CS5512: Machine Learning, Programming Assignment 4

Daney Alex (122003001)

May 31, 2020

1 Objective

1.1 Objective 1

Generate 3 clusters of different size and shape

$X\_1 = np.random.multivariate\_normal(mean = [4, 0], cov = [[1, 0], [0, 1]], size = 75)$

$X\_2 = np.random.multivariate\_normal(mean = [6, 6], cov = [[2, 0], [0, 2]], size = 250)$

$X\_3 = np.random.multivariate\_normal(mean = [1, 5], cov = [[1, 0], [0, 2]], size = 20)$

1.2 Objective 2

Observe the result of K-mean clustering for different initial positions of centres for the generated data.

1.3 Objective 3

Run Gaussian Mixture Models(GMM) on same data. Compare K-Means and GMM methods.

1.4 Objective 4

Implement K-Means to compress an image . Take a high resolution RGB image. For each pixel location we would have three 8-bit integers that specify the red, green, and blue intensity values. Our goal is to reduce the number of colors to 30 and represent (compress) the photo using those 30 colors only. To pick which colors to use, we will use K-Means algorithm on the image and treat every pixel as a data point in 3-dimensional space which is the intensity of RGB. We have to run K-Means to find 30 centroids of colors and finally we will represent the image using the 30 centroids for each pixel. Check if the actual size of the image changes.

2 Algorithm

2.1

- Data set is generated in 2D coordinate, by combining points in X_1, X_2 and X_3.
- Run K-Mean clustering algorithm on data set. Repeat the step for different values of initial centroid values. We will obtain final centroid values for 3 clusters and cluster label for each point in data set. Generate the plot representing points in different clusters in different colours.
- Run GMM algorithm on data set. We will obtain cluster label for each point in data set. Generate the plot representing points in different clusters in different colours.

2.2 Image Compression

- Obtain the Height and Width of the input image. The dimension of RGB color image will be a 3D matrix with dimension (Height, Width, 3). Reshape the 3D matrix of image to a 2D matrix with dimension (Height*Width, 3).
- Each row in the 2D matrix represent a point in 3D coordinate depending on the values of R,G and B.
- Run K-Means algorithm on the 2D matrix for generating 30 different clusters and centroid for each cluster. We will obtain the label of the cluster for each point in the the 2D matrix.
- Now we will replace the value of R, G and B of all pixels with the value of the centroid of the cluster corresponding to the label of that pixel obtained from K-Means algorithm. The 2D matrix back to 3D matrix of dimension (Height, Width, 3). The 3D Matrix is the new compressed image.

3 Programming

- **How to Run the code** : python3 Daney_Alex.py Input_Image.png
- **Output** :
  1. Generated_dataset.png
  2. K_Means1
  3. K_Means2

4. K_Means3
5. K_Means4
6. K_Means5
7. GMM.png
8. Daney_Alex2.png

- **Libraries Used** :

1. sklearn.mixture
2. sklearn.cluster
3. matplotlib.pyplot
4. numpy
5. random
6. math
7. cv2

4 Results and Observations

4.1

- The generated Dataset from X_1, X_2 and X_3 is shown in Figure 1. The plot represent X_1, X_2 and X_3 in different colours.

- Plots obtained after K-Means clustering for 3 clusters and the 3 centroids are shown in Figure. We initialise clustering centroid with 5 different values to obtain different K-Means clusters as shown in Figure 2 to 6 . Three different clusters are shown using different colours.

- Plots obtained after K-Means clustering for 3 clusters and the 3 centroids are shown in Figure 7. Three different clusters are shown using different colours.

- In K-Means algorithm, final clusters is dependent on initial clustering centroid values. From the plots we could observe small difference in clusters formed by varying initial centroid values.

- The K-Means model places a circle at the center of each cluster, with a radius defined by the most distant point in the cluster. This works fine for when your data is circular, however when the data takes on different shape it may result in overlap. Gaussian mixture models can handle even very oblong clusters.

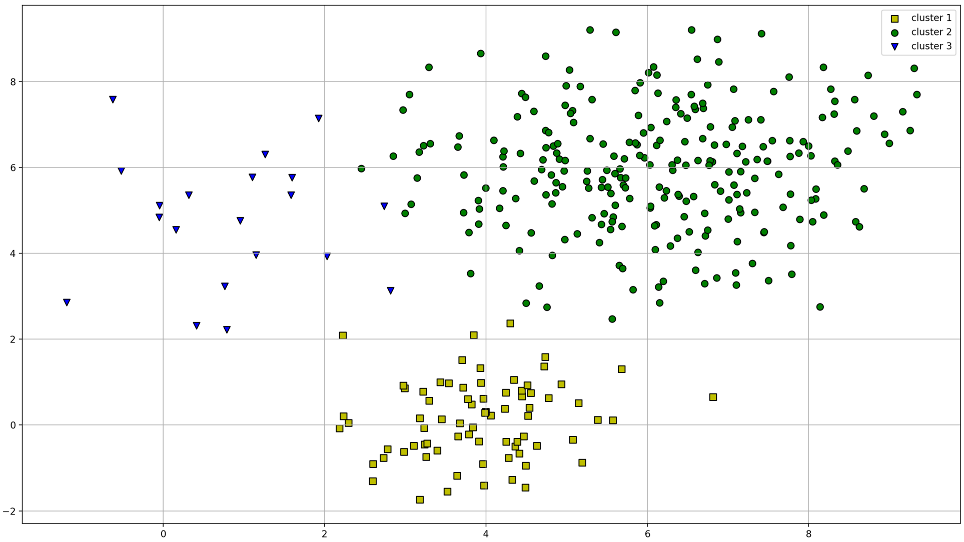

Figure 1: Generated Data set

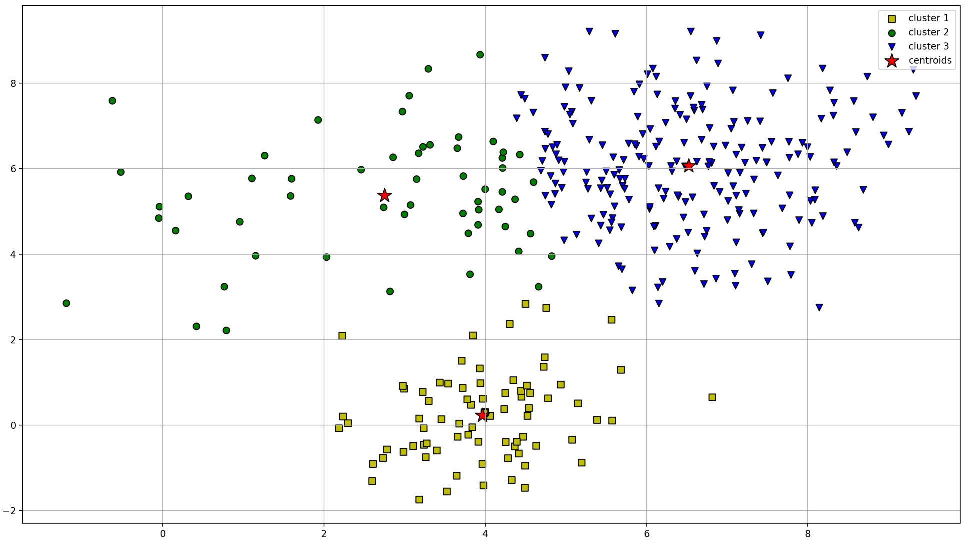

Figure 2: K-Means Clustering Output

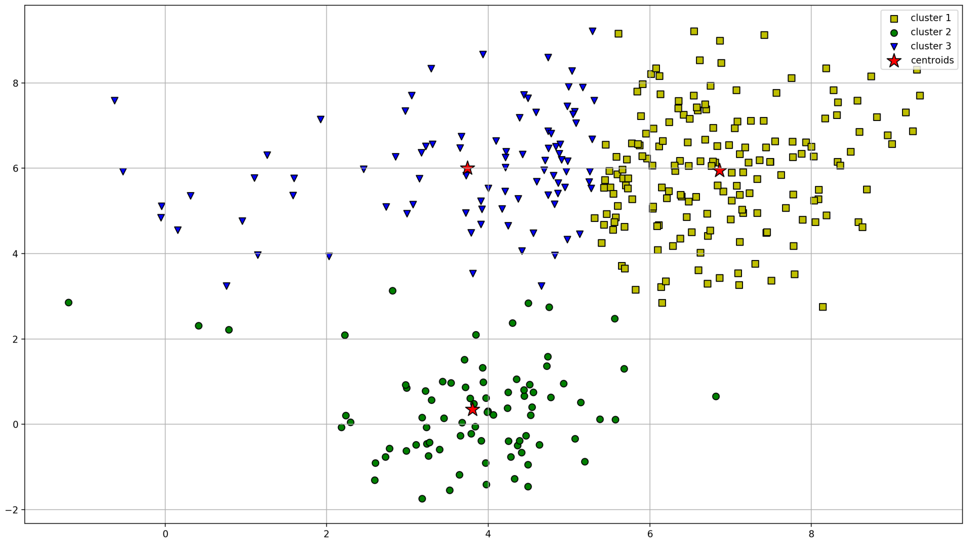

Figure 3: K-Means Clustering Output

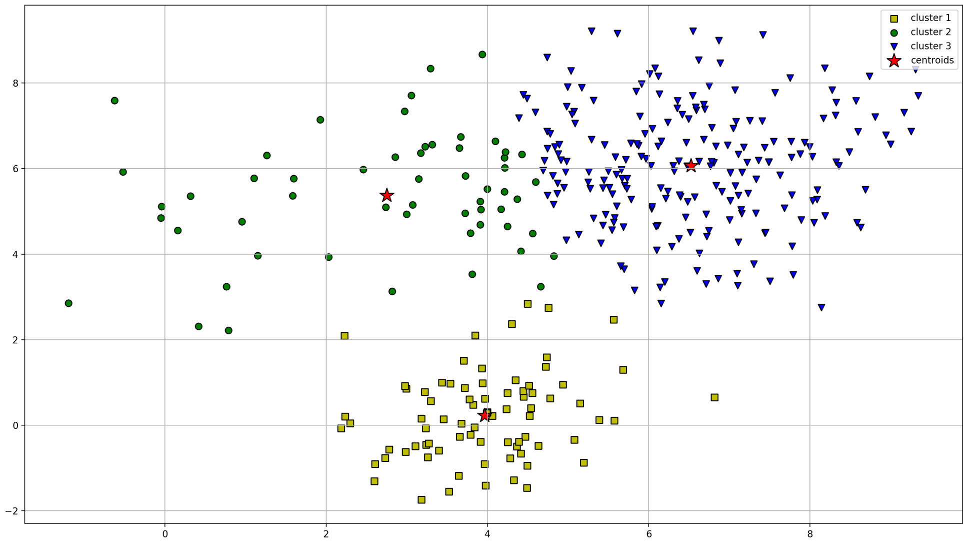

Figure 4: K-Means Clustering Output

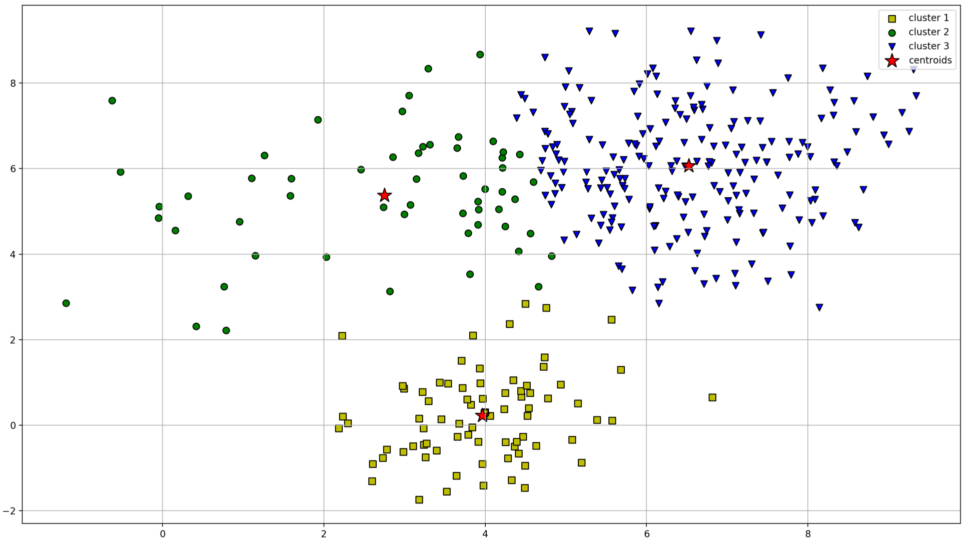

Figure 5: K-Means Clustering Output

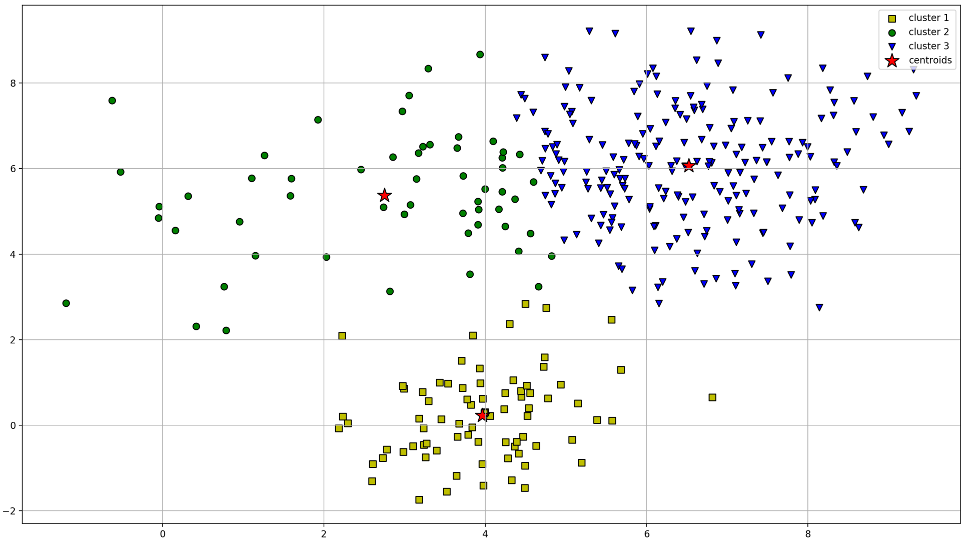

Figure 6: K-Means Clustering Output

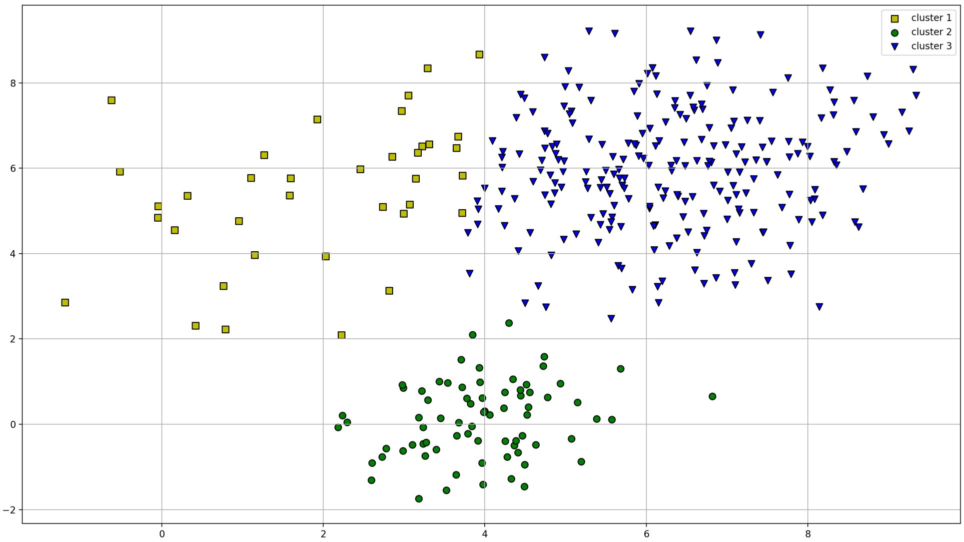

Figure 7: Gaussian Mixture Models Clustering Output

- K-Means provide us what data point belong to which cluster but would not provide us with the probabilities that a given data point belongs to each of the possible clusters. But GMM can provide the probabilities. In GMM model the point with the highest probability of belongs to the cluster.

- In K-Means centroid can be the representative data of the cluster. In GMM we can choose the point with the maximal density to represent its cluster.

4.2 Image Compression

- Original image before compression is shown in Figure 8.

- Compressed image shown in Figure 9.

- In input image, each pixel is represented in 3D coordinate system where R,G and B values are the coordinates. The position of pixel in 3D coordinate represent intensity value of that pixel.

- Using K-means for image results in formation of 30 spherical clusters on 3D coordinate system with 30 centroid value. Now every pixel is grouped with any of the clusters depending upon distance of the pixels

Figure 8: Original Image

Figure 9: Compressed Image

from centroids. So the centroid of a cluster represents intensity of a pixel in that cluster.

- Using K-Means the orginal image was compressed to 40% of its orginal size (2019kB - 777kB).

- Quality of compressed image is reduced by a small amount. Smoothness of edges is reduced as a result of reducing number of colours to represent image. Also adjoined pixels with very small difference in intensity values are now represented by same intensity values.