# CS420 Machine Learning – Assignment 1

**Xiang Gu**
5130309729

## 1 K-means vs. GMM

**Probabilistic K-means**: One possible variant of K-means is to borrow the idea of soft-assignment in GMM to K-means and thus create a probabilistic version of K-means algorithm which I call probabilistic K-means algorithm.

Basically, this new algorithm does exactly the same thing as K-means except that now each data point $x_i$ is assigned to all clusters $\mu_1, \mu_2, \ldots, \mu_K$ according to the distance from $x_i$ to each cluster. Without loss of generality, let us assume, for data point $x_i$, $\mu_1, \mu_2, \ldots, \mu_K$ are the clusters sorted based on some distance metrics (say, square of Euclidean distance) to $x_i$ in ascending order. Then in the assignment step of K-means, instead of assign $x_i$ completely to $\mu_1$, this new algorithm will assign $x_i$ to $\mu_k$ with proportion $a_{ik}$ as follows:

$$\text{E-Step: } A_{ik} = \frac{e^{-||x_i - \mu_k||^2}}{\sum_{j=1}^{K} e^{-||x_i - \mu_j||^2}} \tag{1}$$

Next we need to modify the update of positions of all the clusters. For each cluster $\mu_k$, rather than change it to the mean position of all the data points assigned to $\mu_k$, we now change it to the average position of all data points weighted by their proportions $a_ik$'s for all $i$:

$$\text{M-Step: } \mu_k = \frac{\sum_{i=1}^{N} a_{ik} x_i}{\sum_{i=1}^{N} a_{ik}}, \forall k \tag{2}$$

**Possible Advantages and Limitations**: One possible advantage of this probabilistic K-means is that it can alleviate the 'bad initialization' issue of K-means algorithm. Concretely, if the data points can be indeed clustered into two group and, somehow or another, we have two clusters ($K = 2$). But one cluster $\mu_2$ is unfortunately initialized far away from those data points whereas $\mu_1$ is initialized to be close enough to the center of all the data points. Then the traditional K-means algorithm will get stuck because all data points will be assigned to $\mu_1$ and it will terminate by placing $\mu_1$ at the center of all data points and leave $\mu_2$ alone in the corner.
Probabilistic K-means algorithm, on the other hand, can drag $\mu_2$ to the center of all those data points as well. From there, $\mu_2$ can gradually move to the center of one group of the data points and thus correctly cluster the data points.

The main disadvantage of this algorithm will be the computational cost because now we essentially need to compute the distance of all data points to all clusters, which is then used to compute the proportion matrix $A \in R^{N \times K}$ where $A_{ik} = a_{ik}$.

## 2 K-means vs. Competitive Learning (CL)

**Comparison**: K-means and CL are two different algorithm for the same task – clustering in a unsupervised context. One major difference though is that K-means does the job when all the data points are readily present while CL tries to approach the task when data points arrives one by one in a stream. Of course, one can, in principle, wait until the arrival of all data points and then use K-means to do the clustering but it is more desirable to be able to update the cluster while the data points are coming in, so at any time point, we would have all the clusters to be placed in a reasonably good position. Therefore, CL can be viewed as an online algorithm for clustering.

**Attempt to Incorporate Rival Penalizing Idea to K-means**: Well, Okay, here is one straightforward implementation of carrying the rival penalizing idea to K-means. Before going to the details of the algorithm, let us discuss a bit more about the matrix $A$ in section 1. Recall that $A$ is a $N \times K$ matrix and each entry $A_ik$ represents the assignment proportion of data point $x_i$ to cluster $\mu_k$. In original K-means algorithm, we are in fact implicitly construct this matrix, where each row

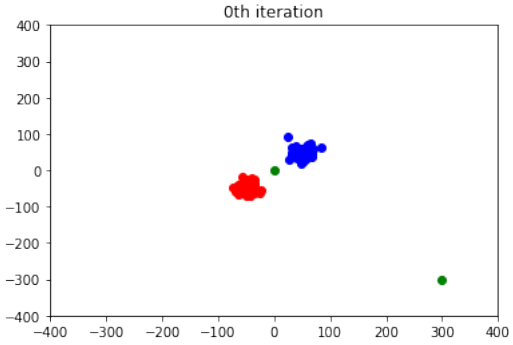


Figure 1: The initial configuration of the problem. Blue and Red are two heap of 100 data points. Two clusters (green) are initialized in the indicated positions.

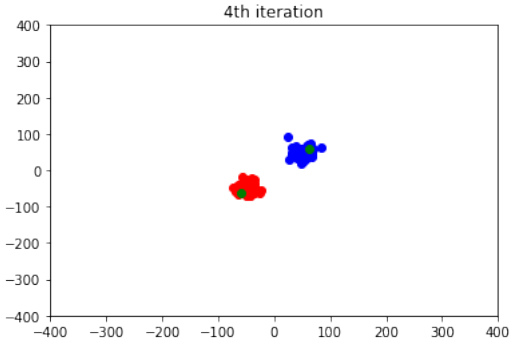


Figure 2: Final result found by our rival penalized K-means algorithm.

$A_i$ has only one entry that is $1$ and $0$'s everywhere else. When updating the positions of clusters, K-means uses exactly the same formula as equation 2.

One rival-penalized version of K-means can thus be implemented by changing A slightly: for each row $A_i$, everything is the same as before, except that now the entry $A_{ij}$ is $-\gamma$ when cluster $\mu_j$ is the second closest cluster to data point $x_i$. The update (M-step) remains the same as equation 2. An example of matrix $A$ when there are three clusters is provided below.

$$A = \begin{bmatrix} 1 & 0 & -\gamma \\ 1 & -\gamma & 0 \\ \vdots & \vdots & \vdots \\ 0 & -\gamma & 1 \end{bmatrix}$$

**Experimental Results**: We tested two scenarios of our algorithm:

- Two data point groups, two clusters, bad initialization: The initialization was set to be the one shown in figure 1. We have already seen in class that conventional K-means algorithm will get stuck in such case. Our rival penalized version ($\gamma = 0.1$), however, gave a final result shown in figure 2.

  Great! Our algorithm successfully finds the correct data centers.

- Two data point groups, three clusters, bad initialization: The initialization is shown in figure 3. Our algorithm found a final result shown in figure 4.

  Again, our algorithm reaches a pretty good solution by successfully kicking out one of three competing clusters! Yay!

## 3 Model Selection of GMM

Reference: https://scikit-learn.org/stable/auto_examples/mixture/plot_gmm_selection.html

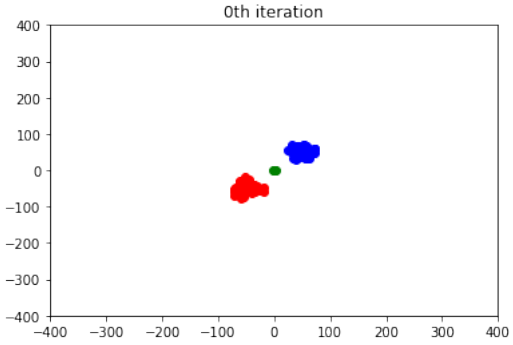


Figure 3: Initialization of scenario 2. Three clusters (green) are initialized roughly at the same position, thus creating a competing context.

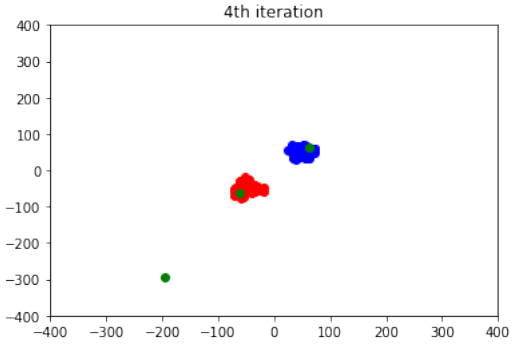


Figure 4: Final results of our algorithm in scenario 2.

- BIC vs. AIC: In this section, we will first see how to use both AIC and BIC to select the best GMM for a set of randomly generated data points. Namely, for a GMM, we concerned both the type of covariance and number of components in the model.
  We can see that in this specific case, BIC gives the same result (figure 5) as AIC (figure 6).
- VBEM: I really don't understand this part so I don't know how to do it. Sorry.

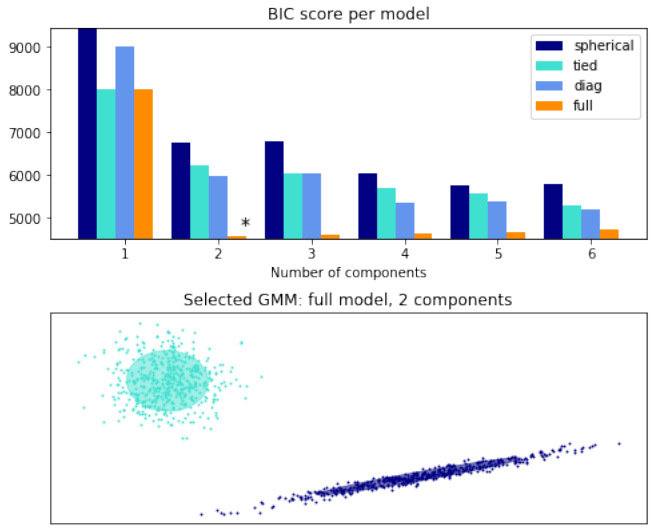


Figure 5: Model selection with BIC. Top figure shows different candidates of GMM that we tested, and figure below shows the optimal model fitting the data set.

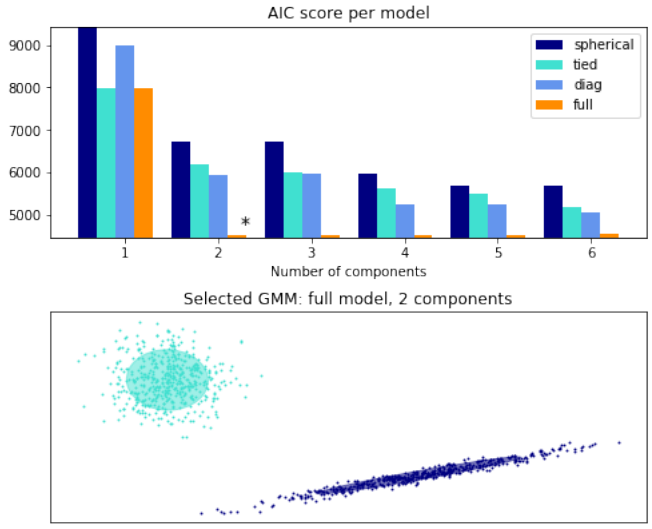


Figure 6: Model selection with AIC.