Anomaly detection using Gaussian Mixture Model

Cao Van Nam

January 25, 2020

1 Expectation maximization

First, K clusters is choosen randomly. The point of EM algorithm is to adjust these clusters to fit the training data. In the E step, the likelihood of each data sample is calculated based on the parameters of the k^th cluster:

$$f(x|\mu_k, \Sigma_k) = \frac{1}{(2\pi)^{d/2}\Sigma_k^{-1/2}} exp\left(-(x-\mu_k)\Sigma_k^{-1}(x-\mu_k)^T\right)$$

where d is the number of dimentions of x, μ_k and Σ_k are k^th cluster's mean and covariance matrix.

Then, using Bayes' theorem, we can calculate the likelihood of a given example x to belong to the k^th cluster:

$$b_k = p(k|x) = \frac{p(x|k)p(k)}{p(x)} = \frac{f(x|\mu_k, \Sigma_k)p(k)}{\sum_{k=1}^{K} f(x|\mu_k, \Sigma_k)p(k)}$$

In the M step: we update the clusters' parameters as follow:

$$\mu_k = \frac{\sum b_k x}{\sum b_k}$$

$$\Sigma_k = \frac{\sum b_k x x^T}{\sum b_k}$$

$$p(k) = \frac{1}{N}\sum b_k$$

2 Applying GMM on cardiotocogrpahy dataset

I split the dataset into 1500 normal samples for training and leave the rest for testing. This seems reasonable since the test data consists of about equal amount of normal and abnormal samples.

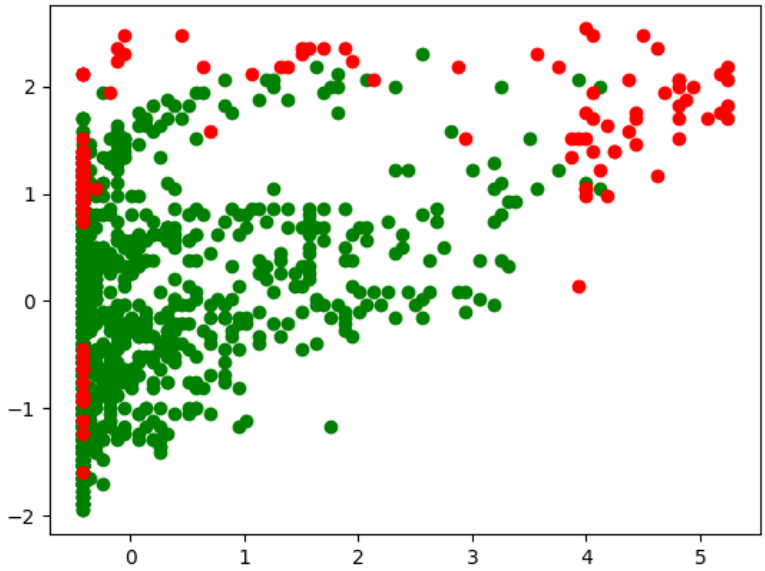

Figure 1: Visualization of the dataset where 2 axises represent 7^{th} and 9^{th} component of data. Red dots mean abnormal samples while green dots mean normal samples.

I choose the number of clusters K = 1 because this dataset is medical related. Usually, a medical measurement is considered normal if the value lies in a defined range and it makes no sense to have multiple discontinuous ranges for a medical measurement to be normal. Thus, 1 cluster should be optimal for the dataset.

However, I had several difficulties fitting the GMM model on the dataset. After running the program, the covarience matrix ended up being singular. After some analyzation, I discovered that 3 components of the dataset are dependent or close enough. Thus I removed one of those from the dataset and try fitting again. Then I faced another program: likelihood of some samples which are far away from the clusters' center are so small that the floating point number can not represent them accurately and end up being zeros, leading to numerical issue.

In order to overcome these difficulties, a few tricks are used. First, to ensure that covarience matrices are invertible, a small value is added to their diagonals. Thus, the formula to calculate them is modified to:

$$\Sigma_k = \frac{\sum b_k x x^T}{\sum b_k} + \epsilon I$$

Second, to solve the numerical instabilities, a small constant is added to all the result of the gaussian distribution's results. The point is to ensure that no sample's likelihood is zero.

In the training process, the E steps and M steps are repeated until the condition for convergence is met or the number of iterations reach a maximum value. The convergence is determined by calculating the difference in the mean of all samples' likelihood; if it is small enough the algorithm is stopped.

In the testing process, a threshold is calculated based on the training data. If the likelihood of a new sample is lower than the threshold, it is classified as anomaly. In the implementation, the threshold value is the smallest value of the training samples' likelihood. This may not be the optimal solution, but it works well enough.

3 Results

The precision and recall values on test dataset is in table 1. It can be seen that the model perform best with 1 cluster. This suggests that my prediction about the data earlier is correct.

Performance of the model with different value of ϵ (that is added to the covarience matrices' diagonals) is in table 2. We can see that the value of ϵ does not affect the results too much, as long as it remains small enough.

k	precision	recall
1	0.84	0.73
2	0.79	0.64
3	0.77	0.59
4	0.78	0.61
5	0.76	0.58

Table 1: Performance of model with different number of clusters

ϵ	precision	recall
1	0.71	0.47
1e-1	0.83	0.72
1e-2	0.86	0.80
1e-3	0.85	0.78
1e-4	0.84	0.77
1e-5	0.85	0.75
1e-6	0.84	0.73

Table 2: Performance with different value of ϵ

Overall, the GMM model performs decently with around 80% precision and recall. However this is still not good enough to be applied in practice. The fact that precision and recall is not even near 100% means that Gaussian mixtures are not the actual distribution of the data. Other kinds of distribution can be explored to find the best fitting model for our dataset.