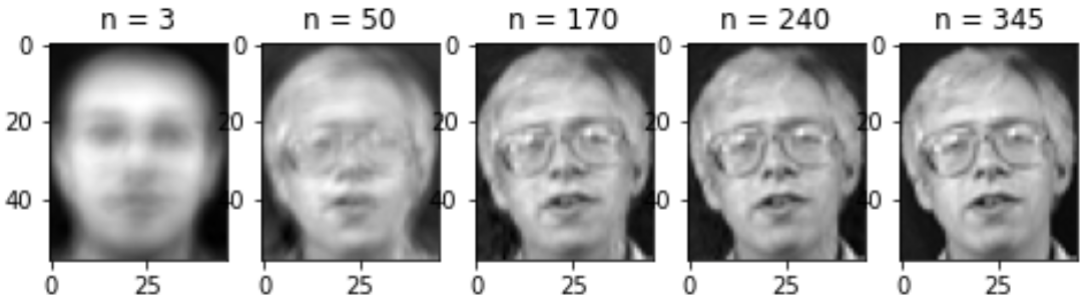
   eigenface_n: pca.fit(train_X – mean_face) .components_[n]).reshape(56,46)



2. 2_1.png

   先得到 space: pca.transform(img - mean_face)

   reconstruct face: mean_face + np.dot(eigenspace[:,:n], output.components_[:n])



3. Mse 計算: np.mean((original_img – reconstruct_face)**2)

```
n= 3 , mse: 746.7994084720323
n= 50 , mse: 236.55430882685195
n= 170 , mse: 46.71704972752627
n= 240 , mse: 13.366611255635418
n= 345 , mse: 0.21543397076007192
```

4. 使用 sklearn 的 cross_val_score
   然後計算 acc:
   cross_val_score(knn,X=trainsform_X[:, :n],y=train_Y,cv=3,scoring='accuracy')
   最後取 mean 當作最後 acc

```
------- k = 1 -------
n = 3  , acc: 0.65
n = 50  , acc: 0.9611111111111111
n = 170  , acc: 0.9555555555555556

------- k = 3 -------
n = 3  , acc: 0.6111111111111112
n = 50  , acc: 0.9
n = 170  , acc: 0.8888888888888888

------- k = 5 -------
n = 3  , acc: 0.5611111111111111
n = 50  , acc: 0.8472222222222222
n = 170  , acc: 0.8222222222222223
```

   從平均 acc 決定 -> (k, n) = (1, 50)

5. 先 pca 去 fit k=1, n=50
   然後直接 pca.predict 轉換後的 test (n=50)
   再計算 acc 即可

```
acc in testing data: 0.925
```