Assignment 2 Report

(1) Report the Centroid of each cluster in K-means:

Using random seed = 0 to initialize we begin with the following centroids in red:

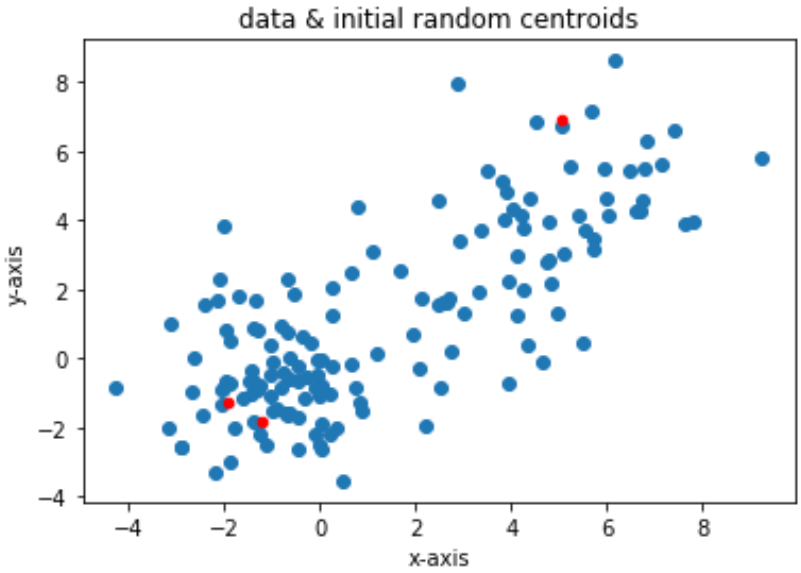

Figure 1.

After running the implementation of the K-means algorithm:

- The resulting centroid of cluster 1 has coordinates: (2.88349711, 1.35826195)
- The resulting centroid of cluster 2 has coordinates: (5.43312387, 4.86267503)
- The resulting centroid of cluster 3 has coordinates: (−1.03940862, −0.6791968)

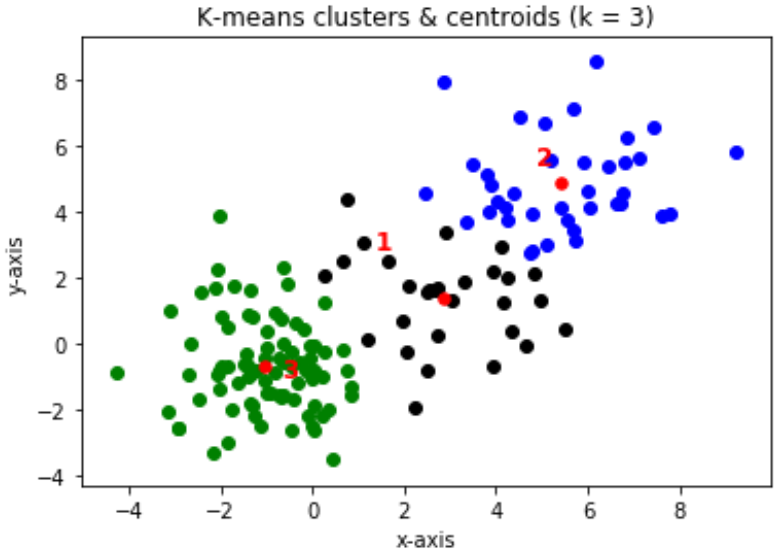

Figure 2.

(2) *Report the mean, amplitude and covariance matrix of each Gaussian in GMM:*

Using random seed $= 9$ to initialize we begin with the following 2D gaussians as our initial hypothesized generative models:

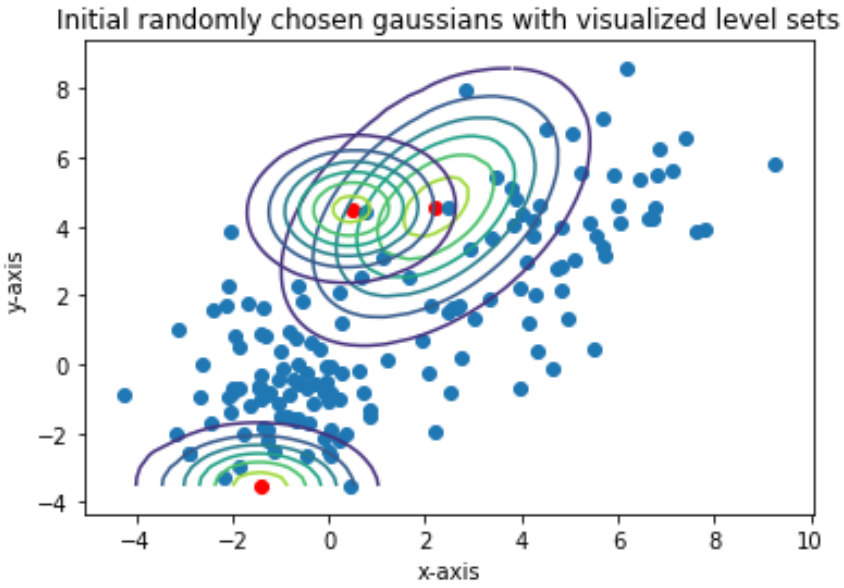

Figure 3.

After running the implemented EM clustering, we obtain Gaussians with the following:

The x-y mean of Gaussian 1 is: (4.23971922,3.29079133)
with amplitude of: 0.04139088101121953
and Covariance Matrix:
$$\begin{bmatrix} 4.09524727 & 2.45398838 \\ 2.45398838 & 5.08086183 \end{bmatrix}$$

The x-y mean of Gaussian 2 is: (−1.13390322,−1.29997857)
with amplitude of: 0.1495206525238713
and Covariance Matrix:
$$\begin{bmatrix} 1.45749471 & 0.21739236 \\ 0.21739236 & 0.80980072 \end{bmatrix}$$

The x-y mean of Gaussian 3 is: (−0.90808405,−0.04092778)
with amplitude of: 0.13205888136675678
and Covariance Matrix:
$$\begin{bmatrix} 0.82833928 & -0.67469114 \\ -0.67469114 & 2.30300679 \end{bmatrix}$$

Visually:

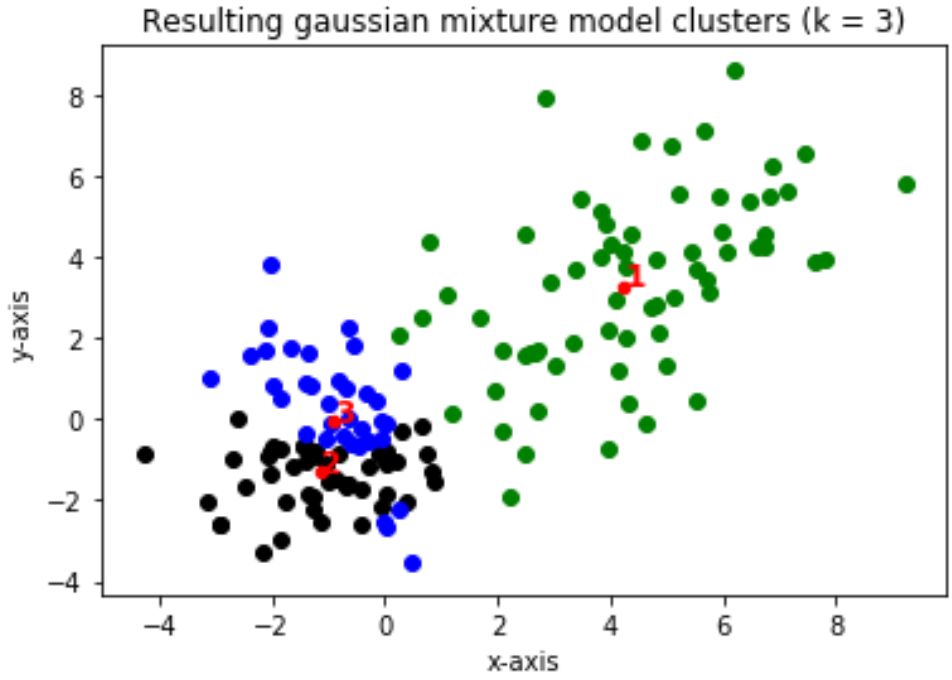

Figure 4.

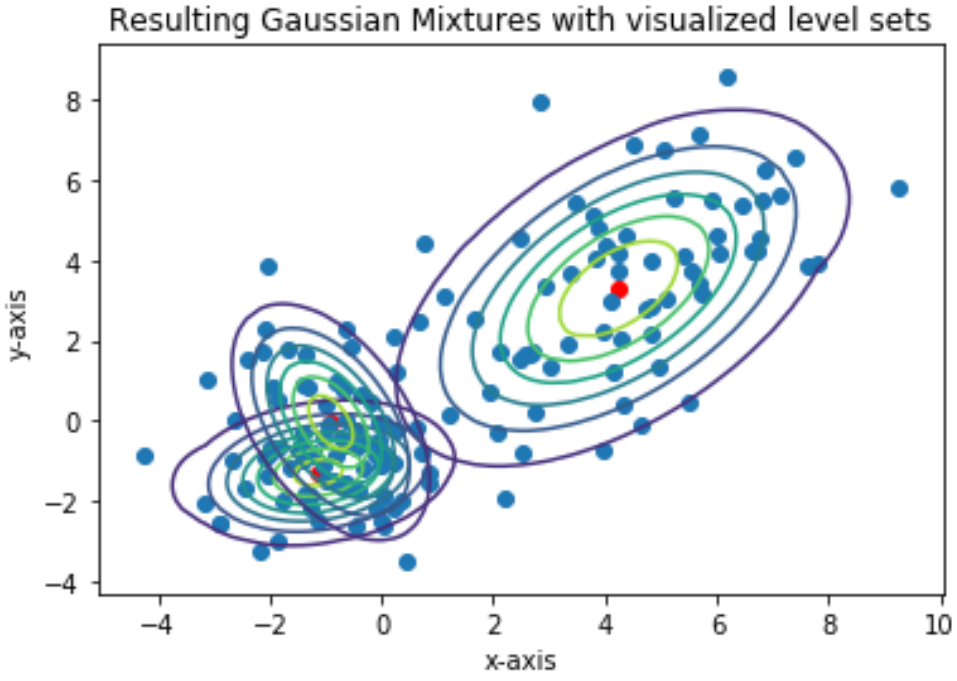

Figure 5.

(3) *Compare the results of the two algorithms*

We can see the different resulting clusters of the two Algorithms by comparing Figure 2. (K-means) with Figure 4. (EM Clustering). In the case of the K-means, the clusters are very cleanly cleanly grouped in their own regions. In the case of the EM algorithm, our clusters can overlap with each other, as illustrated by the crossover occurring between clusters 2 and 3 (black & blue respectively) and by the the overlapping level sets of their gaussians in Figure 5. If we use the EM implementation with random state 0 (for direct comparison with our k-means), we also get a similar result where overlapping clusters again occurs and the centers do not seem to move as drastically as in k-means.

After experimenting with several random states and multiple runs, the EM clustering is observed to be slower than K-means in terms of converging on the final values. For the purposes of this assignment, an epsilon tolerance level of 0.001 was chosen as the terminating criteria for both implementations.

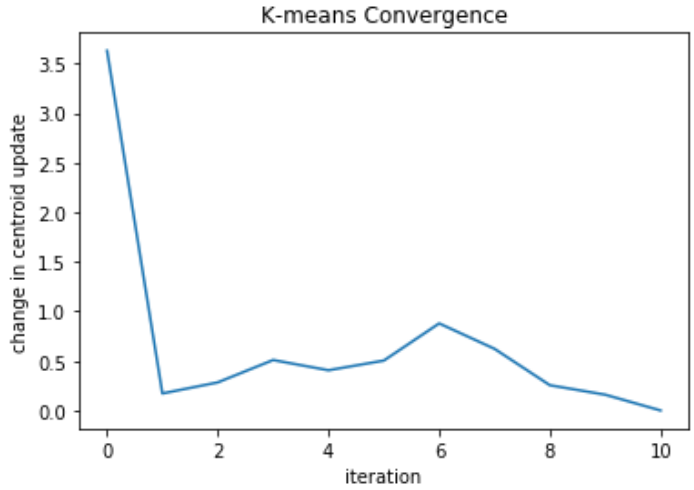

Figure 6.

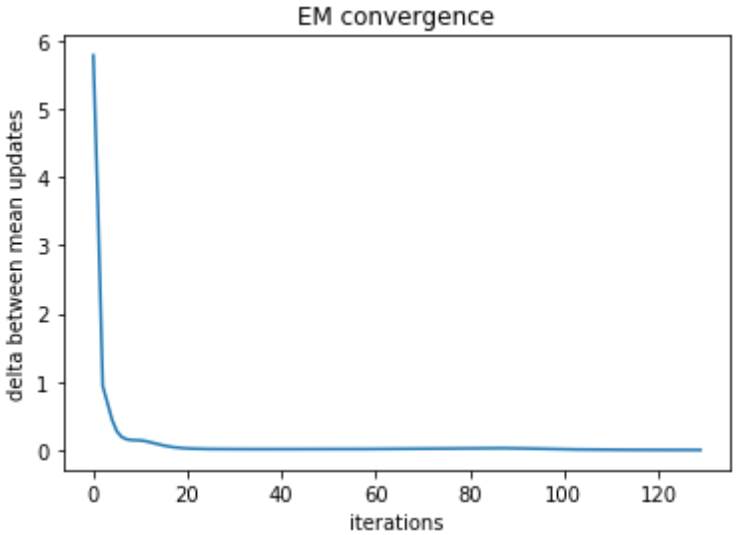

Figure 7.

Additional descriptions and challenges:

The data structures used were primarily numpy arrays and lists. A dataframe was used at the beginning to ensure proper read-in of the data, but otherwise was not used in the main implementations of either. A number of helper functions, (initialize_centroids, nearest_centroid, and update_centroid) were written to aid in implementing the K-means algorithm. Initialize_centroid and Nearest_centroid were also re-used in initializing the starting coordinates and initial covariances for the Gaussians in the EM implementation and also to help determine the final class memberships of each data point once convergence was obtained. One challenge in this assignment was the bivariate normal, in a previous version I attempted to implement a bivariate_normal helper class but struggled with issues related to matrix non-invertibility and numerical precision, among others. To overcome this challenge, I used the multivariate_normal class implementation from the basic scipy statistics library.