《机器学习》课程系列

EM 算法*

武汉纺织大学数学与计算机学院

杜小勤

2020/05/04

Contents

*本系列文档属于讲义性质，仅用于学习目的。Last updated on: June 22, 2020。

1 EM 算法的引入

EM(Expectation Maximization) 算法于 1977 年由 Dempster 等人提出，是一种求解隐变量 (Hidden Variable) 概率模型的极大似然估计或极大后验估计方法。它在机器学习中有着极为广泛的用途，例如，它常被用来学习高斯混合模型 (Gaussian Mixture Model，GMM) 的参数、隐马尔科夫模型 (Hidden Markov Model, HMM) 的参数等。

EM 算法是一种迭代优化策略，其每次迭代都分两步，即期望步 (E 步) 和极大步 (M 步)，EM 算法由此得名。EM 算法最初是为了解决数据缺失情况下的参数估计问题而提出的，它的基础算法、收敛性与有效性等问题，在 Dempster、Laird 和 Rubin 三人于 1977 年所写的论文“Maximum Likelihood from Incomplete Data Via the EM Algorithm”中给出了详细的阐述。

EM 算法的基本思想是，在数据缺失的情况下，引入隐变量，然后交替地执行 E 步和 M 步——在 E 步，利用初始或上一阶段推断出的模型参数 $\boldsymbol{\theta}^t$，针对隐变量 $\boldsymbol{Z}^t$ 求和，计算出关于隐变量 $\boldsymbol{Z}^t$ 的期望值，从而得到 Q 函数；在 M 步，对 Q 函数使用最大似然方法，从数据集中计算出最优的模型参数 $\boldsymbol{\theta}^{t+1}$。上述过程，循环往复，直至算法收敛到局部最优解。

下面，通过一个简单的实例来说明 EM 算法。假设有三枚硬币：A、B、C，它们的正面朝上概率分别是 π、p、q。我们按如下方法进行试验：首先，抛出硬币 A，如果正面朝上，选择 B 硬币继续开展试验，否则，选择 C 硬币继续开展试验；然后，再抛出所选的硬币 B 或 C，并记录 B 或 C 的试验结果。上述试验独立重复地执行 $N = 10$ 次，最后得到如下的试验结果：1101001011，其中 1 表示正面朝上，0 表示反面朝上。假设只能观测到抛硬币的结果，不能观测到抛硬币的过程，试估计三硬币模型的参数：π、p、q。

在这个问题中，需要估计 A、B 和 C 三个模型的参数 π、p 和 q，而我们只能观测到 B 和 C 模型的数据 (观测数据)，A 模型的数据对我们而言是隐藏的，无法直接观测 (不可观测数据)。显然，直接对观测数据进行最大似然求解，会非常困难，甚至不可行。

为了解决该问题，可以引入隐变量。一般地，使用 Y 表示观测随机变量，使用 Z 表示隐 (随机) 变量。Y 和 Z 合并在一起，被称为完全数据 (Complete Data)，观测数据 Y 又被称为不完全数据 (Incomplete Data)。

对于此问题，设观测变量 Y 的取值为 1 或 0，分别表示试验的观测结果为正面或反面，即 B 和 C 的试验结果；设隐变量 Z 的取值为 1 或 0，分别表示未观测到的硬币 A 的试验结果为正面或反面。在引入了隐变量 Z 之后，三硬币的概率模型可以表示为：

$$\begin{aligned} P(y;\boldsymbol{\theta}) &= \sum_z P(y,z;\boldsymbol{\theta}) = \sum_z P(z;\boldsymbol{\theta})P(y|z;\boldsymbol{\theta}) \\ &= \pi p^y(1-p)^{1-y} + (1-\pi)q^y(1-q)^{1-y} \end{aligned} \tag{1}$$

其中，$\boldsymbol{\theta}$ 表示模型参数 π、p 和 q。上式就是观测数据 1101001011 的生成模型。

设观测数据集为 $\boldsymbol{Y} = (y_1, y_2, \ldots, y_N)$，引入不可观测随机变量 z，那么完全数据集的似然函数为：

$$\begin{aligned} P(\boldsymbol{Y};\boldsymbol{\theta}) &= \prod_{j=1}^N P(y_j;\boldsymbol{\theta}) = \prod_{j=1}^N \sum_z P(y_j,z;\boldsymbol{\theta}) \\ &= \prod_{j=1}^N \left(\pi p^{y_j}(1-p)^{1-y_j} + (1-\pi)q^{y_j}(1-q)^{1-y_j}\right) \end{aligned} \tag{2}$$

其对数似然函数为：

$$L(\boldsymbol{\theta}) = \log P(\boldsymbol{Y};\boldsymbol{\theta}) = \log \prod_{j=1}^N \sum_z P(y_j,z;\boldsymbol{\theta}) = \sum_{j=1}^N \log \sum_z P(y_j,z;\boldsymbol{\theta}) \tag{3}$$

可以看出，上式中 log 内部含有求和项。实际上，这正是隐变量模型求解的困难根源。因此，需要对该式进行某种变形处理。此处的关键技巧是，利用 Jensen 不等式找到上式的下界，然后通过 E 步与 M 步交替迭代的方式，不断地提高下界，从而间接地找到上式的局部最优值！

首先，为了应用 Jensen 不等式，引入一个关于 z 的任意分布 $q(z)$，它满足如下条件：对 $\forall z$，$q(z) \geqslant 0$，且 $\sum\limits_z q(z) = 1$。然后，利用 log 函数的凸性及 Jensen 不等式对公式 (3) 进行变形，找到其下界：

$$\begin{aligned} L(\boldsymbol{\theta}) &= \sum_{j=1}^N \log \sum_z P(y_j,z;\boldsymbol{\theta}) = \sum_{j=1}^N \log \sum_z q(z)\frac{P(y_j,z;\boldsymbol{\theta})}{q(z)} \\ &\geqslant \sum_{j=1}^N \sum_z q(z) \log \frac{P(y_j,z;\boldsymbol{\theta})}{q(z)} \end{aligned} \tag{4}$$

对于上式，仅当 $\frac{P(y_j,z;\boldsymbol{\theta})}{q(z)} = c$ 时 (其中 $c > 0$ 为常数)，等号成立：此时，不等式右端是似然函数 $L(\boldsymbol{\theta})$ 的精确下界。现在，通过该等式条件来分析 $q(z)$ 应该具有什

么形式。令 $\frac{P(y_j,z;\boldsymbol{\theta})}{q(z)}=c$，变形且两端对 z 求和，得到：

$$\begin{aligned}
&\frac{P(y_j,z;\boldsymbol{\theta})}{q(z)}=c \quad\Rightarrow\quad P(y_j,z;\boldsymbol{\theta})=cq(z) \quad\Rightarrow\\
&\sum_z P(y_j,z;\boldsymbol{\theta})=\sum_z cq(z) \quad\Rightarrow\quad c=\sum_z P(y_j,z;\boldsymbol{\theta}) \quad\Rightarrow\\
&q(z)=\frac{P(y_j,z;\boldsymbol{\theta})}{\sum\limits_z P(y_j,z;\boldsymbol{\theta})}=P(z|y_j;\boldsymbol{\theta})
\end{aligned} \tag{5}$$

下面，需要将 $q(z)=P(z|y_j;\boldsymbol{\theta})$ 代入公式 (4)。然而，需要注意的是，如果仅将它原样代入公式 (4)，那么只能得到 $L(\boldsymbol{\theta})$ 的另一种精确形式，而不能让 $L(\boldsymbol{\theta})$ 极大化。为此，需要引入迭代更新的过程。于是，将时间步 t 引入到 $\boldsymbol{\theta}$ 中，令 $q^t(z)=P(z|y_j;\boldsymbol{\theta}^t)$，令 $P(y_j,z;\boldsymbol{\theta})\Rightarrow P(y_j,z;\boldsymbol{\theta}^{t+1})$，将 $L(\boldsymbol{\theta})$ 改成 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)$[1]，得到：

$$\begin{aligned}
L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)&=\sum_{j=1}^N\log\sum_z P(y_j,z;\boldsymbol{\theta}^{t+1})=\sum_{j=1}^N\log\sum_z q^t(z)\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{q^t(z)}\\
&\geqslant\sum_{j=1}^N\sum_z q^t(z)\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{q^t(z)}=\sum_{j=1}^N\sum_z P(z|y_j;\boldsymbol{\theta}^t)\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^t)}
\end{aligned} \tag{6}$$

注意，此时 $\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^t)}$ 不再是常数。只有当 $\boldsymbol{\theta}^t\Rightarrow\boldsymbol{\theta}^{t+1}$ 时 (即相当于 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t+1})$)，等号才成立。

令 $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)=\sum\limits_{j=1}^N\sum\limits_z P(z|y_j;\boldsymbol{\theta}^t)\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^t)}$，它是 EM 算法的核心，被称为 Q 函数。其功能是，使用不完全数据集 $\boldsymbol{Y}$，对隐变量 z 求期望。Q 函数的确定，意味着 E 步的结束。在 M 步，再次使用不完全数据集 $\boldsymbol{Y}$，并利用最大似然方法估计 $\boldsymbol{\theta}^{t+1}$：

$$\boldsymbol{\theta}_*^{t+1}=\arg\max_{\boldsymbol{\theta}^{t+1}}Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t) \tag{7}$$

E 步和 M 步交替执行，直至 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)$ 收敛至局部最优值，算法停止。

回到本例，首先，需要确定 Q 函数 $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)$：

$$Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)=\sum_{j=1}^N\sum_z P(z|y_j;\boldsymbol{\theta}^t)\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^t)} \tag{8}$$

从参数和时间步的角度看，它包含了两类参数，即 $\boldsymbol{\theta}^t$ 和 $\boldsymbol{\theta}^{t+1}$，前者是上一轮的模型参数，本轮可以直接使用；后者是本轮需要极大化的模型参数，为本轮的求解

[1]注意，$L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)$ 是 $\boldsymbol{\theta}^{t+1}$ 的函数，$\boldsymbol{\theta}^t$ 固定。

目标[2]。于是，对 Q 函数按 z 求和并展开，可得：

$$
\begin{aligned}
Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) &= \sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^{t})} \\
&= \sum_{j=1}^{N}P(z=1|y_j;\boldsymbol{\theta}^{t})\log\frac{P(y_j,z=1;\boldsymbol{\theta}^{t+1})}{P(z=1|y_j;\boldsymbol{\theta}^{t})}+ \\
&\quad \sum_{j=1}^{N}P(z=0|y_j;\boldsymbol{\theta}^{t})\log\frac{P(y_j,z=0;\boldsymbol{\theta}^{t+1})}{P(z=0|y_j;\boldsymbol{\theta}^{t})}
\end{aligned}
\tag{9}
$$

其中，$P(z=1|y_j;\boldsymbol{\theta}^{t})$ 为：

$$
\begin{aligned}
P(z=1|y_j;\boldsymbol{\theta}^{t}) &= \frac{P(y_j,z=1;\boldsymbol{\theta}^{t})}{P(y_j,z=1;\boldsymbol{\theta}^{t})+P(y_j,z=0;\boldsymbol{\theta}^{t})} \\
&= \frac{\pi_t p_t^{y_j}(1-p_t)^{1-y_j}}{\pi_t p_t^{y_j}(1-p_t)^{1-y_j}+(1-\pi_t)q_t^{y_j}(1-q_t)^{1-y_j}}
\end{aligned}
\tag{10}
$$

$P(z=0|y_j;\boldsymbol{\theta}^{t})$ 为：

$$
P(z=0|y_j;\boldsymbol{\theta}^{t}) = 1-P(z=1|y_j;\boldsymbol{\theta}^{t}) \tag{11}
$$

$P(y_j,z=1;\boldsymbol{\theta}^{t+1})$ 为：

$$
P(y_j,z=1;\boldsymbol{\theta}^{t+1}) = \pi_{t+1}p_{t+1}^{y_j}(1-p_{t+1})^{1-y_j} \tag{12}
$$

$P(y_j,z=0;\boldsymbol{\theta}^{t+1})$ 为：

$$
P(y_j,z=1;\boldsymbol{\theta}^{t+1}) = (1-\pi_{t+1})q_{t+1}^{y_j}(1-q_{t+1})^{1-y_j} \tag{13}
$$

为简化表达，令 $\mu_j^t = P(z=1|y_j;\boldsymbol{\theta}^{t})$，于是，Q 函数为：

$$
Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) = \sum_{j=1}^{N}\mu_j^t\log\frac{\pi_{t+1}p_{t+1}^{y_j}(1-p_{t+1})^{1-y_j}}{\mu_j^t}+\left(1-\mu_j^t\right)\log\frac{(1-\pi_{t+1})q_{t+1}^{y_j}(1-q_{t+1})^{1-y_j}}{1-\mu_j^t} \tag{14}
$$

于是，完成 E 步，得到：

$$
\begin{aligned}
&L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) \geqslant Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) \\
&\quad = \sum_{j=1}^{N}\mu_j^t\log\frac{\pi_{t+1}p_{t+1}^{y_j}(1-p_{t+1})^{1-y_j}}{\mu_j^t}+\left(1-\mu_j^t\right)\log\frac{(1-\pi_{t+1})q_{t+1}^{y_j}(1-q_{t+1})^{1-y_j}}{1-\mu_j^t}
\end{aligned}
\tag{15}
$$

[2]即 $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$ 是 $\boldsymbol{\theta}^{t+1}$ 的函数，$\boldsymbol{\theta}^{t}$ 固定。

对 Q 函数进行极大化：

$$
\begin{aligned}
\boldsymbol{\theta}_*^{t+1} &= \arg\max_{\boldsymbol{\theta}^{t+1}} Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t) \\
&= \arg\max_{\boldsymbol{\theta}^{t+1}} \sum_{j=1}^{N} \mu_j^t \log \frac{\pi_{t+1} p_{t+1}^{y_j} (1-p_{t+1})^{1-y_j}}{\mu_j^t} + \left(1-\mu_j^t\right) \log \frac{(1-\pi_{t+1}) q_{t+1}^{y_j} (1-q_{t+1})^{1-y_j}}{1-\mu_j^t}
\end{aligned}
\tag{16}
$$

首先，求解 $\frac{\partial Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t)}{\partial \pi_{t+1}}$：

$$
\frac{\partial Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t)}{\partial \pi_{t+1}} = \sum_{j=1}^{N} \left(\frac{\mu_j^t}{\pi_{t+1}} - \frac{1-\mu_j^t}{1-\pi_{t+1}} \right) = 0 \quad \Rightarrow \quad \pi_{t+1} = \frac{1}{N} \sum_{j=1}^{N} \mu_j^t \tag{17}
$$

然后，求解 $\frac{\partial Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t)}{\partial p_{t+1}}$：

$$
\frac{\partial Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t)}{\partial p_{t+1}} = \sum_{j=1}^{N} \left(\frac{\mu_j^t y_j}{p_{t+1}} - \frac{\mu_j^t (1-y_j)}{1-p_{t+1}} \right) = 0 \quad \Rightarrow \quad p_{t+1} = \frac{\sum\limits_{j=1}^{N} \mu_j^t y_j}{\sum\limits_{j=1}^{N} \mu_j^t} \tag{18}
$$

最后，求解 $\frac{\partial Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t)}{\partial q_{t+1}}$：

$$
\frac{\partial Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t)}{\partial q_{t+1}} = \sum_{j=1}^{N} \left(\frac{(1-\mu_j^t) y_j}{q_{t+1}} - \frac{(1-\mu_j^t)(1-y_j)}{1-q_{t+1}} \right) = 0 \quad \Rightarrow \quad q_{t+1} = \frac{\sum\limits_{j=1}^{N} (1-\mu_j^t) y_j}{\sum\limits_{j=1}^{N} (1-\mu_j^t)} \tag{19}
$$

最后，完成 M 步，得到了新的模型参数 π_{t+1}、p_{t+1} 和 q_{t+1}。它们的更新公式，具有可解释性，意义非常明确。

需要注意的是，在进行算法迭代时，首先需要为模型参数 $\boldsymbol{\theta}_0 = \{\pi_0, p_0, q_0\}$ 设置初始值，一般可以取 $\boldsymbol{\theta}_0 = \{\pi_0 = 0.5, p_0 = 0.5, q_0 = 0.5\}$，并据此利用观测数据集 $\boldsymbol{Y}$ 计算 $\mu_j^0 (j = 1, 2, \cdots, N)$，然后使用上面的更新公式，再次利用观测数据集 $\boldsymbol{Y}$ 计算出新的模型参数 $\boldsymbol{\theta}_1$。上述过程反复迭代，直至收敛为止。注意，算法的求解结果与初始值有很大的关系，不同的初始值可能会得到不同的参数估计值。

2 通用 EM 算法

实际上，在上一节，我们几乎已经推导出了通用 EM 算法的核心公式。为了完整性，将公式 (6) 的主要部分重新列出：

$$
\begin{aligned}
L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) &= \sum_{j=1}^{N}\log\sum_{z}P(y_j,z;\boldsymbol{\theta}^{t+1}) \\
&\geqslant \sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^{t})}
\end{aligned} \tag{20}
$$

继续变形：

$$
\begin{aligned}
L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) &= \sum_{j=1}^{N}\log\sum_{z}P(y_j,z;\boldsymbol{\theta}^{t+1}) \\
&\geqslant \sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^{t})} \\
&= \sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log P(y_j,z;\boldsymbol{\theta}^{t+1}) - \sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log P(z|y_j;\boldsymbol{\theta}^{t}) \\
&\geqslant \sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log P(y_j,z;\boldsymbol{\theta}^{t+1})
\end{aligned} \tag{21}
$$

其中，熵 $H\left(P(Z|y_j;\boldsymbol{\theta}^{t})\right) = -\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log P(z|y_j;\boldsymbol{\theta}^{t}) \geqslant 0$。上式表明，我们又获得了一个新下界。将 2 个下界列出如下，并分别命名为 $B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$ 和 $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$：

$$
\begin{aligned}
B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) &= \sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^{t})} \\
Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) &= \sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^{t})\log P(y_j,z;\boldsymbol{\theta}^{t+1})
\end{aligned} \tag{22}
$$

由于 $B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$ 关于 $\boldsymbol{\theta}^{t+1}$ 的最优化与其 log 函数中 $P(z|y_j;\boldsymbol{\theta}^{t})$ 分母项无关，因此：

$$
\arg\max_{\boldsymbol{\theta}^{t+1}} B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) = \arg\max_{\boldsymbol{\theta}^{t+1}} Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) \tag{23}
$$

这表明，两个下界关于 $\boldsymbol{\theta}^{t+1}$ 的最优化具有等价性。在 2 个下界中，$B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$ 是 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$ 的紧下界，当 $\boldsymbol{\theta}^{t} \Rightarrow \boldsymbol{\theta}^{t+1}$ 时，两者相等：

$$
L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t+1}) = B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t+1}) \tag{24}
$$

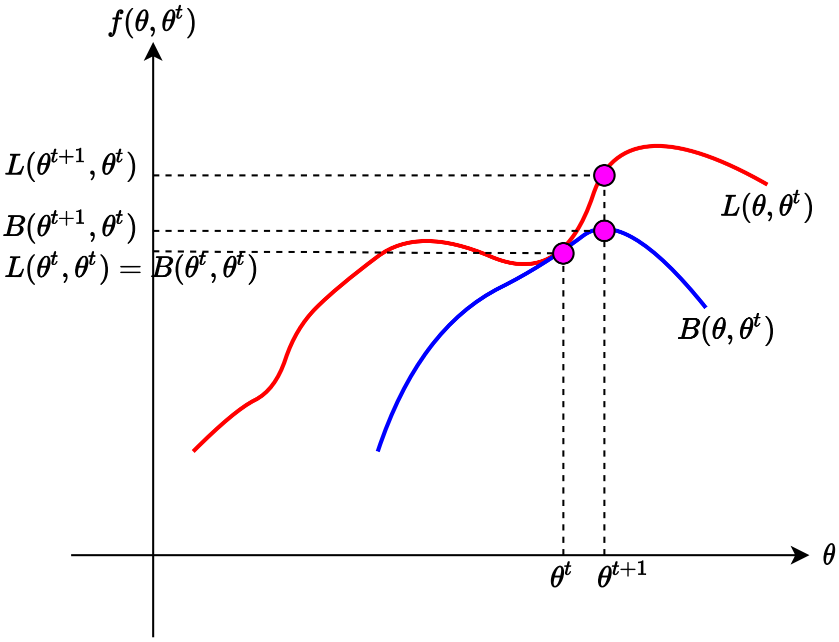


图 2-1: EM 算法的解释

于是，我们得到一个简化版的 Q 函数：

$$Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)=\sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log P(y_j,z;\boldsymbol{\theta}^{t+1}) \tag{25}$$

下面给出 Q 函数的正式定义。

定义 2.1 （Q 函数）（完全数据）对数似然函数 $\log P(y,z;\boldsymbol{\theta}^{t+1})$ 关于观测数据集 $\boldsymbol{Y}$ 的 Q 函数被定义为：

$$Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)=\sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log P(y_j,z;\boldsymbol{\theta}^{t+1}) \tag{26}$$

从上面的讨论可以看出，EM 算法通过迭代地极大化完全数据的对数似然函数 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)$ 的（非紧）下界，即 Q 函数，而逐步地逼近 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)$ 的局部最优值。图2-1给出了 EM 算法某次迭代的直观解释。注意，图中展示了紧下界 B 函数，没有展示非紧下界 Q 函数。但是，前面的讨论已经表明，两者在极大化时，具有等价性。

在图2-1中，当 $\boldsymbol{\theta}=\boldsymbol{\theta}^t$ 时，$L(\boldsymbol{\theta},\boldsymbol{\theta}^t)=B(\boldsymbol{\theta},\boldsymbol{\theta}^t)$。在迭代时间步 $t+1$，对 $B(\boldsymbol{\theta},\boldsymbol{\theta}^t)$ 进行极大化求解，获得了极值点 $\boldsymbol{\theta}=\boldsymbol{\theta}^{t+1}$，它同时也是 Q 函数的极值点 $\boldsymbol{\theta}^{t+1}$。该

极值点 $\boldsymbol{\theta}^{t+1}$ 将使得 L 函数的值升高。在 $\boldsymbol{\theta}^{t+1}$ 点处，获得一个新的 Q 函数和 B 函数，并进入下一轮迭代。上述迭代过程，一直进行到 L 函数获得局部最优时停止。

下面给出通用 EM 算法。

算法 2.1（通用 *EM* 算法）

Input:

  Dataset: $\boldsymbol{Y}$(the number of training samples: N)

  Parameters: $\boldsymbol{\theta}$(Initialization:$\boldsymbol{\theta}^0$)

  max_steps: T

Output:

  Parameters: $\boldsymbol{\theta}^T$

Algorithm:

  for $t = 0 \cdots T-1$:

    E-step: $Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t) = \sum\limits_{j=1}^{N} \sum\limits_{z} P(z|y_j; \boldsymbol{\theta}^t) \log P(y_j, z; \boldsymbol{\theta}^{t+1})$

      where: $P(z|y_j; \boldsymbol{\theta}^t) = \frac{P(y_j, z; \boldsymbol{\theta}^t)}{\sum\limits_z P(y_j, z; \boldsymbol{\theta}^t)}$

    M-step: $\boldsymbol{\theta}_*^{t+1} = \arg\max\limits_{\boldsymbol{\theta}^{t+1}} Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t)$

3 EM 算法的收敛性

下面，将表明 EM 算法 (算法2.1) 是收敛的。

定理 3.1 对于 $\forall \boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t \in \Omega$(其中 Ω 表示参数集合)，$L(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t) - L(\boldsymbol{\theta}^t, \boldsymbol{\theta}^{t-1}) \geqslant Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t) - Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^t)$。

证明：

$$
\begin{aligned}
&L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})-L(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t-1})\\
&=\sum_{j=1}^{N}\log\sum_{z}P(y_j,z;\boldsymbol{\theta}^{t+1})-\sum_{j=1}^{N}\log\sum_{z}P(y_j,z;\boldsymbol{\theta}^{t})\\
&=\sum_{j=1}^{N}\log\frac{\sum_{z}P(y_j,z;\boldsymbol{\theta}^{t+1})}{\sum_{z}P(y_j,z;\boldsymbol{\theta}^{t})}=\sum_{j=1}^{N}\log\frac{\sum_{z}P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(y_j;\boldsymbol{\theta}^{t})}\\
&=\sum_{j=1}^{N}\log\sum_{z}\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(y_j;\boldsymbol{\theta}^{t})}=\sum_{j=1}^{N}\log\sum_{z}\frac{P(z|y_j,\boldsymbol{\theta}^{t})P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j,\boldsymbol{\theta}^{t})P(y_j;\boldsymbol{\theta}^{t})}\\
&=\sum_{j=1}^{N}\log\sum_{z}\frac{P(z|y_j,\boldsymbol{\theta}^{t})P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(y_j,z;\boldsymbol{\theta}^{t})}\geqslant\sum_{j=1}^{N}\sum_{z}P(z|y_j,\boldsymbol{\theta}^{t})\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(y_j,z;\boldsymbol{\theta}^{t})}
\end{aligned}
\tag{27}
$$

其中，上式最后一行，利用了 Jensen 不等式，继续变形：

$$
\begin{aligned}
&L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})-L(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t-1})\\
&\geqslant\sum_{j=1}^{N}\sum_{z}P(z|y_j,\boldsymbol{\theta}^{t})\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(y_j,z;\boldsymbol{\theta}^{t})}\\
&=\sum_{j=1}^{N}\sum_{z}P(z|y_j,\boldsymbol{\theta}^{t})\log P(y_j,z;\boldsymbol{\theta}^{t+1})-\sum_{j=1}^{N}\sum_{z}P(z|y_j,\boldsymbol{\theta}^{t})\log P(y_j,z;\boldsymbol{\theta}^{t})\\
&=Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})-Q(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t})
\end{aligned}
\tag{28}
$$

□

此定理表明，当参数从 $\boldsymbol{\theta}^{t}$ 变化到 $\boldsymbol{\theta}^{t+1}$ 时，对数似然函数 L 的值将从 $L(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t-1})$ 提高到 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$，提高的下界由 $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})-Q(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t})$ 保证。因为 $\boldsymbol{\theta}^{t+1}$ 是通过最优化 $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$ 而得，所以 $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})\geqslant Q(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t})$[3]，从而有 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})\geqslant L(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t-1})$。

推论 3.1 对于 $t=0,1,\ldots,T-1$，对数似然函数的值构成非递减序列，即 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})\geqslant L(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t-1})$。

虽然上述结论表明，在迭代过程中，对数似然函数的值形成一个非递减序列，但是，该保证相对较弱[4]。然而，在满足一定条件下，当 $T\to\infty$，EM 算法确实收敛到对数似然函数的局部最优[5]。

[3] $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$ 是 $\boldsymbol{\theta}^{t+1}$ 的函数，以 $\boldsymbol{\theta}^{t+1}$ 为优化目标，而 $\boldsymbol{\theta}^{t}$ 是固定常数。

[4] 简单地，令 $\boldsymbol{\theta}^{t+1}=\boldsymbol{\theta}^{t}(t=0,1,\ldots,T-1)$，则 $L(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})\geqslant L(\boldsymbol{\theta}^{t},\boldsymbol{\theta}^{t-1})$ 成立。

[5] 证明详见《On the Convergence Properties of the EM Algorithm》，美国数学家吴建福，1983。

4 朴素贝叶斯的 EM 算法

在朴素贝叶斯模型中，如果样本点 $\boldsymbol{x}$ 的类别 y 不可观测，那么就不能直接使用最大似然估计。此时，EM 算法就可以派上用场了。

设给定的观测数据集为 $\boldsymbol{X}=(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_N)$，类别 $y\in\mathcal{Y}$ 不可观测 (其中 $\mathcal{Y}=\{c_1,c_2,\ldots,c_K\}$)，朴素贝叶斯模型的似然函数为：

$$\mathcal{L}(\boldsymbol{\theta})=\prod_{i=1}^{N}P(\boldsymbol{x}_i)=\prod_{i=1}^{N}\sum_{y}P(\boldsymbol{x}_i,y)=\prod_{i=1}^{N}\sum_{y}P(y)\prod_{j=1}^{d}P(x_i^{(j)}|y) \tag{29}$$

上式中，除了 y 不可观测外，其余符号的意义与常规朴素贝叶斯模型一致，$\boldsymbol{\theta}$ 表示模型参数 $P(y)$ 和 $P(x^{(j)}|y)$。

下面，对照通用 EM 算法，直接写出其 Q 函数：

$$\begin{aligned}
Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) &= \sum_{i=1}^{N}\sum_{y}P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^{t})\log P(\boldsymbol{x}_i,y;\boldsymbol{\theta}^{t+1}) \\
&= \sum_{i=1}^{N}\sum_{y}P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^{t})\log P^{t+1}(y)\prod_{j=1}^{d}P^{t+1}(x_i^{(j)}|y) \\
&= \sum_{i=1}^{N}\sum_{y}P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^{t})\log P^{t+1}(y)+ \\
&\quad \sum_{i=1}^{N}\sum_{y}P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^{t})\sum_{j=1}^{d}\log P^{t+1}(x_i^{(j)}|y)
\end{aligned} \tag{30}$$

上式中，$P^{t+1}(y)$ 和 $P^{t+1}(x^{(j)}|y)$ 是函数 $Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t})$ 的优化目标：

$$\boldsymbol{\theta}_*^{t+1}=\arg\max_{\boldsymbol{\theta}^{t+1}}Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^{t}) \tag{31}$$

约束条件为[6]：

$$\begin{aligned}
&\sum_{y}P^{t+1}(y)=1 \\
&\sum_{x^{(j)}}P^{t+1}(x^{(j)}|y)=1
\end{aligned} \tag{32}$$

[6]其它的约束条件，例如，需要满足的概率性质——$P^{t+1}(y)\geqslant 0$ 和 $P^{t+1}(x^{(j)}|y)\geqslant 0$，可以在求解完毕之后，再进行验证，这样可以简化求解过程。

利用拉格朗日乘数法，其优化函数为：

$$
\begin{aligned}
L(\boldsymbol{\theta}^{t+1}) = -\sum_{i=1}^{N}\sum_{y} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\log P^{t+1}(y) - \sum_{i=1}^{N}\sum_{y} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\sum_{j=1}^{d}\log P^{t+1}(x_i^{(j)}|y) + \\
\alpha\left(\sum_{y} P^{t+1}(y) - 1\right) + \sum_{j=1}^{d}\beta_j\sum_{i=1}^{N}\left(\sum_{x_i^{(j)}} P^{t+1}(x_i^{(j)}|y) - 1\right)
\end{aligned}
\tag{33}
$$

首先，求解 $\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial P^{t+1}(y)}$：

$$
\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial P^{t+1}(y)} = -\sum_{i=1}^{N}\frac{P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}{P^{t+1}(y)} + \alpha = 0 \quad \Rightarrow \quad P^{t+1}(y)\alpha = \sum_{i=1}^{N} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t) \tag{34}
$$

利用公式 $\sum\limits_{y} P^{t+1}(y) = 1$，可得：

$$
\alpha = \sum_{y}\sum_{i=1}^{N} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t) = \sum_{i=1}^{N}\sum_{y} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t) = N \tag{35}
$$

于是，得到：

$$
P^{t+1}(y) = \frac{\sum\limits_{i=1}^{N} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}{\alpha} = \frac{1}{N}\sum_{i=1}^{N} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t) \tag{36}
$$

然后，求解 $\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial P^{t+1}(x_i^{(j)}|y)}$：

$$
\begin{aligned}
&\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial P^{t+1}(x_i^{(j)}|y)} = -\sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\}\frac{P(y|\boldsymbol{x}_n;\boldsymbol{\theta}^t)}{P^{t+1}(x_i^{(j)}|y)} + \beta_j\sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\} = 0 \quad \Rightarrow \\
&P^{t+1}(x_i^{(j)}|y)\left(\beta_j\sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\}\right) = \sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\}P(y|\boldsymbol{x}_n;\boldsymbol{\theta}^t)
\end{aligned}
\tag{37}
$$

利用公式 $\sum\limits_{x_i^{(j)}} P^{t+1}(x_i^{(j)}|y) = 1$，对上式最后一行两端按 $x_i^{(j)}$ 求和，可得：

$$
\begin{aligned}
\beta_j\sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\} &= \sum_{x_i^{(j)}}\sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\}P(y|\boldsymbol{x}_n;\boldsymbol{\theta}^t) \\
&= \sum_{n=1}^{N}\sum_{x_i^{(j)}} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\}P(y|\boldsymbol{x}_n;\boldsymbol{\theta}^t) \\
&= \sum_{n=1}^{N} P(y|\boldsymbol{x}_n;\boldsymbol{\theta}^t)
\end{aligned}
\tag{38}
$$

于是，得到：

$$P^{t+1}(x_i^{(j)}|y) = \frac{\sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\} P(y|\boldsymbol{x}_n;\boldsymbol{\theta}^t)}{\beta_j \sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\}} = \frac{\sum_{n=1}^{N} I\{\boldsymbol{x}_n^{(j)} = x_i^{(j)}\} P(y|\boldsymbol{x}_n;\boldsymbol{\theta}^t)}{\sum_{n=1}^{N} P(y|\boldsymbol{x}_n;\boldsymbol{\theta}^t)} \tag{39}$$

因为 $P^{t+1}(x_i^{(j)}|y)$ 属于模型参数[7]，所以将 $x_i^{(j)}$ 中的下标 i 去掉，得到：

$$P^{t+1}(x^{(j)}|y) = \frac{\sum_{i=1}^{N} I\{\boldsymbol{x}_i^{(j)} = x^{(j)}\} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}{\sum_{i=1}^{N} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)} \tag{40}$$

而 $P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)$ 自身，可以利用贝叶斯公式得到：

$$P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t) = \frac{P(\boldsymbol{x}_i, y;\boldsymbol{\theta}^t)}{\sum_y P(\boldsymbol{x}_i, y;\boldsymbol{\theta}^t)} = \frac{P^t(y) \prod_{j=1}^{d} P^t(x_i^{(j)}|y)}{\sum_y P^t(y) \prod_{j=1}^{d} P^t(x_i^{(j)}|y)} \tag{41}$$

容易验证，上述求解结果符合概率 ($\geqslant 0$) 性质。于是，在 $t=0$ 时间步，为模型参数 $P^0(y)$ 和 $P^0(x^{(j)}|y)$ 设置随机值[8]，然后算法以迭代的方式运行，最终将收敛于局部最优，获得了类别不可见时的朴素贝叶斯模型参数。

需要注意的是，在公式 (36) 和公式 (40) 中，如果每个样本点的类别标签 y 是可见的，那么可以简单地设置：$P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)=1$，如果 $y=y_i$；否则，$P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)=0$。于是，朴素贝叶斯的 EM 算法就还原为最大似然方法——直接利用最大似然方法的结论计算出模型参数，而不需要迭代求解。一般而言，对于同一个问题，如果类别标签可见时，模型参数容易求解，那么相应地，类别标签不可见的模型参数也容易求解。

因此，从这个角度来看，在迭代求解的过程中，EM 算法使用已有的模型参数来推断不可见的类别概率，然后再利用该推断更新已有的模型参数，周而复始，直至算法收敛为止。

下面给出朴素贝叶斯模型的 EM 算法。

算法 4.1 (朴素贝叶斯模型的 *EM* 算法)

[7]在公式 (37) 的推导过程中，我们也是把它当作模型参数来处理的。

[8]当然，需要服从概率约束。

Input:

Dataset: $\boldsymbol{X}$(the number of training samples: N)

Parameters: $\boldsymbol{\theta}$(Initialization:$\boldsymbol{\theta}^0$)

max_steps: T

Output:

Parameters: $\boldsymbol{\theta}^T$

Algorithm:

for $t = 0 \cdots T-1$:

E-step: $P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t) = \frac{P^t(y)\prod\limits_{j=1}^{d} P^t(x_i^{(j)}|y)}{\sum\limits_y P^t(y)\prod\limits_{j=1}^{d} P^t(x_i^{(j)}|y)}, \quad i = 1, 2, \ldots, N$

M-step: $P^{t+1}(y) = \frac{1}{N}\sum\limits_{i=1}^{N} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)$

$P^{t+1}(x^{(j)}|y) = \frac{\sum\limits_{i=1}^{N} I\{\boldsymbol{x}_i^{(j)}=x^{(j)}\}P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}{\sum\limits_{i=1}^{N} P(y|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}$

5 高斯混合模型的 EM 算法

高斯混合模型 (Gaussian Mixture Model, GMM) 是一类重要的模型，应用非常广泛。

图5-2表示的是由四个高斯分布组成的高斯混合模型。如果数据集带有类别标签数据，那么就可以直接利用最大似然方法估计出每个高斯分布的模型参数，即均值 $\boldsymbol{\mu_k}$ 与方差 $\boldsymbol{\Sigma_k}$。

但是，如果数据集没有提供类别数据，那么如何求解出每个高斯分布的参数呢？与朴素贝叶斯模型的 EM 算法一样，在没有类别数据的情况下，可以引入隐变量，对类别进行建模，然后使用 EM 算法进行求解，最后可以得到每个高斯分布的模型参数。

定义 **5.1**（高斯混合模型）高斯混合模型是一种具有如下形式的概率模型：

$$P(\boldsymbol{x};\boldsymbol{\theta}) = \sum_{k=1}^{K} \alpha_k \phi(\boldsymbol{x};\boldsymbol{\theta}_k) \tag{42}$$

其中，$\alpha_k(\alpha_k \geqslant 0)$ 为权重系数，满足条件 $\sum\limits_{k=1}^{K}\alpha_k = 1$，$K$ 表示基模型或分模型的

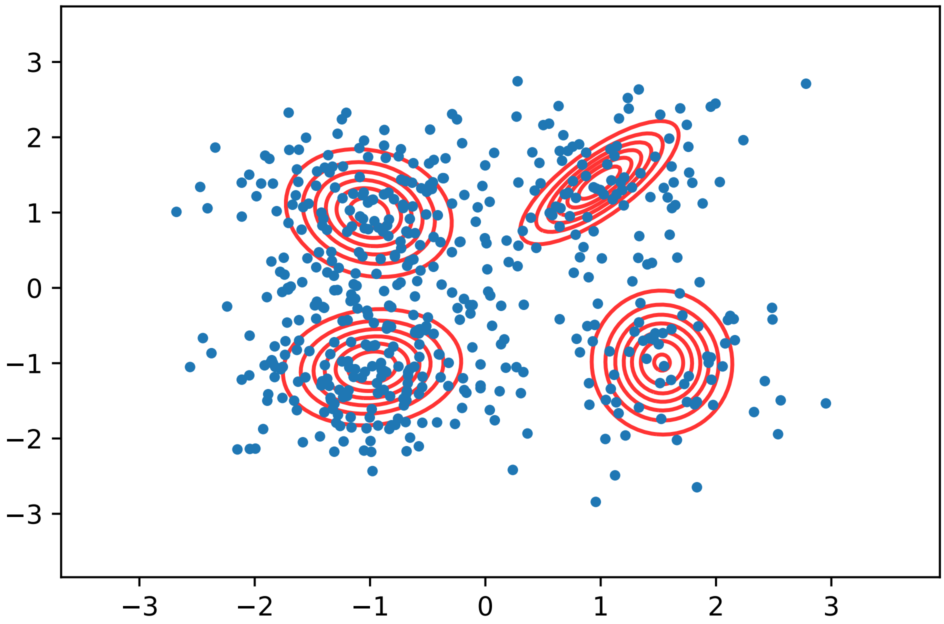


图 5-2: 高斯混合模型示意图

个数；$\phi(\boldsymbol{x};\boldsymbol{\theta}_k)$ 是高斯分布，$\boldsymbol{\theta}_k=(\boldsymbol{\mu_k},\boldsymbol{\Sigma_k})$：

$$\phi(\boldsymbol{x};\boldsymbol{\theta}_k)=\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)=\frac{1}{(2\pi)^{\frac{d}{2}}}\frac{1}{|\boldsymbol{\Sigma}_k|^{\frac{1}{2}}}\exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\right\} \tag{43}$$

其中，d 维向量 $\boldsymbol{\mu}_k$ 被称为均值向量，$d\times d$ 维矩阵 $\boldsymbol{\Sigma}_k$ 被称为协方差矩阵，$|\boldsymbol{\Sigma}_k|$ 表示 $\boldsymbol{\Sigma}_k$ 的行列式；$\boldsymbol{\theta}=(\alpha_1,\alpha_2,\ldots,\alpha_K,\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\ldots,\boldsymbol{\theta}_K)$。

假设观测数据 $\boldsymbol{X}=(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_N)$ 由高斯混合模型 $P(\boldsymbol{x};\boldsymbol{\theta})$ 生成：

$$P(\boldsymbol{x};\boldsymbol{\theta})=\sum_{k=1}^{K}\alpha_k\phi(\boldsymbol{x};\boldsymbol{\theta}_k) \tag{44}$$

其类别数据 k 未知，试估计模型参数 $\boldsymbol{\theta}=(\alpha_1,\alpha_2,\ldots,\alpha_K,\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\ldots,\boldsymbol{\theta}_K)$。

针对此问题，将类别 k 设为隐变量，它用来表示样本点 $\boldsymbol{x}$ 所属的高斯分模型[9]。下面使用 EM 算法求解模型参数 $\boldsymbol{\theta}$。对照通用 EM 算法，直接写出其 Q 函数：

$$Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)=\sum_{i=1}^{N}\sum_{k=1}^{K}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\log P(\boldsymbol{x}_i,k;\boldsymbol{\theta}^{t+1}) \tag{45}$$

[9]即我们把此问题当作一个类别未知（无监督）的高斯混合模型（多分类）的参数估计问题来处理。

其中，$P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)$ 表示 $\boldsymbol{x}_i$ 来自于类别或高斯分模型 k 的概率，称为分模型 k 对观测数据 $\boldsymbol{x}_i$ 的响应度；$P(\boldsymbol{x}_i,k;\boldsymbol{\theta}^{t+1})=\alpha_k^{t+1}\phi(\boldsymbol{x}_i;\boldsymbol{\theta}_k^{t+1})$。于是，得到：

$$
\begin{aligned}
&Q(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\log P(\boldsymbol{x}_i,k;\boldsymbol{\theta}^{t+1})\\
&=\sum_{i=1}^{N}\sum_{k=1}^{K}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\log\left(\alpha_k^{t+1}\frac{1}{(2\pi)^{\frac{d}{2}}}\frac{1}{|\boldsymbol{\Sigma}_k^{t+1}|^{\frac{1}{2}}}\exp\left\{-\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T(\boldsymbol{\Sigma}_k^{t+1})^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})\right\}\right)\\
&\propto\sum_{i=1}^{N}\sum_{k=1}^{K}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left[\log\alpha_k^{t+1}-\frac{1}{2}\log|\boldsymbol{\Sigma}_k^{t+1}|-\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T(\boldsymbol{\Sigma}_k^{t+1})^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})\right]
\end{aligned}
\tag{46}
$$

上式中，去掉了与优化无关项。这是一个带约束的最优化问题，写出拉格朗日优化函数如下[10]：

$$
\begin{aligned}
&L(\boldsymbol{\theta}^{t+1})=\\
&-\sum_{i=1}^{N}\sum_{k=1}^{K}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left[\log\alpha_k^{t+1}-\frac{1}{2}\log|\boldsymbol{\Sigma}_k^{t+1}|-\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T(\boldsymbol{\Sigma}_k^{t+1})^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})\right]\\
&+\gamma\left(\sum_{k=1}^{K}\alpha_k^{t+1}-1\right)
\end{aligned}
\tag{47}
$$

其中，γ 为拉格朗日乘数。首先，求解 $\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial\alpha_k^{t+1}}$：

$$
\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial\alpha_k^{t+1}}=-\sum_{i=1}^{N}\frac{P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}{\alpha_k^{t+1}}+\gamma=0\quad\Rightarrow\quad\alpha_k^{t+1}\gamma=\sum_{i=1}^{N}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\tag{48}
$$

利用公式 $\sum\limits_{k=1}^{K}\alpha_k^{t+1}=1$，对上式两端按 k 求和，得到：

$$
\gamma=\sum_{k=1}^{K}\sum_{i=1}^{N}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)=\sum_{i=1}^{N}\sum_{k=1}^{K}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)=N\tag{49}
$$

于是，得到：

$$
\alpha_k^{t+1}=\frac{\sum\limits_{i=1}^{N}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}{\gamma}=\frac{\sum\limits_{i=1}^{N}P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}{N}\tag{50}
$$

[10] 为求解方便，没有列出不等式约束，求解完毕再验证是否满足。

然后，求解 $\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial \boldsymbol{\mu}_k^{t+1}}$，利用微分算子求解[11]：

$$\begin{aligned} dL(\boldsymbol{\theta}^{t+1}) = &-\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(\frac{1}{2}d\left(\boldsymbol{\mu}_k^{t+1}\right)^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}\left(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1}\right)+\right.\\ &\left.\frac{1}{2}\left(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1}\right)^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}d\boldsymbol{\mu}_k^{t+1}\right) \\ = &-\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1}\right)^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}d\boldsymbol{\mu}_k^{t+1} \end{aligned} \tag{51}$$

其中，最后一步的推导，利用了向量内积等式 $\boldsymbol{u}^T\boldsymbol{v}=\boldsymbol{v}^T\boldsymbol{u}$ 以及 $\boldsymbol{\Sigma}_k^{t+1}$ 为对称矩阵的事实。于是，得到：

$$\begin{aligned} &\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial \boldsymbol{\mu}_k^{t+1}} = -\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}\left(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1}\right)=\mathbf{0} \quad \Rightarrow \\ &\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}\boldsymbol{\mu}_k^{t+1} = \sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}\boldsymbol{x}_i \quad \Rightarrow \\ &\boldsymbol{\mu}_k^{t+1}\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t) = \sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\boldsymbol{x}_i \quad \Rightarrow \quad \boldsymbol{\mu}_k^{t+1} = \frac{\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\boldsymbol{x}_i}{\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)} \end{aligned} \tag{52}$$

最后，求解 $\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial \boldsymbol{\Sigma}_k^{t+1}}$，利用微分算子求解：

$$\begin{aligned} dL(\boldsymbol{\theta}^{t+1}) &= -\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(-\frac{1}{2}\frac{1}{|\boldsymbol{\Sigma}_k^{t+1}|}d|\boldsymbol{\Sigma}_k^{t+1}|-\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T d(\boldsymbol{\Sigma}_k^{t+1})^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})\right) \\ &= -\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(-\frac{1}{2}\frac{1}{|\boldsymbol{\Sigma}_k^{t+1}|}|\boldsymbol{\Sigma}_k^{t+1}|\mathrm{tr}\left(\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}d\boldsymbol{\Sigma}_k^{t+1}\right)\right. \\ &\qquad \left.+\frac{1}{2}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}d\boldsymbol{\Sigma}_k^{t+1}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})\right) \end{aligned} \tag{53}$$

[11]具体方法，详见《机器学习》课程系列之“判别函数的线性分类基础”附录：标量函数对矩阵的求导，Chapter2-CN.pdf。

两端应用迹 tr 算子：

$$
\begin{aligned}
dL(\boldsymbol{\theta}^{t+1}) &= \mathrm{tr}\left(dL(\boldsymbol{\theta}^{t+1})\right) \\
&= -\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(-\frac{1}{2}\mathrm{tr}\left(\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1} d\boldsymbol{\Sigma}_k^{t+1}\right)\right. \\
&\qquad\qquad \left.+\frac{1}{2}\mathrm{tr}\left((\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1} d\boldsymbol{\Sigma}_k^{t+1}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})\right)\right) \\
&= -\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(-\frac{1}{2}\mathrm{tr}\left(\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1} d\boldsymbol{\Sigma}_k^{t+1}\right)\right. \\
&\qquad\qquad \left.+\frac{1}{2}\mathrm{tr}\left(\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1} d\boldsymbol{\Sigma}_k^{t+1}\right)\right) \\
&= \mathrm{tr}\left(-\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\right. \\
&\qquad \left.\left(-\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1} d\boldsymbol{\Sigma}_k^{t+1}+\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1} d\boldsymbol{\Sigma}_k^{t+1}\right)\right) \\
&= \mathrm{tr}\left(-\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\right. \\
&\qquad \left.\left(-\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}+\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}\right) d\boldsymbol{\Sigma}_k^{t+1}\right)
\end{aligned} \tag{54}
$$

于是，利用 $\boldsymbol{\Sigma}_k^{t+1}$ 为对称矩阵的事实，得到：

$$
\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial \boldsymbol{\Sigma}_k^{t+1}} = -\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(-\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}+\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}\right) \tag{55}
$$

令 $\frac{\partial L(\boldsymbol{\theta}^{t+1})}{\partial \boldsymbol{\Sigma}_k^{t+1}} = \mathbf{0}$，得到：

$$
\begin{aligned}
&-\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(-\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}+\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}\right) = \mathbf{0} \quad \Rightarrow \\
&\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(-\frac{1}{2}\boldsymbol{I}+\frac{1}{2}\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T\right) = \mathbf{0} \quad \Rightarrow \\
&\sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\boldsymbol{I} = \sum_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\left(\boldsymbol{\Sigma}_k^{t+1}\right)^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T \quad \Rightarrow \\
&\boldsymbol{\Sigma}_k^{t+1} = \frac{\sum\limits_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T}{\sum\limits_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}
\end{aligned} \tag{56}
$$

其中，$\boldsymbol{I}$ 为单位矩阵。而 $P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)$ 自身，可以利用贝叶斯公式得到：

$$P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t) = \frac{P(\boldsymbol{x}_i,k;\boldsymbol{\theta}^t)}{\sum\limits_{k=1}^{K} P(\boldsymbol{x}_i,k;\boldsymbol{\theta}^t)} = \frac{\alpha_k^t \phi(\boldsymbol{x}_i;\boldsymbol{\theta}_k^t)}{\sum\limits_{k=1}^{K} \alpha_k^t \phi(\boldsymbol{x}_i;\boldsymbol{\theta}_k^t)} \tag{57}$$

其中，$\boldsymbol{\theta}_k^t = (\boldsymbol{\mu}_k^t, \boldsymbol{\Sigma}_k^t)$。最后，容易验证，$\boldsymbol{\alpha}_k^{t+1} \geqslant 0$，符合概率性质。至此，高斯混合模型的参数求解完毕。

下面给出高斯混合模型的 EM 算法。

算法 **5.1** (高斯混合模型的 *EM* 算法)

Input:

Dataset: $\boldsymbol{X}$(the number of training samples: N)

Parameters: $\boldsymbol{\theta}$(Initialization:$\boldsymbol{\theta}^0$)

max_steps: T

Output:

Parameters: $\boldsymbol{\theta}^T$

Algorithm:

for $t = 0 \cdots T-1$:

E-step: $P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t) = \frac{\alpha_k^t \phi(\boldsymbol{x}_i;\boldsymbol{\theta}_k^t)}{\sum\limits_{k=1}^{K} \alpha_k^t \phi(\boldsymbol{x}_i;\boldsymbol{\theta}_k^t)}, \quad i = 1,2,\ldots,N$

M-step: $\alpha_k^{t+1} = \frac{\sum\limits_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}{N}$

$\boldsymbol{\mu}_k^{t+1} = \frac{\sum\limits_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)\boldsymbol{x}_i}{\sum\limits_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}, \quad k = 1,2,\ldots,K$

$\boldsymbol{\Sigma}_k^{t+1} = \frac{\sum\limits_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})(\boldsymbol{x}_i-\boldsymbol{\mu}_k^{t+1})^T}{\sum\limits_{i=1}^{N} P(k|\boldsymbol{x}_i;\boldsymbol{\theta}^t)}$

6 EM 算法与 F 函数

在 Q 函数的定义 (2.1) 中，2 个 (迭代) 概率之间存在如下的关系：

$$P(z|y_j;\boldsymbol{\theta}^t) = \frac{P(y_j,z;\boldsymbol{\theta}^t)}{\sum\limits_{z} P(y_j,z;\boldsymbol{\theta}^t)} \tag{58}$$

这一关系是必然的吗？直观上理解，确实如此——隐变量 z 的条件概率与完全数据 (y_j,z) 的联合概率，两者之间当然服从贝叶斯定理。当时，我们从隐变量 z 的

分布应该满足什么条件的角度进行了讨论，并建立了上述关系。下面，从另一个角度对此问题进行讨论。

在公式 (21) 和公式 (22) 中，我们定义了 $B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)$ 函数：

$$\begin{aligned}B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)&=\sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log\frac{P(y_j,z;\boldsymbol{\theta}^{t+1})}{P(z|y_j;\boldsymbol{\theta}^t)}\\&=\sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log P(y_j,z;\boldsymbol{\theta}^{t+1})-\sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log P(z|y_j;\boldsymbol{\theta}^t)\end{aligned}\tag{59}$$

其中，熵 $H\left(P(Z|y_j;\boldsymbol{\theta}^t)\right)=-\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log P(z|y_j;\boldsymbol{\theta}^t)$。从数据集的角度看，其熵为 $H\left(P(Z|\boldsymbol{Y};\boldsymbol{\theta}^t)\right)=-\sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log P(z|y_j;\boldsymbol{\theta}^t)$，其中 $\boldsymbol{Y}$ 表示观测数据集或不完全数据集。

为了表达方便，引入如下符号：Z 表示隐 (随机) 变量，$\boldsymbol{Y}$ 表示观测数据集，令 $\tilde{P}(Z;\boldsymbol{\theta}^t)=P(Z|\boldsymbol{Y};\boldsymbol{\theta}^t)$ 表示 Z 服从的分布。在不引起歧义的情况下，将公式 (59) 简化为[12]：

$$\begin{aligned}B(\boldsymbol{\theta}^{t+1},\boldsymbol{\theta}^t)&=\sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log P(y_j,z;\boldsymbol{\theta}^{t+1})-\sum_{j=1}^{N}\sum_{z}P(z|y_j;\boldsymbol{\theta}^t)\log P(z|y_j;\boldsymbol{\theta}^t)\\&=\mathbb{E}_{P\left(Z|\boldsymbol{Y};\boldsymbol{\theta}^t\right)}\log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}^{t+1}\right)+H\left(P\left(Z|\boldsymbol{Y};\boldsymbol{\theta}^t\right)\right)\\&=\mathbb{E}_{\tilde{P}\left(Z;\boldsymbol{\theta}^t\right)}\log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}^{t+1}\right)+H\left(\tilde{P}\left(Z;\boldsymbol{\theta}^t\right)\right)\end{aligned}\tag{60}$$

实际上，上式最后一行就是 F 函数。于是，问题就变成：给定 F 函数，在什么条件下，隐变量 Z 的分布满足：$\tilde{P}(Z;\boldsymbol{\theta}^t)=P(Z|\boldsymbol{Y};\boldsymbol{\theta}^t)$。

定义 6.1（*F* 函数）假设隐变量 Z 服从分布 $\tilde{P}(Z)$，F 函数定义为：

$$F(\tilde{P},\boldsymbol{\theta})=\mathbb{E}_{\tilde{P}}\left[\log P(\boldsymbol{Y},Z;\boldsymbol{\theta})\right]+H(\tilde{P})\tag{61}$$

其中 $H(\tilde{P})=-\mathbb{E}_{\tilde{P}}\left[\log\tilde{P}(Z)\right]$ 是分布 $\tilde{P}(Z)$ 的熵。

一般情况下，假设 $P(\boldsymbol{Y},Z;\boldsymbol{\theta})$ 是 $\boldsymbol{\theta}$ 的连续函数，因而 $F(\tilde{P},\boldsymbol{\theta})$ 也是 $\tilde{P}$ 和 $\boldsymbol{\theta}$ 的连续函数。

[12]应该把这些公式理解为：对大写字母所表示的随机变量或数据集，按照它们的具体取值逐一按相应的公式求值叠加。

引理 6.1 当 $\boldsymbol{\theta}$ 固定时，关于 $\tilde{P}$ 极大化求解 $F(\tilde{P},\boldsymbol{\theta})$，将得到唯一的最优分布 $\tilde{P}^*$：

$$\tilde{P}^*(Z) = P(Z|\boldsymbol{Y};\boldsymbol{\theta}) \tag{62}$$

并且 $\tilde{P}^*$ 随 $\boldsymbol{\theta}$ 连续变化。

证明：这是一个带约束的最优化问题，列出拉格朗日优化函数如下[13]：

$$\begin{aligned} L(\tilde{P}) &= -\mathbb{E}_{\tilde{P}}\left[\log P(\boldsymbol{Y},Z;\boldsymbol{\theta})\right] - H(\tilde{P}) + \lambda\left(\sum_Z \tilde{P}(Z) - 1\right) \\ &= -\sum_Z \tilde{P}(Z)\log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right) + \sum_Z \tilde{P}(Z)\log\tilde{P}(Z) + \lambda\left(\sum_Z \tilde{P}(Z) - 1\right) \end{aligned} \tag{63}$$

其中 λ 为拉格朗日乘数。求解 $\frac{\partial L(\tilde{P})}{\partial \tilde{P}}$：

$$\begin{aligned} &\frac{\partial L(\tilde{P})}{\partial \tilde{P}} = -\log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right) + \left(\log\tilde{P}(Z)+1\right) + \lambda = 0 \quad \Rightarrow \\ &\log\tilde{P}(Z) = \log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right) - (1+\lambda) \quad \Rightarrow \\ &\tilde{P}(Z) = \frac{P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right)}{e^{1+\lambda}} \end{aligned} \tag{64}$$

利用公式 $\sum_Z \tilde{P}(Z) = 1$, 对上式两端关于 Z 求和，得到：

$$e^{1+\lambda} = \sum_Z P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right) \tag{65}$$

于是，得到：

$$\begin{aligned} &\tilde{P}^*(Z) = \frac{P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right)}{\sum_Z P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right)} = \frac{P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right)}{P\left(\boldsymbol{Y};\boldsymbol{\theta}\right)} \quad \Rightarrow \\ &\tilde{P}^*(Z) = P(Z|\boldsymbol{Y};\boldsymbol{\theta}) \end{aligned} \tag{66}$$

由假设 $P(\boldsymbol{Y},Z;\boldsymbol{\theta})$ 是 $\boldsymbol{\theta}$ 的连续函数，因而 $\tilde{P}^*(Z)$ 也是 $\boldsymbol{\theta}$ 的连续函数。 □

从上面的推导可以看出，对于参数 $\boldsymbol{\theta}$，在考虑时序关系 t 的情况下，F 函数 (定义6.1) 关于 $\tilde{P}(Z)$ 的最优化，使得 F 函数的第 1 项实际上成为了 Q 函数[14]：

$$\begin{aligned} \mathbb{E}_{\tilde{P}^*}\left[\log P(\boldsymbol{Y},Z;\boldsymbol{\theta}^{t+1})\right] &= \sum_Z \tilde{P}^*(Z)\log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}^{t+1}\right) \\ &= \sum_Z P(Z|\boldsymbol{Y};\boldsymbol{\theta}^t)\log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}^{t+1}\right) \end{aligned} \tag{67}$$

[13] 按照惯例，转换为等价的最小化问题。

[14] 参看 Q 函数的定义2.1。更进一步地，此时，Q 函数关于参数 $\boldsymbol{\theta}^{t+1}$ 的最优化将等价于 F 函数关于参数 $\boldsymbol{\theta}^{t+1}$ 的最优化，因为 F 函数的第 2 项对于参数 $\boldsymbol{\theta}^{t+1}$ 的最优化没有影响。

另一方面，对于参数 $\boldsymbol{\theta}$，如果不考虑时序关系 t，那么将 $\tilde{P}^*(Z)=P(Z|\boldsymbol{Y};\boldsymbol{\theta})$ 代入 F 函数 (定义6.1) 后，得到：

$$\begin{aligned}
F(\tilde{P}^*,\boldsymbol{\theta}) &= \mathbb{E}_{\tilde{P}^*}\left[\log P(\boldsymbol{Y},Z;\boldsymbol{\theta})\right]+H(\tilde{P}^*) \\
&= \sum_Z \tilde{P}^*(Z)\log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right)-\sum_Z \tilde{P}^*(Z)\log \tilde{P}^*(Z) \\
&= \sum_Z P(Z|\boldsymbol{Y};\boldsymbol{\theta})\log P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right)-\sum_Z P(Z|\boldsymbol{Y};\boldsymbol{\theta})\log P(Z|\boldsymbol{Y};\boldsymbol{\theta}) \\
&= \sum_Z P(Z|\boldsymbol{Y};\boldsymbol{\theta})\log \frac{P\left(\boldsymbol{Y},Z;\boldsymbol{\theta}\right)}{P(Z|\boldsymbol{Y};\boldsymbol{\theta})} \\
&= \sum_Z P(Z|\boldsymbol{Y};\boldsymbol{\theta})\log P(\boldsymbol{Y};\boldsymbol{\theta}) \\
&= \log P(\boldsymbol{Y};\boldsymbol{\theta})
\end{aligned} \tag{68}$$

上式的最后一行，得到了观测数据集 $\boldsymbol{Y}$ 的对数似然函数。

引理 **6.2** 在 F 函数关于 $\tilde{P}(Z)$ 最优化后，得到最优分布 $\tilde{P}^*(Z)=P(Z|\boldsymbol{Y};\boldsymbol{\theta})$，那么，此时的 F 函数将等价于观测数据集 $\boldsymbol{Y}$ 的对数似然函数：

$$F(\tilde{P}^*,\boldsymbol{\theta})=\log P(\boldsymbol{Y};\boldsymbol{\theta}) \tag{69}$$

下面来总结一下。在 F 函数中，有 2 类参数，即 $\tilde{P}(Z)$ 和 $\boldsymbol{\theta}$。如果针对 $\tilde{P}(Z)$ 最优化，而固定 $\boldsymbol{\theta}$，那么将得到隐变量 Z 的最优分布 $\tilde{P}^*(Z)=P(Z|\boldsymbol{Y};\boldsymbol{\theta})$，此时的 F 函数将等价于观测数据集 $\boldsymbol{Y}$ 的对数似然函数 (即不考虑参数 $\boldsymbol{\theta}$ 的时序关系，引理6.2)。然后，针对 $\boldsymbol{\theta}$ 最优化 Q 函数 (即考虑参数 $\boldsymbol{\theta}$ 的时序关系，此时 F 函数的第 1 项成为 EM 算法中的 Q 函数，公式 (67))，而固定 $\tilde{P}^*(Z)$，那么将得到与当前最优隐变量分布 $\tilde{P}^*(Z)=P(Z|\boldsymbol{Y};\boldsymbol{\theta})$ 相称的 (局部) 最优模型参数 $\boldsymbol{\theta}^*$。可以看出，F 函数的 2 阶段与 EM 算法中的 2 阶段非常相似。

实际上，F 函数中的第 1 阶段，相当于 EM 算法中的 E 步阶段，用于确定隐变量 Z 的最优分布 $\tilde{P}^*(Z)=P(Z|\boldsymbol{Y};\boldsymbol{\theta})$，它是 Q 函数的核心。F 函数的第 2 阶段，相当于 EM 算法中的 M 步阶段，用于确定模型的 (局部) 最优参数 $\boldsymbol{\theta}^*$。图6-3展示了 EM 算法与 F 函数的 2 阶段优化示意图。可以看出，在横轴方向，F 函数关于分布 $\tilde{P}(Z)$ 的极大化相当于 EM 算法的 E 步；在纵轴方向，F 函数关于模型参数 $\boldsymbol{\theta}$ 的极大化相当于 EM 算法的 M 步。实际上，下面的定理可以证明这一点。

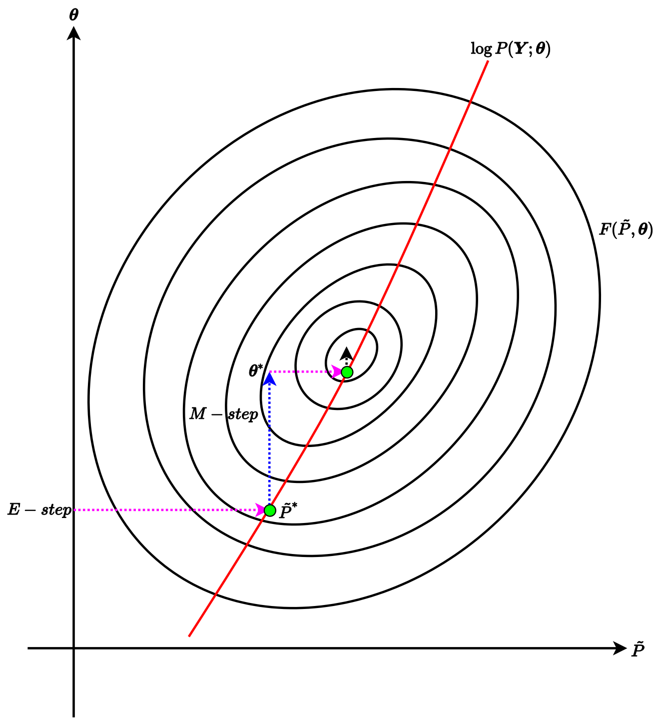


图 6-3: EM 算法的 F 函数极大-极大优化观点

定理 6.1 *EM 算法的一次 E 步与 M 步，等价于 F 函数的一次极大-极大过程。假设 $\boldsymbol{\theta}^t$ 为第 t 次迭代的模型参数，$\tilde{P}^t(Z)$ 为第 t 次迭代的隐变量分布，那么在第 $t+1$ 次迭代，2 阶段优化分别为：*

1. *固定 $\boldsymbol{\theta}^t(=\boldsymbol{\theta}^{t+1})$，关于 $\tilde{P}^t(=\tilde{P}^{t+1})$ 极大化 $F(\tilde{P}^{t+1},\boldsymbol{\theta}^{t+1})$，得到 $\tilde{P}_*^{t+1}$。此步相当于 EM 算法的 E 步；*

2. *固定 $\tilde{P}_*^{t+1}$，关于 $\boldsymbol{\theta}^{t+1}$ 极大化 $F(\tilde{P}_*^{t+1},\boldsymbol{\theta}^{t+1})$，得到 $\boldsymbol{\theta}_*^{t+1}$。此步相当于 EM 算法的 M 步；*

证明：

1. 在第 $t+1$ 次迭代的第 1 阶段，固定 $\boldsymbol{\theta}^{t+1}(=\boldsymbol{\theta}^t)$ 时，关于 $\tilde{P}^{t+1}$ 极大化 $F(\tilde{P}^{t+1},\boldsymbol{\theta}^{t+1})$，根据引理6.1，得到：

$$\tilde{P}_*^{t+1}(Z) = P(Z|\mathbf{Y};\boldsymbol{\theta}^{t+1}) = P(Z|\mathbf{Y};\boldsymbol{\theta}^t) \tag{70}$$

将其代入 F 函数，得到：

$$
\begin{aligned}
F(\tilde{P}_*^{t+1}, \boldsymbol{\theta}^{t+1}) &= \mathbb{E}_{\tilde{P}_*^{t+1}}\left[\log P(\boldsymbol{Y}, Z; \boldsymbol{\theta}^{t+1})\right] + H\left(\tilde{P}_*^{t+1}\right) \\
&= \sum_Z P(Z|\boldsymbol{Y}; \boldsymbol{\theta}^t) \log P(\boldsymbol{Y}, Z; \boldsymbol{\theta}^{t+1}) + H\left(\tilde{P}_*^{t+1}\right) \\
&= Q\left(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t\right) + H\left(\tilde{P}_*^{t+1}\right)
\end{aligned} \tag{71}
$$

由此，确定了 Q 函数，完成了 EM 算法的 E 步；

2. 在第 $t+1$ 次迭代的第 2 阶段，固定 $\tilde{P}_*^{t+1}$，关于 $\boldsymbol{\theta}^{t+1}$ 极大化 $F(\tilde{P}_*^{t+1}, \boldsymbol{\theta}^{t+1})$，得到：

$$
\boldsymbol{\theta}_*^{t+1} = \arg\max_{\boldsymbol{\theta}^{t+1}} F(\tilde{P}_*^{t+1}, \boldsymbol{\theta}^{t+1}) = \arg\max_{\boldsymbol{\theta}^{t+1}} Q(\boldsymbol{\theta}^{t+1}, \boldsymbol{\theta}^t) \tag{72}
$$

此时，$\tilde{P}_*^{t+1}$ 固定，$H(\tilde{P}_*^{t+1})$ 对 $\boldsymbol{\theta}^{t+1}$ 的优化不起作用。由此，确定了 Q 函数的 (局部) 最优模型参数 $\boldsymbol{\theta}_*^{t+1}$，完成了 EM 算法的 M 步。

因此，EM 算法的 E 步和 M 步，可以分别由 F 函数的 2 次极大过程实现。 □

EM 算法的目标是，解决给定观测数据集 $\boldsymbol{Y}$ 时模型的参数估计问题——其方法的总体思路是，针对观测数据集 $\boldsymbol{Y}$ 的对数似然函数 $L(\boldsymbol{\theta}) = \log P(\boldsymbol{Y}; \boldsymbol{\theta})$，引入隐变量 Z，进而确定 Q 函数，并利用 2 阶段迭代优化，最后完成模型参数的 (局部) 最优估计。既然 F 函数与 EM 算法之间的关系如此密切，我们自然会提出这样一个问题：对数似然函数 $L(\boldsymbol{\theta})$ 与 F 函数的最优值之间具有什么样的关系。下面的定理，建立了它们之间的关系。

定理 6.2 设 $L(\boldsymbol{\theta}) = \log P(\boldsymbol{Y}; \boldsymbol{\theta})$ 为观测数据集 $\boldsymbol{Y}$ 的对数似然函数，现使用 *EM* 算法对其进行参数估计，得到了估计序列 $\boldsymbol{\theta}^t$，其中 $t = 0, 1, \ldots, T-1$。根据定理*6.1*，与 *EM* 算法等价的 $F(\tilde{P}, \boldsymbol{\theta})$ 函数，也将得到这样的估计序列 $\boldsymbol{\theta}^t$——如果 $F(\tilde{P}, \boldsymbol{\theta})$ 在 $\tilde{P}^*$ 和 $\boldsymbol{\theta}^*$ 取得局部最优，那么 $L(\boldsymbol{\theta})$ 也在 $\boldsymbol{\theta}^*$ 取得局部最优；类似地，如果 $F(\tilde{P}, \boldsymbol{\theta})$ 在 $\tilde{P}^*$ 和 $\boldsymbol{\theta}^*$ 取得全局最优，那么 $L(\boldsymbol{\theta})$ 也在 $\boldsymbol{\theta}^*$ 取得全局最优。

证明：由引理6.1和6.2知，观测数据集 $\boldsymbol{Y}$ 的对数似然函数与 F 函数存在如下关系：

$$
F(\tilde{P}^*, \boldsymbol{\theta}) = \log P(\boldsymbol{Y}; \boldsymbol{\theta}) = L(\boldsymbol{\theta}) \quad \forall \boldsymbol{\theta} \tag{73}
$$

如果 $\boldsymbol{\theta}^*$ 为 $F(\tilde{P}^*, \boldsymbol{\theta})$ 的局部最优，那么相应地，$L(\boldsymbol{\theta})$ 也将在 $\boldsymbol{\theta}^*$ 取得局部最优；否则，令 $\boldsymbol{\theta}^{**}$ 是一个接近 $\boldsymbol{\theta}^*$ 的真正局部最优点，有 $L(\boldsymbol{\theta}^{**}) > L(\boldsymbol{\theta}^*)$ 成立，那么：

$$
F(\tilde{P}^*, \boldsymbol{\theta}^{**}) = L(\boldsymbol{\theta}^{**}) > L(\boldsymbol{\theta}^*) = F(\tilde{P}^*, \boldsymbol{\theta}^*) \tag{74}
$$

这与 $F(\tilde{P}^*, \boldsymbol{\theta}^*)$ 是局部最优相矛盾。因此，$L(\boldsymbol{\theta})$ 必将在 $\boldsymbol{\theta}^*$ 取得局部最优。

类似地，可以证明关于全局最优的结论。 $\square$

7 EM 算法实验

三硬币模型的 EM 算法验证

```
In [1]: import numpy as np
```

```
In [2]: class ThreeCoinsEM:
            def __init__(self, pi, p, q, threshold):
                self.pi = pi #A 模型, 正面朝上概率
                self.p = p #B 模型, 正面朝上概率
                self.q = q #C 模型, 正面朝上概率
                self.threshold = threshold # 算法停止阈值

            # 依据当前模型参数, 计算 B 模型对观测数据 y 的响应度,
            # 即依据 y 测算选择 B 硬币的概率
            def CalcResponse(self, Y):
                prob_b = self.pi * np.power(self.p, Y) *
                 np.power(1 - self.p, 1 - Y)
                prob_c = (1 - self.pi) * np.power(self.q, Y) *
                 np.power(1 - self.q, 1 - Y)
                return prob_b / (prob_b + prob_c)

            # 执行 E-step 和 M-step
            def fit(self, Y):
                while (True):
                    # 执行 E-step
                    response = self.CalcResponse(Y)
                    # 执行 M-step
                    new_pi = np.mean(response)
                    new_p = np.dot(response, Y) / np.sum(response)
                    new_q = np.dot(1 - response, Y) / np.sum(1 - response)
                    #EM 算法停止条件
                    max_error = max(abs(new_pi - self.pi), abs(new_p - self.p),
                     abs(new_q - self.q))
                    self.pi = new_pi
                    self.p = new_p
                    self.q = new_q
                    if max_error < self.threshold:
                        break
                print(self.pi, self.p, self.q)
                print('Three-Coins model training completed!')
```

定义观测序列

```
In [3]: Y = [1, 1, 0, 1, 0, 0, 1, 0, 1, 1]
```

测试不同的初值

```
In [4]: em = ThreeCoinsEM(0.5, 0.5, 0.5, 1e-6)
        em.fit(np.array(Y))
```

```
0.5 0.6 0.6
Three-Coins model training completed!
```

```
In [5]: em = ThreeCoinsEM(0.4, 0.6, 0.7, 1e-6)
        em.fit(np.array(Y))
```

```
0.406417112299 0.536842105263 0.643243243243
Three-Coins model training completed!
```

高斯混合模型的 EM 算法实现

```
In [6]: from scipy.stats import multivariate_normal
        import copy
```

```
In [7]: class GaussianMixtureEM:
            def __init__(self, K, threshold):
                self.K = K # 高斯混合模型中分模型的个数
                self.threshold = threshold # 算法停止阈值

            # 依据当前的模型参数，计算每个分模型 k 对观测数据 y 的响应度
            # 计算结果：K*N 矩阵
            def CalcResponse(self, Y):
                for k in range(self.K): # 遍历每个分模型
                    self.response[k] = self.alpha[k] * multivariate_normal.pdf(Y,
                     mean = self.mu[k], cov = self.cov[k])
                self.response = self.response / np.sum(self.response, axis = 0)

            # 执行 E-step 和 M-step
            def fit(self, Y):
                # 参数初始化
                N, D = Y.shape
                self.alpha = np.random.rand(self.K)
                self.alpha = self.alpha / np.sum(self.alpha)
                self.mu = np.random.rand(self.K, D)
                self.cov = np.empty((self.K, D, D))
                for k in range(self.K): # 随机生成协方差矩阵，必须是半正定矩阵
                    self.cov[k] = np.eye(D) * np.random.rand() * self.K
                self.response = np.zeros((self.K, N))
                # 保存参数，用于生成动画
                self.paras = [(copy.deepcopy(self.mu), copy.deepcopy(self.cov))]
                while (True):
                    # 执行 E-step
                    self.CalcResponse(Y)
                    # 执行 M-step
                    old_alpha = copy.deepcopy(self.alpha)
                    old_mu = copy.deepcopy(self.mu)
                    old_cov = copy.deepcopy(self.cov)
```

```
                r = np.sum(self.response, axis = 1) # 每个分模型的响应度
                for k in range(self.K): # 遍历每个分模型
                    self.mu[k] = np.dot(self.response[k], Y) / r[k]
                    self.cov[k] = np.dot(self.response[k] *
                     (Y - self.mu[k]).T, (Y - self.mu[k])) / r[k]
                self.alpha = r / N
                self.paras.append((copy.deepcopy(self.mu),
                 copy.deepcopy(self.cov))) # 保存参数，用于生成动画
                if max(np.abs(self.alpha - old_alpha).max(),
                       np.abs(self.mu - old_mu).max(),
                       np.abs(self.cov - old_cov).max()) < self.threshold:
                    break
            print('alpha:', self.alpha)
            print('\nGaussian mixture model training completed!')
```

载入数据集

```
In [8]: with open('points.dat') as file:
            allines = file.readlines()
        Y = np.array([line.strip().split() for line in allines]).
         astype(np.float32)[:500]
```

使用 EM 算法训练高斯混合模型

```
In [9]: gm = GaussianMixtureEM(4, 1e-3)
        gm.fit(Y)
```

```
alpha: [ 0.14134577  0.2932097   0.17946101  0.38598352]

Gaussian mixture model training completed!
```

绘制数据集

```
In [10]: import matplotlib.pyplot as plt
         %matplotlib inline
         %config InlineBackend.figure_format = 'svg'
```

```
In [11]: # 本函数来源于 matplotlib.mlab，在较新版本中被禁用
         # 单独列出，供使用
         def bivariate_normal(X, Y, sigmax=1.0, sigmay=1.0,
                              mux=0.0, muy=0.0, sigmaxy=0.0):
             """
             Bivariate Gaussian distribution for equal shape *X*, *Y*.
             See `bivariate normal
             <http://mathworld.wolfram.com/BivariateNormalDistribution.html>`_
             at mathworld.
             """
             Xmu = X-mux
```

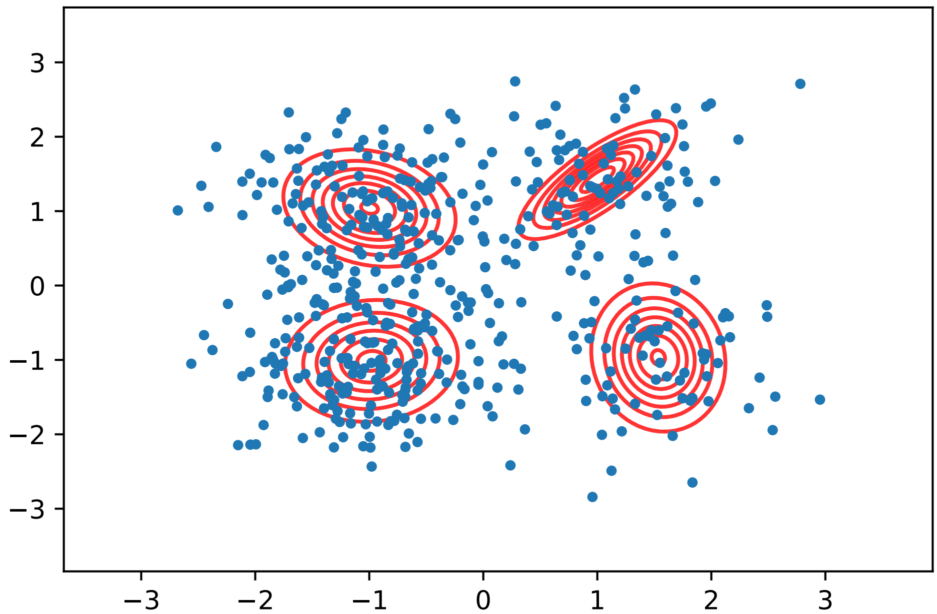
```
    Ymu = Y-muy

    rho = sigmaxy/(sigmax*sigmay)
    z = Xmu**2/
    sigmax**2 + Ymu**2/sigmay**2 - 2*rho*Xmu*Ymu/(sigmax*sigmay)
    denom = 2*np.pi*sigmax*sigmay*np.sqrt(1-rho**2)
    return np.exp(-z/(2*(1-rho**2))) / denom
```

In [12]:
```
plt.plot(Y[:, 0], Y[:, 1], '.')

# 生成绘图用的网格
x0_min, x0_max = Y[:, 0].min() - 1, Y[:, 0].max() + 1
x1_min, x1_max = Y[:, 1].min() - 1, Y[:, 1].max() + 1

xx0, xx1 = np.meshgrid(np.arange(x0_min, x0_max, 0.02),
                       np.arange(x1_min, x1_max, 0.02))
for k in range(gm.K):
    xx2 = bivariate_normal(xx0, xx1, gm.cov[k, 0, 0], gm.cov[k, 1, 1],
     gm.mu[k, 0], gm.mu[k, 1], gm.cov[k, 0, 1])
    # 检验数组元素是否都是有效数据, 避免绘制 contour 时出现警告信息
    if np.isnan(xx2).sum() == 0:
        plt.contour(xx0, xx1, xx2, colors = 'r', alpha = 0.8)
plt.savefig('EM_OUTPUT1.pdf', bbox_inches='tight')
```



绘制动画

In [13]:
```
import matplotlib.animation as animation
from IPython.display import HTML
```

```
# 动画播放，如果是%matplotlib inline，则会嵌入图片
%matplotlib notebook

fig = plt.figure()

# 生成绘图用的网格
x0_min, x0_max = Y[:, 0].min() - 1, Y[:, 0].max() + 1
x1_min, x1_max = Y[:, 1].min() - 1, Y[:, 1].max() + 1
xx0, xx1 = np.meshgrid(np.arange(x0_min, x0_max, 0.02),
                       np.arange(x1_min, x1_max, 0.02))

ax = plt.axes(xlim=(x0_min, x0_max), ylim=(x1_min, x1_max))
ax.plot(Y[:, 0], Y[:, 1], '.')
def update(para):
    ax.collections = [] # 清除原有的图元对象
    mu, cov = para[0], para[1]
    for k in range(gm.K):
        xx2 = bivariate_normal(xx0, xx1, cov[k, 0, 0], cov[k, 1, 1],
         mu[k, 0], mu[k, 1], cov[k, 0, 1])
        # 检验数组元素是否都是有效数据，避免绘制 contour 时出现警告信息
        if np.isnan(xx2).sum() == 0:
            contour = ax.contour(xx0, xx1, xx2, colors = 'r', alpha = 0.8)
    return ax.collections
ani = animation.FuncAnimation(fig, update, gm.paras)
```

生成动画

```
        # 不显示动画生成过程中出现的警告信息
In [14]: np.seterr(divide = 'ignore', invalid = 'ignore')
         ani.save('Fitting-duxiaoqin.mp4')
```

使用 sklearn.mixture.GaussianMixture 类

```
In [15]: from sklearn.mixture import GaussianMixture
```

使用数据集进行训练

```
In [16]: n_components = 4
         skgm = GaussianMixture(n_components)
         skgm.fit(Y)

Out[16]: GaussianMixture(covariance_type='full', init_params='kmeans',
                 max_iter=100, means_init=None, n_components=4, n_init=1,
                 precisions_init=None, random_state=None, reg_covar=1e-06,
                 tol=0.001, verbose=0, verbose_interval=10, warm_start=False,
                 weights_init=None)
```

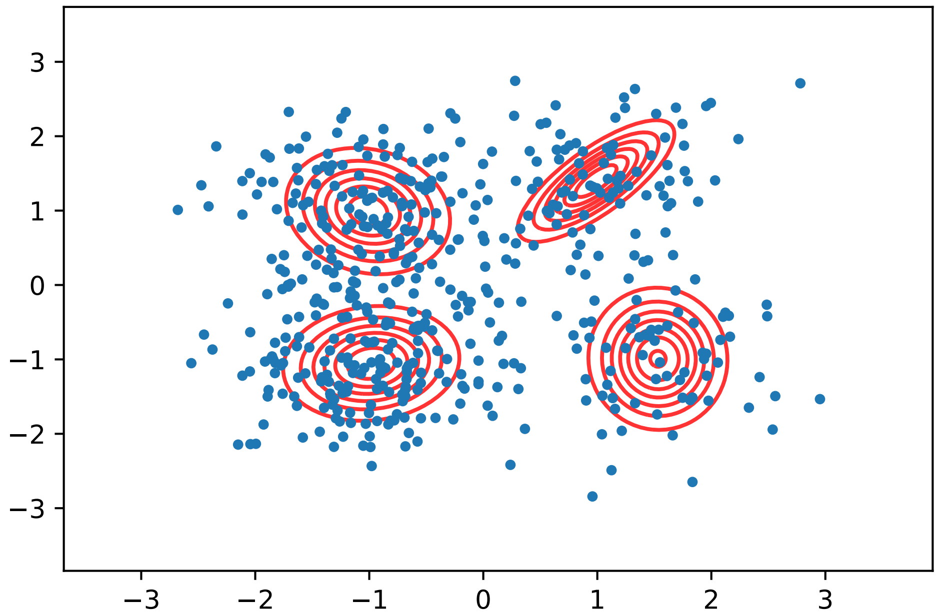
绘制训练结果

In [17]:
```
%matplotlib inline
%config InlineBackend.figure_format = 'svg'

plt.plot(Y[:, 0], Y[:, 1], '.')

# 生成绘图用的网格
x0_min, x0_max = Y[:, 0].min() - 1, Y[:, 0].max() + 1
x1_min, x1_max = Y[:, 1].min() - 1, Y[:, 1].max() + 1

xx0, xx1 = np.meshgrid(np.arange(x0_min, x0_max, 0.02),
                       np.arange(x1_min, x1_max, 0.02))
for k in range(n_components):
    xx2 = bivariate_normal(xx0, xx1, skgm.covariances_[k, 0, 0],
     skgm.covariances_[k, 1, 1], skgm.means_[k, 0], skgm.means_[k, 1],
     skgm.covariances_[k, 0, 1])
    # 检验数组元素是否都是有效数据，避免绘制 contour 时出现警告信息
    if np.isnan(xx2).sum() == 0:
        plt.contour(xx0, xx1, xx2, colors = 'r', alpha = 0.8)
plt.savefig('EM_OUTPUT2.pdf', bbox_inches='tight')
```



8 参考文献

1. Trevor Hastie, Robert Tibshirani, Jerome Friedman. The Elements of Statistical Learning, Data Mining, Inference, and Prediction. Second Edition. Springer Series in Statistics.

2. 周志华。《机器学习》，清华大学出版社，2016 年 1 月第 1 版。

3. Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer, 2006.

4. Kevin P. Murphy. Machine Learning: A Probabilistic Perspective, The MIT Press, 2012.

5. 李航。《统计学习方法》，清华大学出版社，2019 年 5 月第 2 版。

6. Michael Collins. The Naive Bayes Model, Maximum-Likelihood Estimation, and the EM Algorithm. http://www.cs.columbia.edu/ mcollins/em.pdf.

7. EM 算法与简单的三硬币模型。https://zhuanlan.zhihu.com/p/57679630.

8. https://github.com/wzyonggege/statistical-learning-method.

9. http://www.cnblogs.com/Determined22/p/5776791.html.

10. http://www.cnblogs.com/nightbreeze/p/10218117.html.

11. https://www.cs.rochester.edu/ gildea/, itanveer@cs.rochester.edu.