HW 2

Arnab Dey
Student ID: 5563169
Email: dey00011@umn.edu

Solution 1.a

In multi-variate case when $\boldsymbol{x}$ is d-dimensional and normal distributed, we have

$$P(\boldsymbol{x}|C_i) = \prod_{t=1}^{N_i} \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_i|^{1/2}} exp\left[-\frac{1}{2}\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)\right]$$

where N_i is the total number of samples in class C_i, $\boldsymbol{\Sigma}_i$ is the covariance matrix for the variables belonging to each sample of class C_i, $\boldsymbol{\mu}_i$ is the mean vector for samples in class C_i.
The log-likelihood function to estimate $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ is given as follows:

$$\begin{aligned} L(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i|\boldsymbol{x}) &= \sum_{t=1}^{N_i} \ln\left(\frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_i|^{1/2}} exp\left[-\frac{1}{2}\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)\right]\right) \\ &= -\frac{N_i d}{2}\ln(2\pi) - \frac{N_i}{2}\ln|\boldsymbol{\Sigma}_i| - \frac{1}{2}\sum_{t=1}^{N_i}\left(\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)\right) \end{aligned} \tag{1}$$

From Eq.1, to find the estimate of $\boldsymbol{\mu}_i$, what we denote as $\boldsymbol{m}_i$, we set the derivative of log-likehood function w.r.t $\boldsymbol{\mu}_i$ to 0.

$$\begin{aligned} &\frac{\partial L}{\partial \boldsymbol{\mu}_i} = 0 \\ &\implies 0 = -\frac{1}{2}\sum_{t=1}^{N_i}\left(2\boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{x}^t - \boldsymbol{m}_i\right)(-1)\right) \\ &\implies 0 = \sum_{t=1}^{N_i}\left(\boldsymbol{\Sigma}_i^{-1}\boldsymbol{x}^t - \boldsymbol{\Sigma}_i^{-1}\boldsymbol{m}_i\right) \\ &\implies N_i\boldsymbol{m}_i = \sum_{t=1}^{N_i}\boldsymbol{x}^t \qquad [Pre - multiplying\ by\ \boldsymbol{\Sigma}_i] \\ &\implies \boldsymbol{m}_i = \frac{1}{N_i}\sum_{t=1}^{N_i}\boldsymbol{x}^t \end{aligned} \tag{2}$$

Similarly, we can find the estimate of $\boldsymbol{\Sigma}_i$. Before, doing that let us write the terms of log-likelihood function which depends on $\boldsymbol{\Sigma}_i$ as the other terms will eventually become 0 when we will take the derivative. We will also use the fact that

$$\boldsymbol{x}^T A\boldsymbol{x} = trace[\boldsymbol{x}^T A\boldsymbol{x}] = trace[\boldsymbol{x}\boldsymbol{x}^T A]$$

The log-likelihhod function involving the terms that depend on $\boldsymbol{\Sigma}_i$ can be written as:

$$\begin{aligned} L' &= -\frac{N_i}{2}\ln|\boldsymbol{\Sigma}_i| - \frac{1}{2}\sum_{t=1}^{N_i}\left(\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)\right) \\ &= \frac{N_i}{2}\ln|\boldsymbol{\Sigma}_i^{-1}| - \frac{1}{2}\sum_{t=1}^{N_i}\left(trace\left[\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)\right]\right) \\ &= \frac{N_i}{2}\ln|\boldsymbol{\Sigma}_i^{-1}| - \frac{1}{2}\sum_{t=1}^{N_i}\left(trace\left[\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\right]\right) \end{aligned} \tag{3}$$

From Eq.1, to find the estimate of $\mathbf{\Sigma}_i$, what we denote as $\boldsymbol{S}_i$, we can equivalently set the derivative of L' w.r.t $\mathbf{\Sigma}_i^{-1}$ to 0, i.e.

$$\begin{aligned}
&\frac{\partial L'}{\partial \mathbf{\Sigma}_i^{-1}} = 0 \\
\implies & 0 = \frac{N_i}{2}\boldsymbol{S}_i - \frac{1}{2}\sum_{t=1}^{N_i}\left(\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)^T\right) \\
\implies & \boldsymbol{S}_i = \frac{1}{N_i}\sum_{t=1}^{N_i}\left(\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)\left(\boldsymbol{x}^t - \boldsymbol{\mu}_i\right)^T\right)
\end{aligned} \tag{4}$$

Using the estimate of $\boldsymbol{\mu}_i$, $\boldsymbol{m}_i$, we can write

$$\boldsymbol{S}_i = \frac{1}{N_i}\sum_{t=1}^{N_i}\left(\left(\boldsymbol{x}^t - \boldsymbol{m}_i\right)\left(\boldsymbol{x}^t - \boldsymbol{m}_i\right)^T\right) \tag{5}$$

For model 1, where $\boldsymbol{S_1}$ and $\boldsymbol{S_2}$ are independent, we have to use the equations as shown in Eq.5 and Eq.2. For model 2, we assume that $\boldsymbol{S}$ is shared between two classes, therefore, we need to take the expectation of what is given in Eq.5. Hence,

$$\boldsymbol{S_1} = \boldsymbol{S_2} = P(C_1)\boldsymbol{S_1} + P(C_2)\boldsymbol{S_2}$$

where $\boldsymbol{S_i}$ is given in Eq.5.
For model 3, we assume that variables in the samples of each class are independent. Therefore, in this case we have to take only the diagonal terms of corresponding $\boldsymbol{S}_i$ from Eq.5 setting all the off-diagonal terms to 0.

Solution 1.c

Table 1 shows the error rates for different models on different test sets.

Model	1	2	3
test set 1	30.0%	24.5%	25.0%
test set 2	4.5%	21.0%	14.5%
test set 3	23.5%	25.5%	21.5%

Table 1: Q1.c: Error-rates for different models and different test sets

From the table if we match the data pair to the model which gives lowest error rates on the test data then we can conclude the following:

data pair	Chosen model
data pair 1	2
data pair 2	1
data pair 3	3

Table 2: Q1.c: Chosen model for each data pair based on lowest error rate on test data set

Explanantion of different error rates with different models

When we choose independent S_1 and S_2, the discriminant is non-linear. Moreover, when model 2 is chosen, the discriminant becomes linear and finally if model 3 is chosen, we assume that the variables are independent. Therefore, as data pair 2 gives lowest error rate with model 1, we can say that the data in the data pair 2 is not linearly separable and does not have independent variables. Data pair 1 can be linearly separable but variables are not independent. For data pair 3, the data are linearly separable and variables are independent also.

Solution 2.a

Error rates for different k in k-nearest neighbor algorithm on Optdigit dataset have been tabulated in the table below:

k	1	3	5	7
Error rate(%)	5.387	4.040	4.377	5.387

Table 3: Q2.a: Error-rate Vs. k table

Solution 2.b

We performed PCA on Optdigits training data and found the following proportion of variance plot:

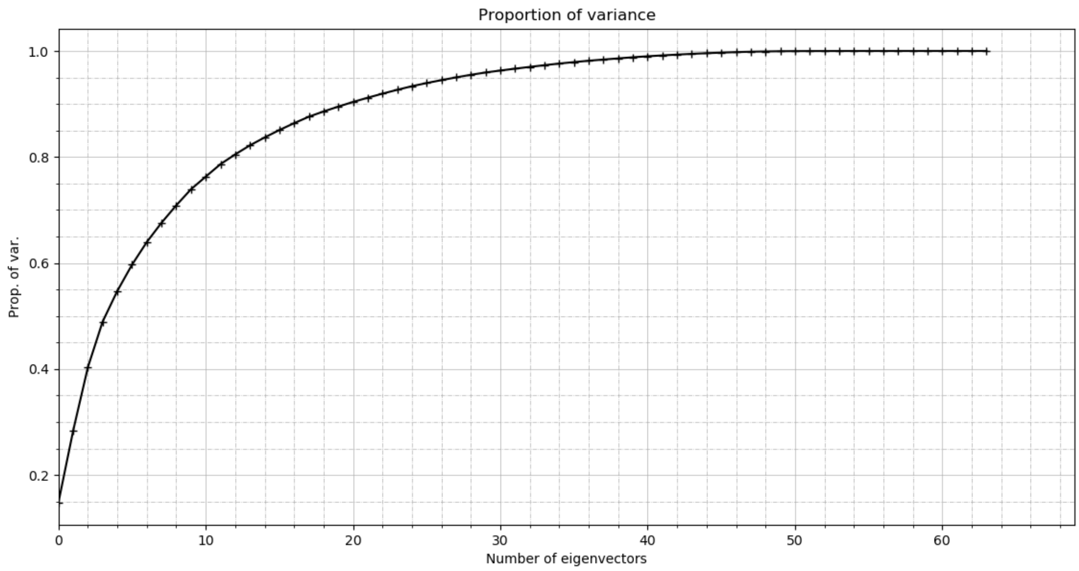

Figure 1: Proportion of variance plot for Optdigits training data

We can see from Fig.1 that the minimum number of eigenvectors that explain at least 90% of the variance is 20.
Therefore, we used 20 principal components for PCA and reduced the dimension of the original Optdigits data to 20. Then, we used KNN on this reduced dimension Optdigits test data. The following table shows the error rates for different k in k-nearest neighbor algorithm on reduced Optdigits test data.

k	1	3	5	7
Error rate(%)	4.040	4.040	4.040	4.377

Table 4: Q2.b: Error-rate Vs. k table

Solution 2.c

Fig.2 shows the both Optdigits training and test data after PCA with 2 components.

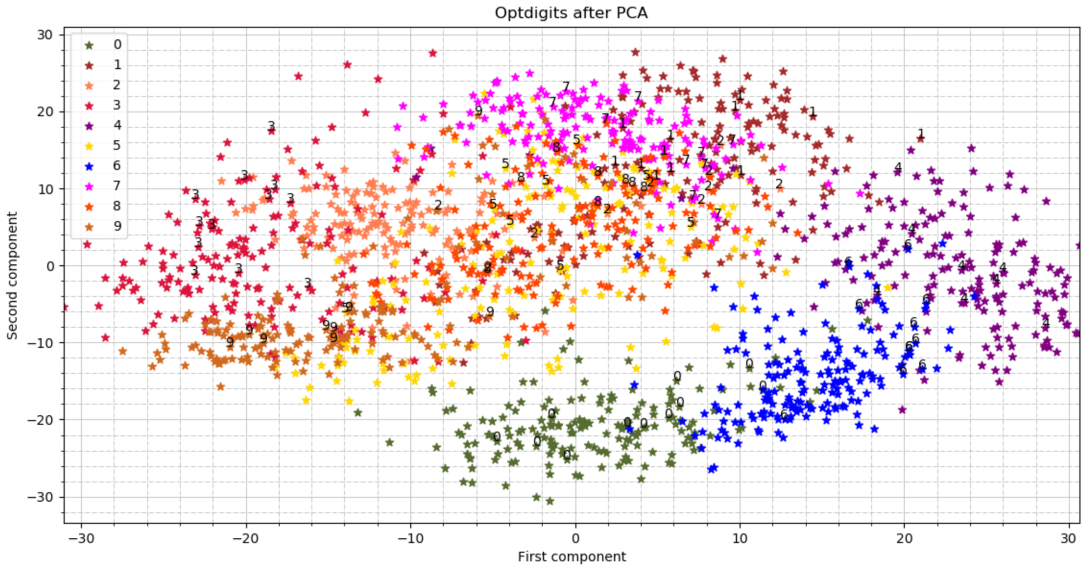

Figure 2: Optdigits data after PCA

Solution 2.d

Table 5 shows error rates for different L dimensions and different k neighbors for KNN algorithm on Optdigits test data.

L	2	4	9
k=1	44.781%	19.191%	9.764%
k=3	41.414%	18.518%	9.427%
k=5	40.740%	15.824%	9.427%

Table 5: Q2.d: Error-rates for different L dimensions and k neighbors

Solution 2.e

Fig.3 shows the both Optdigits training and test data after LDA with 2 components.

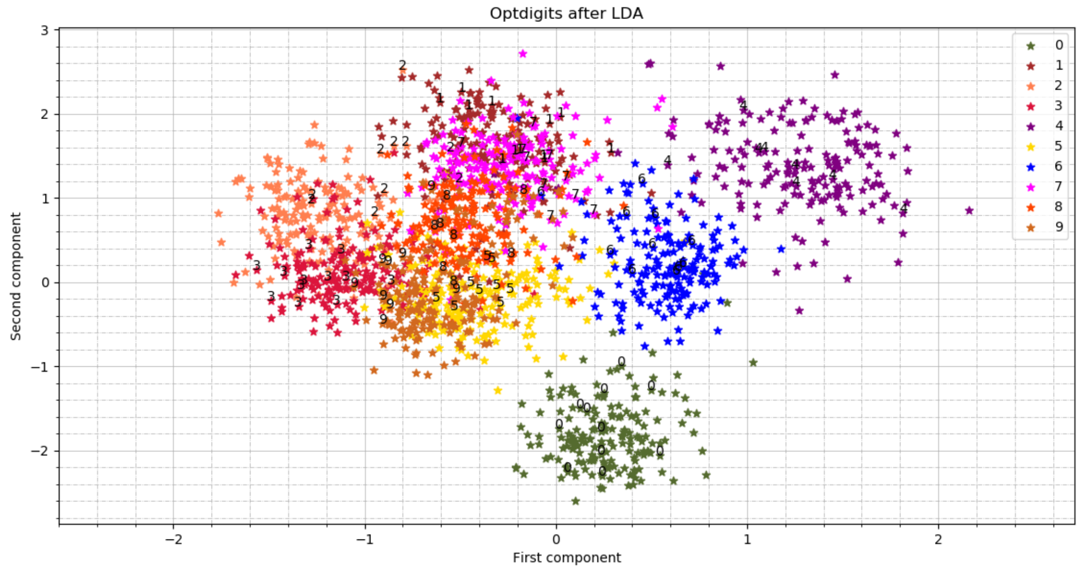

Figure 3: Optdigits data after LDA

Solution 3.a

The mean face is shown in Fig.4 The first 5 eigen-faces are shown in Fig.5

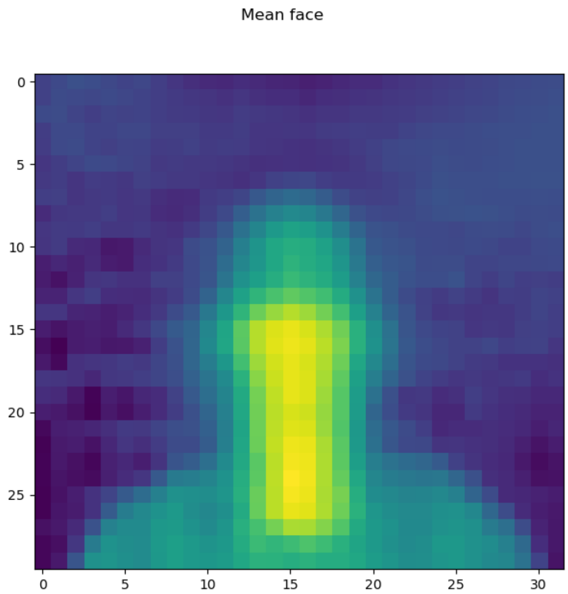

Figure 4: Mean face

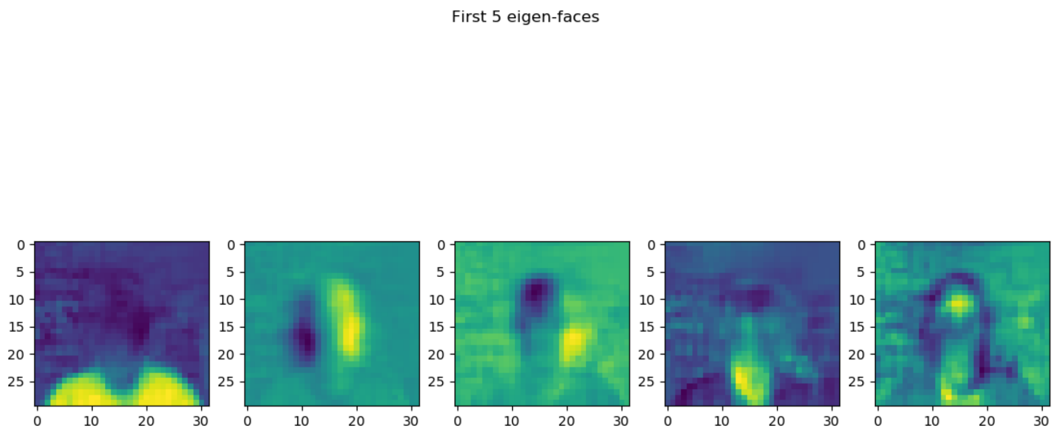

Figure 5: First 5 eigen-faces

Solution 3.b

We performed PCA on face training data and found the following proportion of variance plot:

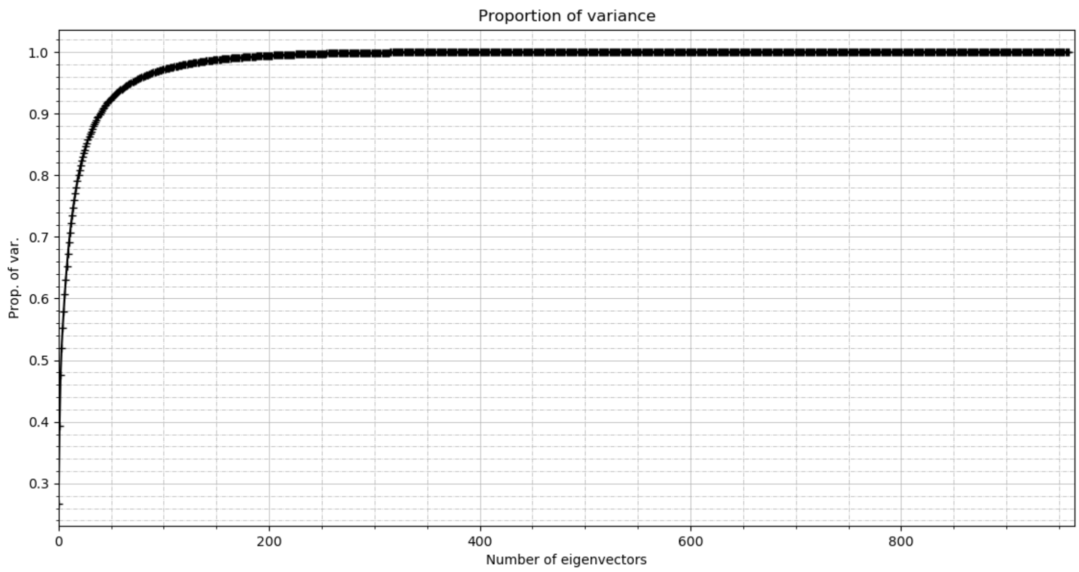

Figure 6: Proportion of variance plot for face training data

We can see from Fig.6 that the minimum number of eigenvectors that explain at least 90% of the variance is 40.
Therefore, we used 40 principal components for PCA and reduced the dimension of the original face data to 40. Then, we used KNN on this reduced dimension face test data. The following table shows the error rates for different k in k-nearest neighbor algorithm on reduced face test data.

k	1	3	5	7
Error rate(%)	10.483	24.193	39.516	39.516

Table 6: Q3.b: Error-rate Vs. k table

Solution 3.c

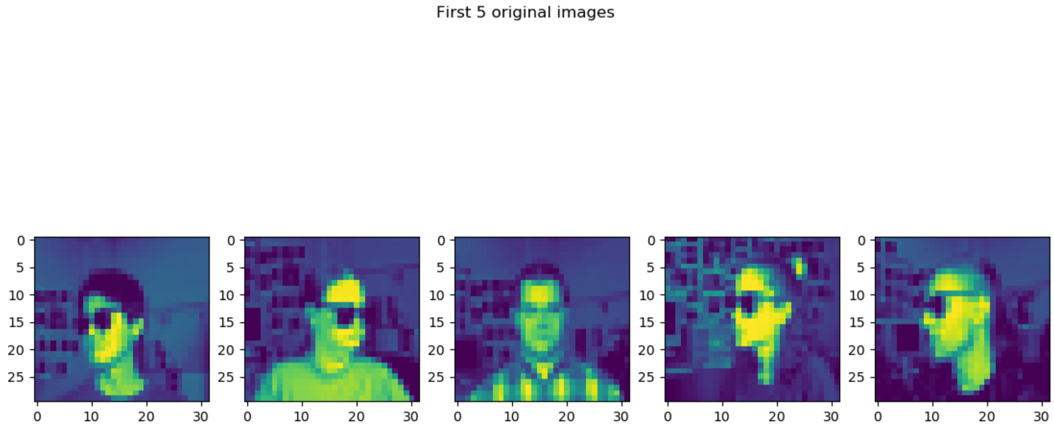

Figure 7: Original first 5 faces from training data

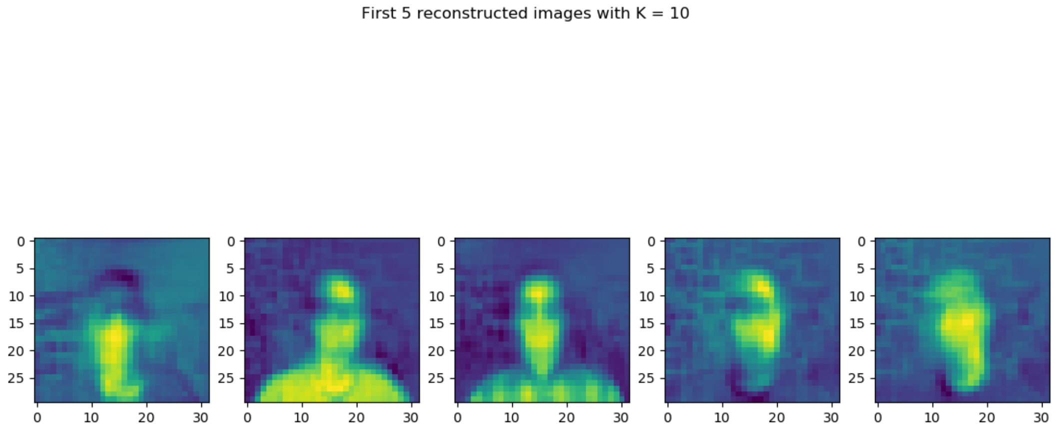

Figure 8: First 5 reconstructed faces using 10 principal components

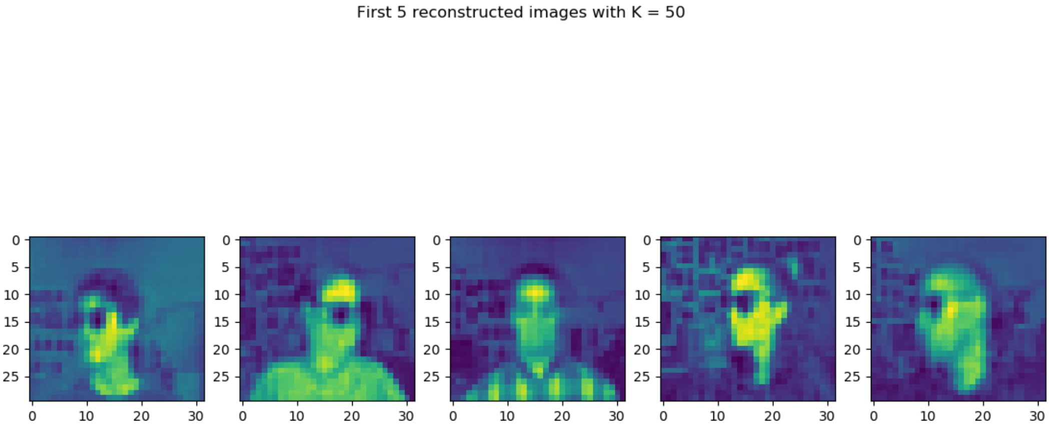

Figure 9: First 5 reconstructed faces using 50 principal components

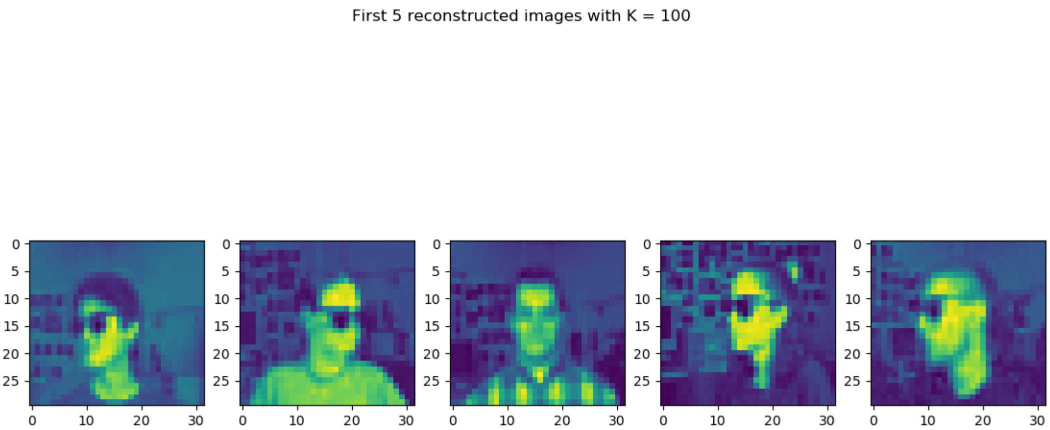

Figure 10: First 5 reconstructed faces using 100 principal components

Thus we can see that as we increase the number of principal components, reconstructed images become closer to the original images but at a cost of increased complexity and processing time.