Anomaly detection using Gaussian Mixture Model

Bùi Quang Long

February 2020

Contents

1 Expectation maximization

1.1 Proof of equation (2)

$$b_k = \frac{f(x \mid \mu_k, \sigma_k^2)\phi_k}{\sum_{k=1}^{K} f(x \mid \mu_k, \sigma_k^2)\phi_k}$$

First, we define:

- $p(x)$: the likelihood of an random observation equals to x.
- ϕ_k: the prior beliefs that an example was drawn from the k_{th} (or k^{th} Gaussian distribution).

- $p(z_k)$: the likelihood of an random observation belongs to the k^{th} cluster. (*prior probability*). Here,
$$p(z_k) = \phi_k$$
- $p(x \mid z_k) = f(x \mid \mu_k, \sigma_k^2)$: the likelihood of an observation x using the estimated parameters of the k^{th} cluster. (*prior probability*)
- $p(z_k \mid x)$ (or b_k in equation (2)): the likelihood of a given observation x belongs to the k_{th} cluster. (*posterior probability*). Thus,
$$p(x) = \sum_{j=1}^{K} p(z_j, x)$$
- $p(z_k, x) = p(x, z_k)$: the 1ikelihood of a random observation satisfies both 2 conditions: equals to x and belongs to the k_{th} cluster. Applying Bayes' Rule,
$$p(z_k, x) = p(x \mid z_k)p(z_k)$$

Then, we obtain:
$$p(z_k \mid x) = \frac{p(z_k, x)}{p(x)}$$
$$p(z_k \mid x) = \frac{p(z_k, x)}{\sum_{j=1}^{K} p(z_j, x)}$$
$$p(z_k \mid x) = \frac{p(x \mid z_k)p(z_k)}{\sum_{j=1}^{K} p(x \mid z_j)p(z_j)}$$
$$p(z_k \mid x) = \frac{f(x \mid \mu_k, \sigma_k^2)\phi_k}{\sum_{j=1}^{K} f(x \mid \mu_j, \sigma_j^2)\phi_j}$$
$$b_k = \frac{f(x \mid \mu_k, \sigma_k^2)\phi_k}{\sum_{j=1}^{K} f(x \mid \mu_j, \sigma_j^2)\phi_j}$$

1.2 Formula for n-D data

1.2.1 E step

- $f(x \mid \mu, \Sigma)$: The likelihood of an observation x using the estimated parameters of the k^{th} cluster. Here,
  - μ is a N-dimensional vector denoting the mean of the distribution.
  - Σ is the NxN symmetric covariance matrix and positive definite.

$$f(x \mid \mu, \Sigma) = \frac{exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu))}{\sqrt{(2\pi)^k \, |\Sigma|}}$$

- b_k: The likelihood of a given observation belongs to the k_{th} cluster.

$$b_k = \frac{f(x \mid \mu_k, \Sigma_k)\phi_k}{\sum_{j=1}^{K} f(x \mid \mu_j, \Sigma_j)\phi_j}$$

  For further calculation, we define b_{nk} equals to the likelihood of the n_{th} observation belongs to the k_{th} cluster.

$$b_{nk} = \frac{f(x_n \mid \mu_k, \Sigma_k)\phi_k}{\sum_{j=1}^{K} f(x_n \mid \mu_j, \Sigma_j)\phi_j}$$

1.2.2 M step

Here we define N is number of observation in our data set. After E step, we recalculate the following parameters:

$$\mu_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} b_{nk} x_n$$

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

$$\phi_k^{new} = \frac{N_k}{N}$$

where

$$N_k = \sum_{n=1}^{N} b_{nk}$$

2 Anomaly detection

2.1 Optimal number of clusters (K)

To define a good number of clusters, we can use the Bayesian information criterion (BIC) or the Akaike information criterion (AIC). Here is my quick comparison between these two criteria:

- AIC is more suitable for unstable data (small, large noise levels, ...).
- AIC tries to select the model that most adequately describes an unknown, high dimensional reality. On the contrary, BIC tries to find the TRUE model among the set of candidates.
- BIC penalizes models more for free parameters than does AIC.

Hence our training data has more than normal 1500 samples, I use the BIC and get the result: The model works best at K around 5 to 20. (I choose 7)

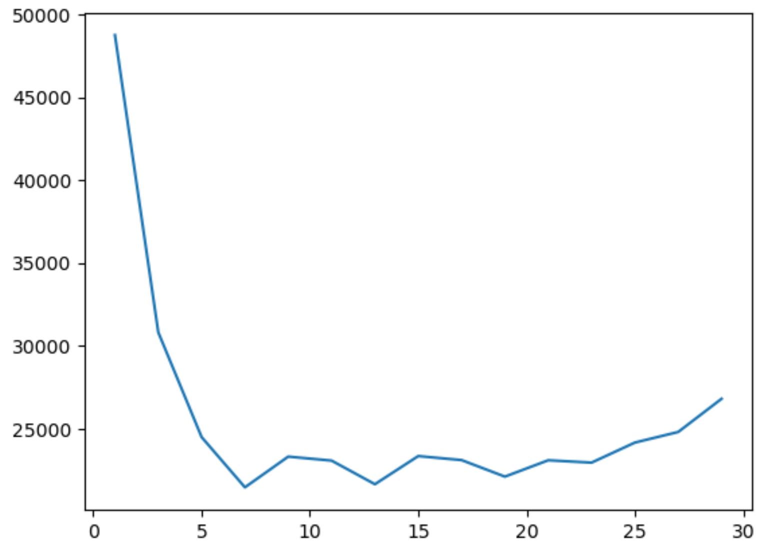


Figure 1: Using BIC to define number of clusters

2.2 The likelihood of an abnormal sample

The likelihood of an sample x:

$$p(x) = \sum_{k=1}^{K} f(x \mid \mu_k, \Sigma_k)\phi_k$$

2.3 Threshold

I come up with a simple threshold formula based on normal samples: find the min likelihood among normal samples.

$$threshold = min(p(x))$$

3 Implementation and Explanation

3.1 Visualization

The data has 21 dimensions for each sample so the first thing I come up with is dimensionality reduction, which led me to Principal component analysis (PCA). With 3 dimensions, I get:

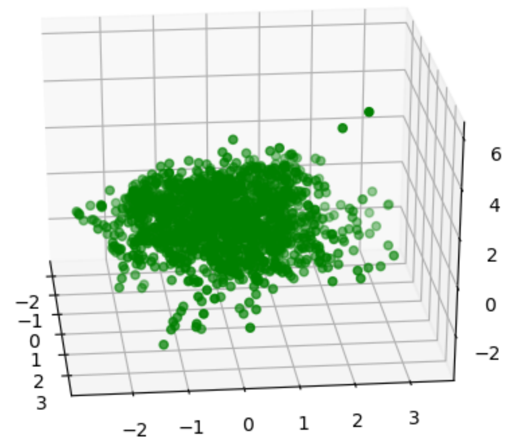


Figure 2: 3D illustration

It looks like lots of spatial information can't be illustrated. Thus I haven't recognized the separation clearly. More detail will be illustrated in Result Section.

3.2 GMM Implementation

3.2.1 Data preparation

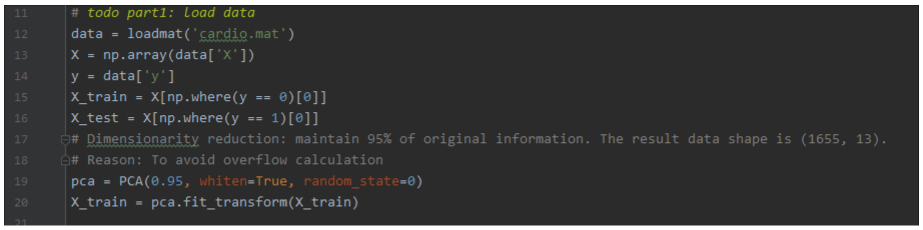
– After several attempts, I was stuck with overflow error. The covariance matrices became too small and overflow the float type. Thus I use PCA to reduce the numbers of features to 13 and still maintain 95% of original information.

– From the data set of 1831 samples, I also split the data into 2 parts: normal and suspect part. Now my samples for training has the shape of (1655, 13)

3.2.2 Model building

– I apply the Expectation-Maximization algorithm. The goal is to maximize the Log-Likelihood funtion:

$$\ln p(X \mid \mu, \Sigma, \phi) = \sum_{n=1}^{N} \ln (\sum_{k=1}^{k} \phi_k f(x_n \mid \mu_k, \Sigma_k)$$



```python
# todo part1: load data
data = loadmat('cardio.mat')
X = np.array(data['X'])
y = data['y']
X_train = X[np.where(y == 0)[0]]
X_test = X[np.where(y == 1)[0]]
# Dimensionarity reduction: maintain 95% of original information. The result data shape is (1655, 13).
# Reason: To avoid overflow calculation
pca = PCA(0.95, whiten=True, random_state=0)
X_train = pca.fit_transform(X_train)
```

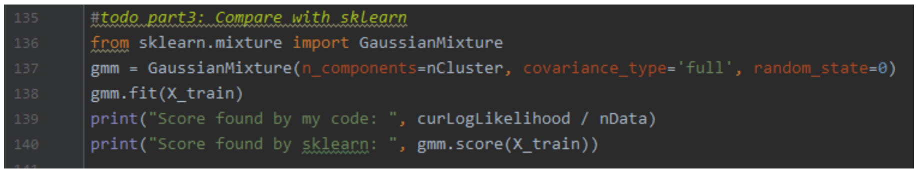
Figure 3: Data preparation

- I set the convergence criterion to 0.1, the loop will stop when the difference between the Log-Likelihood of this loop and previous loop less than 0.1:

$$curLogLikelihood - preLogLikelihood < 0.1$$

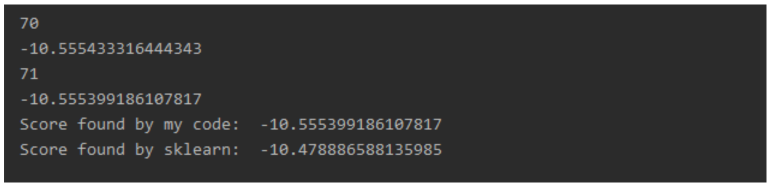
3.2.3 Result and Comparison with sklearn

After 71 iterations, my GMM converged and release the score approximately the score found by sklearn.



```python
#todo part3: Compare with sklearn
from sklearn.mixture import GaussianMixture
gmm = GaussianMixture(n_components=nCluster, covariance_type='full', random_state=0)
gmm.fit(X_train)
print("Score found by my code: ", curLogLikelihood / nData)
print("Score found by sklearn: ", gmm.score(X_train))
```

Figure 4: GMM using sklearn



```
70
-10.555433316444343
71
-10.555399186107817
Score found by my code:  -10.555399186107817
Score found by sklearn:  -10.478886588135985
```

Figure 5: Compare

3.2.4 Anomaly detection

With my threshold formula, I detected 30 anomlies, which will be marked as black points in 3D illustration.

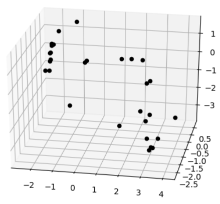


Figure 6: Anomalies distribution

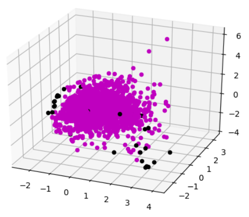


Figure 7: Anomalies and normal samples