第八章、聚类与EM算法

赵涵

2023 年 5 月 9 日

这一章，我们讨论无监督学习里面最重要的一类，**聚类**（cluster）还有如何让聚类模型更新的算法——**期望最大化算法**（Expectation Maximization）。

1 聚类

所谓聚类，就是把数据里面类似的归为一类，首先我们用一个简单的相似性标准，欧几里得距离，即越近的数据，越像（类似中国古话，物以类聚，人以群分）。我们下面介绍四种常见的聚类算法。

1.1 K-means聚类

在聚类问题，我们给顶一个训练集$\{x^{(i)},\dots,x^{(n)}\}$，我们想把这些数据点分成$k$组。这里的数据没有标签，$x^{(i)}\in\mathbb{R}^d$，所以这是一个无监督学习任务。

下面我们给出k-means聚类算法：

1. 随机初始化每一个**类别中心**（cluster centroids）的向量$\mu_1,\mu_2,\dots,\mu_k\in\mathbb{R}^d$；
2. 重复下面过程，直到收敛：

   (a) 对每一个i，求得每个数据点的归属类别：

   $$c^{(i)}:=\arg\min_j||x^{(i)}-\mu_j||^2; \tag{1}$$

   (b) 对每一个j，更新类别中心：

   $$\mu_j:=\frac{\sum_{i=1}^n\mathbb{I}\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^n\mathbb{I}\{c^{(i)}=j\}} \tag{2}$$

算法的超参数，k是类别的个数，μ_j表示第j类的类别中心。在第一步随机初始化时，我们一般随机挑选k 个训练样本，然后把这k个数据点当成k个类别的中心（也可选择其他初始化方法）。

算法的第二步是两个循环套在一个大循环里，先看第一个小循环(i)，给每一个训练样本赋值它的所属类别中心，规则是离当前样本欧氏距离最小的类别中心；(ii)，在每个数据点的类别都标记后，对当前每一类的数据点，计算新的类别中心。图（1）展示了这个过程。 在k-means算法的迭代过程，会收敛吗？答案是肯定的。我们定义如下的目标函数：

$$J(c,\mu)=\sum_{i=1}^n||x^{(i)}-\mu_{c^{(i)}}||^2 \tag{3}$$

这里J表示了每个数据点$x^{(i)}$距离$\mu_{c^{(i)}}$的距离，我们需要对目标函数求最小值，我们可以看到，我们采用的更新方法是坐标下降算法。具体来说，我们先固定类别中心μ，把其当作常数，更新类别标记c，下一步，更新类别标记，把类别中心μ固定，这种交替更新的规则，就是坐标下降，换句话说，目标函

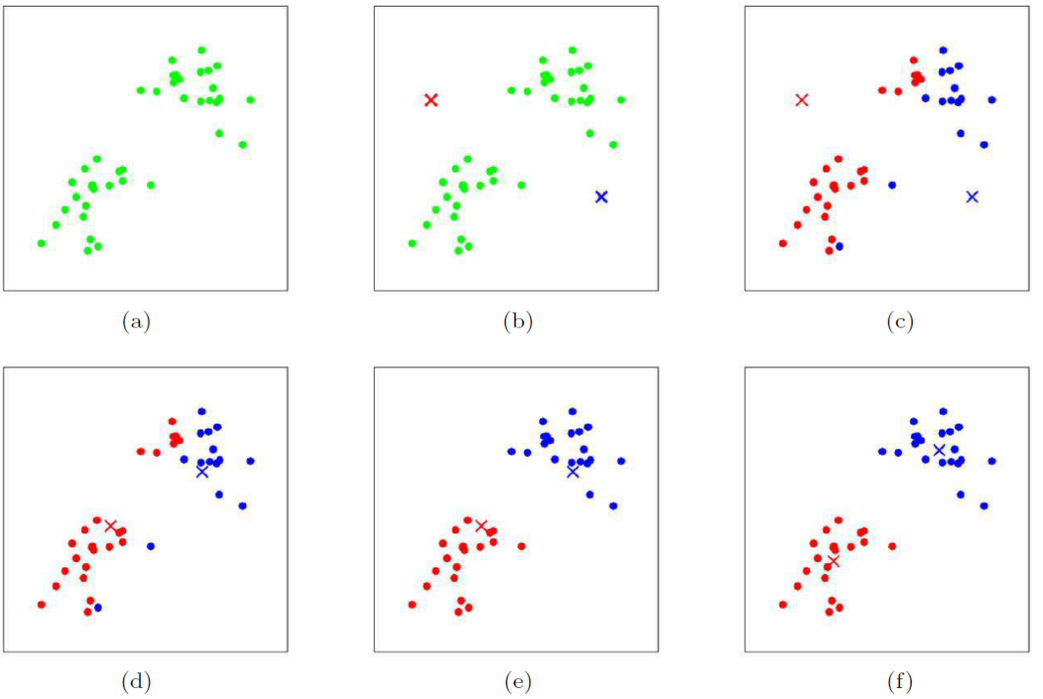

Figure 1: k-means算法。训练样本用点来表示。（a）表示原始数据集；（b）表示随机选择两个类别中心；（c-f）通过两次迭代的k-means算法。在每一步都是让数据点选择最近的类别中心标记类别记号。

数一定是单调下降的，那么算法也一定会收敛（存在多个极小值，可能收敛到不同的解，但是在一般情况下，不会发生）。

目标函数J是一个非凸函数，那就从理论讲，坐标下降不一定收敛到全局最优解。换句话说，会得到局部最优解。但是总的来说，k-means算法简单有效，是相当好的算法。如果说，你在使用时，担心算法会落入局部最优解，那么考虑多运行几次，然后计算不同解的目标函数值，选择最小目标函数对应的解。

1.2 基于密度的空间聚类

基于密度的空间聚类（DBSCAN，Density-Based Spatial Clustering of Applications with Noise），具有噪声的基于密度的聚类方法)是一种很典型的密度聚类算法。

1.2.1 密度聚类原理

DBSCAN是一种基于密度的聚类算法，这类密度聚类算法一般假定类别可以通过样本分布的紧密程度决定。同一类别的样本，他们之间的紧密相连的，也就是说，在该类别任意样本周围不远处一定有同类别的样本存在（主要思想是高内聚，低耦合）。

通过将紧密相连的样本划为一类，这样就得到了一个聚类类别。通过将所有各组紧密相连的样本划为各个不同的类别，则我们就得到了最终的所有聚类类别结果。

1.2.2 DBSCAN密度定义

接下来，我们就看看DBSCAN是如何描述密度聚类的。DBSCAN是基于一组邻域来描述样本集的紧密程度的，参数（ϵ, MinPts）用来描述邻域的样本分布紧密程度。其中，ϵ描述了某一样本的邻域距

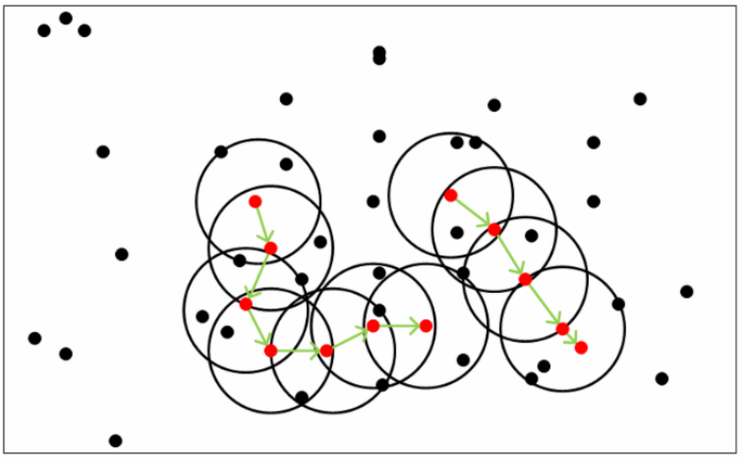

Figure 2: 密度聚类相关定义的可视化。

离阈值，MinPts 描述了某一样本的距离为ϵ的邻域中样本个数的阈值。

假设我的样本集是$D = (x_1, x_2, \ldots, x_m)$，则DBSCAN具体的密度描述定义如下：

1. ϵ-邻域：对于$x_j \in D$，其ϵ-邻域包含样本集D中与x_j的距离不大于ϵ 的子样本集，即$N_\epsilon(x_j) = \{x_i \in D | distance(x_i, x_j) \leq \epsilon\}$, 这个子样本集的个数记为$|N_\epsilon(x_j)|$。

2. 核心对象：对于任一样本$x_j \in D$，如果其ϵ-邻域对应的$N_\epsilon(x_j)$至少包含MinPts 个样本，即如果$|N_\epsilon(x_j)| \geq MinPts$，则$x_j$是核心对象。

3. 密度直达：如果x_i位于x_j的ϵ-邻域中，且x_j是核心对象，则称x_i由x_j密度直达。注意反之不一定成立，即此时不能说x_j由x_i密度直达，除非且x_i也是核心对象。

4. 密度可达：对于x_i和x_j，如果存在样本样本序列$p_1, p_2, \ldots, p_T$，满足$p_1 = x_i$,$p_T = x_j$, 且p_{t+1}由p_t密度直达，则称x_j由x_i密度可达。也就是说，密度可达满足传递性。此时序列中的传递样本$p_1, p_2, \ldots, p_{T-1}$均为核心对象，因为只有核心对象才能使其他样本密度直达。注意密度可达也不满足对称性，这个可以由密度直达的不对称性得出。

5. 密度相连：对于x_i和x_j，如果存在核心对象样本x_k，使x_i和x_j均由x_k密度可达，则称x_i和x_j密度相连。注意密度相连关系是满足对称性的。

从图（2）可以很容易看出理解上述定义，图中$MinPts = 5$，红色的点都是核心对象，因为其ϵ-邻域至少有5个样本。黑色的样本是非核心对象。所有核心对象密度直达的样本在以红色核心对象为中心的超球体内，如果不在超球体内，则不能密度直达。图中用绿色箭头连起来的核心对象组成了密度可达的样本序列。在这些密度可达的样本序列的ϵ-邻域内所有的样本相互都是密度相连的。

1.2.3 DBSCAN密度聚类思想

DBSCAN的聚类定义很简单：由密度可达关系导出的最大密度相连的样本集合，即为我们最终聚类的一个类别，或者说一个簇。

这个DBSCAN的簇里面可以有一个或者多个核心对象。如果只有一个核心对象，则簇里其他的非核心对象样本都在这个核心对象的ϵ-邻域里；如果有多个核对象，则簇里的任意一个核心对象的ϵ-邻域中一定有一个其他的核心对象，否则这两个核心对象无法密度可达。这些核心对象的ϵ-邻域里所有的样本的集合组成的一个DBSCAN聚类簇。

DBSCAN使用的方法很简单，它任意选择一个没有类别的核心对象作为种子，然后找到所有这个核心对象能够密度可达的样本集合，即为一个聚类簇。接着继续选择另一个没有类别的核心心对象去寻找密度可达的样本集合，这样就得到另一个聚类簇。一直运行到所有核心对象都有类别为止。基本上这就是DBSCAN算法的主要内容了，但是我们还是有三个问题没有考虑。

1. 一些异常样本点或者说少量游离于簇外的样本点，这些点不在任何一个核心对象在周围，在DBSCAN中，我们一般将这些样本点标记为噪音点。

2. 距离的度量问题，即如何计算某样本和核心对象样本的距离。在DBSCAN中，一般采用最近邻思想，采用某一种距离度量来衡量样本距离，比如欧式距离。对应少量的样本，寻找最近邻可以直接去计算所有样本的距离，如果样本量较大，则一般采用KD树或者球树来快速的搜索最近邻。度量方式的选择，我们在这里给出一些常用的指标：

   - 欧几里得距离：
   $$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{4}$$
   - 曼哈顿距离：
   $$\sum_{i=1}^{n}|x_i - y_i| \tag{5}$$
   - 切比雪夫距离：
   $$\max|x_i - y_i|(i = 1, 2, \ldots, n) \tag{6}$$
   - 闵可夫斯基距离：
   $$\sqrt[p]{\sum_{i=1}^{n}(w \times |x_i - y_i|)^p} \tag{7}$$
   w是特征权重；
   - 马氏距离：
   $$\sqrt{(x-y)^T S^{-1}(x-y)} \tag{8}$$
   S^{-1}为样本协方差矩阵的逆矩阵，有时也称为精度矩阵。

3. 某些样本可能到两个核心对象的距离都小于ϵ，但是这两个核心对象由于不是密度直达，又不属于同一个聚类簇，那么如果界定这个样本的类别呢。一般来说，此时DBSCAN采用先来后到，先进行聚类的类别簇会标记这个样本为它的类别。也就是说DBSCAN的算法不是完全稳定的算法。

1.2.4 DBSCAN聚类算法

我们对DBSCAN聚类算法的流程做一个总结。

1.2.5 DBSCA的总结

和传统的K-Means算法相比，DBSCAN最大的不同就是不需要输入类别数k，当然它最大的优势是可以发现任意形状的聚类簇，而不是像K-Means，一般仅仅使用于凸的样本集聚类。同时它在聚类的同时还可以找出异常点。一般来说，如果数据集是稠密的，并且数据集不是凸的，那么用DBSCAN会比K-Means聚类效果好很多。如果数据集不是稠密的，则不推荐用DBSCAN 来聚类。

DBSCAN的主要优点有：

Algorithm 1 algorithm caption

Input: 样本集$D = (x_1, x_2, \ldots, x_m)$，邻域参数$(\epsilon, MinPts)$，样本距离度量方式

Output: 簇划分C。

初始化核心对象集合$\Omega = \emptyset$，初始化聚类簇数$k = 0$，初始化未访问样本集合$\Gamma = D$，簇划分$C = \emptyset$。

for $j = 1, 2, \ldots, m$， **do**

　　通过距离度量方式，找到样本x_j的ϵ-邻域子样本集$N_\epsilon(x_j)$。

　　如果子样本集样本个数满足$|N_\epsilon(x_j)| \geq MinPts$，将样本$x_j$加入核心对象样本集合：$\Omega = \Omega \cup \{x_j\}$

如果核心对象集合$\Omega = \emptyset$，则算法结束，否则转入步骤6。

在核心对象集合Ω中，随机选择一个核心对象o，初始化当前簇核心对象队列$\Omega_{cur} = \{o\}$, 初始化类别序号$k = k + 1$，初始化当前簇样本集合$C_k = \{o\}$，更新未访问样本集合$\Gamma = \Gamma - \{o\}$。

如果当前簇核心对象队列$\Omega_{cur} = \emptyset$，则当前聚类簇C_k生成完毕，更新簇划分$C = \{C_1, C_2, \ldots, C_k\}$，更新核心对象集合$\Omega = \Omega - C_k$，转入步骤5。否则更新核心对象集合$\Omega = \Omega - C_k$。

在当前簇核心对象队列Ω_{cur}中取出一个核心对象o'，通过邻域距离阈值ϵ找出所有的ϵ-邻域子样本集$N_\epsilon(o')$，令$\Delta = N_\epsilon(o') \cap \Gamma$，更新当前簇样本集合$C_k = C_k \cup \Delta$，更新未访问样本集合$\Gamma = \Gamma - \Delta$，更新$\Omega_{cur} = \Omega_{cur} \cup (\Delta \cap \Omega) - o'$，转入步骤5。

return 簇划分$C = \{C_1, C_2, \ldots, C_k\}$。

1. 可以对任意形状的稠密数据集进行聚类，相对的，K-Means之类的聚类算法一般只适用于凸数据集。
2. 可以在聚类的同时发现异常点，对数据集中的异常点不敏感。
3. 聚类结果没有偏倚，相对的，K-Means之类的聚类算法初始值对聚类结果有很大影响。

DBSCAN的主要缺点有：

1. 如果样本集的密度不均匀、聚类间距差相差很大时，聚类质量较差，这时用DBSCAN聚类一般不适合。
2. 如果样本集较大时，聚类收敛时间较长，此时可以对搜索最近邻时建立的KD树或者球树进行规模限制来改进。
3. 调参相对于传统的K-Means之类的聚类算法稍复杂，主要需要对距离阈值ϵ，邻域样本数阈值MinPts 联合调参，不同的参数组合对最后的聚类效果有较大影响。

1.3 谱聚类

聚类问题一般可以分为两种思路：

1. 紧致型（Compactness），这类有K-means，GMM 等，但是这类算法只能处理凸集，为了处理非凸的样本集，必须引入核技巧。
2. 连接性（Connectivity），这类以谱聚类为代表。

谱聚类是一种基于无向带权图的聚类方法。图$G = (V, E)$表示，其中$V = \{1, 2, \ldots, N\}$，$E = \{w_{ij}\}$，

这里w_{ij}就是边的权重，这里权重取为相似度，$W=(w_{ij})$是相似度矩阵，定义相似度（径向核）：

$$w_{ij}=k(x_i,k_j)=\exp(-\frac{||x_i-x_j||_2^2}{2\sigma^2}),(i,j)\in E \tag{9}$$

$$w_{ij}=0,(i,j)\notin E \tag{10}$$

下面定义图的分割，这种分割就相当于聚类的结果。定义$w(A,B)$：

$$A\subset V,B\subset V,A\cap B=\emptyset,w(A,B)=\sum_{i\in A,j\in B}w_{ij} \tag{11}$$

假设一共有K个类别，对这个图的分割：

$$CUT(V)=CUT(A_1.A_2,\ldots,A_K)=\sum_{k=1}^{K}w(A_k,\overline{A_k})=\sum_{k=1}^{K}[w(A_k,V)-w(A_k,A_k)] \tag{12}$$

于是，我们的目标就是$\min_{A_k}CUT(V)$。为了平衡每一类内部的权重不同，我们做归一化的操作，定义每一个集合的度，首先，对单个节点的度定义：

$$d_i=\sum_{j=1}^{N}w_{ij} \tag{13}$$

其次，每个集合：

$$\Delta_k=degree(A_k)=\sum_{i\in A_k}d_i \tag{14}$$

于是：

$$N(CUT)=\sum_{k=1}^{k}\frac{w(A_k,\overline{A_k})}{\sum_{i\in A_k}d_i} \tag{15}$$

所以目标函数就是最小化这个式子。谱聚类的模型就是：

$$\{\hat{A}_k\}_{k=1}^{K}=\arg\min_{A_k}N(CUT) \tag{16}$$

引入指示向量：

$$y_i\in\{0,1\}^K \tag{17}$$

$$\sum_{j=1}^{K}y_{ij}=1 \tag{18}$$

其中，y_{ij}表示第i个样本属于j 个类别，记：$Y=(y_1,y_2,\ldots,y_N)^T$。所以：

$$\hat{Y}=\arg\min_{Y}N(CUT) \tag{19}$$

将$N(CUT)$写成对角矩阵的形式，于是：

$$N(CUT) = Trace[diag(\frac{w(A_1,\overline{A}_1)}{\sum_{i\in A_1}d_i},\frac{w(A_2,\overline{A}_2)}{\sum_{i\in A_2}d_i},\ldots,\frac{w(A_K,\overline{A}_K)}{\sum_{i\in A_K}d_i})] \tag{20}$$

$$= Trace[diag(w(A_1,\overline{A}_1),w(A_2,\overline{A}_2),\ldots,w(A_K,\overline{A}_K))\cdot diag(\sum_{i\in A_1}d_i,\ldots,\sum_{i\in A_K}d_i)^{-1}] \tag{21}$$

$$= Trace[O\cdot P^{-1}] \tag{22}$$

我们已经知道Y,w这两个矩阵，我们希望求得O,P。由于：

$$Y^TY=\sum_{i=1}^{N}y_iy_i^T \tag{23}$$

对于$y_i y_i^T$，只在对角线上的$k \times k$处为1，所以：

$$Y^T Y = diag(N_1, N_2, \ldots, N_K) \tag{24}$$

其中，N_i表示有N_i个样本属于i，即$N_k = \sum_{k \in A_k} 1$。引入对角矩阵，根据d_i的定义，$D = diag(d_1, d_2, \ldots, d_N) = diag(w_{NN}\mathbf{1}_{N1})$ ，于是：

$$P = Y^T DY \tag{25}$$

对另一项$O = diag(w(A_1, \overline{A}_1), w(A_2, \overline{A}_2), \ldots, w(A_K, \overline{A}_K))$：

$$O = diag(w(A_i, V)) - diag(w(A_i, A_i)) = diag(\sum_{j \in A_i} d_j) - diag(w(A_i, A_i)) \tag{26}$$

其中，第一项已知，第二项可以写成$Y^T wY$，这是由于：

$$Y^T wY = \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j^T w_{ij} \tag{27}$$

于是这个矩阵的第lm项可以写为：

$$\sum_{i \in A_i, j \in A_m} w_{ij} \tag{28}$$

这个矩阵的对角线上的项和$w(A_i, A_i)$ 相同，所以取迹后的取值不会变化。所以：

$$N(CUT) = Trace[(Y^T(D - w)Y) \cdot (Y^T DT)^{-1}] \tag{29}$$

其中，$L = D - w$叫做拉普拉斯矩阵。

1.4 杰森不等式

在介绍混合高斯模型之前，我们先引入一个有用的结论，**杰森不等式**（Jensen's inequality）。

首先定义一个函数f，定义域在整个实数域。我们说，当一个函数是凸函数，那么它的二阶导数，在整个实数域，$f''(x) \geq 0$。如果说函数的自变量是一个向量，那么凸函数的条件是海塞矩阵$H \geq 0$。如果说$f''(x) > 0$，那么f 是严格凸函数（同样对于自变量时向量的函数，海塞矩阵$H > 0$时，也是严格凸函数）。杰森不等式具有如下的表述：

定理：当f是凸函数时，X 是随机变量，那么有：

$$E[f(X)] \geq f(E[X]) \tag{30}$$

特别来说，如果说f是严格凸函数，那么$E[f(X)] = f(E[X])$当且仅当$p(X = E[X]) = 1$（换句话说，X是一个常数）。

方便起见，我们记$f(E[X]) = f(EX)$。作为定理的解释，我们给出如下的图（3），给出直观的解释。 f在图上，是一个实线，很明显是一个凸函数。X是一个随机变量，取a和b的概率分别为0.5，因此X的期望是a和b之间的中点。

在y轴上，可以看到$f(a), f(b)$和$f(EX)$之间的关系。进一步来说，$E[f(X)]$的值是$f(a)$和$f(b)$之间的中点。从上面的例子可以看出，因为函数f是凸函数，所以有：$E[f(X)] \geq f(EX)$。顺便说一句，这个不等式很多人容易记错，通过图片的形式可以帮助记忆，这也是在学习过程中，一种很好的学习方法。

如果f是凹函数，即$f''(x) \leq 0$或者$H \leq 0$。杰森不等式的形式与凸函数的形式相反，即：$E[f(X)] \leq f(EX)$。

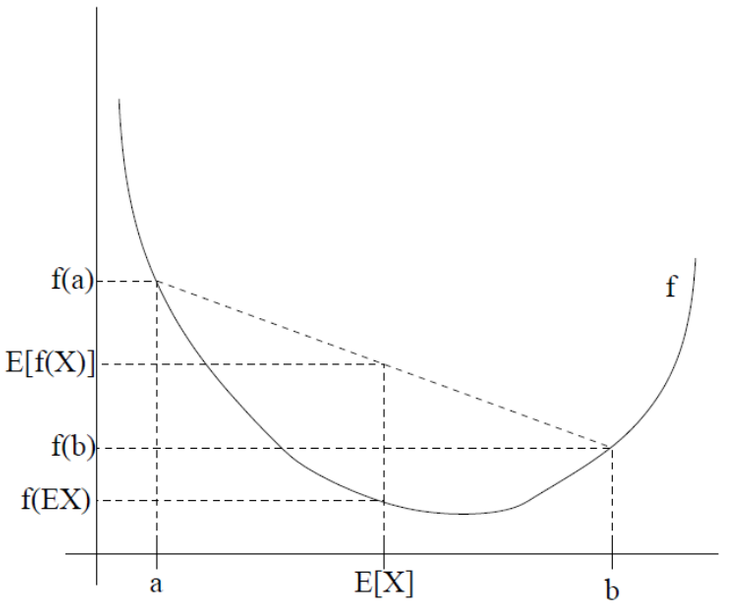

Figure 3: 对于一个凸函数，可以看出$E[f(X)]$与$f(EX)$之间的关系。

1.5 混合高斯模型

与K-means一样，给定的训练样本是$\{x^{(1)},....x^{(n)}\}$，我们将隐含类别标签用$z^{(i)}$表示。与K-means的硬指定不同，我们首先认为$z^{(i)}$ 是满足一定的概率分布的，先验认为满足多项式分布，即：$z^{(i)} \sim Multinomial(\phi)$，其中$p(z^{(i)} = j) = \phi_j, \phi_j \geq 0, \sum_{j=1}^{k} \phi_j = 1, z^{(i)}$有$k$个值$\{1, \ldots, K\}$可以选取。而且认为在给定$z^{(i)}$后，$x^{(i)}$满足多值高斯分布，即$(x^{(i)}|z^{(i)}) \sim N(\mu_j, \Sigma_j)$。由此可以得到联合分布$p(x^{(i)}, z^{(i)}) = p(x^{(i)}|z^{(i)})p(z^{(i)})$。

整个模型简单描述为对于每个样本$x^{(i)}$，我们先从K 个类别中按多项式分布抽取一个$z^{(i)}$然后根据$z^{(i)}$ 所对应的K个多值高斯分布中的一个生成样例$x^{(i)}$，。整个过程称作**混合高斯模型**（Mixtures of Gaussians）。这里的$z^{(i)}$仍然是隐含随机变量。模型中还有三个变量ϕ, μ, Σ。最大似然估计为$p(x, z)$。对数化后，有：

$$\ell(\phi, \mu, \Sigma) = \sum_{i=1}^{n} \log p(x^{(i)}; \phi, \mu, \Sigma) \tag{31}$$

$$= \sum_{i=1}^{n} \log \sum_{z^{(i)}=1}^{K} p(x^{(i)}|z^{(i)}; \mu, \Sigma)p(z^{(i)}; \phi) \tag{32}$$

这个式子的极大值是不能通过前面使用的求导数为0的方法解决的，因为求的结果不是封闭解。但是假设我们知道了单个样本的$z^{(i)}$，那么上式可以简化为：

$$\ell(\phi, \mu, \Sigma) = \sum_{i=1}^{n} \log p(x^{(i)}|z^{(i)}\mu, \Sigma) + \log p(z^{(i)}; \phi) \tag{33}$$

Figure 4: 混合高斯模型的概率图表示。

在确定下来$z^{(i)}$后，我们就能对参数进行求导，得到闭式解了，结果如下：

$$\phi_j := \frac{1}{n}\sum_{i=1}^{n}\mathbb{I}\{z^{(i)}=j\} \tag{34}$$

$$\mu_j := \frac{\sum_{i=1}^{n}\mathbb{I}\{z^{(i)}=j\}x^{(i)}}{\sum_{i=1}^{n}\mathbb{I}\{z^{(i)}=j\}} \tag{35}$$

$$\Sigma_j := \frac{\sum_{i=1}^{n}\mathbb{I}\{z^{(i)}=j\}(x^{(i)}-\mu_j)(x^{(i)}-\mu_j)^T}{\sum_{i=1}^{n}\mathbb{I}\{z^{(i)}=j\}} \tag{36}$$

ϕ_j就是样本类别中$z^{(i)}=j$的比率。μ_j是类别为j的样本特征均值，Σ_j是类别为j的样本的特征的协方差矩阵。实际上，当知道$z^{(i)}$后，最大似然估计就近似于高斯判别分析模型（Gaussian discriminant analysis model）。所不同的是GDA中类别y是伯努利分布，而这里的z是多项式分布，还有这里的每个类别都有不同的协方差矩阵，而GDA中认为只有一个。

作为一个生成式模型，高斯混合模型通过隐变量z的分布来生成样本。用概率图（4）来表示： 其中，节点z就是上面的概率，x就是生成的高斯分布。于是对一个样本来说，它的$p(x)$：

$$p(x)=\sum_z p(x,z)=\sum_{j=1}^{K}p(x,z=j)=\sum_{j=1}^{K}p(z=j)p(x|z=j) \tag{37}$$

因此有：

$$p(x)=\sum_{j=1}^{K}\phi_j N(x|\mu_j,\Sigma_j) \tag{38}$$

对于样本为$X=(x^{(1)},x^{(2)},\ldots,x^{(N)})$，$(X,Z)$为完全参数，参数为$\theta=\{\phi_1,\phi_2,\ldots,\phi_K,\mu_1,\mu_2,\ldots,\mu_K,\Sigma_1,\Sigma_2,\ldots,\Sigma_K\}$。我们通过极大似然估计得到$\theta$的值：

$$\theta_{MLE} = \arg\max_\theta \log p(X) = \arg\max_\theta \sum_{i=1}^{N}\log p(x^{(i)}) \tag{39}$$

$$= \arg\max_\theta \sum_{j=1}^{N}\log\sum_{j=1}^{K}\phi_j N(x^{(i)}|\mu_j,\Sigma_j) \tag{40}$$

这个表达式正好与我们之前提到的最大似然函数对应。直接通过求导，由于连加号的存在，无法得到解析解。因此需要使用期望最大化（Expectation Maximization）算法。这里我们先给出更新算法，作为一个结论，等到介绍了EM 后，我们在回过头来推导该算法。

- 循环下面步骤，直到收敛：
  - （E-step）对于每一个i和j，计算：

$$w_j^{(i)} := p(z^{(i)}=j|x^{(i)};\phi,\mu,\Sigma) \tag{41}$$

– （M-step）更新参数：

$$\phi_j := \frac{1}{n}\sum_{i=1}^{n} w_j^{(i)} \tag{42}$$

$$\mu_j := \frac{\sum_{i=1}^{n} w_j^{(i)} x^{(i)}}{\sum_{i=1}^{n} w_j^{(i)}} \tag{43}$$

$$\Sigma_j := \frac{\sum_{i=1}^{n} w_j^{(i)} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^{n} w_j^{(i)}} \tag{44}$$

在E步中，先将其他参数ϕ, μ, Σ看作常量，计算$z^{(i)}$的后验概率，也就是估计隐变量的具体取值。估计好后，利用上面的公式重新计算其他参数，计算好后发现最大似然估计时，$w_j^{(i)}$值又不对了，需要重新计算，周而复始，直至收敛。$w_j^{(i)}$的具体计算公式如下：

$$p(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma) = \frac{p(x^{(i)} | z^{(i)} = j; \mu, \Sigma) p(z^{(i)} = j; \phi)}{\sum_{l=1}^{k} p(x^{(i)} | z^{(i)} = l; \mu, \Sigma) p(z^{(i)} = l; \phi)} \tag{45}$$

这个式子利用了贝叶斯公式。通过使用$w_j^{(i)}$代替了前面的$\mathbb{I}\{z^{(i)} = j\}$，由简单的0/1值变成了概率值。对比K-means可以发现，这里使用了“软”指定，为每个样本分配的类别$z^{(i)}$是有一定的概率，同时计算量也变大了，每个样例i都要计算属于每一个类别j的概率。与K-means相同的是，结果仍然是局部最优解。对其他参数取不同的初始值进行多次计算不失为一种好方法。

2 最大化期望算法

期望最大算法的目的是解决具有隐变量的混合模型的参数估计（极大似然估计）。MLE对$p(x|\theta)$参数的估计记为：$\theta_{MLE} = \arg\max_\theta \log p(x|\theta)$。EM 算法对这个问题的解决方法是采用迭代的方法：

$$\theta^{t+1} = \arg\max_\theta \int_z \log[p(x, z|\theta)] p(z|x, \theta^t) dz = E_{z|x,\theta^t}[\log p(x, z|\theta)] \tag{46}$$

这个公式包含了迭代的两步：

1. E-step：计算$\log p(x, z|\theta)$在概率分布$p(z|x, \theta^t)$下的期望；
2. M-step：计算使这个期望最大化的参数，进入下一个EM步骤的输入。

接下来我们证明算法的收敛性。即证明：

$$\log p(x|\theta^t) \leq \log p(x|\theta^{t+1}) \tag{47}$$

证明：因为：

$$\log p(x|\theta) = \log p(z, x|\theta) - \log p(z|x, \theta) \tag{48}$$

对左右两边同时乘以隐变量的后验分布$p(z|x, \theta^t)$，并对隐变量z积分：

$$Left: \int_z p(z|x, \theta^t) \log p(x|\theta) dz = \log p(x|\theta) \tag{49}$$

$$Right: \int_z p(z|x, \theta^t) \log p(x, z|\theta) dz - \int_z p(z|x, \theta^t) \log p(z|x, \theta) dz = Q(\theta, \theta^t) - H(\theta, \theta^t) \tag{50}$$

所以：

$$\log p(x|\theta) = Q(\theta, \theta^t) - H(\theta, \theta^t) \tag{51}$$

由于$Q(\theta,\theta^t)=\int_z p(z|x,\theta^t)\log p(x,z|\theta)dz$，而根据方程（46）可知：$Q(\theta^{t+1},\theta^t)\geq Q(\theta^t,\theta^t)$。下面证明$\log p(x|\theta^t)\leq\log p(x|\theta^{t+1})$。需证：$H(\theta^t,\theta^t)\geq H(\theta^{t+1},\theta^t)$：

$$\begin{aligned}H(\theta^{t+1},\theta^t)-H(\theta^t,\theta^t) &= \int_z p(z|x,\theta^t)\log p(z|x,\theta^{t+1})dz-\int_z p(z|x,\theta^t)\log p(z|x,\theta^t)dz &(52)\\ &= \int p(z|x,\theta^t)\log\frac{p(z|x,\theta^{t+1})}{p(z|x,\theta^t)}dz &(53)\\ &= -KL(p(z|x,\theta^{t+1}),p(z|x,\theta^t))\leq 0 &(54)\end{aligned}$$

最后一个不等号，使用了杰森不等式。综合上面的结果：

$$\log p(x|\theta^t)\leq\log p(x|\theta^{t+1}) \tag{55}$$

根据上面的证明，我们看到，似然函数在每一步都会增大。进一步的，我们看EM迭代过程中的式子是怎么来的：

$$\log p(x|\theta)=\log p(z,x|\theta)-\log p(z|x,\theta)=\log\frac{p(z,x|\theta)}{q(z)}-\log\frac{p(z|x,\theta)}{q(z)} \tag{56}$$

之前在变分推断时，已经对此进行过介绍了。分别对两边求期望$E_{q(z)}$：

$$Left:\int_z q(z)\log p(x|\theta)dz=\log p(x|\theta) \tag{57}$$

$$Right:\int_z q(z)\log\frac{p(z,x|\theta)}{q(z)}dz-\int_z q(z)\log\frac{p(z|x,\theta)}{q(z)}dz=ELBO+KL(p(z|x,\theta),q(z)) \tag{58}$$

上式中，Evidence Lower Bound(ELBO)，是一个下界，所以$\log p(x|\theta)\geq ELBO$，等于号取在KL 散度为0，即：$q(z)=p(z|x,\theta)$，EM算法的目的是将ELBO最大化，根据上面的证明过程，在每一步EM后，求得了最大的ELBO，并根据这个使ELBO最大的参数代入下一步中：

$$\begin{aligned}\hat{\theta} &= \arg\max_\theta ELBO=\arg\max_\theta\int_z q(z)\log\frac{p(x,z|\theta)}{q(z)}dz &(59)\\ &= \arg\max_\theta ELBO=\arg\max_\theta\int_z p(z|x,\theta^t)\log\frac{p(x,z|\theta)}{p(z|x,\theta^t)}dz &(60)\\ &= \arg\max_\theta\int_z p(z|x,\theta^t)\log p(x,z|\theta)dz &(61)\end{aligned}$$

最后一个等号时因为后验分布的熵与参数θ无关，所以在求极大值时视为常数。这个公式就是上面EM迭代过程中的式子。从Jensen不等式出发，也可以导出这个公式：

$$\log p(x|\theta)=\log\int_z p(x,z|\theta)dz=\log\int_z\frac{p(x,z|\theta)q(z)}{q(z)}dz \tag{62}$$

$$\log E_{q(z)}[\frac{p(x,z|\theta)}{q(z)}]\geq E_{q(z)}[\log\frac{p(x,z|\theta)}{q(z)}] \tag{63}$$

其中，右边的式子就是ELBO，等号在对数里面的随机变量为常数时成立，即：$p(x,z|\theta)=Cq(z)$。于是：

$$\int_z q(z)dz=\frac{1}{C}\int_z p(x,z|\theta)dz=\frac{1}{C}p(x|\theta)=1 \tag{64}$$

$$\Rightarrow q(z)=\frac{1}{p(x|\theta)}p(x,z|\theta)=p(z|x,\theta) \tag{65}$$

可以发现，这个过程就是上面的最大值取等号的条件。

2.1 广义EM算法

EM 模型解决了概率生成模型的参数估计的问题，通过引入隐变量z，来学习θ，具体的模型对z有不同的假设。对学习任务$p(x|\theta)$，就是学习任务$\frac{p(x,z|\theta)}{p(z|x,\theta)}$。在这个式子中，我们假定了在E步骤中，$q(z)=p(z|x,\theta)$，但是这个$p(z|x,\theta)$如果无法求解，那么必须使用采样（MCMC）或者变分推断等方法来近似推断这个后验。我们观察KL散度的表达式，为了最大化ELBO，在固定的θ时，我们需要最小化KL散度，于是：

$$\hat{q}(z)=\arg\min_q KL(p,q)=\arg\max_q ELBO \tag{66}$$

这就是广义EM的基本思路：

1. E-step：
$$q^{t+1}(z)=\arg\max_q \int_z q^t(z)\log\frac{p(x,z|\theta)}{q^t(z)}dz,\ \theta \text{ is fixed} \tag{67}$$
2. M-step：
$$\hat{\theta}=\arg\max_\theta \int_z q^{t+1}(z)\log\frac{p(x,z|\theta)}{q^{t+1}(z)}dz,\ q^{t+1}(z) \text{ is fixed} \tag{68}$$

对于上面的积分：

$$ELBO=\int_z q(z)\log\frac{p(x,z|\theta)}{q(z)}dz=E_{q(z)}[p(x,z|\theta)]+Entropy(q(z)) \tag{69}$$

因此，我们看到，广义EM相当于在原来的式子中加入熵这一项。

2.2 EM算法的推广

EM算法类似于坐标上升法，固定部分坐标，优化其他坐标，直到迭代到收敛。如果在EM框架中，无法求解z后验概率，那么需要采用一些变种的EM来估算这个后验。

- 基于平均场的变分推断，VBEM/VEM
- 基于蒙特卡洛的EM，MCEM

2.3 EM求解高斯混合模型

下面，我们就通过GMM来具体看一下，如何求解GMM的参数。之前提到的混合高斯模型的参数μ和Σ计算公式都是根据很多假定得出的，有些没有说明来源。为了简单，这里在M步只给出ϕ和μ的推导方法。E步很简单，按照一般EM公式得到：

$$w_j^{(i)}=q_i(z^{(i)}=j)=p(z^{(i)}=j|x^{(i)};\phi,\mu,\Sigma) \tag{70}$$

简单解释就是每个样本i的隐含类别$z^{(i)}$为j的概率可以通过后验概率计算得到。在M步中，我们需要在固定$q_i(z^{(i)})$后最大对数似然估计，即：

$$\sum_{i=1}^n\sum_{z^{(i)}} q_i(z^{(i)})\log\frac{p(x^{(i)},z^{(i)};\phi,\mu,\Sigma)}{q_i(z^{(i)})} \tag{71}$$

$$=\sum_{i=1}^n\sum_{j=1}^K q_i(z^{(i)}=j)\log\frac{p(x^{(i)}|z^{(i)};\phi,\mu,\Sigma)p(z^{(i)}=j;\phi)}{q_i(z^{(i)})} \tag{72}$$

$$=\sum_{i=1}^n\sum_{j=1}^K w_j^{(i)}\log\frac{\frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}}\exp(-\frac{1}{2}(x^{(i)}-\mu_j)^T\Sigma_j^{-1}(x^{(i)}-\mu_j))\cdot\phi_j}{w_j^i} \tag{73}$$

接下来我们先计算参数μ_l的表达式。先对μ_l求偏导得：

$$\nabla_{\mu_l} \sum_{i=1}^{n} \sum_{j=1}^{K} w_j^{(i)} \log \frac{\frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} \exp(-\frac{1}{2}(x^{(i)} - \mu_j)^T \Sigma_j^{-1}(x^{(i)} - \mu_j)) \cdot \phi_j}{w_j^i} \tag{74}$$

$$= -\nabla_{\mu_l} \sum_{i=1}^{n} \sum_{j=1}^{K} w_j^{(i)} \frac{1}{2}(x^{(i)} - \mu_j)^T \Sigma_j^{-1}(x^{(i)} - \mu_j) \tag{75}$$

$$= \frac{1}{2} \sum_{i=1}^{n} w_l^{(i)} \nabla_{\mu_l}(2\mu_l^T \Sigma_l^{-1} x^{(i)} - \mu_l^T \Sigma_l^{-1} \mu_l) \tag{76}$$

$$= \sum_{i=1}^{n} w_l^{(i)}(\Sigma_l^{-1} x^{(i)} - \Sigma_l^{-1} \mu_l) \tag{77}$$

令其等于0，得到：

$$\mu_l := \frac{\sum_{i=1}^{n} w_l^{(i)} x^{(i)}}{\sum_{i=1}^{n} w_l^{(i)}} \tag{78}$$

这就是我们之前模型中的μ的更新公式。然后推导ϕ_j的更新公式。通过观察，与求ϕ导数有关的，只有：

$$\sum_{i=1}^{n} \sum_{j=1}^{K} w_j^{(i)} \log \phi_j \tag{79}$$

需要知道，ϕ_j还需要满足一定的约束条件就是$\sum_{j=1}^{K} \phi_j = 1$。这个优化问题我们很熟悉了，直接构造拉格朗日乘子，有：

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} \sum_{j=1}^{K} w_j^{(i)} \log \phi_j + \beta(\sum_{j=1}^{K} \phi_j - 1) \tag{80}$$

β是拉格朗日乘子，还有一点就是$\phi_j \geq 0$，但这一点会在得到的公式里自动满足。对拉格朗日函数求偏导数，有：

$$\frac{\partial}{\partial \phi_j} \mathcal{L}(\phi) = \sum_{i=1}^{n} \frac{w_j^{(i)}}{\phi_j} + \beta \tag{81}$$

令其等于0，得到ϕ_j的表达式：

$$\phi_j = \frac{\sum_{i=1}^{n} w_j^{(i)}}{-\beta} \tag{82}$$

从上式可以看出，$\phi_j \propto \sum_{i=1}^{n} w_j^i$，再次利用$\sum_{j=1}^{K} \phi_j = 1$，得到：

$$-\beta = \sum_{i=1}^{n} \sum_{j=1}^{K} w_j^{(i)} = \sum_{i=1}^{n} 1 = n \tag{83}$$

这里我们使用了$w_j^{(i)}$是多项式分布这一先验事实。这样就得到了β 的值。因此在M 步对ϕ_j的更新公式为：

$$\phi_j := \frac{1}{n} \sum_{i=1}^{n} w_j^{(i)} \tag{84}$$

对于Σ的更新，同样是上述方法，我们在此不做介绍了，留给读者作为练习。

References