Author：Liu Jian

Time：2020-02-03

机器学习5-EM算法与聚类

1 EM 算法 (expectation maximization algorithm)

1.1 EM 算法的推导

EM算法：用于含有**隐变量**的概率模型参数的极大似然估计，或最大后验估计的迭代算法。个人认为 EM 算法并不能算作一种建模方法 (模型的学习策略依然是极大似然估计)，而是一种优化问题的近似、迭代的求解方法，。

X：可观测到的随机变量，样本空间 $\{q_1, q_2, \cdots, q_r\}$

Z：观测不到的随机变量，隐变量，样本空间 $\{t_1, t_2, \cdots, t_m\}$

θ：概率模型的参数。

EM 算法的背景：按照极大似然估计的流程，应该极大化完全变量 (X, Z) 的似然：$\max \log P(X, Z|\theta)$，然而 Z 为隐变量，观测不到，因此我们选择极大化"边际似然"：$\max \log P(X|\theta) = \max \log \int_Z P(X, Z|\theta) dZ$ (西瓜书中模型的引出方式)；或者，一种更直接的理解方式是，我们的目标就是极大化观测数据的似然：$\max \log P(X|\theta) = \max \log \int_Z P(X, Z|\theta) dZ$ (《统计学习方法》中模型的引出方式)。然而，因为模型 $\log P(X|\theta)$ 是通过 $\log \int_Z P(X, Z|\theta) dZ$ 这样一个积分计算得到的，形式比较复杂，因此我们会发现在对 $\log P(X|\theta)$ 进行极大化的过程中无法得到 θ 的解析求解公式，需要迭代求解 (**直接根据极大似然估计也可以构造出迭代求解的计算公式**，比如高斯混合模型，西瓜书上就是基于极大似然估计得到的迭代公式，而《统计学习方法》中则采用的 EM 法，二者得到的迭代公式一致)，为此，我们就发展出了 EM 算法，这样一种迭代求解的范式。

注：(1) 上述公式中特意采用 $\int_Z$ 而不是像《统计学习方法》上采用 $\sum_Z$ 是为了避免产生混淆：这里的意思是在 Z 的样本空间内进行积分，会遍历其所有可能的取值，而不是代入其采样数据；况且，Z 为隐变量，采样数据根本观测不到。(2) 上文提到"$\log P(X|\theta)$ 是通过 $\log \int_Z P(X, Z|\theta) dZ$ 这样一个积分计算得到的，形式比较复杂"，我们可以这样理解：$P(X|\theta)$ 就是将 $P(X, Z = t_1|\theta), P(X, Z = t_2|\theta), \cdots, P(X, Z = t_n|\theta)$ 叠加而得，因此，取对数后得到的就是 $\log(\sum)$ 的形式，在极大化目标函数时，因对数符号在叠加符号外，目标函数导数的形式就会比较复杂。(3) 上文中提到，根据极大似然估计也可以构造出迭代求解的计算公式，比如我们将对似然函数求导并令其为 0，得到参数 θ 的计算公式 $f(\theta) = 0$，但这个式子是一种"你中有我，我中有你"的形式，无法将 θ 单独分离到等号的一边得到其解析解，为此，我们可以基于 $f(\theta) = 0$ 构造一个迭代公式 $\theta_{i+1} = g(\theta_i)$，只需 $i \to +\infty$ 时，迭代公式等价于原式即 $\theta = g(\theta) \Leftrightarrow f(\theta) = 0$ 即可。

EM 算法的思路：目标是极大化似然函数 $L(\theta)$，为此根据琴生不等式构造出 $L(\theta)$ 的一个下界 $B(\theta|\theta^i)$，其中 θ^i 为一个迭代步中已知的点。**我们希望，可以通过不断提升下界 $B(\theta|\theta^i)$ 的值来不断提升 $L(\theta)$ 的值。**又因为刚好 $B(\theta^i|\theta^i) = L(\theta^i)$，则若我们找到 θ^{i+1}，使得 $B(\theta^{i+1}|\theta^i) > B(\theta^i|\theta^i)$，则这个 θ^{i+1} 也满足 $L(\theta^{i+1}) > L(\theta^i)$ ($L(\theta^{i+1}) \geqslant B(\theta^{i+1}, \theta^i) > B(\theta^i|\theta^i) = L(\theta^i)$)。因此，我们的目标就转化为不断进行如下的迭代操作直至收敛：

$$\theta^{i+1} = \arg\max_{\theta} B(\theta|\theta^i), \quad i = 0, 1, 2, \cdots$$

$$\text{with} \quad B(\theta|\theta^i) = \int_z P(z|x, \theta^i) \log \frac{P(x, z|\theta)}{P(z|x, \theta^i)} dz$$

也就等价于不断极大化如下 Q 函数：

$$\theta^{i+1} = \arg\max_{\theta} Q(\theta|\theta^i),\ \ i = 0, 1, 2, \cdots$$

$$\text{with}\ \ Q(\theta|\theta^i) = \int_z P(z|x, \theta^i) \log P(x, z|\theta) dz$$

上面只表述了对于单个样本的情况，若对于多个样本 $\boldsymbol{x} = (x_1, x_2, \cdots, x_N)$，对应隐变量 $\boldsymbol{z} = (z_1, z_2, \cdots, z_N)$，则存在两种方法得到 Q 函数：

方法1：基于上面的结论，直接代入到Q函数中，并进行化简：

$$Q(\theta|\theta^i) = \int_{\boldsymbol{z}} P(\boldsymbol{z}|\boldsymbol{x}, \theta^i) \log P(\boldsymbol{x}, \boldsymbol{z}|\theta) d\boldsymbol{z}$$

$$= \int_{\boldsymbol{z}} \left(\prod_{j=1}^{N} P(z_j|x_j, \theta^i) \right) \log \prod_{j=1}^{N} P(x_j, z_j|\theta) d\boldsymbol{z}$$

$$= \sum_{j=1}^{N} \int_{z_j} P(z_j|x_j, \theta^i) \log P(x_j, z_j|\theta) dz_j$$

上式第二行到第三行其实并不是看起来那么直接，需要对式子进行恒等变形，然后归纳得到。

因此，这种方法比较困难。

方法2：按照上面的推导，对多个样本的情形作类似的推导，可得：

$$Q(\theta|\theta^i) = \sum_{j=1}^{N} \int_{z_j} P(z_j|x_j, \theta^i) \log P(x_j, z_j|\theta) dz_j$$

这种方法更加简单。

EM 算法的步骤：

- E 步：

计算隐变量的后验分布：$P(Z|X, \theta^i)$；

得到 Q 函数：$Q(\theta|\theta^i) = \sum_{j=1}^{N} \int_{z_j} P(z_j|x_j, \theta^i) \log P(x_j, z_j|\theta) dz_j$。

- M 步：

极大化 Q 函数：$\theta^{i+1} = \arg\min_{\theta} Q(\theta|\theta^i)$。

EM 算法的收敛性：

EM 算法的收敛性包含关于对数似然函数序列 $L(\theta^i)$ 的收敛性和关于参数估计序列 θ^i 的收敛性两层含义，而且前者并不蕴含后者：

- 由上一节可知对数似然函数序列 $L(\theta^i)$ 单调递增有上界，则 $L(\theta^i)$ 收敛于上确界 L^*；
- 在 Q 函数与 $L(\theta)$ 满足一定条件下，由 EM 算法得到的参数估计序列的收敛值 θ^* 是 $L(\theta)$ 的**驻点。**

相关内容可参见文献：On the convergence properties of the EM algorith - Wu CFJ - 1983。上述定理并不能保证参数估计序列 θ^i 收敛到极大值点。EM 算法对初值的选取比较敏感，因此，**通常的做法是选取几个不同的初值进行迭代，然后对得到的各个估计值加以比较，从中选择最好的。**

1.2 EM 算法的另一种理解方式：坐标上升法

> 参考文献：A View Of The Em Algorithm That Justifies Incremental, Sparse, And Other Variants - Radford M. Neal，Geoffrey E. Hinton - 2000

对对数似然函数 $\log L(\theta)$ 进行如下变形，其中 $q(Z)$ 表示隐变量 Z 的某一分布函数，是变量：

$$\log L(\theta) = \log P(X|\theta) = \log \int_Z P(X, Z|\theta) dZ$$

$$= \log \int_Z q(Z) \frac{P(X, Z|\theta)}{q(Z)} dZ$$

$$\geqslant \int_Z q(Z) \log \left(\frac{P(X, Z|\theta)}{q(Z)} \right) dZ$$

$$(1) = \int_Z q(Z) \log \left(P(X, Z|\theta) \right) dZ - \int_Z q(Z) \log \left(q(Z) \right) dZ$$

$$= \mathbb{E}_q(\log \left(P(X, Z|\theta) \right)) + H(q)$$

$$\equiv F(q, \theta)$$

$$(2) = \int_Z q(Z) \log \left(\frac{P(Z|X, \theta) P(X|\theta)}{q(Z)} \right) dZ$$

$$= \int_Z q(Z) \log \left(\frac{P(Z|X, \theta)}{q(Z)} \right) dZ + \int_Z q(Z) \log \left(P(X|\theta) \right) dZ$$

$$= \int_Z q(Z) \log \left(\frac{P(Z|X, \theta)}{q(Z)} \right) dZ + \log \left(P(X|\theta) \right)$$

$$= - \int_Z q(Z) \log \left(\frac{q(Z)}{P(Z|X, \theta)} \right) dZ + \log \left(P(X|\theta) \right)$$

$$= -KL(q(Z)||P(Z|X, \theta)) + \log \left(P(X|\theta) \right)$$

根据上述推导 (1)，可定义 F 函数：$F(q, \theta) \equiv \mathbb{E}_q(\log \left(P(X, Z|\theta) \right)) + H(q)$，其中 $H(q) = - \int_Z q(Z) \log \left(q(Z) \right) dZ$ 是分布 $q(Z)$ 的信息熵。**F 函数中存在两个变量：分布函数 $q(Z)$ 和参数 θ，分布函数存在约束条件 $\int_Z q(Z) dZ = 1$。**为使 F 函数极大化，可采用拉格朗日函数法，其中对分布函数 $q(Z)$ 的导数如下：

$$\text{拉格朗日函数：} \mathcal{L} = \int_Z q(Z) \log \left(P(X, Z|\theta) \right) dZ - \int_Z q(Z) \log \left(q(Z) \right) dZ + \lambda \left(1 - \int_Z q(Z) dZ \right)$$

$$\text{对} q(Z) \text{求偏导：} \frac{\partial \mathcal{L}}{\partial q(Z)} = \log \left(P(X, Z|\theta) \right) - \log \left(q(Z) \right) - 1 - \lambda$$

注：积分 $\int_Z dZ$ 可看做对 Z 各种不同取值的情况进行叠加 $\sum_Z$，而对 $q(Z)$ 求偏导时不要被其所含的变量 Z 所迷惑而不知如何进行了。

事实上，对于不同的 Z，都存在相应的变量 $q(Z)$，对其中任何一个求偏导，均得到上述相同的形式。

$$\text{令偏导等于} 0\text{，可得：} q(Z) = P(X, Z|\theta) \exp \left(1 + \lambda \right)$$

$$\text{则：} \int_Z q(Z) dZ = 1 = \int_Z P(X, Z|\theta) \exp \left(1 + \lambda \right) dZ = P(X|\theta) \exp \left(1 + \lambda \right)$$

$$\text{即：} P(X|\theta) = \exp \left(-1 - \lambda \right)$$

$$\text{则：} q(Z) = P(X, Z|\theta) \exp \left(1 + \lambda \right) = \frac{P(X, Z|\theta)}{P(X|\theta)} = P(Z|X, \theta)$$

事实上由推导 (2) $F(q, \theta) = -KL(q(Z)||P(Z|X, \theta)) + \log \left(P(X|\theta) \right)$，可知，**若要极大化 F 函数，则必有 $q(Z) = P(Z|X, \theta)$，此时 $F(q, \theta) = \log \left(P(X|\theta) \right)$，极大化 F 函数也就等价于极大化对数似然函数 $\log \left(P(X|\theta) \right)$**。

为了极大化 F 函数，我们可以采用"坐标上升"这样一种近似的方法，即先固定其中的一个变量 (坐标) θ 为 θ^i，优化另一个变量 (坐标) $q(Z) = \arg\min_q F(q, \theta^i) = P(Z|X, \theta^i)$，以上就为 EM 算法的 E 步；接着，固定变量 (坐标) $q(Z) = P(Z|X, \theta^i)$，优化另一个变量 (坐标)

$$\theta^{i+1} = \arg\min_\theta F(P(Z|X, \theta^i), \theta) = \arg\min_\theta \int_Z P(Z|X, \theta^i) \log \left(\frac{P(X, Z|\theta)}{P(Z|X, \theta^i)} \right) dZ =$$

$$\arg\min_\theta \int_Z P(Z|X, \theta^i) \log \left(P(X, Z|\theta) \right) dZ = \arg\min_\theta Q(\theta|\theta^i),$$

对应就是 EM 算法的 M 步。因此，**EM 算法就等价于 F 函数的坐标上升法，也就是所谓的 F 函数的极大-极大化。算法里已知的是观察数据，未知的是隐含数据和模型参数，在 E 步，我们所做的事情是固定模型参数的值，优化隐含数据的分布；而在M步，我们所做的事情是固定隐含数据分布，优化模型参数的值。**

注意，我们特意强调了，坐标上升法只是函数极大化的一种近似方法，只是在尽可能地增大函数，并不能保证找到函数的极大值点。也就是说 F 函数的极大化与 F 函数的极大-极大化 (坐标上升法) 不等价。我们或许会产生如下的疑惑，F 函数极大化的必要条件就是对分布 q 和参数 θ 的偏导为 0；对 q 求偏导时，参数 θ 被视为不变，相当于只对坐标 q 进行极大化；而对 θ 求偏导时，分布 q 被视为不变，相当于只对坐标 θ 进行极大化。综合起来看，F 函数的极大化不就和上文阐述的 F 函数的极大-极大化 (坐标上升法) 的过程一样吗，为什么又说二者不等价呢？这里确实很容易搞混淆。注意，在 F 函数极大化的过程中，虽然求偏导时将求导变量视为变量而其他因变量视为固定值，但这些被视为固定值的因变量并没有赋给一个确定的数值，而极大-极大化 (坐标上升法) 中固定一个变量 (坐标) 优化另一个变量 (坐标) 时，被固定的变量是赋给了一个确定的数值了的，这就相当于对原函数进行了改写，得到了一个近似的只含有一个变量 (记此变量为 y) 的便于求极值的新函数，我们对这个新函数进行最优化显然与在原函数中固定其他变量，而优化变量 y (也就是令原函数对 y 的偏导为 0) 不是等价的，而是后者包含前者的关系。F 函数极大化令偏导为 0 最终得到的是描述了极值点处因变量之间关系的方程组，极值点一定满足这个方程组，而这个方程组的解即为极值的可疑点。但是，若这个方程组无法或者很难推出解析解，我们就需要采用数值解法：给定某个初始值，基于方程组迭代地计算出该初始值下函数能达到的最高点。这也就退化为我们这里的极大-极大化 (坐标上升法)，或者说极大-极大化 (坐标上升法) 是极大化问题无法推得解析解时的落地算法。显然，采用迭代法计算出的收敛点一定是方程组的解，但不一定能保证就是极大值点，因此为了提高极大值点的捕捉概率，就需要我们选择多个不同的初始值进行迭代。F 函数的极大化和极大-极大化 (坐标上升法) 的比较如下：

F 函数极大-极大化 (坐标上升法)	**F 函数极大化**
(1) $q^i = P(Z\|X,\theta^i)$ ，θ^i **为给定数值**；	(1) 对 q 偏导为 0 : $q(Z) = P(Z\|X,\theta)$ ，θ **为变量**；记这个关系为 $q = g_1(\theta)$ ，**此公式包含左边 (1) 式**；
(2) $\theta^{i+1} = \arg\min_\theta F(q^i,\theta)$ ，q^i **为已计算出的数值**；	(2) 对 θ 偏导为 0 ，得到因变量间另一个关系式，记为 $\theta = g_2(q)$ ，q **为变量，此公式包含左边 (2) 式**；
迭代算法，需要不断重复(1)(2)两步的计算。	联立(1)(2)中的方程，求解即可；**若根据方程组无法推得解析解，则可采用数值解法给出可疑点，即迭代计算：(1)** $q^i = g_1(\theta^i)$ **；(2)** $\theta^{i+1} = g_2(q^i)$ **直至收敛，此时就退化为左边的极大-极大化 (坐标上升法)。**

综上所述，极大化似然函数 $\log\left(P(X|\theta)\right)$ **与极大化** $F(q,\theta)$ **等价；而 EM 算法与** $F(q,\theta)$ **的坐标上升法 (极大-极大化) 等价。 坐标上升法并不能保证能将 F 函数** $F(q,\theta)$ **极大化，只能尽可能地使之变大，是极大化 F 函数的一种近似解法 (因为只能选择有限个迭代初始值，找到有限个极值的可疑点，无法保证捕捉到所有可疑点)；同样地，EM 算法与极大化似然函数** $\log\left(P(X|\theta)\right)$ **的关系也是如此。四者之间的关系如下**：

两个概念等价		**两个算法等价**
极大化 $\log\left(P(X\|\theta)\right)$	$\approx$	EM 算法
$\Updownarrow$		$\Updownarrow$
极大化 $F(q,\theta)$	$\approx$	坐标上升法

上一节，我们指出，是因为直接求导极大化对数似然函数很困难我们才选用得 EM 算法，因为后者计算难度相比前者更低。**事实上，我们很难从上一节的内容看出 EM 算法的计算难度更低这一结论，但现在我们就能很好地理解这一点了，因为 EM 算法就是极值问题的坐标上升法，相比于直接求极值，显然坐标上升这种近似的方法其计算难度更小。**

由本节的内容我们还可以得到如下两个普遍的结论：

- 迭代求解一个方程组其实就相当于在使用坐标上升法极大化某个函数 (或许可以称之为**势函数**)，令这个函数的偏导为 0 得到的方程组就是我们所要求解的方程组。
- 对多元函数 $f(x_1,x_2)$ ： $\max_{x_1,x_2} f(x_1,x_2) \Leftrightarrow \max_{x_2}\max_{x_1} f(x_1,x_2)$ 。但 $\max_{x_2}\max_{x_1} f(x_1,x_2)$ 并不等价于我们所说的极大-极大化算法或者说坐标上升法。因为，在计算 $\max_{x_2}\max_{x_1} f(x_1,x_2)$ 时，里层最大化时的 x_2 、外层最大化时的 x_1 虽然被视为固定的，但并没有赋给其具体的数值，本质上仍然是变量。但极大-极大化算法或者说坐标上升法在操作过程中 x_1 和 x_2 均被交替赋予了具体数值，计算得到的收敛点是最大值点的可疑点，但因为不可能计算出或者说不知道是否已经找到了所有的可疑点，因此极大-极大化算法或者说坐标上升法只能说是最大化问题的近似解。

《统计学习方法》中还给出了 GEM 算法 1、GEM 算法 2、GEM 算法 3 三种算法。其中，GEM 算法 1 就是前文所提的 EM 算法；GEM 算法 2 和 GEM 算法 3 的 E 步与 EM 算法的 E 步相同，不同的是 M 步。EM 算法中，M 步是极大化 Q 函数 $\arg\min_\theta Q(\theta|\theta^i)$ ，而 GEM 算法 2 的 M 步只是寻找到一个 θ^{i+1} 使得 $Q(\theta^i|\theta^i) > Q(\theta|\theta^i)$ 即可，并没有极大化 Q 函数；GEM 算法 3 的 M 步就是考虑若 θ 存在多个分量，则使用坐标上升法提高 Q 函数的值，也没有极大化 Q 函数 (坐标上升法并不一定能极大化 Q 函数)。

1.3 EM 算法的应用

EM算法的应用包括：

- 支持向量机的SMO算法；
- 高斯混合模型 (人为对模型进行解读，构造了隐变量)；
- k-means；
- 隐马尔可夫模型。

1.4 关于 EM 算法的思考

- 隐变量 Z 和参数 θ 的辩证关系：不考虑实际问题所赋予的意义，隐变量 Z 和参数 θ 其实都可以看做是模型参数。虽然在极大似然估计中参数被认为是某个待定的固定值，但最大后验估计中参数 θ 和隐变量 Z 一样也存在某个概率分布，因此此时隐变量 Z 和参数 θ 一样均可视为模型参数。Kevin P. Murphy 的《Machine Learning: A Probabilistic Perspective 》通篇就是从这个角度在看待一切机器学习模型。
- EM 算法与 BSPCE 算法的联系：

1. EM 算法所解决的问题：

$$\arg\max_{\theta} P(X|\theta) = \arg\min_{\theta} \int_Z P(X,Z|\theta)dZ = \arg\min_{\theta} \int_Z P(X|Z,\theta)P(Z|\theta)dZ$$

BME 所要解决的问题：

$$\arg\max_{\mathcal{A}} P(\boldsymbol{y}|\mathcal{A}) = \arg\min_{\mathcal{A}} \int_{\boldsymbol{a}} P(\boldsymbol{y},\boldsymbol{a}|\mathcal{A})d\boldsymbol{a} = \arg\min_{\mathcal{A}} \int_{\boldsymbol{a}} P(\boldsymbol{y}|\boldsymbol{a},\mathcal{A})P(\boldsymbol{a}|\mathcal{A})d\boldsymbol{a}$$

可观测变量 X 对应模型响应 $\boldsymbol{y}$，隐变量 Z 对应多项式系数 $\boldsymbol{a}$，参数 θ 对应基函数向量 $\mathcal{A}$。
二者之间是否存在着可以借鉴的地方？按 EM 算法求解 BME 的步骤如下：

E 步：计算给定参数 $\mathcal{A}$ 后隐变量 $\boldsymbol{a}$ 的后验分布 $P(\boldsymbol{a}|\boldsymbol{y},\mathcal{A}^i)$，得到 Q 函数：

$$Q(\mathcal{A}|\mathcal{A}^i) = \int_{\boldsymbol{a}} P(\boldsymbol{a}|\boldsymbol{y},\mathcal{A}^i)\log P(\boldsymbol{y},\boldsymbol{a}|\mathcal{A})d\boldsymbol{a}$$

M 步：极大化 Q 函数：

$$\mathcal{A}^{i+1} = \arg\max_{\mathcal{A}} Q(\mathcal{A}|\mathcal{A}^i)$$

显然，因为 $\mathcal{A}$ 是一个离散的集合，上述计算中需用到的模型 $P(\boldsymbol{y},\boldsymbol{a}|\mathcal{A})$ 并不好构建，因此 极大化 BME 并不能采用 EM 算法。

2. BSPCE 中 $\boldsymbol{a}$ 和 σ_ε^2 的最大后验估计 $\hat{\boldsymbol{a}}$ 和 $\hat{\sigma}_\varepsilon^2$ 也是采用迭代计算的。最大化 $\boldsymbol{a}$ 的后验概率 $p(\boldsymbol{a}|\boldsymbol{y},\sigma_\varepsilon^2,\mathcal{A})$，我们得到了关系式：$\boldsymbol{a} = g_1(\sigma_\varepsilon^2)$；最大化 σ_ε^2 的后验概率 $p(\sigma_\varepsilon^2|\boldsymbol{y},\boldsymbol{a},\mathcal{A})$，我们得到了关系式：$\sigma_\varepsilon^2 = g_2(\boldsymbol{a})$；联立这两个方程，我们发现无法解析地求解，则给定一个初值采用迭代的方法求解。根据本章的内容，更稳妥的做法是选择多个初始值进行迭代，若得到多个收敛点，则选择其中使两个后验概率最大的那个点作为参数的最大后验估计。但因为该问题实际上为凸优化问题，只存在一个收敛点，且收敛点就是最优点，因此只用选取一个初始值进行迭代即可。
3. 有人可能还会有这样的疑问：为什么在计算 $\boldsymbol{a}$ 和 σ_ε^2 的最大后验估计时，是分别最大化概率 $p(\boldsymbol{a}|\boldsymbol{y},\sigma_\varepsilon^2,\mathcal{A})$、$p(\sigma_\varepsilon^2|\boldsymbol{y},\boldsymbol{a},\mathcal{A})$ (在 BSPCE 的笔记中我们指出这等价于最大化联合后验概率 $p(\boldsymbol{a},\sigma_\varepsilon^2|\boldsymbol{y},\mathcal{A})$) 而不是 $p(\boldsymbol{a}|\boldsymbol{y},\mathcal{A}) = \int p(\boldsymbol{a}|\boldsymbol{y},\sigma_\varepsilon^2,\mathcal{A})d\sigma_\varepsilon^2$、$p(\sigma_\varepsilon^2|\boldsymbol{y},\mathcal{A}) = \int p(\sigma_\varepsilon^2|\boldsymbol{y},\boldsymbol{a},\mathcal{A})d\boldsymbol{a}$？这个问题其实是涉及到咱们的建模策略问题，上述各种策略没有谁对谁错之分，根据上述不同策略进行计算得到的结果当然也会有略微差别，不过按照策略自身的逻辑也都是正确的。正如本章开头所陈述的西瓜书和《统计学习方法》中对最优化目标 $\log P(X|\theta)$ 的两种提出方式一样，按照西瓜书的提出方式，我们最好的策略是优化 $\log P(X,Z|\theta)$ 而优化边际似然 $\log P(X|\theta)$ 给人的感觉就是一个无奈的次优选择；而按照《统计学习方法》中的提出方式，直接最优化似然 $\log P(X|\theta)$ 就很合理。对于建模策略思路的细微不同，我们不必过于纠结去判断谁对谁错，它们按照自身的逻辑都是合理的。

- 类似于 EM 算法构造了一个似然函数 $L(\theta)$ 的下界，通过不断优化这个下界来提高似然函数的值，是否可以构造一个似然函数 $L(\theta)$ 的上界，通过不断优化这个上界来提高似然函数的值？
- 知乎-EM算法存在的意义是什么？：EM算法理解的九层境界及其他回答。

2 聚类 (Clustering)

聚类是一种**无监督学习**，任务是将样本划分为若干个不相交的子集，也就是“簇”/“类”，但这些簇对应的概念语义需由使用者来把握和命名。也就是说，**聚类就是对样本数据进行分类，而和监督学习中存在明确的类别标记不同，聚类前并不存在客观的类别标记，因此，聚类所形成的各个簇 (或者说各个类) 所代表的客观含义也无从知晓，各簇的名字和含义均由人为给定。**此外，聚类簇数 k 通常需由用户提供，有一些启发式用于自动确定 k，但常用的仍是基于不同的 k 值多次运行后选取最佳结果。

2.1 性能度量和距离计算

性能度量：评估某个聚类算法在样本数据上表现的好坏。对于聚类结果，我们需要通过某种性能度量来评估其好坏；另一方面，若明确了最终要使用的性能度量，则可直接将其作为聚类过程的优化目标，从而开发出更好的聚类算法以更好地得到符合要求的聚类结果。

性能度量：

1. 外部指标：存在可供比较的“参考模型”，有 Jaccard 系数、FM 指数、Rand 指数等；
2. 内部指标：直接考察聚类结果而不利用任何参考模型，有 DB 指数、Dunn 指数等。

总而言之，聚类结果应该**簇内相似度高，而簇间相似度低。**而相似度的衡量则基于样本点间的距离，而“距离度量”是人为设定的，需要满足：(1) 非负性、(2) 同一性 (点到其自身的距离为 0)、(3) 对称性、(4) 直递性 (三角不等式)。

记 n 维 (也就是有 n 个属性) 样本点 $x=\left(x^{(1)}, x^{(2)}, \cdots, x^{(n)}\right)$，则：

1. 若属性为**有序属性**，则样本点 x_i 和 x_j 的距离：

- 闵可夫斯基距离：

$$dist_{mk}(x_i, x_j)=\left(\sum_{k=1}^{n}|x_i^k-x_j^k|^p\right)^{\frac{1}{p}}$$

当 $p=2$ 时为欧式距离、 $p=1$ 时为曼哈顿距离 (街区距离)、 $p=+\infty$ 时为切比雪夫距离

- 豪斯多夫距离：用于计算两个集合之间的距离；

2. 若属性为**无序属性** (如属性定义域为{飞机，火车，轮船}) ，则可采用 VDM (Value Difference Metric) 来衡量两个属性值 (比如飞机和火车) 之间的距离；
3. 闵可夫斯基距离和 VDM 结合可处理**混合属性**；
4. 不同属性的重要性不同时，可采用**加权距离**；

但是，我们需要注意的是，用于相似度度量的距离未必一定要满足距离度量的所有基本性质，尤其是直递性，我们将这样的距离称为**非距离度量。**此外，我们也可以基于数据样本来确定合适的距离计算式，也就是进行**距离度量学习** (distance metric learning)。距离度量学习可直接嵌入到聚类学习过程中。

2.2 原型聚类算法

原型 (prototype) 是指样本空间中具有代表性的点，基于原型的聚类算法假设聚类结构能通过一组原型刻画。

1. k-means 算法

原型向量为均值向量。设定簇的个数 k 和初始均值向量 (含有 k 个分量，每个分量为其对应簇的均值)；考察每一个样本点，将其划入距离最近的均值所对应的簇中；根据划分结果，更新均值向量，并迭代至收敛。

k-means 算法是高斯混合模型 (要求高斯混合模型的混合成分方差相等，且每个样本仅指派给一个混合成分 (硬分类)) EM 算法的特例，解读见下文高斯混合聚类。

2. 学习向量量化 (Learning Vector Quantization, LVQ)

LVQ 和 k-means 算法相似，都是将样本点划入距离最近的原型向量中，不同的是，我们在考察每个样本点时均会更新一个原型向量，而 k-means 算法中则是考察完所有点后通过计算均值来更新原型向量。假设一个样本点的属性共有 n 种，LVQ 中距离的计算只用到前 $n-1$ 个样本属性，而最后一个样本属性被称之为“标记”用于原型向量的更新：若样本点的“标记”与原型向量的“标记”相同，则原型向量以一定的学习率 η 缩短与样本点的距离，完成一次原型向量的更新；否则，增大距离。这里所说的“标记”可以理解为就是一个被选作用于原型向量更新的属性，其意义不同于监督学习中样本的标记，否则我们都知道样本标记了那就不用进行聚类了。

3. 高斯混合聚类

原型为高斯分布的均值和协方差矩阵。

高斯混合模型 (GMM)：

$$P(x) = \sum_{i=1}^{k} \alpha_i \phi_i(x|\mu_i, \Sigma_i)$$

$$\text{with} \quad \sum_{i=1}^{k} \alpha_i = 1, \ \alpha_i > 0 \ \ \text{for all } i$$

$$\phi_i(x|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{n/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right)$$

最直观的理解方式是 k 个高斯分布的加权平均，另一种解读方式是，数据 x 由第 i 个高斯分布 $\phi_i(x|\mu_i, \Sigma_i)$ 生成的概率是 α_i，考虑所有可能生成数据 x 的情况，则数据 x 出现的概率是 $\sum_{i=1}^{k} \alpha_i \phi_i(x|\mu_i, \Sigma_i)$。对于高斯混合聚类而言，我们取第二种理解方式。这样，每一个高斯模型或者说高斯混合成分 (μ_i, Σ_i) 就对应一个簇，则一共有 k 个簇。在对某个数据点 x 进行划分时，若我们只将其划入到其出现概率最大的混合成分所代表的那一个簇中：$\arg\min_i \phi_i(x|\mu_i, \Sigma_i)$，则这种方式称之为硬分类；若我们基于后验概率采用如下表述：某个样本数据有多大的概率属于某个簇，则称之为软分类 (一个数据点可属于多个混合成分)。

模型的参数 $(\alpha_i, \mu_i, \Sigma_i),\ i = 1, \cdots, k$ 的学习：

1. 西瓜书：基于极大似然估计构造迭代计算公式；
2. 《统计学习方法》：**基于对高斯混合模型的第二种理解方式，构造隐变量 Z (高斯混合模型中隐变量 Z 的取值表明数据是由哪一个高斯混合成分生成，即若某个样本点的 $Z = i$，则该样本点由第 i 个混合成分生成)，使用 EM 算法进行迭代求解。**《统计学习方法》中隐变量的含义和我们这里的表述稍有不同，更加麻烦，我们还是取这里的表述方法。这两种方法得到的迭代计算公式是相同的。

对"k-means 算法是高斯混合模型 (要求高斯混合模型的混合成分方差相等，且每个样本仅指派给一个混合成分 (硬分类)) EM 算法的特例"的证明详见知乎-高赞回答、《PRML》EM 算法那一章或论文《Convergence properties of the k-means algorithms - L Bottou, Y Bengio - 1995》，个人总结如下：

- k-means 算法中的均值向量对应高斯混合模型中的参数 μ_i；
- 当混合成分方差相等时，高斯分布的分子中可以看到欧式距离的影子，尤其是取对数后；
- 要知道高斯混合模型中隐变量 Z 的值表明数据来源于哪一个混合成分，也就是属于哪一个簇。那么可想而知，高斯混合模型 EM 算法的 E 步中根据当前参数和样本数据计算隐变量 Z 的后验分布就对应 k-means 算法中基于当前均值向量对样本数据进行分类；
- 隐变量 Z 实际使用的后验分布并不是直接取采用贝叶斯公式计算得到的后验分布，而是在此基础上进行了"加工"。假设基于贝叶斯公式，某个样本点由第 i 个混合成分生成的后验概率最大 (也就是说这个样本点到第 i 个混合成分的均值 μ_i 的距离最小)，则该样本点对应隐变量 Z 实际使用的后验分布如下：当 $Z = i$ 时概率为 1，而当 Z 取其他混合成分即 $Z \neq i$ 时概率为 0。这种类型的分布就是 dirac delta 分布，是高斯分布的一种极限。**可以看到，若隐变量 Z 实际使用的后验分布直接取贝叶斯后验，则对应的是软分类，也就是软 EM 算法，而这里是硬分类，使用的是硬 EM 算法。**可参见知乎-EM算法理解的九层境界中的第三层；
- 高斯混合模型的 M 步对应 k-means 算法中均值向量的更新。

2.3 密度聚类算法

基于密度的聚类算法假设聚类结构能够通过样本分布的紧密程度确定。

DBSCAN 算法 (Density-Based Spatial Clustering of Applications with Noise)：

给定邻域参数 (ϵ, m)，在样本数据中若一个样本点的 ϵ -邻域范围内至少包含 m 个其他的样本点，则该样本点是一个核心对象。**DBSCAN 算法所构建的聚类结构可采用 k 串葡萄来比喻**：一串葡萄表示一个簇；一串葡萄的葡萄藤由核心对象组成，一根葡萄藤上相邻两个核心对象间的距离小于 ϵ，分别从两串葡萄的葡萄藤上任取两个核心对象，则它们的距离均大于 ϵ，否则这两串葡萄就可以合并成一串葡萄了；葡萄藤上的葡萄则表示各核心对象 ϵ -邻域内的样本点。**此外，样本数据中不属于任何簇的样本点被认为是噪声或异常样本。基于上述理解，个人将 DBSCAN 算法成为"葡萄算法"。**

2.4 层次聚类算法

层次聚类试图在不同层次对数据集进行划分，从而形成**树形的聚类结构。数据集的划分可采用"自底向上"的聚合策略，也可采用"自顶向下"的分拆策略。**

AGNES (Agglomerative Nesting) 算法："自底向上"聚合类算法，它先将数据集中的每个样本看作一个初始聚类簇，然后在算法运行的每一步中找出距离最近的两个聚类簇进行合并，该过程不断重复，直至达到预设的聚类簇个数。其中，两个聚类簇之间的距离实际上就是两个集合之间的距离，可采用豪斯多夫距离进行估计。