T1.

Iteration: 0

w: [[1.19202922e-01 8.80797076e-01 1.81545808e-09]
[7.31058579e-01 2.68941421e-01 1.69570706e-16]
[2.68941421e-01 7.31058579e-01 1.01529005e-11]
[9.99983299e-01 1.67014218e-05 2.03105874e-42]
[9.99088949e-01 9.11051194e-04 5.37528453e-32]
[9.99876605e-01 1.23394576e-04 3.30529272e-37]
[2.31952283e-16 1.38879439e-11 1.00000000e+00]
[2.31952283e-16 1.38879439e-11 1.00000000e+00]
[3.30570063e-37 5.90009054e-29 1.00000000e+00]]

m: [0.45757242 0.20909425 0.33333333]

mean: [[5.78992692 5.81887265]
[1.67718211 2.14523106]
[-4. -4.66666666]]

Cov: [[[12.31988634 0.]
[0. 12.23304914]]

[[0.62066718 0.]
[0. 0.15261824]]

[[5.66666667 0.]
[0. 5.66666668]]]

==========

Iteration: 1

w: [[1.81294622e-002 9.81582998e-001 2.87540002e-004]
[5.64494061e-001 4.35380622e-001 1.25316584e-004]
[1.92846943e-002 9.80633501e-001 8.18047710e-005]
[1.00000000e+000 4.70826685e-062 5.03915978e-012]
[9.99999990e-001 3.82283898e-027 1.01750382e-008]
[1.00000000e+000 7.32691444e-043 2.49006494e-010]
[1.59795152e-003 7.69753019e-045 9.98402048e-001]

[1.55069753e-003 5.89612118e-058 9.98449302e-001]
[3.59091243e-006 1.89005654e-144 9.99996409e-001]]
m: [0.40056227 0.26639968 0.33303805]
mean: [[6.30842698 6.31259558]
[1.77218759 2.1815904]
[-4.00062813 -4.66675525]]
Cov: [[[3.24482139 0.]
[0. 3.18737779]]

[[0.54812076 0.]
[0. 0.14993733]]

[[4.67362081 0.]
[0. 2.89766742]]]

==========

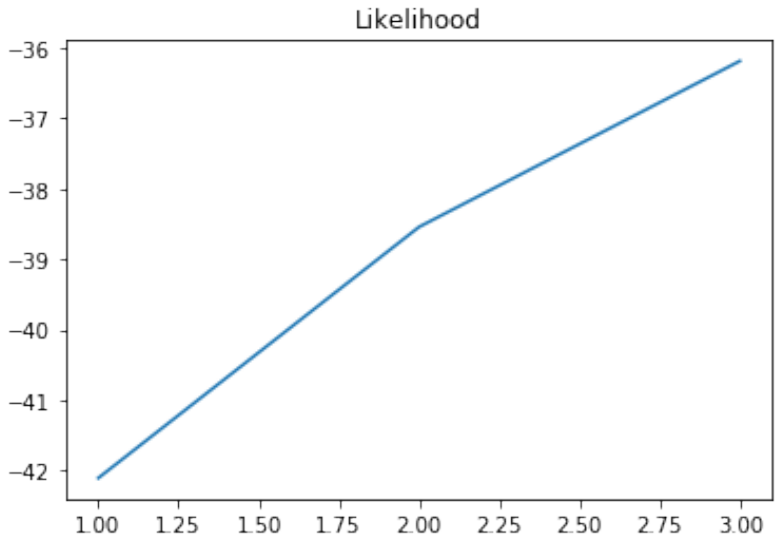
Iteration: 2
w: [[1.81240251e-004 9.99812736e-001 6.02360765e-006]
[1.40770128e-001 8.59229229e-001 6.42663215e-007]
[4.85324724e-004 9.99513546e-001 1.12958651e-006]
[1.00000000e+000 7.30562640e-064 3.40271752e-019]
[1.00000000e+000 4.88274872e-028 5.02342563e-014]
[1.00000000e+000 3.05867029e-044 1.26696752e-016]
[4.85980138e-012 2.29092578e-047 1.00000000e+000]
[3.11260608e-012 1.78986020e-060 1.00000000e+000]
[1.09172834e-023 1.82739807e-151 1.00000000e+000]]
m: [0.34904852 0.31761728 0.3333342]
mean: [[6.81963839 6.81969608]
[1.95082009 2.30058161]
[-3.9999862 -4.66664913]]
Cov: [[[1.58836939 0.]
[0. 1.58356938]]

[[0.67983404 0.   ]
 [0.   0.22439122]]

[[4.6667292 0.   ]
 [0.   2.88899996]]]

==========

T2. Yes it goes up every iteration.



T3.

Iteration: 0

w: [[9.99999985e-01 1.52299795e-08]
 [1.00000000e+00 2.31952283e-16]
 [1.00000000e+00 3.77513454e-11]
 [1.00000000e+00 2.03109266e-42]
 [1.00000000e+00 5.38018616e-32]
 [1.00000000e+00 3.30570063e-37]
 [2.31952283e-16 1.00000000e+00]
 [2.31952283e-16 1.00000000e+00]
 [3.30570063e-37 1.00000000e+00]]

m: [0.66666666 0.33333334]

mean: [[4.50000001 4.66666667]
 [-3.99999997 -4.66666663]]

Cov: [[[9.16666668 0.]
[0. 8.66666669]]

[[5.66666672 0.]
[0. 5.66666677]]]

==========

Iteration: 1
w: [[9.94979696e-01 5.02030393e-03]
[9.99922646e-01 7.73541258e-05]
[9.98623856e-01 1.37614400e-03]
[1.00000000e+00 6.27933891e-12]
[9.99999994e-01 6.33185482e-09]
[1.00000000e+00 2.12626993e-10]
[2.77132409e-03 9.97228676e-01]
[2.45908782e-03 9.97540912e-01]
[1.30217751e-06 9.99998698e-01]]
m: [0.66652866 0.33347134]
mean: [[4.49739004 4.66243446]
[-3.9912655 -4.65434481]]
Cov: [[[6.94971903 0.]
[0. 5.94046426]]

[[4.72011919 0.]
[0. 2.98099996]]]

==========

Iteration: 2
w: [[9.99840572e-01 1.59427561e-04]
[9.99999613e-01 3.86910944e-07]
[9.99967636e-01 3.23639718e-05]
[1.00000000e+00 2.77855078e-18]

[1.00000000e+00 1.61298139e-13]

[1.00000000e+00 7.52961598e-16]

[2.56424026e-04 9.99743576e-01]

[1.65413279e-04 9.99834587e-01]

[6.05550189e-09 9.99999994e-01]]

m: [0.66669218 0.33330782]

mean: [[4.49960686 4.66618544]

[-3.99986439 -4.66641867]]

Cov: [[[6.9196161 0.]

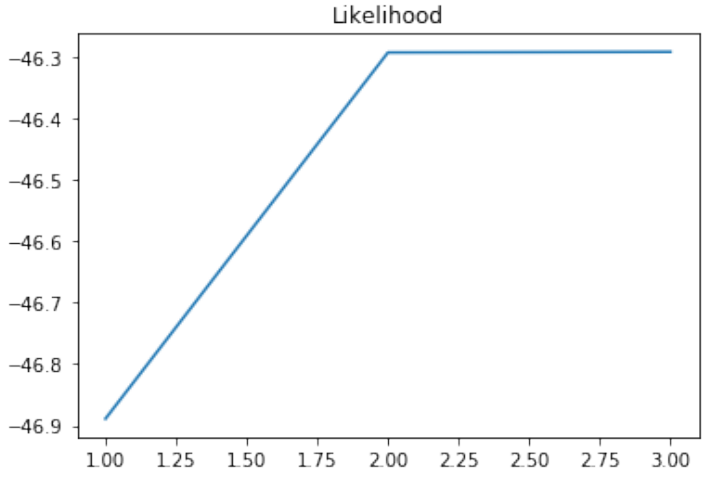
[0. 5.89303254]]

[[4.66851519 0.]

[0. 2.89184297]]]

==========

T4. 3 Mixtures have better log likelihood because it goes up every iteration and the log likelihood value is higher than the case of 2 Mixtures.



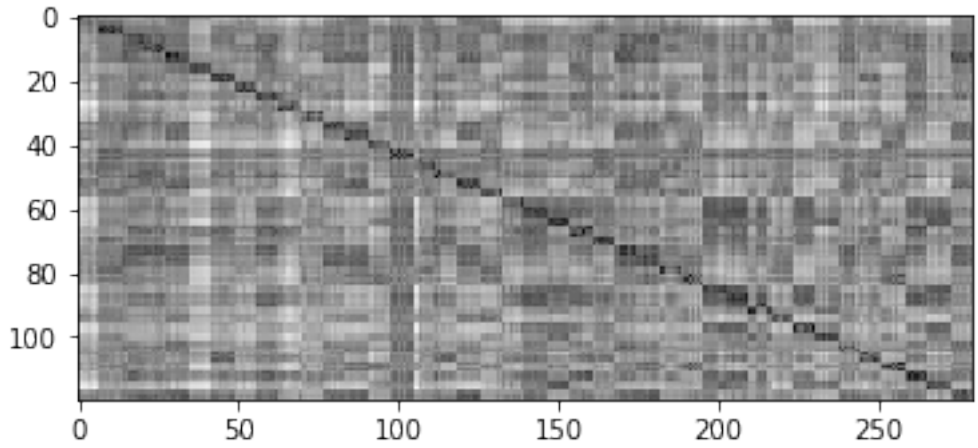
T5. Distance between xf[0, 0] and xf[0, 1] is 10.037616294165492

Distance between xf[0, 0] and xf[1, 0] is 8.173295099737281

No, it does not make sense because the distance between image 0 of person 0 and image 0 of person 1 is shorter than the distance between image 0 of person 0 and image 1 of person 0.

Yes, I think it might be useful, but we have to do some operation to them to reduce the case that it make the prediction wrong.

T6.

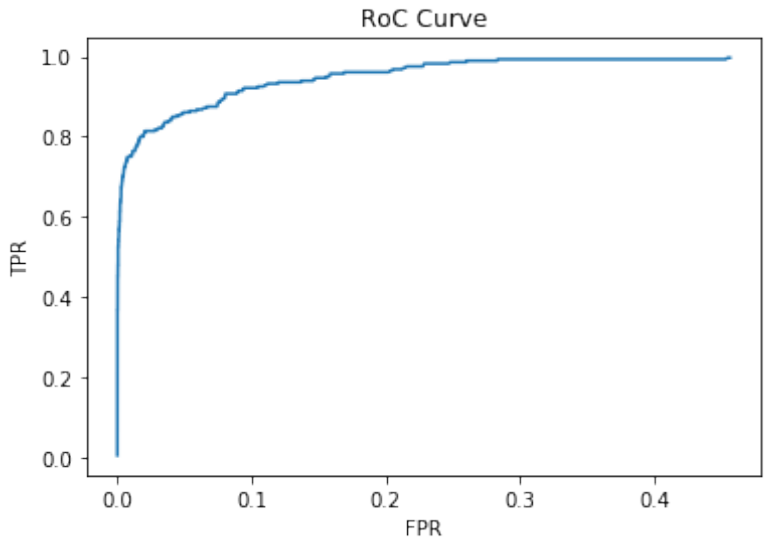


T7. The black square suggest that it is the closest image, which is itself.

The pattern from Person number 2 show that the image of person number 2 is not quite close together compare to the square pattern found form Person 1 because we can observe the darker grey square in the square pattern of Person number 2. Meaning that image of person number 1 is not much difference from each other.

T8. TPR is 0.99642857 and FAR is 0.45641026.

T9.



2 should be minimum threshold

10 should be maximum threshold

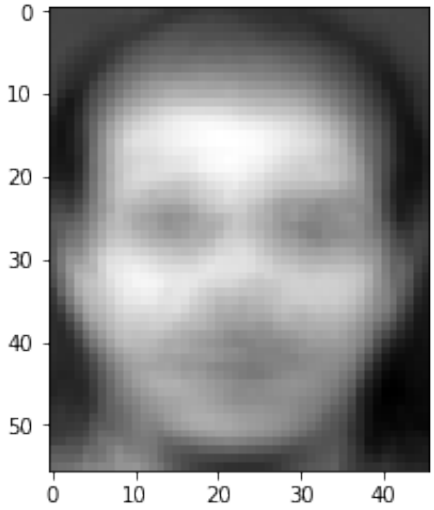
P.S. Minimum and maximum threshold are select by observe FPR and TPR at t = 0 through t = 10.

T10. At 0.1% FPR, Recall is 0.5464285714285714

EER is at TPR = 0.9107142857142857 and FPR = 0.08937728937728938

EER = 0.089

T11.



T12. Size of covariance matrix is 2576 x 2576.

The rank of covariance matrix is 119.

We expect to get 119 non-zero eigenvalues.

T13. The size of gram matrix is 120x120.

The rank of gram matrix is 119.

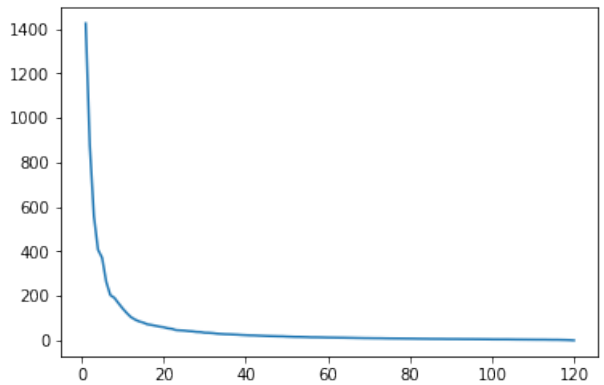
We expect to get 119 non-zero eigenvalues.

T14. Yes.

Consider A^TA, if we transpose this matrix we get $(A^T)^TA^T$ which is A^TA.

Using this observation, Gram matrix calculation is in the form above.

Therefore, Gram matrix is symmetric.



T15. There are 119 non-zero eigenvalues.

T16. We should use 64 eigenvectors.

T17. The images are shown in T17 section in ipynb file.

T18. I think the first eigenvector capture the overall characteristic of person face also with some part of the hair and the second eigenvector capture specific part of the face such as eyes, hairs, nose and mouse.

Yes, the biggest variance have been captured in these 2 eigenvectors because it capture the overall and some specific part of the face that can be use to recognise the face image.

T19. At 0.1% FPR, Recall is 0.5142857142857142

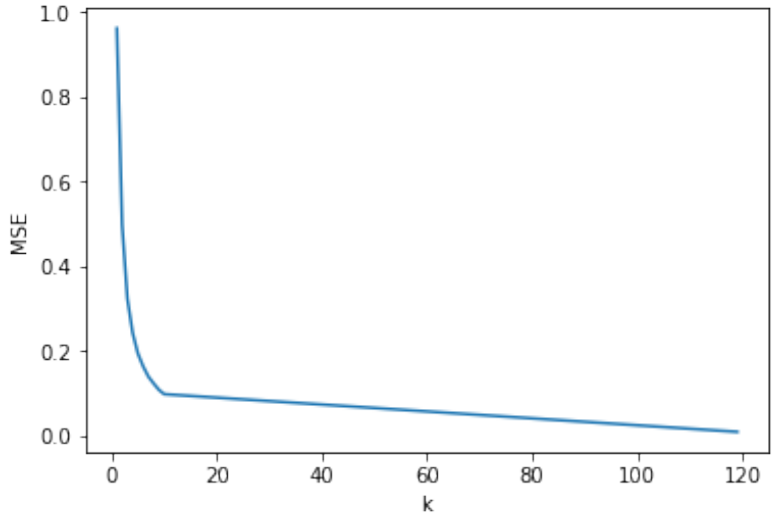
EER is at TPR = 0.9214285714285714 and FPR = 0.07774725274725275

EER = 0.0777

T20. k = 11 give the best EER.

OT1. The MSE is 0.005612765405463091

OT2. The reconstruct images are shown in OT2 section in ipynb file.



OT3. We have to use 10.304 GB (Use 1000 as divider) to store 1000000 images of this type without PCA.

If we compress it with PCA by using first 10 eigenvalues, we can store it in 40 MB.

T21. We need to keep 80 dimensions.

T22. LDA Projection is calculated in T22 section in ipynb file.

T23. The results are shown in the T23 section in ipynb file.

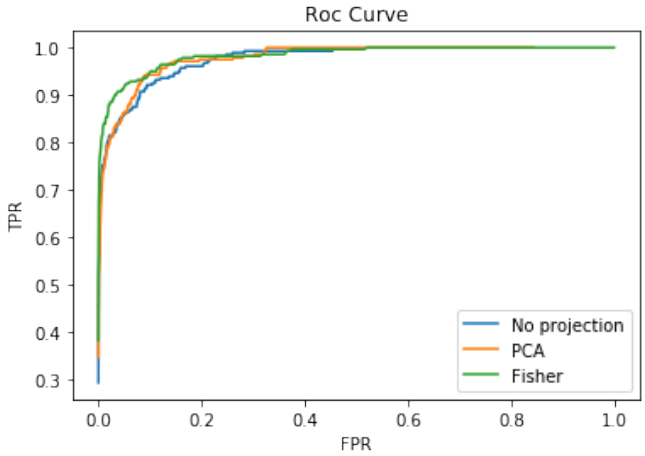
We can observe that the result from LDA can describe the variance better than the case of PCA (Of course, we can see a significant white face for all the plotted image from LDA). Also, LDA plotted faces are quite same for each plotted image and it does not capture some specific part of face like PCA does.

T24. At 0.1% FPR, Recall is 0.6821428571428572

EER is at TPR = 0.9285714285714286 and FPR = 0.07133699633699633

EER = 0.0713

T25.



First you can see that the PCA experiment and no projection experiment give a quite same RoC curve result, at some point (around FPR = 0.1 to 0.2) PCA give us a better TPR, but overall the performance of these 2 experiments have not much difference.

Second, if we observe the RoC curve from the Fisher experiment you can see a quite different result compare to 2 other experiments. Fisher method out perform a better performance, we can see a large gap between Fisher's RoC curve and 2 others curve around FPR = 0.0 through FPR = 0.1, this mean that the Fisher's experiment give us a better model.