Homework 1

CS420, Machine Learning, Shikui Tu, Spring 2020

1 k-mean algorithm

1. Obviously when assigning the points to the nearest μ_i,

$$\|x_n - \mu_k^{new}\| \leqslant \|x_n - \mu_k^{old}\|$$

and because of

$$J(\mu_1, \ldots, \mu_K) = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|x_n - \mu_k\|^2,$$

this step will never increase the k-mean objective function;

2. It can be proved that when μ is the mean of all data points, the sum of squared distances is the smallest.

$$\begin{aligned} f(\mu) &= \sum_{t=1}^{N}\|\mu - x_t\|^2 \\ &= \sum_{t=1}^{N}(\mu - x_t)^T(\mu - x_t) \\ &= \sum_{t=1}^{N}(\mu^T\mu - 2\mu^T\mu + x_t^T x_t) \end{aligned}$$

$$\frac{\partial f}{\partial \mu} = \sum_{t=1}^{N}(2\mu - 2x_t + 0) = 0$$

$$\Downarrow$$

$$\mu = \frac{1}{N}\sum_{t=1}^{N} x_t$$

Therefore the second step will never increase the k-mean objective function.

2 k-mean vs GMM

K-mean can be regarded as a special case of GMM. The main difference between k-means and GMM is that k-mean employs One-in-K assignment while GMM uses soft assignment. Here a variant of K-mean is put up by considering hard assignment instead of soft assignment in GMM, that is,

$$\gamma(z_{nk}) = \begin{cases} 1, k = \arg\max_j \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j) \\ 0, otherwise. \end{cases}$$

Algorithm 1: A variant of k-means

Input : The number of clusters K

Output: $\pi_k, \mu_k, \Sigma_k, (k = 1, 2, ..., K)$

Initialize the means μ_k, covariances Σ_k, mixing coefficients π_k and the initial value of the log likelihood;

while *the convergence criterion of parameters or log likelihood is not satisfied* **do**

E step. Evaluate the responsibilities with the current parameter values:

$$\gamma(z_{nk}) \leftarrow \begin{cases} 1, k = \arg\max_j \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j) \\ 0, otherwise. \end{cases}$$

M step. Re-estimate the parameters with the current responsibilities:

$$N_k \leftarrow \sum_{n=1}^{N} \gamma(z_{nk})$$

$$\mu_k^{new} \leftarrow \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n$$

$$\Sigma_k^{new} \leftarrow \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k^{new})(x_n - \mu_k^{new})^T$$

$$\pi_k^{new} \leftarrow \frac{N_k}{N}$$

Evaluate the log likelihood:

$$\ln p(X|\mu, \Sigma, \pi) \leftarrow \sum_{k=1}^{K} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \right\}.$$

return $\pi_k, \mu_k, \Sigma_k, (k = 1, 2, ..., K)$;

Comparing our algorithm with GMM, since we only consider the most suitable distribution for each data point, our calculation is reduced. Compared with K-mean, our algorithm improves robustness because it uses soft assignment. In addition, since the Mahalanobis distance is used instead of the Euclidean distance, the case of non-circular clusters can be handled.

However, the algorithm still faces similar limitations as GMM and K-mean. Firstly, the result is affected by the initialization. Secondly, it is usually not global optimization but local optimization. Thirdly, the parameter K still needs to be given in advance, but many times do not know what value K is appropriate.

3 k-mean vs CL

3.1 Comparison between k-mean and CL

- **Similarities**

  1. The effects of these two algorithms highly rely on initialization.
  2. They can not estimate the total number of clusters.
  3. They consider distance rather than probability.

- **Differences**

1. k-mean is a batch learning algorithm while CL is an online-learning algorithm.
2. CL has a hyper parameter but k-mean has no hyper parameter.
3. k-mean has higher computation complexity because CL updates the centers with one data point each time while k-mean considers a batch of data each iteration.

3.2 Application of RPCL in k-mean

K-mean cannot estimate the total number of clusters from the data while RPCL can keep other centers away to control the number of clusters. I will use RPCL's competitor punishment ideas for K-mean.

There are a few differences from k-mean:

1.
$$r_{nk} = \begin{cases} 1 & , k = c,\ c = \arg\max_j \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j) \\ -\eta & , k = b,\ b = \arg\max_{j \neq c} \pi_j \mathcal{N}(x_n|\mu_j, \Sigma_j) \\ 0 & , otherwise. \end{cases}$$

2. when $\gamma_{nk} = 1$,
$$\mu_k = \frac{\sum_n \gamma_{nk} x_n}{\sum_n \gamma_{nk}}$$

3. when $\gamma_{nk} < 0$,
$$\mu_i = \mu_i + \sum_k \sum_n \gamma_{nk}(\mu_k - \mu_i)$$

Algorithm 2: RPCL in k-means

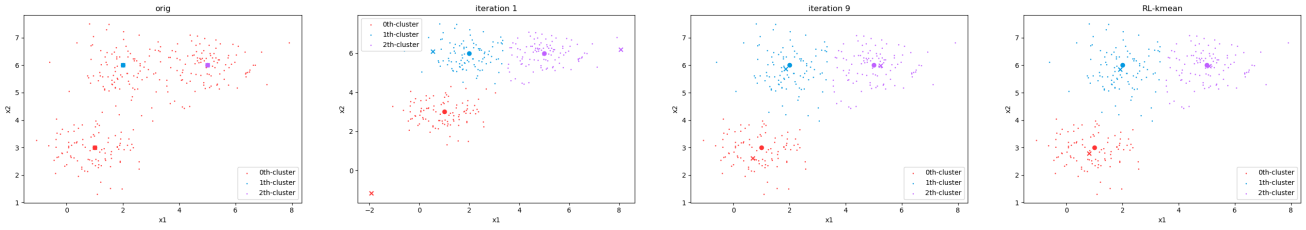
Input : The dataset $x_1, \ldots, x_N$
Output: μ

Initialize the number of centers K and $\mu_1, \ldots, \mu_k$.
while *the convergence criterion is not satisfied* **do**
  for *i = 0 to N* **do**
    for *k = 1 to K* **do**
      Calculating γ_{ik}
  for *k = 1 to K* **do**
    $\mu_k = \frac{\sum_n \gamma_{nk} x_n}{\sum_n \gamma_{nk}}$ when $\gamma_{nk} = 1$
  for *k = 1 to K* **do**
    $\mu_k = \mu_k + \sum_k \sum_n \gamma_{nk}(\mu_k - x_n)$ when $\gamma_{nk} < 0$
  $\eta = \frac{\eta}{t}$ where t is the number of iterations
return μ;

I test this application in a three-cluster dataset generated randomly.It can be seen from the results that to some extent, the algorithm can solve the situation where the initial K value does not match the actual.

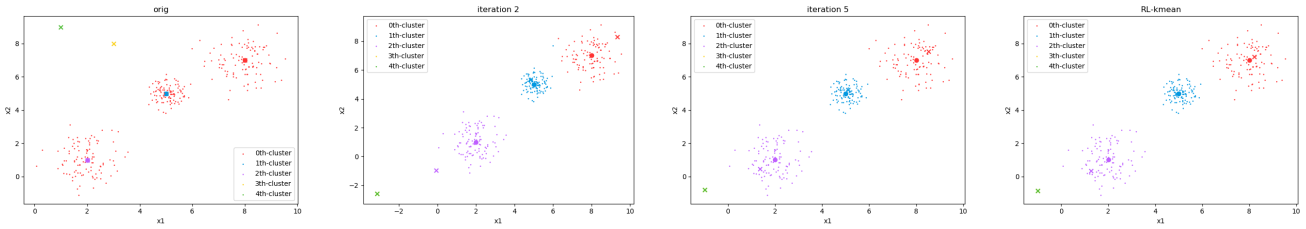
4 Model Selection of GMM

In this section, I compare model selection performances among BIC, AIC and VBEM about the sample sizes and the cluster center distances of the dataset.



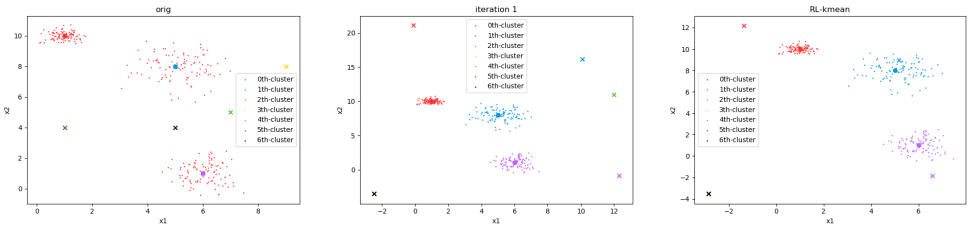
(a) (b) (c) (d)

Figure 1: The results of RPCL in kmeans when K is initialized to 3.



(a) (b) Result_1 (c) Initialization_2 (d) Result_2

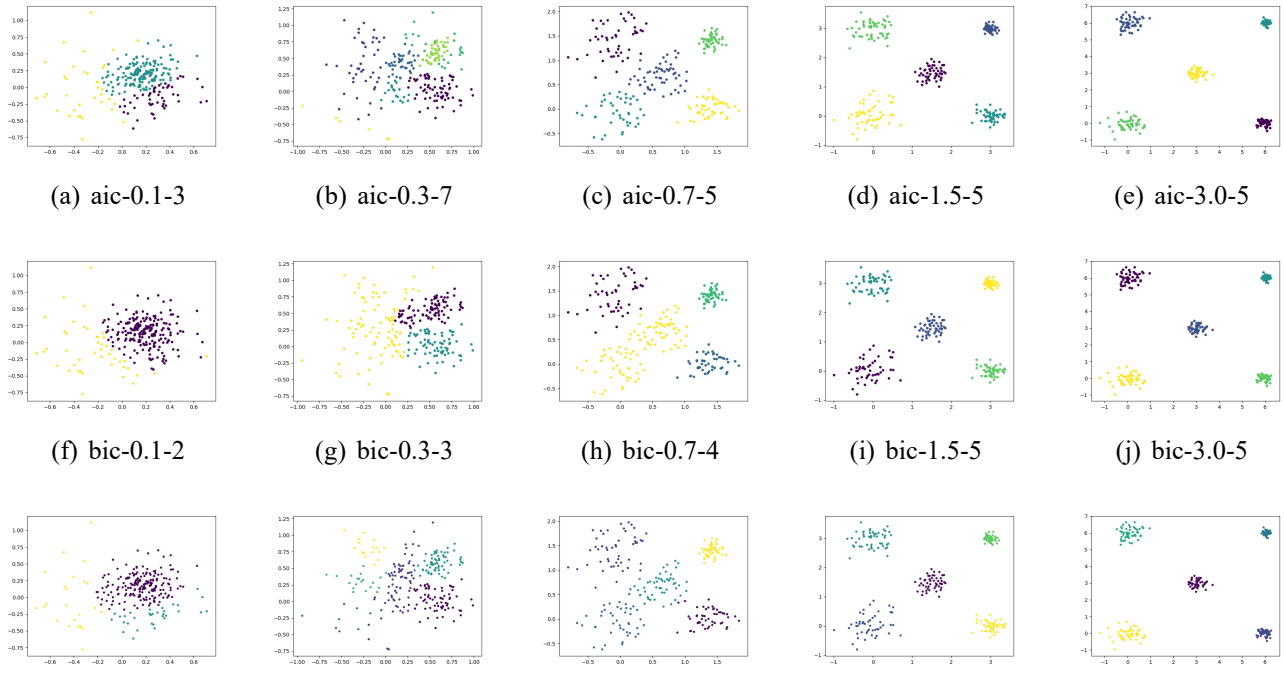
Figure 2: The results of RPCL in kmeans when K is initialized to 5.



(a) (b) Result_1 (c) Initialization_2

Figure 3: The results of RPCL in kmeans when K is initialized to 7.

4.1 Sensitive on Distance Among Cluster Centers



(a) aic-0.1-3 (b) aic-0.3-7 (c) aic-0.7-5 (d) aic-1.5-5 (e) aic-3.0-5

(f) bic-0.1-2 (g) bic-0.3-3 (h) bic-0.7-4 (i) bic-1.5-5 (j) bic-3.0-5

(k) vbem-0.1-3 (l) vbem-0.3-5 (m) vbem-0.7-5 (n) vbem-1.5-5 (o) vbem-3.0-5

Figure 4: Model selection performance of BIC, AIC and VBEM at different clustering center distances. Each line from top to bottom represents AIC, BIC and VBEM. The two numbers below each figure respectively represent the distance between the five cluster centers and the K selected by the corresponding method.

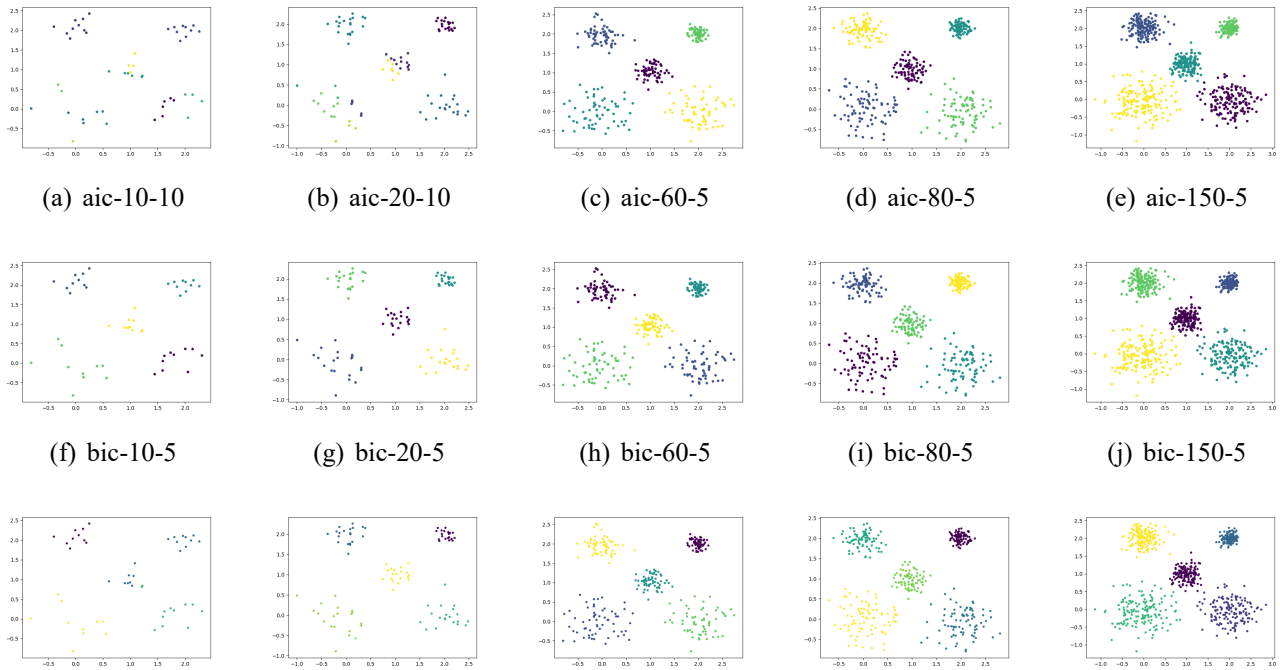
The results show that the performance of VBEM is better than BIC, and it is much better than AIC. Besides, under the high cluster density data set, all three methods will achieve good performance. And BIC will self-adjust faster than AIC. In the face of low cluster density, AIC is easy to collapse but BIC and VBEM can be better applied.

4.2 Different Sample Sizes

The actual K value of the generated original data is 5, and it can be seen from the results that the classification effect of VBEM is much better than AIC and BIC when the sample size is small. In addition, when the number of samples increases, BIC performs better than AIC, which may be related to the number of samples included in the BIC expression. When the sample size is large, the classification performance of the three is excellent.In this experiment, VBEM achieves best clutsering result, BIC is the next and AIC is the worst.

5 Conclusion

In this assignment, I deepened my understanding of k-mean, compared K-mean and GMM, applied the idea of RPCL to k-mean, and conducted experiments related to model selection. I know that these are just a small part of machine learning. I will apply what I have learned in the classroom to practice in future learning, and actively explore more algorithms and applications. Finally, thank teacher Tu and teaching assistant for their patienct guidance and help!



(a) aic-10-10 (b) aic-20-10 (c) aic-60-5 (d) aic-80-5 (e) aic-150-5

(f) bic-10-5 (g) bic-20-5 (h) bic-60-5 (i) bic-80-5 (j) bic-150-5

(k) vbem-10-5 (l) vbem-20-5 (m) vbem-60-5 (n) vbem-80-5 (o) vbem-150-5

Figure 5: The model selection performances of AIC, BIC and VBEM on different sample sizes. Each row from up to bottom denotes AIC, BIC and VBEM respectively. The two numbers below each figure respectively represent the sample numbers and the K selected by the corresponding method.