1 k-means vs GMM

Give a variant of k-mean algorithm somewhat between the original k-mean and Expectation Maximization (EM) for Gaussian Mixture Models (GMM). Please specify the computational details of the formulas. Pseudo-codes of the algorithm would be great. Discuss the advantages or limitations of your algorithm.

1.1 Analysis

Both k-mean and EM for GMM are clustering algorithms. K-mean uses hard margin while GMM uses soft margin. And GMM is more robust and general than k-mean. So I introduce soft margin into k-mean and discuss the advantages with the original k-mean in Section 1.3. The Pesudo code is shown in Algorithm 1.

1.2 Algorithm

Algorithm 1: Variances for K-mean

Input: Data points X, number of clusters K
Output: μ_j, σ_j for j=1 to K

Initialize the means μ_k, covariances Σ_k;
while *the convergence criterion of parameters is not satisfied* **do**
  E-step.
  for $i = 1$ *to* N **do**
    for $j = 1$ *to* K **do**
      $$\gamma_{ij}^t \leftarrow \frac{\mathcal{N}(x_i|\mu_j, \Sigma_j)}{\sum_{j=1}^{K} \mathcal{N}(x_i|\mu_j, \Sigma_j)}$$
  M-step.
  for $j = 1$ *to* k **do**
    $$\mu_j^{t+1} \leftarrow \frac{\sum_{i=1}^{N} \gamma_{ij}^t x_i}{\sum_{n=1}^{N} \gamma_{ij}^t}$$
    $$\Sigma_j^{t+1} \leftarrow \frac{\sum_{i=1}^{N} \gamma_{ij}^t (x_i - \mu_j^t)^T (x_i - \mu_j^t)}{\sum_{i=1}^{N} \gamma_{ij}^t}$$
return μ_k, Σ_k;

1.3 Advantages

There are two mainly differenrce in k-mean and GMM. Firstly, the k-mean has a hard margin, which means the point belongs to a certain cluster whether of not. However, the GMM has a soft margin, which means the point always has a possibility to each cluster although the possibility changes. Secondly, GMM considers the mixing weights and covariance while k-mean consider the fixed mixing weights. Generally, k-mean only consider part of the points when determining a new μ_k but those points with little possibility still has a influence on GMM in its M-step. Using soft margin in k-mean, my algorithm has such two advantages:

1. With a soft margin, the k-mean will be more robust and general.
2. When determining the new μ_k, the variances of k-mean takes all the point into consideration.

2 k-mean vs CL

Compare the k-mean algorithm with competitive learning (CL) algorithm. Could you apply the idea of Rival Penalized Competitive Learning (RPCL) to k-mean so that the number of clusters is automatically determined? If so, give the details of your algorithm and then implement it on a three-cluster dataset generated by yourself. If not, state the reasons.

2.1 Comparison between k-means and CL

- Similarities
    1. They can not estimate the total number of clusters.
    2. The effects of these two algorithms highly rely on initialization.
- Differences
    1. k-means is a batch learning algorithm while CL belongs to adaptive learning.
    2. CL has a hyper parameter η to adjust but k-means has no hyper parameter.
    3. k-means would take more time to converge when dealing with large data size. CL avoids this issue by updating the centers by one data point each time.

2.2 Analysis

RPCL can make extra centers far away to control the number of clusters so applying the idea of RPCL to the initialization of k-means will help to solve the problem. Suppose the initial

number of centers is m and the corresponding weight vectors are $v_i(i = 1, \cdots, m)$. For each center, its output u_i could be defined as follows:

$$u_i = \begin{cases} 1 & \text{if } u_i \text{ is the winner} \\ -1 & \text{if } u_i \text{ is the rival} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Take geometric distribution of the data points into consideration, a density function is defined as:

$$\rho(x_j) = e^{-\sum_{i=1}^{N} \frac{d(x_i, x_j)}{\sum_{i=1}^{N} d(x_1, x_i)}}, j = 1, 2, \cdots, N \tag{2}$$

$d(x_i, x_j)$ means the Euclidean distance between x_i and x_j. Then we have an adjustment for each weight vector v_i according to u_i and $\rho(x_j)$:

$$\Delta v_i = \begin{cases} \alpha\rho(x_j)(x_j - v_i) & u_i = 1 \\ \beta\rho(x_j)(x_j - v_i) & u_i = -1 \\ 0 & otherwise \end{cases} \tag{3}$$

And α and β are hyper parameters. Once the weight vectors converge, test the number of samples in each cluster, and if the ratio of the number of samples to the size of the data set is below the threshold T, delete the redundant class.

Algorithm 2: The idea of RPCL in k-means

Input : The dataset
Output: The initial m centers and corresponding weight vectors $v_i(i = 1, ..., m)$

Initialize the number of centers m and weight vectors v_i, threshold T, density $\rho(x_j), j = 1, 2, ..., N$.
while *the convergence criterion is not satisfied* **do**
  for *i = 0 to n* **do**
    Calculating u_i according to Eqn. 1.
    Adjust the weight vectors of each center via Eqn. 2.
Assign each data point to its closest center.
for *i = 0 to m* **do**
  if $\frac{N_i}{N} < T$ **then**
    Remove center v_i.
  where N_i denotes the number of data points that belong to *ith* center, N is the number of data points in the dataset.
return v;

2.3 Result

I have test three instance for my code, and when the initialization is good the correct centers will always be find like Figure 1,2,3 and Figure 4,5,6. However, when we have a bad initialization, there will be one point which always be the winner and push other competitors away. It is also depends on the order when dealing with the point because I deal with the data one by one and this situation in shown in Figure 7,8,9.

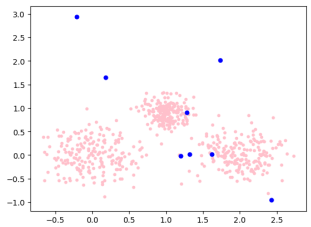

Figure 1: Initialize centers

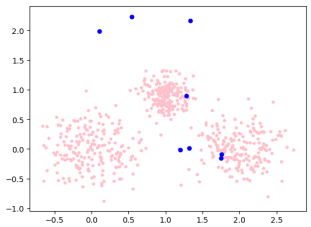

Figure 2: Assign centers

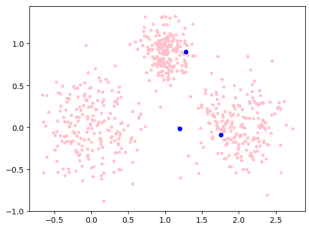

Figure 3: remove redundant centers

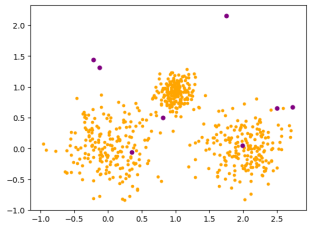

Figure 4: Initialize centers

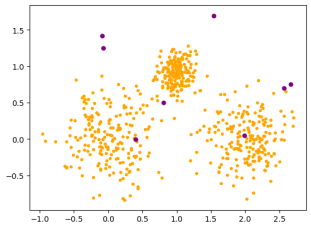

Figure 5: Assign centers

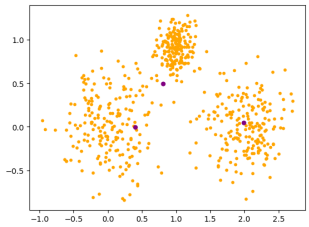

Figure 6: remove redundant centers

Failure instance

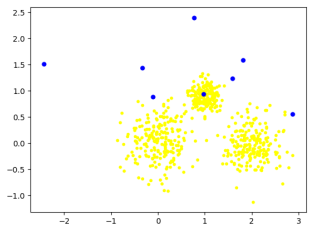

Figure 7: Initialize centers

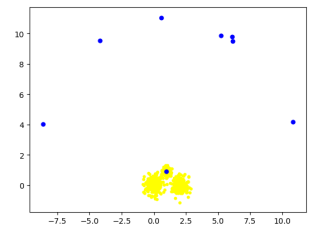

Figure 8: Assign centers

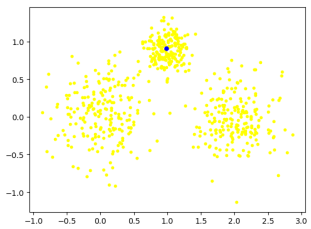

Figure 9: remove redundant centers

3 Model Selection of GMM

I present some visual experimental results among aic selection, bic selection and VBEM.

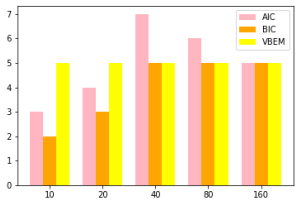

Figure 10: Comparison between clusters distance

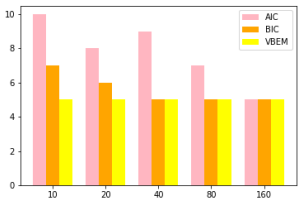

Figure 11: Comparison between sample size

3.1 Cluster distance

Experiment Settings

1. All data are randomly generated based on GMM and the data is generate by $k = 5$ components.
2. The centers are five points pointing to the four corners and the middle of the graph and the center's distance changes in $0.1, 0.4, 0.7, 1.1, 3.0$
3. The sample size is 50

model comparison

The model performance looks like

$$AIC, BIC << VBEM$$

The comparison is showed in Figure 10 and the results is showed from Figure 13 to Figure 27.And we can clearly get that the BIC performs worse then AIC at the beginning. However BIC will adjust itself more quickly than AIC because it will find the correct center number once the cluster gets a slighter separate. I think BIC gives more punishment on extra centers so BIC will find less centers while AIC will find more centers. VBEM performs quite well in this cases.

3.2 Sample size

Experiment Settings

1. All data are randomly generated based on GMM and the data is generate by $k = 5$ components.
2. The centers are $(0, 0), (2, 0), (1, 1), (2, 2), (0, 2)$, pointing to the four corners and the middle of the graph.
3. The sample sizes are $10, 20, 40, 80, 160$

model comparison

The model performance seems look like

$$AIC << BIC << VBEM$$

I second test the influence of data size. The comparison is showed in Figure 11 and the results is showed from Figure 29 to Figure 43. It is obvious that when data size gets larger all the approaches works well. However, when the data size is small, only VBEM can prevent overfitting cases and BIC seems to adjust better than AIC. I still thought it might be BIC punish more than AIC towards the redundant parameters and take the size into consideration while AIC do not.

3.3 Results

After testing for cluster distance and data size, it is obvious that when data size is large enough and the the cluster is separate enough, all the three approaches works well. However, when it comes to small data sizes and small data distance, AIC and BIC seems perform worse than VBEM. But AIC and BIC still have a lot of distance, BIC perfrom better just need a little improvement in the distance and size, while AIC needs more space to adjustment for itself. AIC tends to choose more clusters when it is not sure about number K and BIC and VBEM seems to prefer simple models.

4 Conclusion

In this project I have done some improvement for k-mean utilizing GMM. And I try to find some method for k-mean to deal with the number of clusters is out of range. The two process helps me learn more about the k-mean and GMM. Doing some experiment myself helps me have a deeper understanding for those clustering algorithms. In the third problem, I use different center initialization and data size to test the property of AIC, BIC and VBEM. Although I get a conclusion that $VBEM >> BIC >> AIC$ in total performance, I thought the conclusion might be restricted in some times. There are five centers and five data size chosen in my experiment and the performance changes not quite fiercely. What if there are some tiny differences in the three approaches that I have not tested? And I thought I must figure out the answers in my following study. What's more, thanks to the guidance from the teacher Tu and the assistant for their patience and erudition!

Figure 12: the last label is the cluster distance

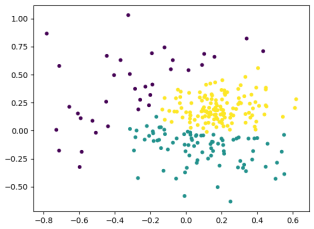

Figure 13: AIC-0.1

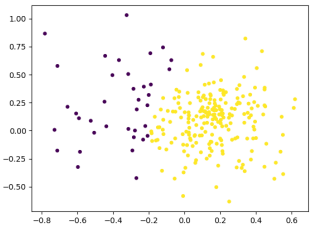

Figure 14: BIC-0.1

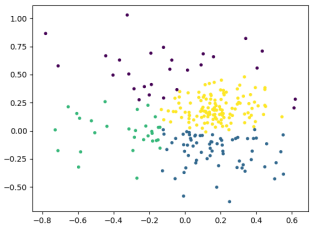

Figure 15: VBEM-0.1

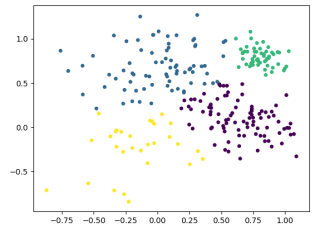

Figure 16: AIC-0.4

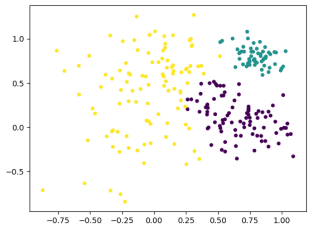

Figure 17: BIC-0.4

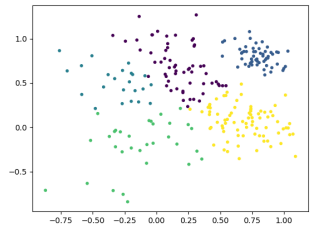

Figure 18: VBEM-0.4

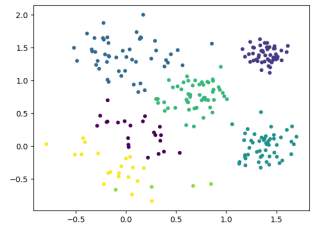

Figure 19: AIC-0.7

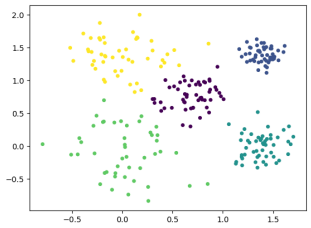

Figure 20: BIC-0.7

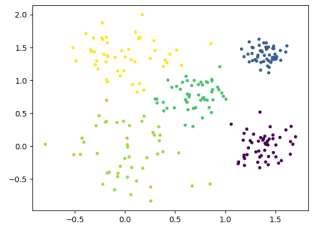

Figure 21: VBEM-0.7

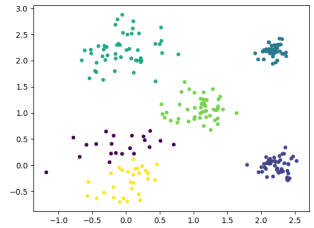

Figure 22: AIC-1.1

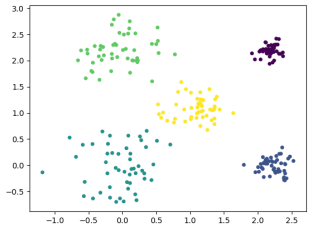

Figure 23: BIC-1.1

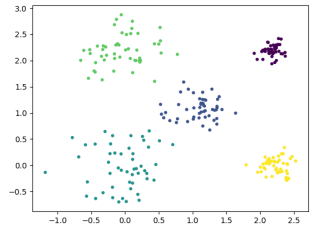

Figure 24: VBEM-1.1

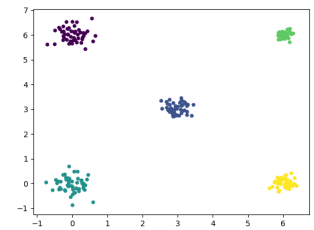

Figure 25: AIC-3.0

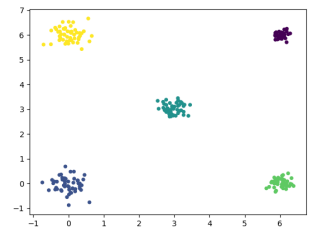

Figure 26: BIC-3.0

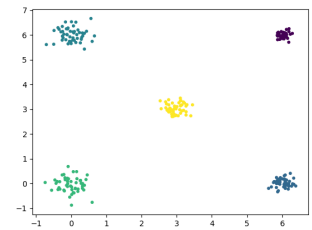

Figure 27: VBEM-3.0

Figure 28: the last label is the data size

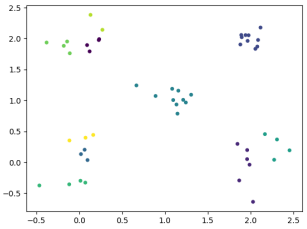

Figure 29: AIC-10

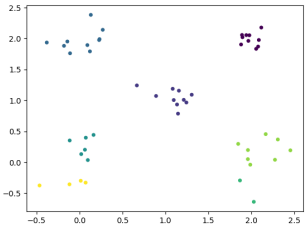

Figure 30: BIC-10

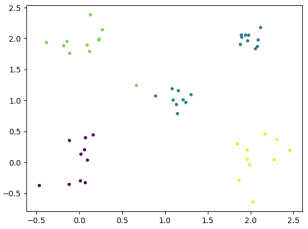

Figure 31: VBEM-10

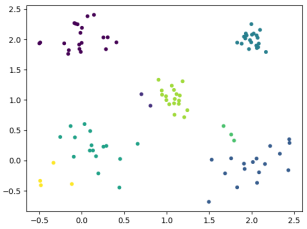

Figure 32: AIC-20

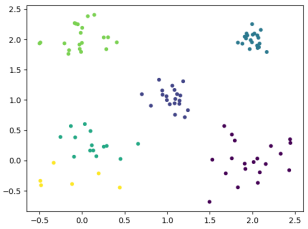

Figure 33: BIC-20

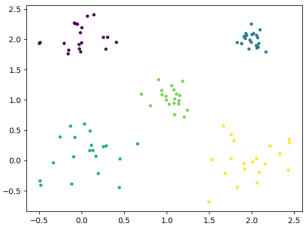

Figure 34: VBEM-20

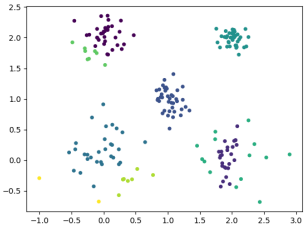

Figure 35: AIC-40

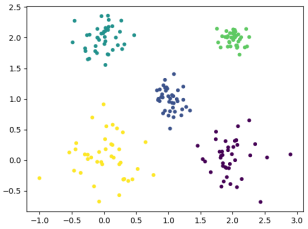

Figure 36: BIC-40

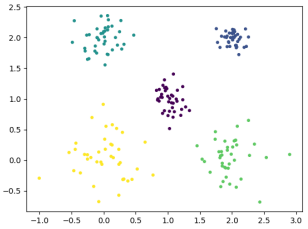

Figure 37: VBEM-40

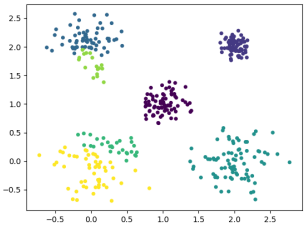

Figure 38: AIC-80

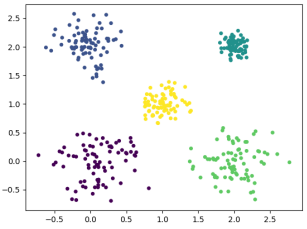

Figure 39: BIC-80

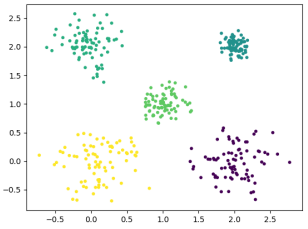

Figure 40: VBEM-80

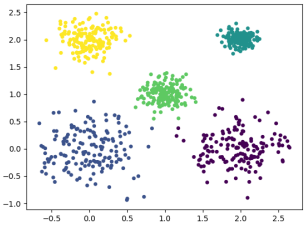

Figure 41: AIC-160

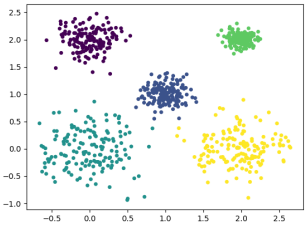

Figure 42: BIC-160

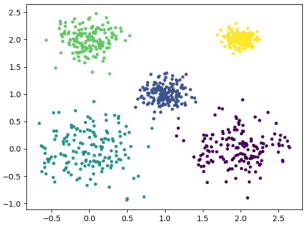

Figure 43: VBEM-160