

E13 EM Algorithm (C++/Python)

17341203 Yixin Zhang

December 8, 2019

Contents

1	Chinese Football Dataset	2
2	EM	3
2.1	The Gaussian Distribution	3
2.2	Mixtures of Gaussians	3
2.2.1	Introduction	3
2.2.2	About Latent Variables	5
2.3	EM for Gaussian Mixtures	6
2.4	EM Algorithm	8
3	Tasks	8
4	Codes and Results	9
4.1	Code	9
4.2	Results	12

1 Chinese Football Dataset

The following Chinese Football Dataset has recored the performance of 16 AFC football teams between 2005 and 2018.

	Country	2006WorldCup	2010WorldCup	2014WorldCup	2018WorldCup	2007AsianCup	2011AsianCup	2015AsianCup
1	China	50	50	50	40	9	9	5
2	Japan	28	9	29	15	4	1	5
3	South_Korea	17	15	27	19	3	3	2
4	Iran	25	40	28	18	5	5	5
5	Saudi_Arabia	28	40	50	26	2	9	9
6	Iraq	50	50	40	40	1	5	4
7	Qatar	50	40	40	40	9	5	9
8	United_Arab_Emirates	50	40	50	40	9	9	3
9	Uzbekistan	40	40	40	40	5	4	9
10	Thailand	50	50	50	40	9	17	17
11	Vietnam	50	50	50	50	5	17	17
12	Oman	50	50	40	50	9	17	9
13	Bahrain	40	40	50	50	9	9	9
14	North_Korea	40	32	50	50	17	9	9
15	Indonesia	50	50	50	50	9	17	17
16	Australia	16	21	30	30	9	2	1
17								

The scoring rules are below:

- For the FIFA World Cup, teams score the same with their rankings if they enter the World Cup; teams score 50 for failing to entering the Asia Top Ten; teams score 40 for entering the Asia Top Ten but not entering the World Cup.
- For the AFC Asian Cup, teams score the same with their rankings if they finally enter the top four; teams score 5 for entering the top eight but not the top four, and 9 for entering the top sixteen but not top eight; teams score 17 for not passing the group stages.

We aim at classifying the above 16 teams into 3 classes according to their performance: the first-class, the second-class and the third-class. In our opinion, teams of Australia, Iran, South Korea and Japan belong to the first-class, while the Chinese football team belongs to the third-class.

2 EM

2.1 The Gaussian Distribution

The Gaussian, also known as the normal distribution, is a widely used model for the distribution of continuous variables. In the case of a single variable x , the Gaussian distribution can be written in the form

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x - \mu)^2\right\} \quad (2.1.1)$$

where μ is the mean and σ^2 is the variance.

For a D -dimensional vector \mathbf{x} , the multivariate Gaussian distribution takes the form

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right\} \quad (2.1.2)$$

where $\boldsymbol{\mu}$ is a D -dimensional mean vector, $\boldsymbol{\Sigma}$ is a $D \times D$ covariance matrix, and $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$.

2.2 Mixtures of Gaussians

2.2.1 Introduction

While the Gaussian distribution has some important analytical properties, it suffers from significant limitations when it comes to modelling real data sets. Consider the example shown in Figure 1. This is known as the ‘Old Faithful’ data set, and comprises 272 measurements of the eruption of the Old Faithful geyser at Yellowstone National Park in the USA. Each measurement comprises the duration of the eruption in minutes (horizontal axis) and the time in minutes to the next eruption (vertical axis). We see that the data set forms two dominant clumps, and that a simple Gaussian distribution is unable to capture this structure, whereas a linear superposition of two Gaussians gives a better characterization of the data set.

Such superpositions, formed by taking linear combinations of more basic distributions such as Gaussians, can be formulated as probabilistic models known as *mixture distributions*. In Figure 1 we see that a linear combination of Gaussians can give rise to very complex densities. By using a sufficient number of Gaussians, and by adjusting their means and covariances as well as the coefficients in the linear combination, almost any continuous density can be approximated to arbitrary accuracy.

We therefore consider a superposition of K Gaussian densities of the form

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (2.2.1)$$

Example of a Gaussian mixture distribution in one dimension showing three Gaussians (each scaled by a coefficient) in blue and their sum in red.

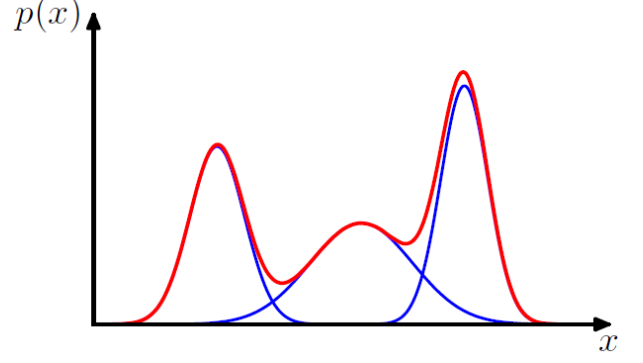


Figure 1: Example of a Gaussian mixture distribution

which is called a mixture of Gaussians. Each Gaussian density $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is called a component of the mixture and has its own mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$.

The parameters π_k in (2.2.1) are called *mixing coefficients*. If we integrate both sides of (2.2.1) with respect to \mathbf{x} , and note that both $p(\mathbf{x})$ and the individual Gaussian components are normalized, we obtain

$$\sum_{k=1}^K \pi_k = 1. \quad (2.2.2)$$

Also, the requirement that $p(\mathbf{x}) \geq 0$, together with $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \geq 0$, implies $\pi_k \geq 0$ for all k . Combining this with condition (2.2.2) we obtain

$$0 \leq \pi_k \leq 1. \quad (2.2.3)$$

We therefore see that the mixing coefficients satisfy the requirements to be probabilities.

From the sum and product rules, the marginal density is given by

$$p(\mathbf{x}) = \sum_{k=1}^K p(k)p(\mathbf{x}|k) \quad (2.2.4)$$

which is equivalent to (2.2.1) in which we can view $\pi_k = p(k)$ as the prior probability of picking the k^{th} component, and the density $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = p(\mathbf{x}|k)$ as the probability of \mathbf{x} conditioned on k . From Bayes' theorem these are given by

$$\gamma_k(\mathbf{x}) = p(k|\mathbf{x}) = \frac{p(k)p(\mathbf{x}|k)}{\sum_l p(l)p(\mathbf{x}|l)} = \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_l \pi_l \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}. \quad (2.2.5)$$

The form of the Gaussian mixture distribution is governed by the parameters $\boldsymbol{\pi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, where we have used the notation $\boldsymbol{\pi} = \{\pi_1, \dots, \pi_K\}$, $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K\}$ and $\boldsymbol{\Sigma} = \{\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_K\}$. One way to set the values of there parameters is to use maximum likelihood. From (2.2.1) the log of the likelihood

function is given by

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \quad (2.2.6)$$

where $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$. One approach to maximizing the likelihood function is to use iterative numerical optimization techniques. Alternatively we can employ a powerful framework called expectation maximization (EM).

2.2.2 About Latent Variables

We now turn to a formulation of Gaussian mixtures in terms of discrete *latent* variables. This will provide us with a deeper insight into this important distribution, and will also serve to motivate the expectation-maximization (EM) algorithm.

Recall from (2.2.1) that the Gaussian mixture distribution can be written as a linear superposition of Gaussians in the form

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (2.2.7)$$

Let us introduce a K -dimensional binary random variable \mathbf{z} having a 1-of- K representation in which a particular element z_k is equal to 1 and all other elements are equal to 0. The values of z_k therefore satisfy $z_k \in \{0, 1\}$ and $\sum_k z_k = 1$, and we see that there are K possible states for the vector \mathbf{z} according to which element is nonzero. We shall define the joint distribution $p(\mathbf{x}, \mathbf{z})$ in terms of a marginal distribution $p(\mathbf{z})$ and a conditional distribution $p(\mathbf{x}|\mathbf{z})$. The marginal distribution over \mathbf{z} is specified in terms of the mixing coefficients π_k , such that

$$p(z_k = 1) = \pi_k \quad (2.2.8)$$

where the parameters $\{\pi_k\}$ must satisfy

$$0 \leq \pi_k \leq 1 \quad (2.2.9)$$

together with

$$\sum_{k=1}^K \pi_k = 1 \quad (2.2.10)$$

in order to be valid probabilities. Because \mathbf{z} uses a 1-of- K representation, we can also write this distribution in the form

$$p(\mathbf{z}) = \prod_{k=1}^K \pi_k^{z_k}. \quad (2.2.11)$$

Similarly, the conditional distribution of \mathbf{x} given a particular value for \mathbf{z} is a Gaussian

$$p(\mathbf{x} | z_k = 1) = \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (2.2.12)$$

which can also be written in the form

$$p(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^K p(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k}. \quad (2.2.13)$$

The joint distribution is given by $p(\mathbf{z})p(\mathbf{x}|\mathbf{z})$, and the marginal distribution of \mathbf{x} is then obtained by summing the joint distribution over all possible states of \mathbf{z} to give

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (2.2.14)$$

where we have made use of (2.2.12) and (2.2.13). Thus the marginal distribution of \mathbf{x} is a Gaussian mixture of the form (2.2.7). If we have several observations $\mathbf{x}_1, \dots, \mathbf{x}_N$, then, because we have represented the marginal distribution in the form $p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z})$, it follows that for every observed data point \mathbf{x}_n there is a corresponding latent variable \mathbf{z}_n .

We have therefore found an equivalent formulation of the Gaussian mixture involving an explicit latent variable. It might seem that we have not gained much by doing so. However, we are now able to work with the joint distribution $p(\mathbf{x}, \mathbf{z})$ instead of the marginal distribution $p(\mathbf{x})$, and this will lead to significant simplifications, most notably through the introduction of the expectation-maximization (EM) algorithm.

Another quantity that will play an important role is the conditional probability of \mathbf{z} given \mathbf{x} . We shall use $\gamma(z_k)$ to denote $p(z_k = 1|\mathbf{x})$, whose value can be found using Bayes' theorem

$$\gamma(z_k) = p(z_k = 1|\mathbf{x}) = \frac{p(z_k = 1)p(\mathbf{x}|z_k = 1)}{\sum_{j=1}^K p(z_j = 1)p(\mathbf{x}|z_j = 1)} = \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \quad (2.2.15)$$

We shall view π_k as the prior probability of $z_k = 1$, and the quantity $\gamma(z_k)$ as the corresponding posterior probability once we have observed \mathbf{x} . As we shall see later, $\gamma(z_k)$ can also be viewed as the responsibility that component k takes for 'explaining' the observation \mathbf{x} .

2.3 EM for Gaussian Mixtures

Initially, we shall motivate the EM algorithm by giving a relatively informal treatment in the context of the Gaussian mixture model.

Let us begin by writing down the conditions that must be satisfied at a maximum of the likelihood function. Setting the derivatives of $\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with respect to the means $\boldsymbol{\mu}_k$ of the Gaussian components to zero, we obtain

$$0 = - \sum_{n=1}^n \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}_{\gamma(z_{nk})} \sum_k (\mathbf{x}_n - \boldsymbol{\mu}_k) \quad (2.3.1)$$

Multiplying by Σ_k^{-1} (which we assume to be nonsingular) and rearranging we obtain

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n \quad (2.3.2)$$

where we have defined

$$N_k = \sum_{n=1}^N \gamma(z_{nk}). \quad (2.3.3)$$

We can interpret N_k as the effective number of points assigned to cluster k . Note carefully the form of this solution. We see that the mean μ_k for the k^{th} Gaussian component is obtained by taking a weighted mean of all of the points in the data set, in which the weighting factor for data point \mathbf{x}_n is given by the posterior probability $\gamma(z_{nk})$ that component k was responsible for generating \mathbf{x}_n .

If we set the derivative of $\ln(\mathbf{X}|\pi, \mu, \Sigma)$ with respect to Σ_k to zero, and follow a similar line of reasoning, making use of the result for the maximum likelihood for the covariance matrix of a single Gaussian, we obtain

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^T \quad (2.3.4)$$

which has the same form as the corresponding result for a single Gaussian fitted to the data set, but again with each data point weighted by the corresponding posterior probability and with the denominator given by the effective number of points associated with the corresponding component.

Finally, we maximize $\ln p(\mathbf{X}|\pi, \mu, \Sigma)$ with respect to the mixing coefficients π_k . Here we must take account of the constraint $\sum_{k=1}^K \pi_k = 1$. This can be achieved using a Lagrange multiplier and maximizing the following quantity

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) + \lambda \left(\sum_{k=1}^K \pi_k - 1 \right) \quad (2.3.5)$$

which gives

$$0 = \sum_{n=1}^N \frac{\mathcal{N}(\mathbf{x}_n | \mu_k, \Sigma_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n | \mu_j, \Sigma_j)} \quad (2.3.6)$$

where again we see the appearance of the responsibilities. If we now multiply both sides by π_k and sum over k making use of the constraint $\sum_{k=1}^K \pi_k = 1$, we find $\lambda = -N$. Using this to eliminate λ and rearranging we obtain

$$\pi_k = \frac{N_k}{N} \quad (2.3.7)$$

so that the mixing coefficient for the k^{th} component is given by the average responsibility which that component takes for explaining the data points.

2.4 EM Algorithm

Given a Gaussian mixture model, the goal is to maximize the likelihood function with respect to the parameters (comprising the means and covariances of the components and the mixing coefficients).

1. Initialize the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k , and evaluate the initial value of the log likelihood.
2. **E step.** Evaluate the responsibilities using the current parameter values

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \quad (2.4.1)$$

3. **M step.** Re-estimate the parameters using the current responsibilities

$$\boldsymbol{\mu}_k^{new} = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) \mathbf{x}_n \quad (2.4.2)$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk}) (\mathbf{x}_n - \boldsymbol{\mu}_k^{new})(\mathbf{x}_n - \boldsymbol{\mu}_k^{new})^T \quad (2.4.3)$$

$$\pi_k^{new} = \frac{N_k}{N} \quad (2.4.4)$$

where

$$N_k = \sum_{n=1}^N \gamma(z_{nk}). \quad (2.4.5)$$

4. Evaluate the log likelihood

$$\ln p(\mathbf{X} | \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \sum_{n=1}^N \ln \left\{ \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \quad (2.4.6)$$

and check for convergence of either the parameters or the log likelihood. If the convergence criterion is not satisfied return to step 2.

3 Tasks

- Assume that score vectors of teams in the same class are normally distributed, we can thus adopt the Gaussian mixture model. Please classify the teams into 3 classes by using EM algorithm. If necessary, you can refer to page 430-439 in the book [Pattern Recognition and Machine Learning.pdf](#) and the website https://blog.csdn.net/jinping_shi/article/details/59613054 which is a Chinese translation.
- You should show the values of these parameters: $\boldsymbol{\gamma}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. If necessary, you can plot the clustering results. [Note that \$\boldsymbol{\gamma}\$ is essential for classifying.](#)
- Please submit a file named [E13_YourNumber.pdf](#) and send it to ai_201901@foxmail.com

4 Codes and Results

4.1 Code

```
1  #!/usr/bin/env python
2  # coding: utf-8
3
4  # In[1]:
5
6
7  # coding: utf-8
8  import numpy as np
9  import matplotlib.pyplot as plt
10
11
12  # In[2]:
13
14
15  def loadData(filename):
16      """从文件中读取数据。"""
17      dataSet = []
18      id2country = [] # 将索引对应到国家名
19      with open(filename) as fr:
20          for i, line in enumerate(fr.readlines()):
21              curLine = line.strip().split(' ')
22              fltLine = list(map(int, curLine[1:])) # 去掉第一列国家名
23              dataSet.append(fltLine)
24              id2country.append(curLine[0])
25      return dataSet, id2country
26
27
28  # In[3]:
29
30
31  def prob(x, mu, sigma):
32      """高斯分布的概率密度函数。"""
33      n = np.shape(x)[1]
34      exp0n = float(-0.5 * (x - mu) * (sigma.I) * ((x - mu).T))
```

```

35     divBy = pow(2 * np.pi, n / 2) * pow(np.linalg.det(sigma), 0.5)
36     return pow(np.e, exp0n) / divBy
37
38
39 # In[4]:
40
41
42 def EM(dataMat, maxIter=50):
43     m, n = np.shape(dataMat)
44     # 1.初始化各高斯混合成分参数
45     alpha = [1/3, 1/3, 1/3] # 初始化 alpha
46     mu = [dataMat[1, :], dataMat[13, :], dataMat[11, :]] # 初始化mu
47     sigma = [np.mat((np.eye(7, dtype=float))) for x in range(3)] # 初始化协方差矩阵
48     gamma = np.mat(np.zeros((m, 3)))
49
50     for i in range(maxIter):
51         for j in range(m):
52             sumAlphaMulP = 0
53             for k in range(3):
54                 gamma[j, k] = alpha[k] * prob(dataMat[j, :], mu[k], sigma[k]) #
55                                     4.计算混合成分生成的后验概率, 即gamma
56                 sumAlphaMulP += gamma[j, k]
57             for k in range(3):
58                 gamma[j, k] /= sumAlphaMulP
59             sumGamma = np.sum(gamma, axis=0)
60
61             for k in range(3):
62                 mu[k] = np.mat(np.zeros((1, n)))
63                 sigma[k] = np.mat(np.zeros((n, n)))
64                 for j in range(m):
65                     mu[k] += gamma[j, k] * dataMat[j, :]
66                 mu[k] /= sumGamma[0, k] # 7.计算新均值向量
67                 for j in range(m):
68                     sigma[k] += gamma[j, k] * (dataMat[j, :] - mu[k]).T * (dataMat[j, :] - mu[k])
69                 sigma[k] /= sumGamma[0, k] # 8. 计算新的协方差矩阵
70                 alpha[k] = sumGamma[0, k] / m # 9. 计算新混合系数

```

```

71         for s in sigma:
72             s += np.eye(7)
73
74     print('gamma')
75     [print(g) for g in gamma]
76     print('\nmu')
77     [print(m) for m in mu]
78     print('\nsigma')
79     [print(s) for s in sigma]
80
81     return gamma
82
83
84 # In[5]:
85
86
87 def initCentroids(dataMat, k):
88     """Init centroids with random samples."""
89     numSamples, dim = dataMat.shape
90     centroids = np.zeros((k, dim))
91     for i in range(k):
92         index = int(np.random.uniform(0, numSamples))
93         centroids[i, :] = dataMat[index, :]
94     return centroids
95
96
97 # In[6]:
98
99
100 def gaussianCluster(dataMat):
101     """进行聚类。"""
102     m, n = np.shape(dataMat)
103     centroids = initCentroids(dataMat, m) ## step 1: init centroids
104     clusterAssign = np.mat(np.zeros((m, 2)))
105     gamma = EM(dataMat)
106     for i in range(m):
107         # amx返回矩阵最大值, argmax返回矩阵最大值所在下标

```

```

108         clusterAssign[i, :] = np.argmax(gamma[i, :]), np.amax(gamma[i, :]) #
           15.确定x的簇标记lambda
109         ## step 4: update centroids
110         for j in range(m):
111             pointsInCluster = dataMat[np.nonzero(clusterAssign[:, 0].A == j)[0]]
112             centroids[j, :] = np.mean(pointsInCluster, axis=0) # 计算出均值向量
113         return centroids, clusterAssign
114
115
116 # In[7]:
117
118
119 dataMat, id2country = loadData('football.txt')
120 dataMat = np.mat(dataMat)
121 centroids, clusterAssign = gaussianCluster(dataMat)
122
123
124 # In[8]:
125
126
127 result = ([], [], [])
128 for i, assign in enumerate(clusterAssign):
129     result[int(assign[0, 0])].append(id2country[i])
130 print('\n-----\n')
131 print('First-class:', result[0])
132 print('Second-class:', result[1])
133 print('Third-class:', result[2])
134
135
136 # In[ ]:

```

4.2 Results

```

1 gamma
2 [[1.94167338e-114 1.25382308e-012 1.00000000e+000]]
3 [[1.00000000e+000 1.30157637e-090 6.99254748e-192]]
4 [[1.00000000e+000 1.77308433e-091 3.22118568e-308]]

```

```

5 [[1.00000000e+000 2.86831156e-033 3.89329040e-153]]
6 [[4.68941135e-103 1.00000000e+000 9.29410695e-122]]
7 [[1.73360486e-117 2.63382865e-004 9.99736617e-001]]
8 [[8.07501018e-70 1.52126772e-04 9.99847873e-01]]
9 [[1.73742960e-121 9.99999999e-001 9.91726551e-010]]
10 [[6.80507222e-64 1.00000000e+00 3.09330574e-23]]
11 [[2.99372863e-167 4.28991995e-061 1.00000000e+000]]
12 [[3.25570891e-236 7.31233936e-046 1.00000000e+000]]
13 [[1.25721563e-166 9.78317131e-042 1.00000000e+000]]
14 [[5.72371065e-127 1.00000000e+000 6.74033915e-028]]
15 [[1.13539296e-97 1.00000000e+00 3.40151214e-35]]
16 [[6.94426982e-210 5.86062654e-050 1.00000000e+000]]
17 [[1.00000000e+000 2.57084628e-074 2.29152462e-282]]
18
19 mu
20 [[21.5 21.25 28.5 20.5 5.25 2.75 3.25]]
21 [[39.60086419 38.40065967 47.99933524 41.19990029 8.39962848 7.99975071
22 7.79983635]]
23 [[50. 48.5715611 45.71462493 44.28596869 7.28591355 12.4290124
24 11.14317249]]
25
26 sigma
27 [[ 2.725000e+01 3.875000e+00 0.000000e+00 -2.250000e+01 -4.875000e+00
28 -1.250000e-01 8.875000e+00]
29 [ 3.875000e+00 1.361875e+02 -1.625000e+00 6.875000e+00 5.937500e+00
30 1.556250e+01 4.937500e+00]
31 [ 0.000000e+00 -1.625000e+00 2.250000e+00 3.750000e+00 2.125000e+00
32 -8.750000e-01 -3.750000e-01]
33 [-2.250000e+01 6.875000e+00 3.750000e+00 3.325000e+01 1.162500e+01
34 -8.750000e-01 -8.375000e+00]
35 [-4.875000e+00 5.937500e+00 2.125000e+00 1.162500e+01 6.187500e+00
36 -4.375000e-01 -2.062500e+00]
37 [-1.250000e-01 1.556250e+01 -8.750000e-01 -8.750000e-01 -4.375000e-01
38 3.187500e+00 5.625000e-01]
39 [ 8.875000e+00 4.937500e+00 -3.750000e-01 -8.375000e+00 -2.062500e+00
40 5.625000e-01 4.187500e+00]]
41 [[ 49.64494506 -0.63308678 -0.80684646 34.07613119 16.55476046

```

```

42     -0.40255911 -12.48066475]
43 [ -0.63308678 11.24631413 -3.20501105 -14.07962157 -13.76334855
44     -1.60184591 -1.92210373]
45 [ -0.80684646 -3.20501105 17.00398812 2.40059822 6.80240687
46     8.00132936 -2.3984915 ]
47 [ 34.07613119 -14.07962157 2.40059822 78.75365819 36.31742778
48     1.2001994 1.4400767 ]
49 [ 16.55476046 -13.76334855 6.80240687 36.31742778 26.44078121
50     3.40083195 -0.71843719]
51 [ -0.40255911 -1.60184591 8.00132936 1.2001994 3.40083195
52     5.00041541 -1.19940939]
53 [-12.48066475 -1.92210373 -2.3984915 1.4400767 -0.71843719
54     -1.19940939 6.76032574]]
55 [[ 1.00000000e+00 3.78808699e-22 2.52540627e-22 1.89403069e-22
56     1.01019058e-22 3.72504158e-22 9.47146495e-23]
57 [ 3.78808699e-22 1.32439513e+01 8.16299251e+00 6.12224439e+00
58     -2.44846776e+00 1.06118903e+01 3.06139095e+00]
59 [ 2.52540627e-22 8.16299251e+00 2.54893112e+01 4.08042109e+00
60     4.08073623e+00 1.46922298e+01 1.63256977e+01]
61 [ 1.89403069e-22 6.12224439e+00 4.08042109e+00 2.54901593e+01
62     1.63189594e+00 1.95911097e+01 1.36729295e+01]
63 [ 1.01019058e-22 -2.44846776e+00 4.08073623e+00 1.63189594e+00
64     9.48874928e+00 5.87641959e+00 4.81500277e+00]
65 [ 3.72504158e-22 1.06118903e+01 1.46922298e+01 1.95911097e+01
66     5.87641959e+00 3.03862236e+01 2.29377945e+01]
67 [ 9.47146495e-23 3.06139095e+00 1.63256977e+01 1.36729295e+01
68     4.81500277e+00 2.29377945e+01 2.96935612e+01]]
69
70 -----
71
72 First-class: ['Japan', 'South_Korea', 'Iran', 'Australia']
73 Second-class: ['Saudi_Arabia', 'United_Arab_Emirates', 'Uzbekistan', 'Bahrain',
74     'North_Korea']
75 Third-class: ['China', 'Iraq', 'Qatar', 'Thailand', 'Vietnam', 'Oman', 'Indonesia']

```