HW1

Machine Learning

1 K-means vs GMM

As a special case of GMM, k-means uses One-in-K (hard) assignment while GMM utilizes soft assignment. In this section I will give a variant of k-mean algorithm which carries out soft clustering.

The algorithm is shown in Alg. 1.

Algorithm 1: Soft K-means.

```
1 Initialize $\mu_1, \dots, \mu_k$;
2 while $\mu_i$ is not converged do
3     E step. Soft Assignment:
4     for all $x^t$ do
5         $$b_i^t = \frac{1/\|x^t - \mu_i\|^2}{\sum_{j=1}^k 1/\|x^t - \mu_j\|^2}$$
6     M step. Re-estimate $\mu_1, \dots, \mu_k$:
7     for all $\mu_i, i = 1, 2, \dots, k$ do
8         $$\mu_i = \frac{\sum_t b_i^t x^t}{\sum_t b_i^t}$$
```

Compared with GMM, this algorithm does not consider the mixing weight and covariance. So there are less parameters to be initialized in Alg. 1 than in GMM and the model of Alg. 1 is simpler than that of GMM. The only difference between Alg. 1 and K-mean is that Alg. 1 uses soft assignment instead of hard assignment. Therefore this algorithm is easier to calculate than GMM and more robust than K-means. However, this algorithm suffers from the same problem as K-means and GMM do: it finds the nearest local optimal and the optimum depends on the initialization. Besides, the value of k will affect the results of the model. Since this algorithm does not consider mixing weight and covariance, it is less general than GMM.

2 K-mean vs CL

2.1 Comparison between k-means and CL

- **Similarities**
  1. The performance depends largely on initialization.
  2. They consider data in distance framework instead of probability.
- **Differences**
  1. k-means is a batch learning algorithm while CL is adaptive (online) learning algorithm. k-means deals with a given batch of data points and CL deals with data points that come one by one.

2. CL updates the centers by one data point each time, which makes the computation much easier than k-means.
3. CL can be more sensitive to outliers since it updates centers using one data point each time.

2.2 Rival Penalized k-means

In my opinion, there are two ways to apply the idea of RPCL in k-means algorithm:

- Preprocess the data with RPCL to define the value of k, then adopt k-means with the obtained k.
- After each EM step, we find centers that are close and do rival penalty to the one with less points belonging to it.

Since I am not sure whether the second method can result in convergence, here I implement the first method. The algorithm is shown in Alg. 2. I use $\alpha = 0.05$ and $T = 100$ in the experiment. The dataset is 3-cluster 2-dimension, with a sample size of 100. The initial k is set to be 8. The initialization and result are shown in Fig. 1. Using RPCL to initialize the centers can result in a better performance of k-means and automatically define the value of k.

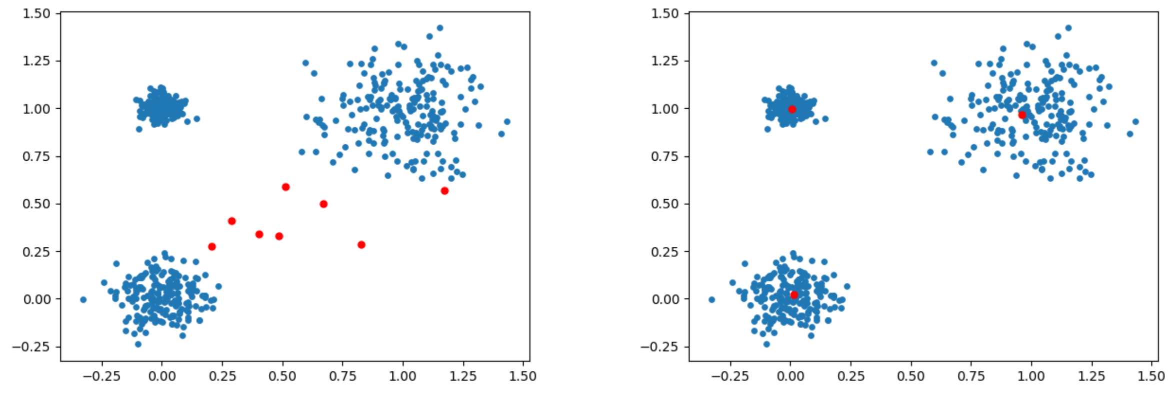

Figure 1: The result of Rival Penalized k-means algorithm.

3 Model Selection of GMM

In this section, the performances of BIC, AIC and VBEM on model selection are evaluated in terms of sample size, the number of clusters and data dimensionality.

3.1 Sample Size

In this part we evaluate the performance of BIC, AIC and VBEM with different sample size. The datasets for evaluation are randomly generated 4-cluster 2-dimension data points. The results are shown in Fig. 2, Fig. 3, Fig. 4 and Fig 5. As we can see from the result, the three algorithm perform well with the sample size increases. Among them, AIC performs worse when sample size is small while VBEM performs best.

Algorithm 2: Rival Penalized k-means.

```
1  Initialize $\boldsymbol{\mu_1}, \dots, \boldsymbol{\mu_m}$;
2  **RPCL step.**
3  **while** *the convergence criterion is not satisfied* **do**
4      **for** $i = 0$ *to* $n$ **do**
5          $$u_i = \begin{cases} 1 & \text{if } \boldsymbol{\mu_i} \text{ is the winner} \\ -1 & \text{if } \boldsymbol{\mu_i} \text{ is the rival} \\ 0 & \text{otherwise} \end{cases},$$
6          $$\Delta v_i = \begin{cases} \alpha\rho(\boldsymbol{x_j})(\boldsymbol{x_j} - \boldsymbol{\mu_i}) & u_i = 1 \\ -\alpha\rho(\boldsymbol{x_j})(\boldsymbol{x_j} - \boldsymbol{\mu_i}) & u_i = -1, \\ 0 & \text{otherwise} \end{cases}$$
           where
           $$\rho(\boldsymbol{x_j}) = \exp[-\sum_{i=1}^{N} \frac{d(\boldsymbol{x_i}, \boldsymbol{x_j})}{\sum\limits_{i=1}^{N} d(\boldsymbol{x_t}, \boldsymbol{x_j})}], j = 1, 2, ..., N$$
7  **for** *i = 0 to m* **do**
8      $$\eta = \frac{N_i}{N},$$
9      **if** $\eta < T$ **then**
10         Remove center $\boldsymbol{\mu_i}$.
11 **EM step.**
12 **while** $\boldsymbol{\mu_i}$ *is not converged* **do**
13     **for** *all* $\boldsymbol{x^t}$ **do**
14         $$b_i^t = \begin{cases} 1 & i = arg\min_j \|\boldsymbol{x^t} - \boldsymbol{\mu_j}\|^2 \\ 0 & otherwise \end{cases}$$
15     **for** *all* $\boldsymbol{\mu_i}$,$i = 1, 2, \dots, k$ **do**
16         $$\boldsymbol{\mu_i} = \frac{\sum_t b_i^t \boldsymbol{x^t}}{\sum_t b_i^t}$$
17 Return $\boldsymbol{\mu_1}, \dots, \boldsymbol{\mu_k}$
```

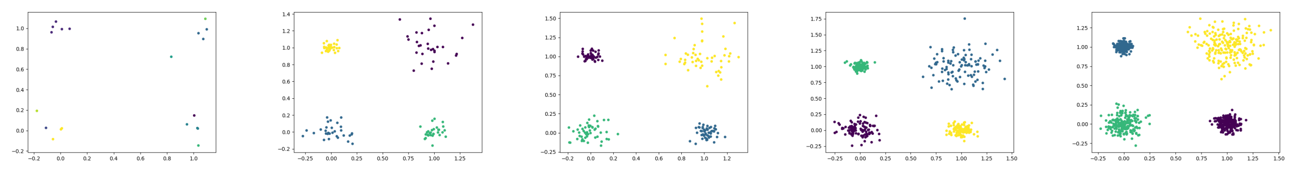

Figure 2: The model selection performances of BIC in terms of different sample sizes. (From left to right the sample size is (5, 30, 50, 100, 200) and the k selected by the algorithm is (10, 4, 4, 4, 4).

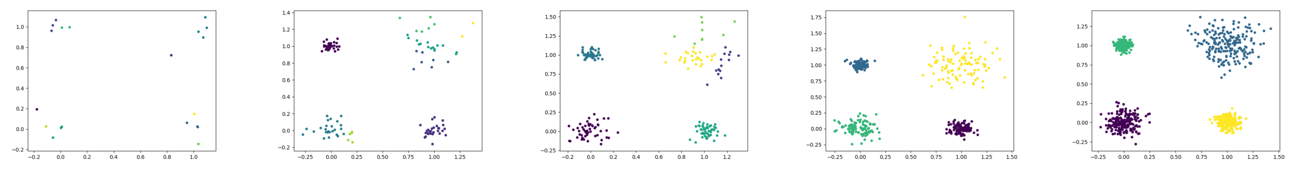

Figure 3: The model selection performances of AIC in terms of different sample sizes. (From left to right the sample size is (5, 30, 50, 100, 200) and the k selected by the algorithm is (10, 8, 6, 4, 4).

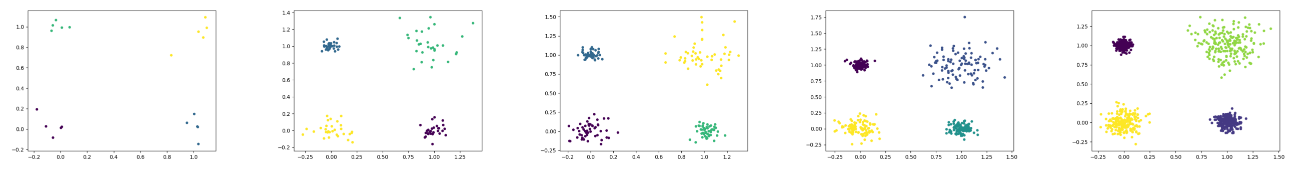

Figure 4: The model selection performances of VBEM in terms of different sample sizes. (From left to right the sample size is (5, 30, 50, 100, 200) and the k selected by the algorithm is (4, 4, 4, 4, 4).

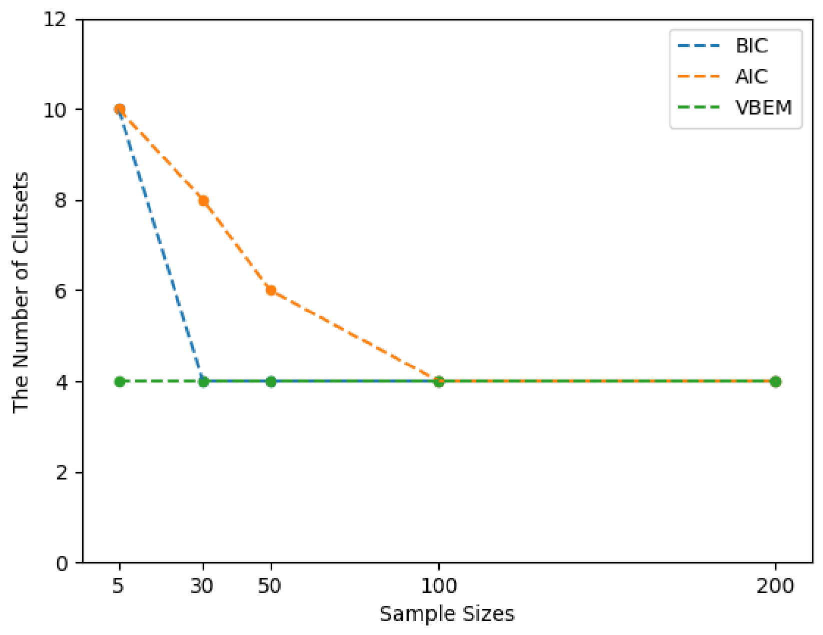

Figure 5: The model selection performances of BIC, AIC and VBEM in terms of different sample sizes.

3.2 Dimensionality

In this part we evaluate the performance of BIC, AIC and VBEM with different number of clusters. The datasets for evaluation are randomly generated 4-cluster data points with a sample size of 80. The results are shown in Fig. 6, Fig. 7, Fig. 8 and Fig 5. As shown in Fig. 9, the performance of BIC and VBEM are almost the same when the sample size is 80 and the number of clusters is 4, while AIC performs worse compared to the other two algorithms.

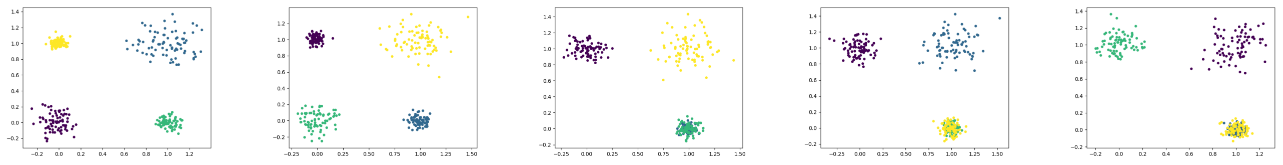

Figure 6: The model selection performances of BIC in terms of different data dimension sizes. (From left to right the dimension size is (2, 3, 4, 5, 6) and the k selected by the algorithm is (4, 4, 4, 4, 4).

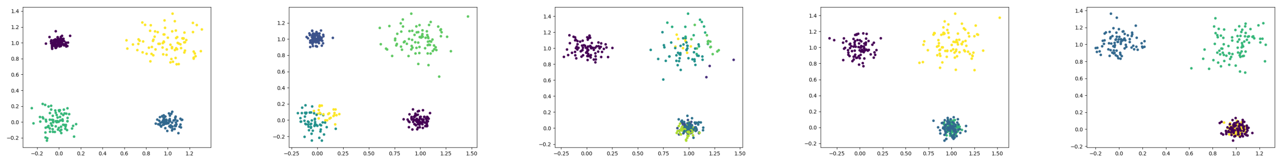

Figure 7: The model selection performances of AIC in terms of different data dimension sizes. (From left to right the dimension size is (2, 3, 4, 5, 6) and the k selected by the algorithm is (4, 5, 9, 4, 4).

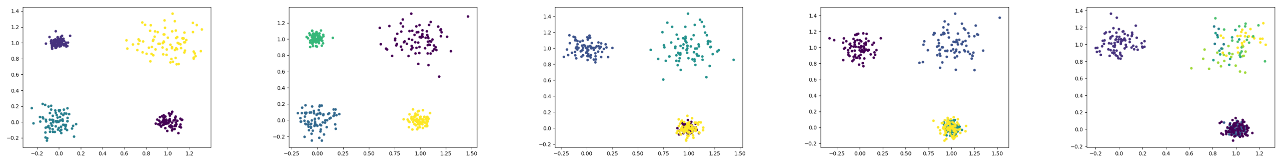

Figure 8: The model selection performances of VBEM in terms of different data dimension sizes. (From left to right the dimension size is (2, 3, 4, 5, 6) and the k selected by the algorithm is (4, 4, 4, 4, 4).

3.3 Number of Clusters

In this part we evaluate the performance of BIC, AIC and VBEM with different number of clusters. The datasets for evaluation are randomly generated 2-dimension data points with a sample size of 80. The results are shown in Fig. 10, Fig. 11, Fig. 12 and Fig 5. As shown in Fig. 13, the performance of BIC and VBEM are almost the same when the sample size is 80 while AIC performs worse compared to the other two algorithms.

3.4 Conclusion

With the evaluation of the performance of the three algorithms in terms of sample size, number of clusters and data dimensionality, we can see that averagely AIC performs worse than the other two algorithms and VBEM has a better performance when there are fewer sample datas. As the sample size increases,

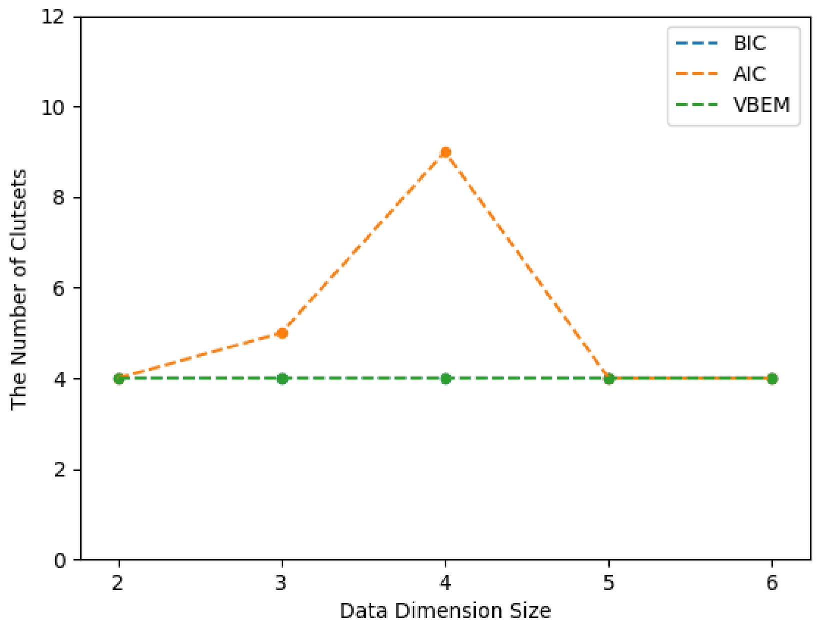

Figure 9: The model selection performances of BIC, AIC and VBEM in terms of different sample sizes.

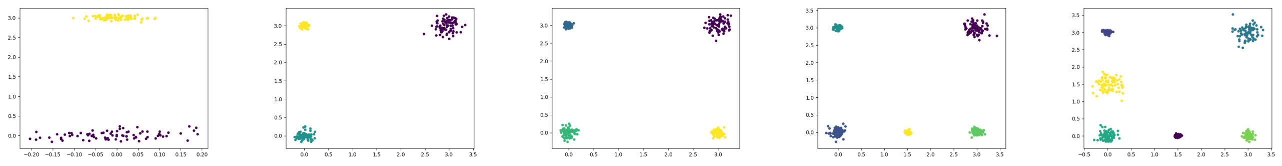

Figure 10: The model selection performances of BIC in terms of different number of clusters. (From left to right the number of clusters is (2, 3, 4, 5, 6) and the k selected by the algorithm is (2, 3, 4, 5, 6).

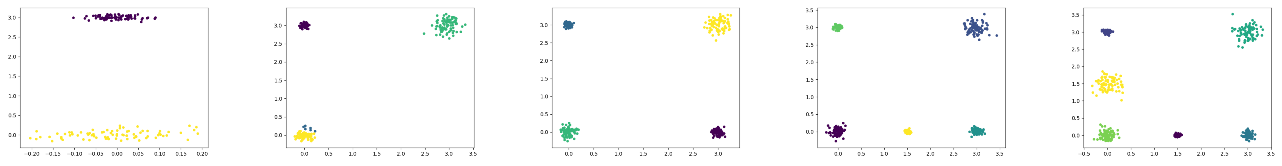

Figure 11: The model selection performances of AIC in terms of different number of clusters. (From left to right the number of clusters is (2, 3, 4, 5, 6) and the k selected by the algorithm is (2, 4, 4, 5, 6).

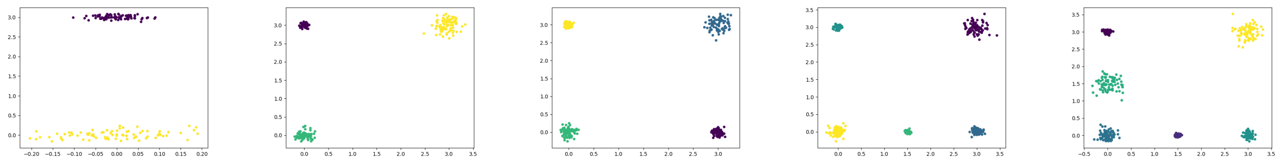

Figure 12: The model selection performances of VBEM in terms of different number of clusters. (From left to right the number of clusters is (2, 3, 4, 5, 6) and the k selected by the algorithm is (2, 3, 4, 5, 6).

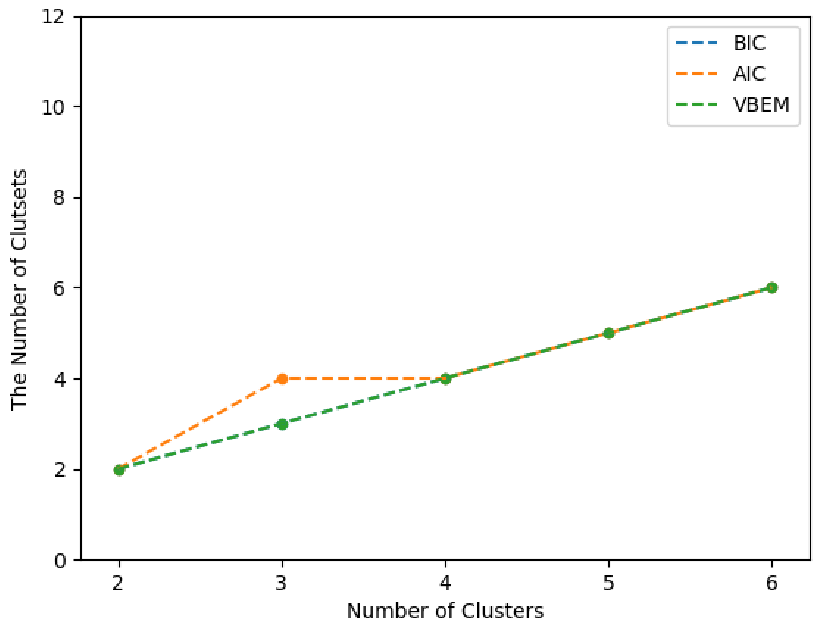

Figure 13: The model selection performances of BIC, AIC and VBEM in terms of different sample sizes.

the gap between the performances of the three algorithms decreases. I did not see any distinct effect of data dimensionality on the performance of BIC and VBEM, which may be attributed to the relatively large sample size setting of the experiment. In brief, the most important factor is the sample size: the larger the sample size is, the better the performance is.