第四次作业

1 1. EM算法理论练习

1.0.1 1.1 推导三硬币模型的EM算法中隐变量后验分布的计算公式以及参数更新公式

三硬币模型:

假设有三枚硬币，分别记为A、B、C，这些硬币正面向上的概率分别是π, p, q。

进行如下掷硬币试验：先掷硬币 A ，根据其结果选择硬币 B 或硬币 C ，正面选硬币 B ，反面选硬币 C ；然后掷选出的硬币，出现正面记为1，出现反面记为0；独立地重复 n 次试验。

假设只能观测到掷硬币的结果，不能观测到掷硬币的过程。问如何估计三硬币正面出现的概率，即三硬币模型的参数 $\theta = (\pi, p, q)$。

模型构建:

引入随机变量 $x \in \{0, 1\}$ 表示一次试验观测的结果是 1 或者 0 ， x 是观测变量，可以观测。

引入随机变量 $z \in \{0, 1\}$ 表示未观测到的掷硬币 A 的结果，z 是隐变量，不可观测。其中$z = 1$表示硬币A为正面，$z = 0$表示硬币A为反面。

写出似然函数的对数$LL(\theta)$，并引入隐变量：

$$\begin{aligned}
LL(\theta) &= \sum_{i=1}^{n} log\left(P(x_i \mid \theta)\right) \\
&= \sum_{i=1}^{n} log\left(P(x_i, z_i = 1 \mid \theta) + P(x_i, z_i = 0 \mid \theta)\right) \\
&= \sum_{i=1}^{n} log\left(Q(z_i = 1)\frac{P(x_i, z_i = 1 \mid \theta)}{Q(z_i = 1)} + Q(z_i = 0)\frac{P(x_i, z_i = 0 \mid \theta)}{Q(z_i = 0)}\right) \\
&\geq \sum_{i=1}^{n} \left[Q(z_i = 1) log\left(\frac{P(x_i, z_i = 1 \mid \theta)}{Q(z_i = 1)}\right) + Q(z_i = 0) log\left(\frac{P(x_i, z_i = 0 \mid \theta)}{Q(z_i = 0)}\right)\right]
\end{aligned} \tag{17}$$

E（expectation）步:

由EM算法知，隐变量的后验分布计算公式为：

$$\begin{aligned}
Q(z_i = 1) &= P(z_i = 1 \mid x_i, \theta^{(t)}) \\
&= \frac{P(z_i = 1 \mid \theta^{(t)}) P(x_i \mid z_i = 1, \theta^{(t)})}{\sum_{z_i=0}^{z_i=1} P(z_i \mid \theta^{(t)}) P(x_i \mid z_i, \theta^{(t)})} \\
&= \frac{\pi^{(t)} [p^{(t)}]^{x_i} [1 - p^{(t)}]^{1-x_i}}{\pi^{(t)} [p^{(t)}]^{x_i} [1 - p^{(t)}]^{1-x_i} + [1 - \pi^{(t)}][q^{(t)}]^{x_i} [1 - q^{(t)}]^{1-x_i}} \\
&\triangleq \mu_i^{(t)} \\
Q(z_i = 0) &= P(z_i = 0 \mid x_i, \theta^{(t)}) \\
&= \frac{P(z_i = 0 \mid \theta^{(t)}) P(x_i \mid z_i = 0, \theta^{(t)})}{\sum_{z_i=0}^{z_i=1} P(z_i \mid \theta^{(t)}) P(x_i \mid z_i, \theta^{(t)})} \\
&= 1 - \mu_i^{(t)}
\end{aligned} \tag{18}$$

M（maximize）步:

由于$Q^{(t)}(z) log Q^{(t)}(z)$是常数，所以参数 θ 更新公式为：

$$\begin{aligned}
\theta^{(t+1)} &= \arg\max_{\theta} \sum_{i=1}^{n} \left[Q(z_i = 1) log\left(\frac{P(x_i, z_i = 1 \mid \theta)}{Q(z_i = 1)}\right) + Q(z_i = 0) log\left(\frac{P(x_i, z_i = 0 \mid \theta)}{Q(z_i = 0)}\right)\right] \\
&= \arg\max_{\theta} \sum_{i=1}^{n} \sum_{z_i} Q(z_i) log\left(P(x_i, z_i \mid \theta)\right) \\
&= \arg\max_{\theta} \sum_{i=1}^{n} \left[\mu_i^{(t)} log(\pi p^{x_i} (1 - p)^{1-x_i}) + (1 - \mu_i^{(t)}) log((1 - \pi) q^{x_i} (1 - q)^{1-x_i})\right] \\
&= \arg\max_{\theta} \sum_{i=1}^{n} \left\{\mu_i^{(t)} \left[log(\pi) + x_i log(p) + (1 - x_i) log(1 - p)\right] + (1 - \mu_i^{(t)}) \left[log(1 - \pi) + x_i log(q) + (1 - x_i) log(1 - q)\right]\right\} \\
&\triangleq \arg\max_{\pi, p, q} f(\pi, p, q)
\end{aligned} \tag{19}$$

将f(x)分别对$\theta = (\pi, p, q)$求偏导，当$\sum_{i=1}^{n} \mu_i^{(t)} \neq 0$且$\sum_{i=1}^{n} (1 - \mu_i^{(t)}) \neq 0$时：

$$\begin{cases}
\dfrac{\partial f(\pi, p, q)}{\partial \pi} = \sum_{i=1}^{n} \left(\dfrac{\mu_i^{(t)}}{\pi} - \dfrac{1 - \mu_i^{(t)}}{1 - \pi}\right) \equiv 0 & \Longrightarrow \pi(\sum_{i=1}^{n} (\mu_i^{(t)} - 1)) = (\pi - 1)(\sum_{i=1}^{n} \mu_i^{(t)}) & \Longrightarrow \pi = \dfrac{1}{n} \\
\dfrac{\partial f(\pi, p, q)}{\partial p} = \sum_{i=1}^{n} \left(\mu_i^{(t)} \dfrac{x_i}{p} - \mu_i^{(t)} \dfrac{x_i - 1}{p - 1}\right) \equiv 0 & \Longrightarrow (p - 1) \sum_{i=1}^{n} x_i \mu_i^{(t)} = p \sum_{i=1}^{n} (x_i - 1) \mu_i^{(t)} & \Longrightarrow p = \sum \\
\dfrac{\partial f(\pi, p, q)}{\partial q} = \sum_{i=1}^{n} \left((1 - \mu_i^{(t)}) \dfrac{x_i}{q} - (1 - \mu_i^{(t)}) \dfrac{x_i - 1}{q - 1}\right) \equiv 0 & \Longrightarrow (q - 1) \sum_{i=1}^{n} x_i (1 - \mu_i^{(t)}) = q \sum_{i=1}^{n} (x_i - 1)(1 - \mu_i^{(t)}) & \Longrightarrow q = \sum
\end{cases}$$

当$\sum_{i=1}^{n} \mu_i^{(t)} = 0$ 时，即$\pi = 0$ 时，参数 p 的取值对结论无影响

当$\sum_{i=1}^{n} (1 - \mu_i^{(t)}) = 0$ 时，即$\pi = 1$ 时，参数 q 的取值对结论无影响

1.0.1 1.2 假设硬币A、B、C正面向上的概率分别是0.7、0.3、0.6，按照三硬币模型的数据生成过程独立地重复100次试验并记录观测结果的序列

```python
import numpy as np
def three_coins_model(pi,p,q): #分别为ABC硬币正面向上概率
    z = np.random.choice([0,1],p=[1-pi,pi]) #投硬币A，其中z=1表示正面，z=0表示反面
    if z == 1:
        x = np.random.choice([0,1],p=[1-p,p]) #投硬币B，其中x=1表示正面，x=0表示反面
    elif z == 0:
        x = np.random.choice([0,1],p=[1-q,q]) #投硬币C，其中x=1表示正面，x=0表示反面
    else:
        return "error: something went wrong with z"
    return x
x_list = []
for i in range(0,100):
    x_list.append(three_coins_model(0.7,0.3,0.6))
print("100次独立重复试验结果：\n",x_list)
```

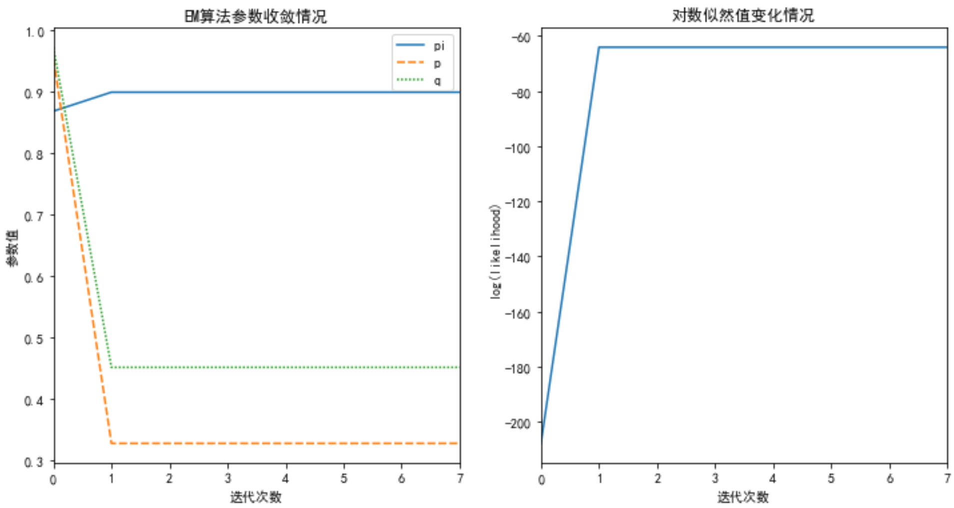
```
100次独立重复试验结果：
 [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1,
0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1]
```

1.0.1 1.3 利用EM算法根据上述观测序列估计各硬币正面向上的概率

```python
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import numpy as np
plt.rcParams['font.sans-serif'] = ['SimHei']  # 显示汉字
plt.rcParams['axes.unicode_minus']=False


def get_loglikelihood(theta_list, x): #获得指定参数下似然函数值
    [pi, p, q] = theta_list
    x = np.array(x)
    likelihood = (pi*(p**x)*((1-p)**(1-x)))+((1-pi)*(q**x)*((1-q)**(1-x)))
    loglikelihood = np.log(likelihood).sum()
    # for x_i in x:
    #     loglikelihood += np.log(pi*(p**x_i)*((1-p)**(1-x_i))+(1-pi)*(q**x_i)*((1-q)**(1-x_i)))
    return loglikelihood


# x表示结果序列; t表示循环迭代次数;initial_theta_list表示起始参数,初始值都为0.5
def EM_algorithm(x, t, initial_theta_list=[0.5, 0.5, 0.5]):
    [pi, p, q] = initial_theta_list
    loglikelihood = get_loglikelihood([pi, p, q], x)
    result_list = [[pi, p, q, loglikelihood]]
    for i in range(t):
        mu = []
        for x_i in x:
            mu.append((pi*p**x_i*(1-p)**(1-x_i))/(pi*(p**x_i) *
                      ((1-p)**(1-x_i))+(1-pi)*(q**x_i)*((1-q)**(1-x_i))))
        pi = sum(mu)/len(mu)
        p = sum([mu[i]*x[i] for i in range(len(x))]) / sum(mu)
        q = sum([(1-mu[i])*x[i] for i in range(len(x))]) / \
            sum([1-mu[i] for i in range(len(mu))])
        loglikelihood = get_loglikelihood([pi, p, q], x)
        result_list.append([pi, p, q, loglikelihood])
        if result_list[i+1] == result_list[i]:
            break
    return result_list, pi, p, q, loglikelihood


def draw_plot(result_list):
    data = pd.DataFrame(result_list, columns=["pi", "p", "q", "loglikelihood"])
    fig = plt.figure(figsize=(12, 6))
    ax1=fig.add_subplot(121)
    plt.title('EM算法参数收敛情况')
    plt.xlabel('迭代次数')
    plt.ylabel('参数值')
    plt.margins(x=0)
    sns.lineplot(data=data.iloc[:,0:3])
    ax2=fig.add_subplot(122)
    plt.title('对数似然值变化情况')
    plt.xlabel('迭代次数')
    plt.ylabel('log(likelihood)')
    plt.margins(x=0)
    sns.lineplot(data=data["loglikelihood"])
    plt.savefig("pic.png", dpi=300)

```

```python

for i in range(10):
    initial_theta_list = np.random.rand(3)
    result_list, pi, p, q, loglikelihood = EM_algorithm(
        x_list, 100, initial_theta_list=initial_theta_list)
    print(pi, p, q, loglikelihood)
print("EM算法估计的各硬币正面向上的概率分别是：", (pi, p, q))
draw_plot(result_list)

```

```
0.37791574858448046 0.741185877323549 0.09627985951218022 -64.10354778811555
0.8794275954353623 0.2739006813969091 0.8221133411822208 -64.10354778811556
0.5186876268611077 0.23051970499195382 0.45798174652242046 -64.10354778811556
0.533677915017902 0.03893590769298998 0.6845499628766888 -64.10354778811558
0.3676406814907094 0.09656613275127825 0.48152743580060614 -64.10354778811555
0.8785501870434601 0.26983609645585255 0.8475554180101406 -64.10354778811556
0.1395277206185737 0.07204563246756793 0.3834494672605473 -64.10354778811556
0.6373205893289423 0.48732703584085674 0.08110867464891316 -64.10354778811556
0.07238132728958667 0.6375894792064936 0.3167793360288392 -64.10354778811555
0.8988653112771975 0.3274905536772214 0.4511815095772829 -64.10354778811555
EM算法估计的各硬币正面向上的概率分别是： (0.8988653112771975, 0.3274905536772214, 0.4511815095772829)
```



2 2. 推导高斯混合模型的EM算法

多元高斯（正态）分布

$$P(\boldsymbol{x} \mid \boldsymbol{\Sigma}, \boldsymbol{\mu}) = \frac{1}{(2\pi)^{\frac{m}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\} \tag{21}$$

其中 x 是 m 维随机变量, μ 是 m 维均值向量, Σ 是 $m \times m$ 的协方差矩阵

高斯混合模型对数似然函数

高斯混合模型是由一系列高斯分布按照参数α_k组成的混合模型，其中$\alpha_k \geq 0$且$\sum_{k=1}^{K} \alpha_k = 1$。

观测数据 $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ 的似然函数为：

$$\begin{aligned} L(\theta) = P(X) &= \prod_{i=1}^{n} P(\boldsymbol{x}_i) \\ &= \prod_{i=1}^{n}\left(\sum_{z_i \in \{1,\ldots,K\}} P(\boldsymbol{x}_i, z_i)\right) \\ &= \prod_{i=1}^{n}\left(\sum_{z_i \in \{1,\ldots,K\}} P(\boldsymbol{x}_i \mid z_i) P(z_i)\right) \\ &= \prod_{i=1}^{n}\left(\sum_{k=1}^{K} \alpha_k P(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right) \end{aligned} \tag{22}$$

对数似然函数为

$$LL(\theta) = \sum_{i=1}^{n} \log\left(\sum_{k=1}^{K} \alpha_k P(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right) \tag{23}$$

E(expectation)步：

计算隐变量 z_i 的分布 $Q(z_i)$，即 z_i 的后验分布 $P(z_i = k \mid \boldsymbol{x}_i)$

$$\begin{aligned} Q(z_i = k) &= P\left(z_i = k \mid \boldsymbol{x}_i, \alpha^{(t)}\right) \\ &= \frac{P\left(\boldsymbol{x}_i \mid z_i = k, \alpha^{(t)}\right) P\left(z_i = k \mid \alpha^{(t)}\right)}{P\left(\boldsymbol{x}_i \mid \alpha^{(t)}\right)} \end{aligned}$$

$$
\begin{aligned}
&= \frac{P\left(\boldsymbol{x}_i \mid z_i=k, \alpha^{(t)}\right) P\left(z_i=k \mid \alpha^{(t)}\right)}{\sum_{k=1}^K P\left(\boldsymbol{x}_i \mid z_i=k, \alpha^{(t)}\right) P\left(z_i=k \mid \alpha^{(t)}\right)} \\
&= \frac{\alpha_k P\left(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)}{\sum_{k=1}^K \alpha_k P\left(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)} \\
&= \gamma_{ik}^{(t)}
\end{aligned} \tag{24}
$$

且有$\sum_{k=1}^K \gamma_{ik}^{(t)} = 1$

M(maximize)步：

参数$\theta = (\alpha, \boldsymbol{\mu}, \boldsymbol{\Sigma})$的更新公式为：

$$
\begin{aligned}
\theta^* &= \arg\max_\theta \sum_{i=1}^n \sum_{k=1}^K \gamma_{ik} \log\left(P\left(\boldsymbol{x}_i \mid z_i=k, \theta\right) P\left(z_i=k \mid \theta\right)\right) \\
&= \arg\max_\theta \sum_{i=1}^n \sum_{k=1}^K \gamma_{ik} \left\{\log P\left(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) + \log \alpha_k\right\} \\
&= \arg\max_{\alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k} \sum_{i=1}^n \sum_{k=1}^K \gamma_{ik} \left\{\log\left\{\frac{1}{(2\pi)^{\frac{n}{2}} |\boldsymbol{\Sigma}_k|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)^\top \boldsymbol{\Sigma}_k^{-1}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)\right\}\right\} + \log \alpha_k\right\} \\
&= \arg\min_{\alpha_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k} \sum_{i=1}^n \sum_{k=1}^K \gamma_{ik} \left\{\frac{1}{2}\log|\boldsymbol{\Sigma}_k| + \frac{1}{2}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)^\top \boldsymbol{\Sigma}_k^{-1}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right) - \log \alpha_k\right\} \\
&\triangleq \arg\min_\theta f(\alpha, \boldsymbol{\mu}, \boldsymbol{\Sigma})
\end{aligned} \tag{25}
$$

将$f(\cdot)$分别对$\boldsymbol{\mu}, \boldsymbol{\Sigma}$求偏导，当$\sum_{i=1}^n \gamma_{ik}^{(t)} \neq 0$对所有$k = 1, 2, \cdots, K$成立时：

$$
\begin{cases}
\dfrac{\partial f(\alpha, \boldsymbol{\mu}, \boldsymbol{\Sigma})}{\partial \boldsymbol{\mu_k}} = -\sum_{i=1}^n \gamma_{ik}^{(t)}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)\boldsymbol{\Sigma}_k^{-1} \equiv 0 & \Longrightarrow \boldsymbol{\Sigma}_k^{-1} \sum_{i=1}^n \gamma_{ik}^{(t)} \boldsymbol{x}_i = \boldsymbol{\mu}_k \boldsymbol{\Sigma}_k^{-1} \sum_{i=1}^n \gamma_{ik}^{(t)} & \Longrightarrow \boldsymbol{\mu}_k = \\
\dfrac{\partial f(\alpha, \boldsymbol{\mu}, \boldsymbol{\Sigma})}{\partial \boldsymbol{\Sigma_k}} = \sum_{i=1}^n \left(\boldsymbol{\Sigma}_k^{-1} - (\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-2}\right)\gamma_{ij} \equiv 0 & \Longrightarrow \boldsymbol{\Sigma}_k \sum_{i=1}^n \gamma_{ik}^{(t)} = \sum_{i=1}^n \gamma_{ik}^{(t)}\left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T & \Longrightarrow \boldsymbol{\Sigma}_k =
\end{cases}
$$

下面计算α_k的参数迭代过程：

$$
\begin{aligned}
\alpha_k &= \arg\min_\alpha \sum_{i=1}^n \sum_{k=1}^K (-\log \alpha_k)\gamma_{ik} \\
&= \arg\max_\alpha \sum_{i=1}^N \sum_{k=1}^K \log \alpha_k \gamma_{ik}
\end{aligned} \tag{27}
$$

由约束 $\sum_{k=1}^K \alpha_k = 1$ 构建拉格朗日对偶函数：

$$
L(\alpha, \lambda) = \sum_{i=1}^n \sum_{k=1}^K \log \alpha_k \gamma_{ik} + \lambda\left(\sum_{k=1}^K \alpha_k - 1\right) \tag{28}
$$

将$L(\alpha, \lambda)$对α_k求偏导：

$$
\frac{\partial L(\alpha, \lambda)}{\partial \alpha_k} = \sum_{i=1}^n \frac{1}{\alpha_k}\gamma_{ik} + \lambda \equiv 0 \Longrightarrow \alpha_k = \frac{\sum_{i=1}^n \gamma_{ik}}{\lambda} \tag{29}
$$

由$\sum_{k=1}^K \gamma_{ik}^{(t)} = 1$，将上式代入得：

$$
\sum_{i=1}^n \sum_{k=1}^K \gamma_{ik}^{(t)} = \lambda \sum_{k=1}^K \alpha_k = n \tag{30}
$$

所以得到$\lambda = n$，从而得到：

$$
\alpha_k = \frac{\sum_{i=1}^n \gamma_{ik}}{n} \tag{31}
$$

当$\sum_{i=1}^n \gamma_{ik}^{(t)} = 0$对$k$成立时，$\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$的取值对结果无影响。

3. 高斯混合模型EM算法代码实现

按下列参数生成高斯混合模型的数据, 共有三个高斯混合成分, 每个成分 、生成 300 个数据点。

$$
\begin{aligned}
\boldsymbol{\mu}_1 &= (3, 1) \quad \boldsymbol{\Sigma}_1 = ((1, -0.5); (-0.5, 1)) \\
\boldsymbol{\mu}_2 &= (8, 10) \quad \boldsymbol{\Sigma}_2 = ((2, 0.8); (0.8, 2)) \\
\boldsymbol{\mu}_3 &= (12, 2) \quad \boldsymbol{\Sigma}_3 = ((1, 0); (0, 1))
\end{aligned} \tag{32}
$$

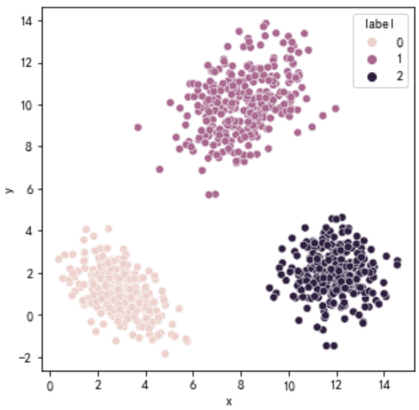
使用 scikit-learn 库中的高斯混合模型实现上述数据的学习过程, 计算在不同个数的高斯混合成分下模型的 AIC 和 BIC 值, 并将学习得到的模型参数与真实模型参数进行对比。

1 3.1 根据参数生成数据点

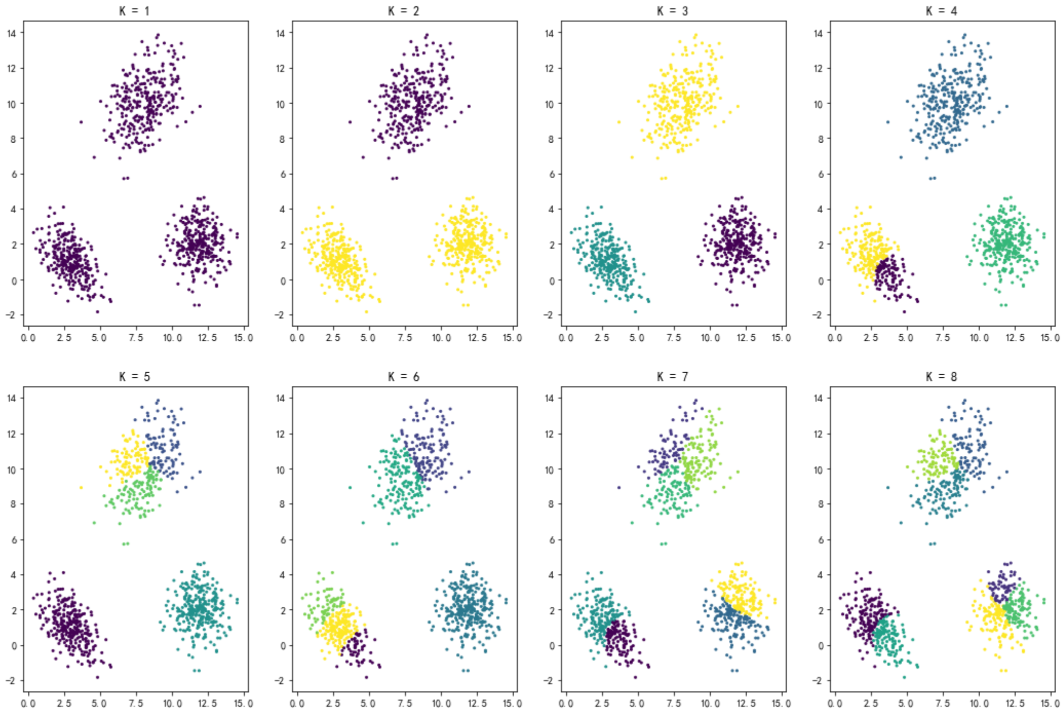
```
import numpy as np
from numpy.random import multivariate_normal
import matplotlib.pyplot as plt
def make_data(len,mu,sigma): #alpha:隐变量权值（K）;mu多元高斯分布均值（n*K）;sigma多元高斯分布协方差矩阵（K*k）,len表示生成点数量
```

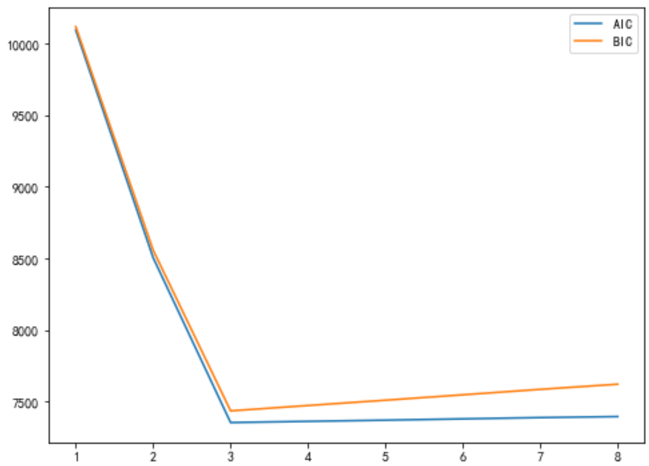
```
    data = pd.DataFrame()
    K = mu.shape[0]
    for k in range(K):
        x_k = multivariate_normal(mean = mu[k], cov = sigma[k], size=(len), check_valid="raise")
        points = pd.DataFrame(x_k,columns =["x","y"])
        points["label"] = k
        data = data.append(points)
        result = np.array(data.iloc[:,0:2])
    return result,data
mu = np.array([[3,1],[8,10],[12,2]])
sigma = np.array([[[1,-0.5],[-0.5,1]],[[2,0.8],[0.8,2]],[[1,0],[0,1]]])
result,data = make_data(300,mu,sigma) #生成300个数据点
plt.figure(figsize=(5,5))
sns.scatterplot(x="x",y ="y",data = data,hue = "label")
```

```
<AxesSubplot:xlabel='x', ylabel='y'>
```




```
from sklearn.mixture import GaussianMixture as GMM
AIC_list = []
BIC_list = []
plt.figure(figsize=(18,12))
for K in range(1, 9):
    gmm = GMM(n_components=K).fit(result)
    labels = gmm.predict(result)
    AIC_list.append(gmm.aic(result))
    BIC_list.append(gmm.bic(result))
    plt.subplot(2, 4, K)
    plt.scatter(result[:, 0], result[:, 1], c=labels, s=4)
    plt.title("K = {}".format(K))
    plt.savefig("Gaussian_result.png",dpi=300)
plt.figure(figsize=(8,6))
plt.plot(range(1,9),AIC_list,label = "AIC")
plt.plot(range(1,9),BIC_list,label = "BIC")
plt.legend()
plt.savefig("AIC-BIC.png",dpi=100)
```


K = 1
K = 2
K = 3
K = 4
K = 5
K = 6
K = 7
K = 8


AIC
BIC