Experimental Report on Implementing and Evaluation of GMM for Clustering

Yuxiang Zhang

I. Methods

In this experiment, we present the results performed by different models in clustering unlabeled data. The selected models contains Gaussian Mixture Models(GMM) algorithm, which is implemented by us, and a well-developed toolbox: sklearn. This section includes two parts, in the first part, we will briefly explain the process of GMM using to cluster unlabeled data. In the other part, we will present the Algorithmic description of GMM Clustering.

For the convenience of the following description, the matrix of input sample features is called $\mathbf{X}$, the mean of sample features in the k^{th} cluster is called μ_k, the variance of them in is called σ_k, the weight of different cluster is called α_k, a latent variable γ_{jk} is also introduced to confirm the cluster for samples.

A. Process of GMM for Clustering Unlabeled Data

The GMM algorithm is a probability distribution model by mixing k Gaussian distributions with a certain proportion of α like Equation(1). By using multiple Gaussian distributions, GMM can approximate the probability distribution of any shape. Each Gaussian distribution is called a ”Component”, and the linear addition of these ”Component” is the probability density function of GMM.

$$P(X,\gamma|\theta) = \prod_{k=1}^{K} \alpha_k^{n_k} \prod_{j=1}^{N} [\frac{1}{\sqrt{2\pi}\sigma_k} exp(-\frac{(X_i-\mu_k)^2}{2\sigma_k^2})]^{\gamma_{jk}}. \quad (1)$$

The input datas to be clustered are regarded as distributed sampling points. The parameters of Gaussian distribution will be solved by EM algorithm. The data is compared with the probability of different ”Component”, then it will become the cluster with the maximal probability.

Finally, GMM can cluster the unlabeled data.

B. Algorithmic Description of GMM Clustering

In this section, we will introduce the GMM algorithm used in clustering. The general idea can be divided into 4 steps, the first step is determining the number of clusters k, which is also a hyperparameter. The second step is initializing the parameters in Gaussian distributions. Then, the third step is using EM algorithm iteration to solve the parameters. The last step is using the parameters to calculate the probability, confirming the cluster. The detail is in Algorithm1).

$$\widehat{\gamma_{jk}} = \frac{\alpha_k P(X_i|\theta_k)}{\sum_{k=1}^{K} \alpha_k P(X_i|\theta_k)}. \quad (2)$$

$$\widehat{\mu_k} = \frac{\sum_{k=1}^{K} \gamma_{jk} X_i}{\sum_{k=1}^{K} \gamma_{jk}}. \quad (3)$$

$$\widehat{\sigma_k^2} = \frac{\sum_{k=1}^{K} \gamma_{jk} (X_i-\mu_k)^2}{\sum_{k=1}^{K} \gamma_{jk}} \quad (4)$$

$$\widehat{\alpha_k} = \frac{\sum_{k=1}^{K} \gamma_{jk}}{N} \quad (5)$$

Algorithm 1 GMM for Clustering

```
Input : The input data features $\mathbf{X}$.
Output : Parameters of Gaussian Mixture Models.
Initializing parameters $\mu_k$, $\sigma_k$, $\alpha_k$ and $\gamma_{jk}$.
Randomly select the clustering centers.
Setting the iteration times maxItem (default 50).
while step<maxItem do
    for Every sample do
        Calculating the $\gamma_{jk}$ using Equation(2).
        Renewing the $\mu_k$, $\sigma_k$ and $\alpha_k$ by Equation(3), (4) and
        (5), respectively.
    end for
    $step + 1$
end while
for Every sample do
    Find the maximal $\gamma_{ji}$ in $(\gamma_{j1}, \gamma_{j2}..., \gamma_{jk})$.
    Saving the i in result, the i means the data is in the $i^{th}$
    cluster.
end for
```

II. Experimental settings

The data comes from ”GMM_EM_data_for_clustering.csv”. We use the entire data to employ the GMM model. By drawing the data scatter diagram for preliminary observation, we choose to divide the data into four clusters. So we set the hyperparameter k as 4. Then, we randomly set the initial parameters μ_k, σ_k, α_k and γ_{jk} with 0 or 0.25.The environment we use is python 3.9, numpy 1.20.3, pandas 1.3.4, scikit-learn 0.21.2 and matplotlib 3.5.0.

III. Results & Discussion

A. Performance of GMM on Clustering

Firstly, we employ our implementation on test data, drawing the scatter plot where the same color points belong to the same cluster. We find that the GMM implemented by us can sometimes give the right answer (see Fig.1).

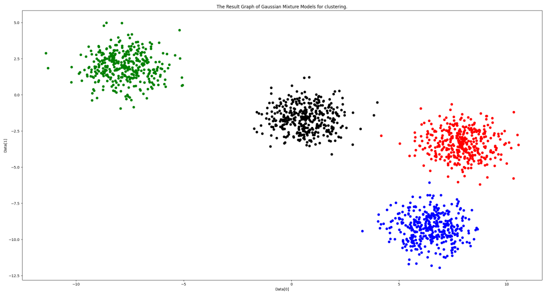

Fig. 1. The Result Graph of Gaussian Mixture Models for clustering.

However, when changing the initial parameters by using random initialization, some wrong results were presented. (see

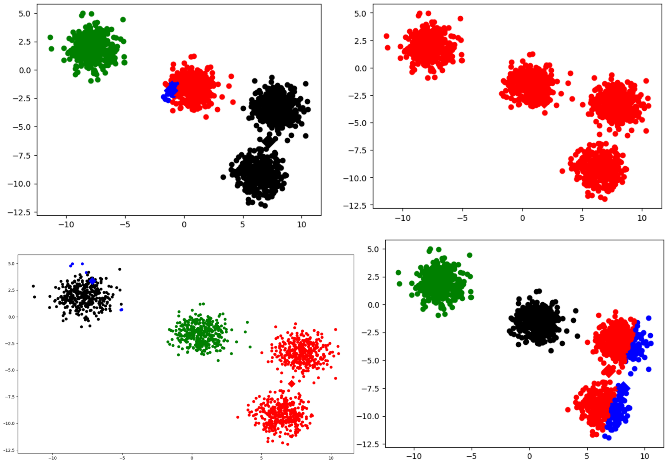

Fig. 2. The Wrong Result Graph of GMM for clustering.

Fig.2) It can be seen that when the four initial samples are selected in the left upper, middle, middle, right lower, and the right lower point is just in the center of the two sets of data points, GMM may judge the two sets of data points in the right lower corner as the same cluster. When the initial samples are selected far from the data points, GMM may even divide all data points into the same cluster. When the selected initial points are evenly located on both sides of the midline of the two sets of data, the GMM may present the result like the right lower graph in Fig.2.

Overall, under the same maximum number of iterations, the results indicated that GMM is significantly affected by the initial selection centers, in other word, the initial parameters. But this problem can be alleviated by further increasing the maximum number of iterations.

B. Comparing the GMM with Sklearn GMM

We use the GMM and the well-developed toolbox, sklearn, employing on the same dataset, drawing the result graph of sklearn GMM (see Fig3).

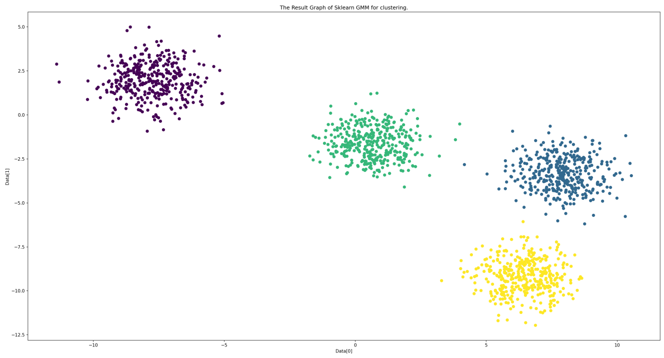

Fig. 3. The Result Graph of Sklearn GMM for clustering.

When clustering the same dataset, both models mis-cluster only one data point. Considering that the number of errors is too small, the statistical test can't be used. By comparing the result graph, we think that GMM performs well in this cluster task. But sklearn GMM's results keep high accuracy in many attempts and having a significantly shorter time than that of GMM. We are supposing that there 's an early stop strategy that ensures accuracy and a shorter run time.

References

[1] hktxt, EM.ipynb, 2021, GitHub repository [Online]. Available: https://github.com/hktxt/Learn-Statistical-Learning-Method/tree/master/EM