Assignment 2

Anita Bahmanyar

November 14, 2016

Question 3

3.1

3.2

3.3

Question 4

4.2

In the code provided (mogEM.py), inside function mogEM there are a few lines that include "randConst" value.

```
p  = randConst + np.random.rand(K, 1)
p  = p / np.sum(p)   # mixing coefficients
mu = mn + np.random.randn(N, K) * (np.sqrt(vr) / randConst)
```

The first line will shows π_k values which are the mixing coefficients in the Gaussian mixture method. The mixing coefficients will be dominated by random values if randConst is small and it will be dominated by the value of randConst if it is large. In the third line, as we increase randConst, "(np.sqrt(vr) / randConst)" decreases and as we decrease the randConst value, this part of "mu" expression increases. This means that as we increase the randConst value, the "mu" values will be dominated by the mean values since the second part of "mu" expression will be very small.

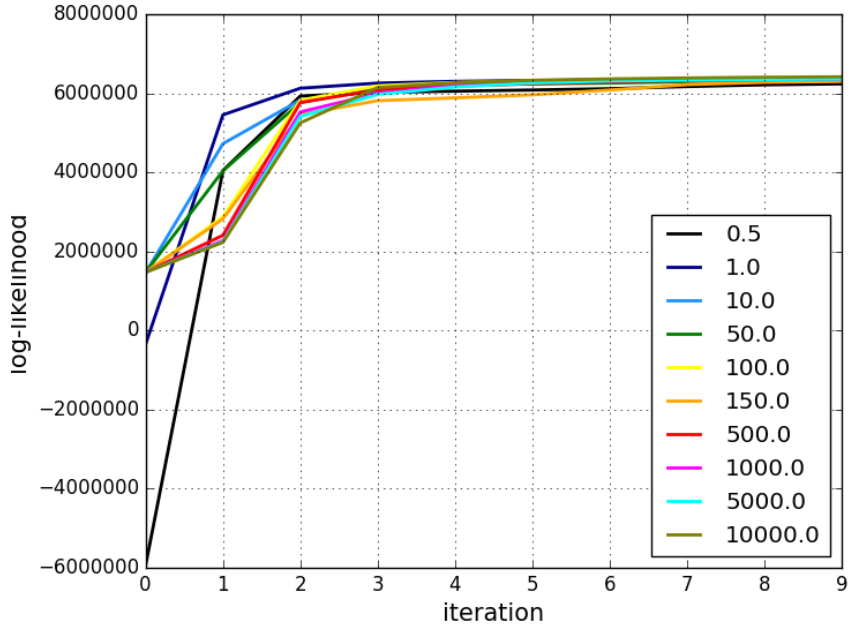

Figure 1: Log-likelihood as a function of number of iterations for different values of randConst parameter.

As we see in Figure 1, randConst=1.0 converges faster to the similar log-likelihood values compared to the other randConst values. So the model I choose has randConst=1.0. Below are variance and mean of the images shown for randConst = 1.0:

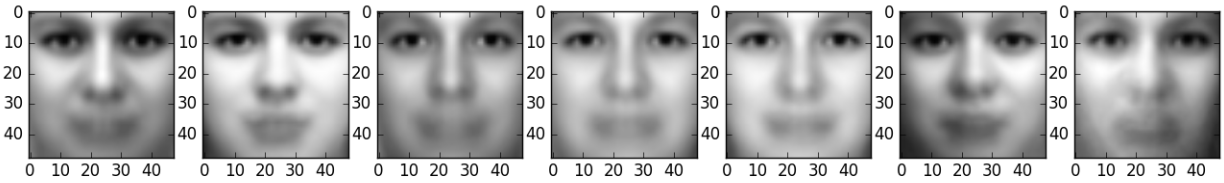

Figure 2: Mean of the images- Is is blurry because it is average.

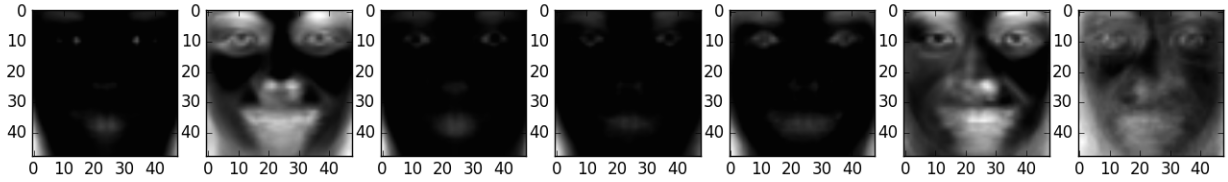

Figure 3: Variance of the images-black means lower variance and white means higher variance.

We see that most of these images look the same to us even though they are not and this is because there are small variations in the images. So variances are the most helpful ones in this case to see the variation between images. Also, black in the variance images mean lower variance and whiter means higher variance. Areas with lower variance are better so it is better if we see a lot of black areas in the variance images. Lower variance means better classifications so the variance images with more black area are better classifiers.

Below in Figure 4, the values of Gaussian mixture coefficients are given vs. cluster number.

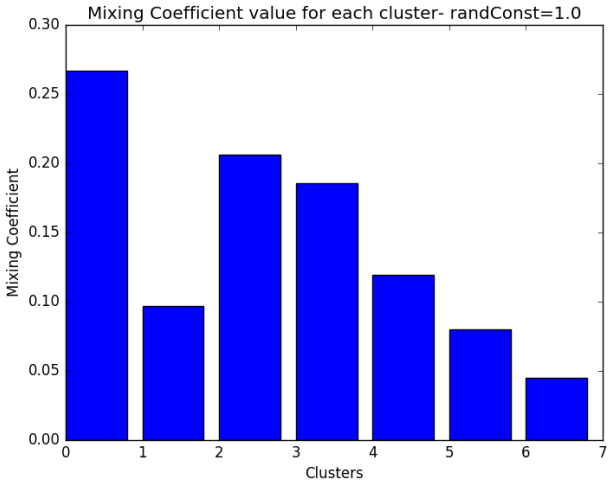

Figure 4: Value of mixture of coefficients for each cluster.

4.3

Initializing the means using K-means makes the code converge much faster than just using random values for the means. This can be seen in Figure 7 below.

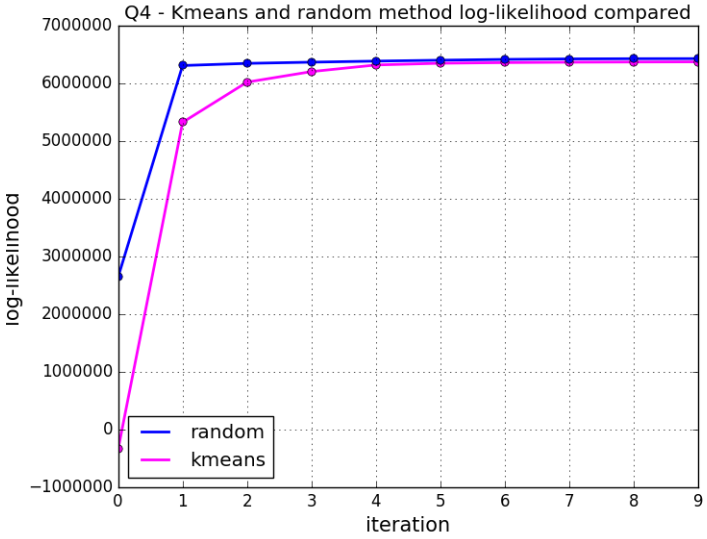

Figure 5: Comparison between log-likelihood using two different methods: Kmeans shown in magenta and randomized method (using randConst) shown in blue. We can see that using means method the log-likelihood converges earlier to a similar value compared to the randomized method.

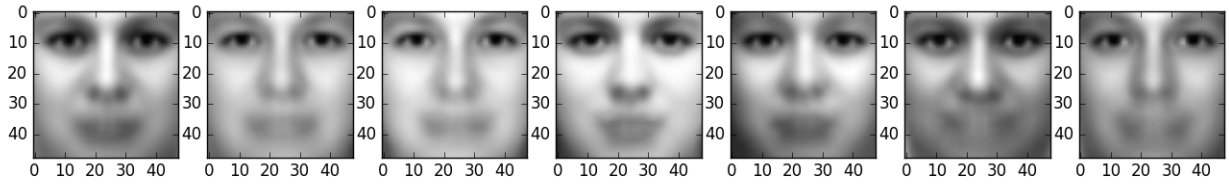

Figure 6:

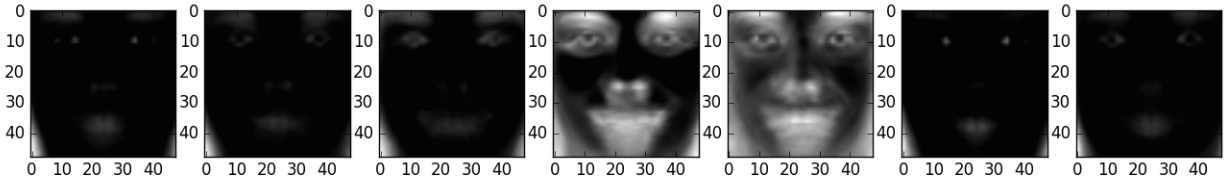

Figure 7:

4.4

In this question, we are trying to compute $p(d|x)$ value using Baye's rule. using Baye's rule we know that:

$$p(d|x) = \frac{p(x|d)p(d)}{p(x)}, \tag{1}$$

where $p(x|d)$, $p(d)$ is the prior and $p(x)$ is the evidence. Since $p(x)$ is constant for all the values we can ignore it here. Then, we can take log of both sides of Equation (1) and write it as:

$$\log(p(d|x)) = \log(p(x|d)p(d)) = \log(p(x|d)) + \log(p(d)). \tag{2}$$

We do have value of $p(d)$ from "log_likelihood_class" function in the code provided and value of $p(x|d)$ is given in the function called "mogLogLikelihood". So we can use these functions to compute $\log p(d|x)$.

Figure **??** shows the for all three cases of training set, validation set and test set.

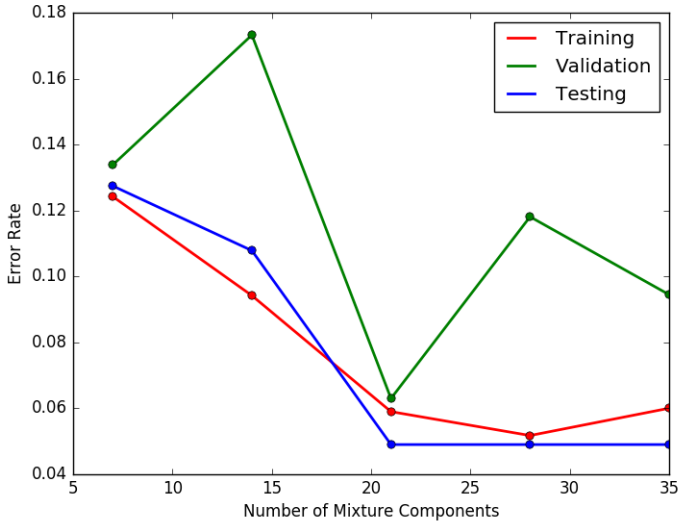

Figure 8:

Just to see the results because we have some randomness in the values we start with, I ran this part of the code again and the result is shown below for the same values as Figure **??**. Figure **??** shows the for all three cases of training set, validation set and test set.

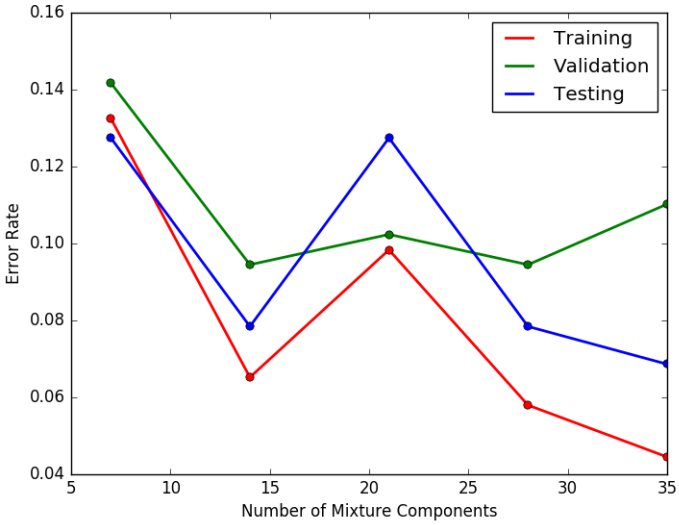
Training
Validation
Testing
Error Rate
Number of Mixture Components
0.16
0.14
0.12
0.10
0.08
0.06
0.04
5
10
15
20
25
30
35

Figure 9: