INF 552 HW2

Teammates:
He Chang 5670527576
Ziqiao Gao 2157371827
Fanlin Qin 5317973858

Part 1.

K Means

Data Structure
We use python dictionaries to keep track of group of points for each centroid.
One dictionary is used to keep track of centroids changes and the other one is used to record the corresponding member points for each centroid in every iteration

Optimization

1. The python dictionary data structure we use provides a fast checking and storing solution because the time complexity of finding and storing the corresponding member points will be constant
2. In order to visually confirm if the classification result makes sense, we use **matplotlib** python library to plot the 3 different point groups centered by our predicted centroids

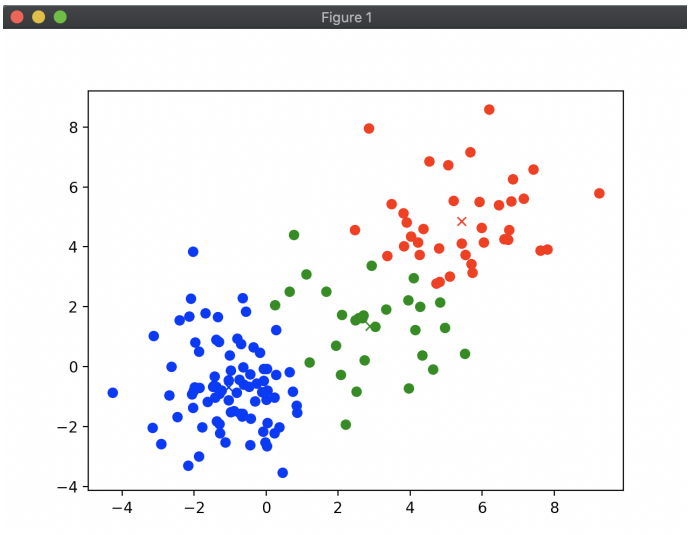
Challenge

1. It took some time to come up with an efficient data structure to keep track of new centroids and its member points

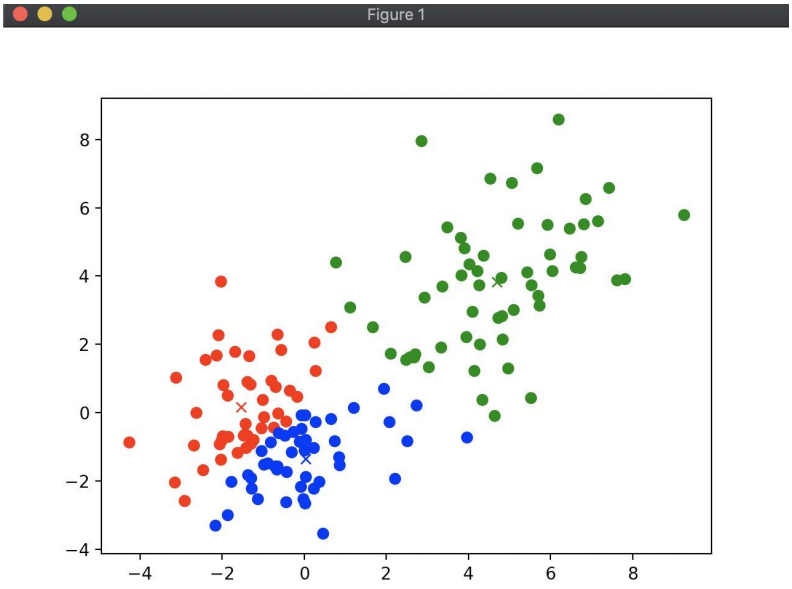
K Means Result

We found two typical local optimal clustering solutions.

Type A:
Centroid 1: [5.433123874157893, 4.862675026605261]
Centroid 2: [2.883497109068966, 1.358261948]
Centroid 3: [-1.0394086163975909, -0.6791968002650602]



Type B:
Centroid 1: [-1.5342584855652175, 0.16235877886956532]
Centroid 2: [4.699491259327586, 3.832118786431033]
Centroid 3: [0.03944794597826092, -1.3463800078260868]]



GMM

Data Structure
We use a dictionary *parameter* to keep track of parameters of each Gaussian Model coefficients and the *ric* of each data point.
E.g.

parameter["Mu_1"] represents the Mean of the first Gaussian Model

parameter["Mu_2"] represents the Mean of the second Gaussian Model
parameter["Sigma_1"] represents the Sigma matrix of the first Gaussian Model

Optimization

1. Using Dictionary to store all parameters of Gaussian Models makes the code human readable.
2. To better visually understand the gaussian model this algorithm generated, we use **matplotlib** python library to plot the 3 different 3D Gaussian Models.

Challenge

1. It took us some time to finish the programming design of the calculation of Gaussian Probability Density Function.
2. Our algorithm produces two types results, indicating that there may be two local optimums can be found by our algorithm, but it is uncertain whether there are other optimal values.
3. Debugging took us time too. Because we have to verify the matrix calculation is actually an outer product and other math are correct.

GMM Clustering Result:

We also found two typical local optimal clustering solutions.

Type A:
Gaussian 1:

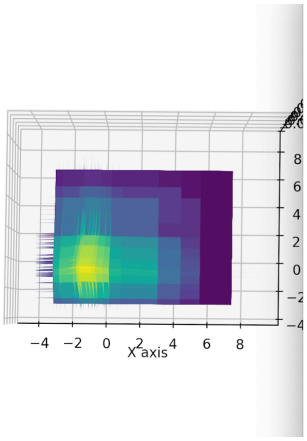
Mu_1:
[[6.48845471], [4.60933322]]
Sigma_1:
[[1.4489933 0.64827428]
 [0.64827428 1.07672521]]
Pi_1:
0.10121508179956039
[6.48845471 4.60933322]

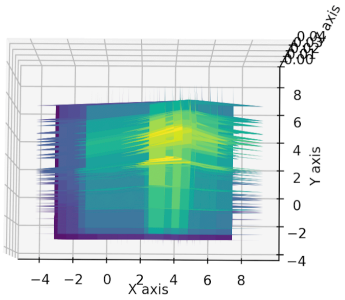
Gaussian 2:

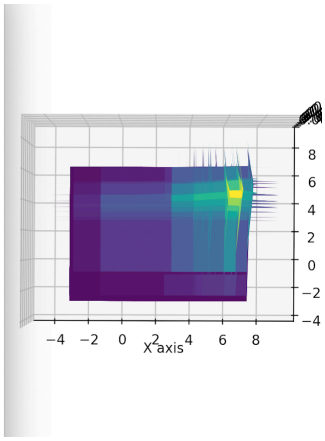
Mu_2:
[[-0.99189256]
 [-0.6211066]]
Sigma_2:
[[1.17588358 -0.06721133]
 [-0.06721133 2.04685835]]
Pi_2:
0.5628861876876167
[-0.99189256 -0.6211066]

Gaussian 3:

Mu_3:
[[3.75208682]
 [2.98224123]]
Sigma_3:
[[2.50119735 1.65960039]
 [1.65960039 5.65209614]]
Pi_3:
0.3358987305128232
[3.75208682 2.98224123]







Type B:
Gaussian 2:
Mu_1:
[[-1.34479988]
 [1.38053608]]
Sigma_1:
[[1.60479263 0.74380268]
 [0.74380268 1.74852305]]
Pi_1:
0.12967411685519553
[-1.34479988 1.38053608]

Gaussian 2:
Mu_2:
[[4.45405955], [3.35431455]]
Sigma_2:
[[3.44040892 2.29942335]
 [2.29942335 5.17207114]]
Pi_2:
0.42812078687400673
[4.45405955 3.35431455]

Gaussian 3:

Mu_3:

[[-0.84522248], [-1.12259861]]

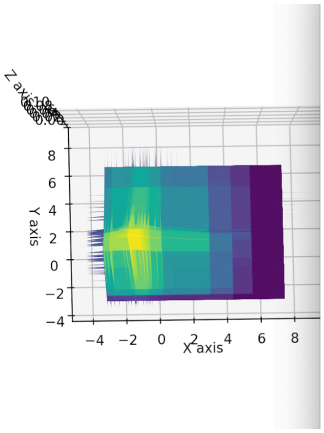
Sigma_3:

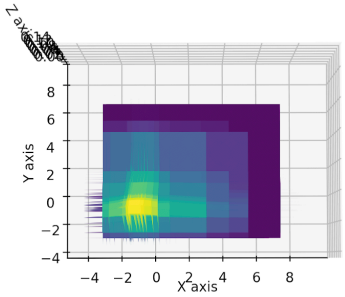
[[1.05768657 0.13953605]
[0.13953605 1.0226497]]

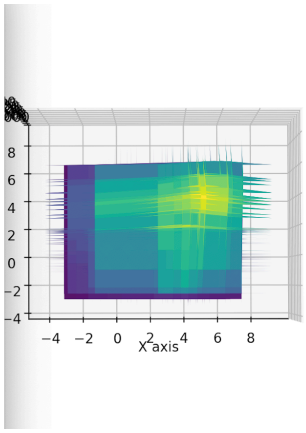
Pi_3:

0.44220509627079774

[-0.84522248 -1.12259861]







Summary:

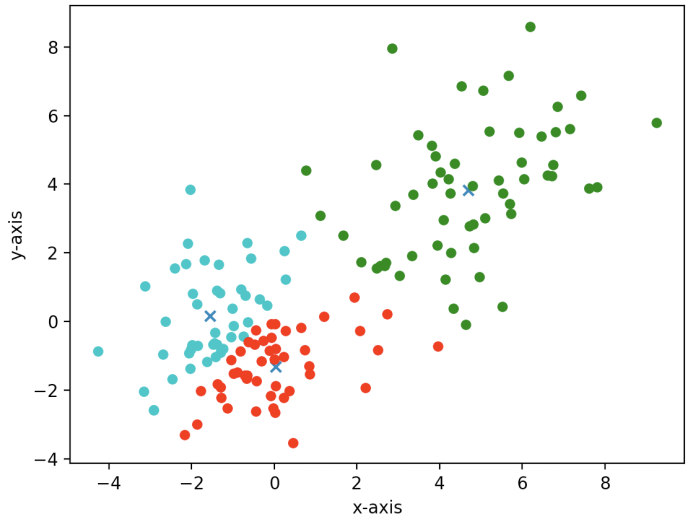
The K-means algorithm and the GMM algorithm produces similar clustering solutions to the 150 points. They both have 2 types of local optimal clustering and we can visually see they group points in similar ways. The thing is that GMM gives more freedom when classifying point membership, which means each point could belong to different Gaussian models with different probability.

Part 2.

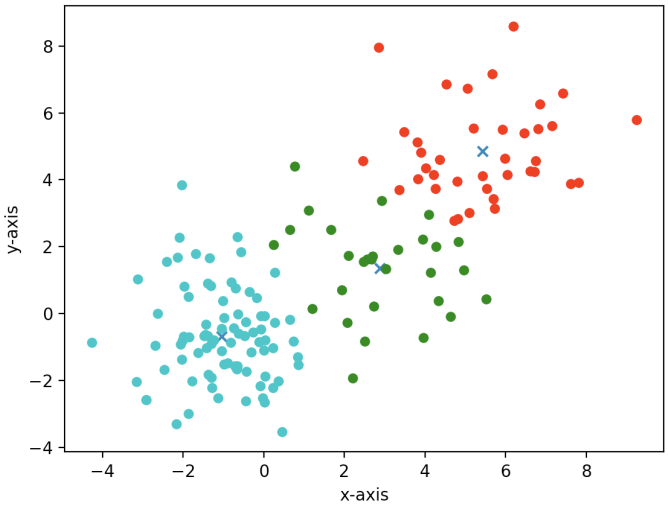
K-Means Result:

Type A:

centroids=[[0.02920441,-1.32307409]
[4.69949126,3.83211879]
[-1.55853093,0.17154457]]



Type B:
centroids=[[5.43312387,4.86267503]
[2.88349711,1.35826195]
[-1.03940862,-0.6791968]]



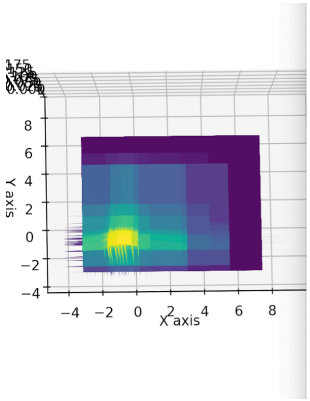
Gaussian Mixture Model Result:

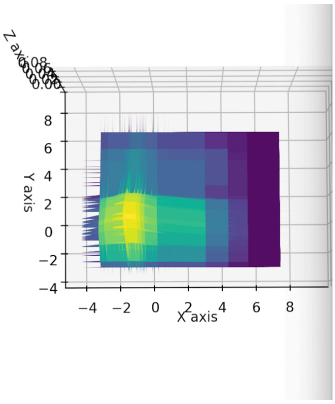
Type A:
GMM weights: [0.29537046,0.43495198,0.26967755]

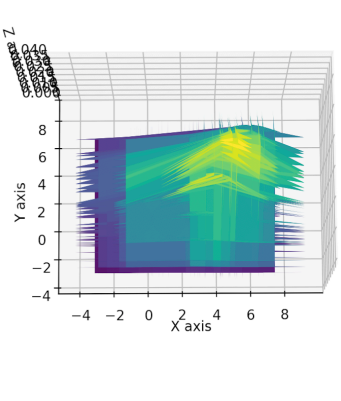
mu1
[-0.62614069 -1.23417536]
sigma1
[[0.9254042 0.05294352]
 [0.05294352 0.83460351]]

mu2
[4.39482854 3.35742974]
sigma2
[[3.61249967 2.26739768]
 [2.26739768 5.06317929]]

mu3
[-1.3641027   0.08480859]
sigma
[[1.22142023 0.37424155]
 [0.37424155 2.63684172]]





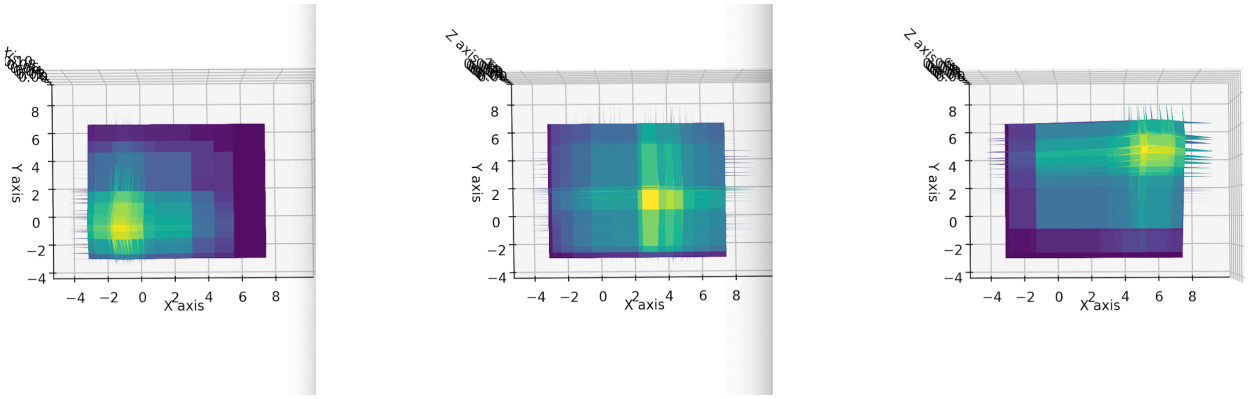


Type B:
GMM weights: [0.17477342, 0.26025477, 0.56497181]
mu1
[2.95794243 1.5172694]
sigma1
[[2.05537151 -0.26590085]
 [-0.26590085 2.08519189]]

mu2
[5.35474763 4.71166217]
sigma2
[[2.3640313 0.45069033]
 [0.45069033 2.33636882]]

mu3

[-0.97675717 -0.65978187]
sigma3
[[1.21681206 -0.11996906]
 [-0.11996906 1.97577318]]



Summary of comparing part 1 and part 2:

The results of self-implemented K-Means and sklearn K-Means match very well, because they both detect two local optimal clustering solutions and the plotted points distribution looks very similar.

The major discovery is found during the implementation of GMM. Although our self-implemented GMM is able to detect the two local optimal solutions as the sklearn GMM does, the ‘bell’ shape of our 3 Gaussians models are not as steady and ‘nice-looking’ as the sklearn Gaussians. After a series of analysis, we believe we could improve our GMM algorithm by applying a stricter standard for checking model convergence. Our current method is to check if the MU of each Gaussian converges by examining if the change of values of new MUs is less than 0.00001. In order to stabilize the shape of our Gaussians, we might have to also check the convergences of SIGMAs and Nk. On the other hand, since we have two types of local optimal clustering results, one thing we learn from online sklearn tutorials is that we could run GMM algorithm multiple times and pick the best answer from all local optimal solutions. The best answer almost lies in just one type of clustering and it is quite steady when the trail number is high.

Part 3.

1.K-means: Text Documents Clustering using K-means Algorithm. The tags, topics and the content of documents can be used to classify documents into multiple categories. Each document is represented as a vector and uses term frequency to classify documents.The document vectors are clustered to help to classify the documents.
2. GMM: GMM algorithm can be used in data stream clustering analysis. Data stream is infinite data and quick stream speed. The common algorithm cannot be used to analyse the data stream. But GMM algorithm is very effective in data stream mining.

Reference:
[1] Samir Kunwar, “Text Documents Clustering using K-means Algorithm”, *CPOL,* 26 Jan 2013
[2] Ming-ming Gao, Chang Tai-hua, Xiang-xiang Ga0, “Application of Gaussian Mixture Model Genetic Algorithm in data stream clustering analysis”, *IEEE International Conference on Intelligent Computing and Intelligent Systems,* Dec-2010

Individual contribution:

Ziqiao Gao: Optimize, implement and debug the K Means & Gaussian algorithm and printing method in part 1.

He Chang: Optimize, implement and debug the K Means & Gaussian algorithm and printing method in part 1.

Fanlin Qin: Research online and use open source python library to implement K Means & Gaussian algorithm applications and complete part 2&3.