

INF552 Homework 2

Group members and contribution :

Krishna Akhil Maddali (Code development and report)

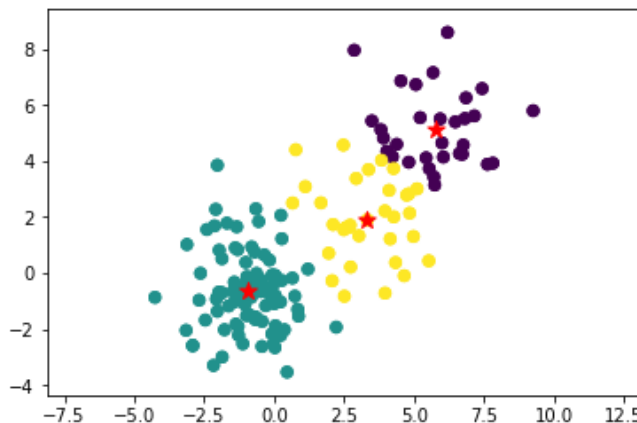
Yash Shahapurkar (Code development and research)

Myungjin Lee (Code development and research)

Part 1: Implementation

- Output after running the program

- **K-means**



Centroids

C1 : 5.43312387, 4.86267503

C2 :-1.03940862, -0.6791968

C3 : 2.88349711, 1.35826195

- **GMM**

Means of Gaussian Mixtures : M1 ([-0.81202416, -1.08534434])

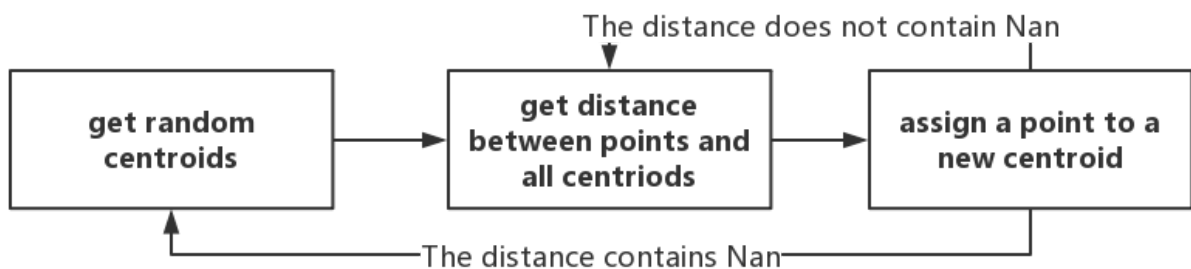
M2 ([-0.60559687, 0.2256695])

M3 ([4.81476245, 3.70697734])

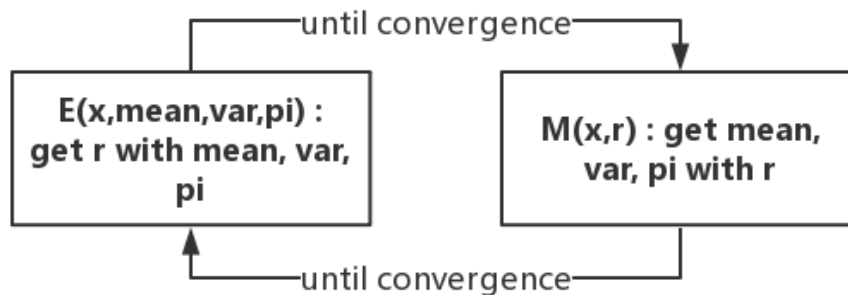
Covariances of Gaussian Mixtures: C1 ([[0.73398193, -0.05868001],
 [-0.05868001, 0.89062805]])
 C2([[3.02798556, 0.73030555],
 [0.73030555, 3.28413644]])
 C3([[2.71766974, 1.58647391],
 [1.58647391, 4.48937379]])
 Amplitudes: [0.3128173 0.31287191 0.37431079]

- Data structure

- **K-means**



- **GMM**



- Code-level optimization

- **K-means**

- We have used isnan function to catch infinite value of distance when the centroid and the data point is too close to compute the distance.
 - Since numpy array does not have function to select/store two values all at once such as 2-dimensional coordinate data set, we have separated 2D coordinate into two arrays (x and y array) to store the new clustered data points. It has been applied in a same way when we calculate the average of the data points of each cluster to update the centroids.
 - Hardcoded the number of clusters (K=3).

- **GMM**

- Use of arrays from numpy over lists because it offers better matrix operations such as matrix multiplication.
 - Hardcoded the number of data points and number of clusters.

- Challenges

- **K-means**

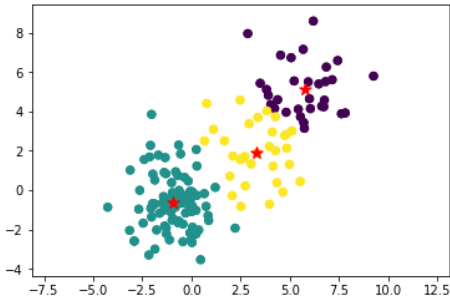
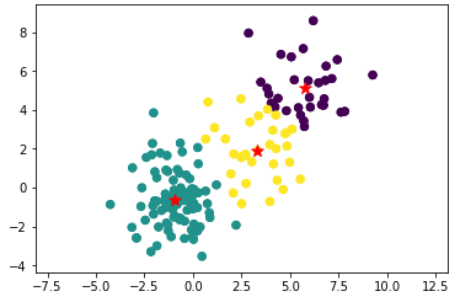
- **Nan value** : When the code calculate the distance between centroids and the data point, sometimes the distance gives me Nan value since the normalization value is zero. In this case, the code will generate the new centroids randomly and recompute the distance between data point and itself.
 - **Recursion** : K-means algorithm is designated to search the closest centroid from a certain data point until it converges. To find out when would stop the recursion was challenging since there are several possibilities such as distance between old and new centroid is zero or the old and new should be exactly same. This code is designed to break recursion when new centroid values are the same as the old centroids.

- **GMM**

- Working with row or column matrices which are sometimes intermediate results for calculating Covariance of a matrix since they have sometimes reshaped.
 - **Recursion**: Similar to K-Means convergence condition must be careful chosen to break the loop

Part 2: Software Familiarization

- K-means

	Our code	Sklearn Kmeans
Output	 [[5.43312387 4.86267503] [-1.03940862 -0.6791968] [2.88349711 1.35826195]]	 [[5.73849535 5.16483808] [-0.96065291 -0.65221841] [3.28884856 1.93268837]]
Library	Our own code	Python Scikit (sklearn)
Comparison	The output of our code and sklearn library does not show huge difference but there must be a certain discrepancy. It comes from the implementation of the random number generator. Sklearn library set a certain seed to generate random number which provides consistent result regardless of the number of running time the code, however, our code has not set a specific seed.	

- GMM

	Our code	SciKit
Output	Means: ([-0.81202416, -1.08534434]), ([-0.60559687, 0.2256695]), ([4.81476245, 3.70697734]) Covariance: C1-([[0.73398193,-0.05868001], [-0.05868001,0.89062805]]) C2([[3.02798556, 0.73030555] [0.73030555, 3.28413644]]), C3([[2.71766974, 1.58647391], [1.58647391,4.48937379]])]	Means: ([5.62092993, 4.99410684]), ([-0.96174678, -0.63618636]), ([3.2842769 , 1.9291939]]) Covariance: C1([2.2216601 , 0.15446244], [0.15446244, 2.094973]) C2([1.23856256, -0.09316563], [-0.09316563,2.02291629]) C3([1.91930032, 0.14344635], [0.14344635,2.57669337])]
Library	Our own code	Python Scikit
Comparison	The outputs show some difference this might be due to the difference in the initial assignment of the parameters.	

- How to improve our code

- **K-means**

- **Precision** : Whenever running the code, the code gives us inconsistent centroid values in the third decimal. The floating point precision could be improved in this code by setting the same seed when it generates random centroids initially.

- **GMM**

- **Initialisation** : The number of iterations and convergence depends upon initialisation. The library offers better initialisation compared to our code. Having heuristics might help in faster convergence

Part 3: Applications

- **K-means**

- **Branch Geo Segmentation:** As for bank, they want to establish ATM service provider based on geo-location of ATMs. The ATMs' Latitude and longitude information were used for creating the clusters.
- **Healthcare Fraud Detection Segmentation:** K-means algorithm is also useful when it comes to healthcare fraud detection. One of healthcare provider's aim was to find claims which could be fraudulent. K-means clustering was used for anomaly detection and claim routing to right claim adjudicator. In addition, claim type, age group, hospital type healthcare setting could also make use of K-means algorithm.

- **GMM**

- **Handwriting recognition:** Gaussian Mixture Models can be used to train models to recognise handwriting such as handwritten numerical digits. Such models are able to cluster handwritten digits effectively.
- **Fuzzy Image Segmentation :** In most traditional image segmentation models each pixel is associated with only one pattern. However in Fuzzy Segmentation each pixel would have a certain 'ownership', thus GMM can associate each pixel with more than one cluster.