14.1 试写出分裂聚类算法，自上而下地对数据进行聚类，并给出其算法复杂度。

书中第14章的分裂聚类介绍：

> 分裂法开始将所有样本分到一个类，之后将已有类中相距最远的样本分到两个新的类，重复此操作直到满足停止条件，得到层次化的类别

输入：n个样本组成的样本集合
输出：满足设定的样本类别数的样本集合的一个层次化聚类

1. 计算n个样本两两之间的欧氏距离$\{d_{ij}\}$，记作矩阵 $D=[d_{ij}]_{n\times n}$
2. 构造**1个**类，该类包含全部样本。
3. 在同一个簇中，计算任意两个样本之间的距离，找到距离最远的两个样本点
4. 分裂类中**距离最大**的两个样本，将其分到两个类，并设置为各自的类中心，根据样本距离将原簇划分为两个子簇
5. 如果每个类只包含一个样本，终止计算，否则回到步骤3

复杂度分析

计算距离矩阵：

- 需要计算 n 个样本两两之间的距离，时间复杂度为 $O(n^2m)$，其中m为样本的维数。

初始化一个簇：

- 初始化时将所有样本放入一个簇，时间复杂度为 $O(1)$。

在簇中找到距离最远的两个点：

- 对于一个簇，找到距离最远的两个点，需要检查簇中任意两点之间的距离，时间复杂度为 $O(k^2)$，其中 k 是簇中样本的数量。在最坏情况下，这一操作可能需要执行 $n-1$ 次（因为每次分裂减少一个簇），所以**总时间复杂度为 $O(n^3)$**。

分裂簇：

- 将一个簇分裂成两个簇，根据距离将样本划分到两个子簇，这一步对距离的检查可以在 `在簇中找到距离最远的两个点` 步骤中完成，所以不在此处作单独分析

综上所述，整个分裂聚类算法的时间复杂度为 $O(n^2(m+n))$

14.3 证明式(14.21)成立，即k均值的可能解的个数是指数级的

- 根书中第14.3.2节的描述：

  k均值聚类就是求解最优化问题：
  $$\begin{aligned} C^* &= \arg\min_C W(C) \\ &= \arg\min_C \sum_{l=1}^{k}\sum_{C(i)=l}\|x_i-\bar{x}_l\|^2 \end{aligned}$$

  相似的样本被聚到同类中，损失函数值最小，这个目标函数的最优化能达到聚类的目的。但是，这是一个组合优化问题，n个样本分到k类，所有可能分法的数目是：

  $$S(n,k)=\frac{1}{k!}\sum_{l=0}^{k}(-1)^l\binom{k}{l}(k-l)^n \tag{14.21}$$

  这个数字是指数级的。

> 《应用组合数学(第2版)》第216页5.5.3节：
> $S(n,k)$的定义是把n个可区分球分配到k个可区分盒子里，其每一个盒子不为空时的方法数量。
> 首先，考虑把n个可区分球放入标有标签$1,2,\cdots,k$的可区分盒子里，且每一个盒子都不为空的方法数量$T(n,k)$的问题，注意
>
> $$T(n,k)=k!S(n,k)$$
>
> 确定把n个可区分球放入k个不为空的不可区分盒子的分配，然后再标记（排序）这些盒子，接下来计算$T(n,k)$。
> 假设球i放入到盒子$C(i)$中，可以通过给出序列$C(1)C(2)\cdots C(n)$编码把球放入可区分盒子里的分配，这是k集合$\{1,2,\cdots,k\}$的一个n排列，且k集合中的每一个标签j至少使用一次，因此，$T(n,k)$是这一排列的数量，且对于固定的k，$T(n,k)$的指数生成函数由下式给出：
>
> $$H(x)=\left(x+\frac{x^2}{2!}+\frac{x^3}{3!}+\cdots\right)^k=(e^x-1)^k$$

> 其中$T(n,k)$是$H(x)$的展开式中$\frac{x^n}{n!}$的系数。

> 根据《应用组合数学(第2版)》第213页指数生成函数定义：
> 记$P(n,k)$是n集合的k排序的数量，固定n，则$P(n,k)$的普通生成函数由下式给出：
>
> $$\begin{aligned} G(x) &= P(n,0)x^0 + P(n,1)x^1 + P(n,2)x^2 + \cdots + P(n,n)x^n \\ &= \sum_{k=0}^{\infty} \frac{(n-k)!}{n!} x^k \end{aligned}$$
>
> 结合从n集合中取k个元素的数量$C(n,k)$，则有下面的表达式：
>
> $$C(n,0)x^0 + C(n,1)x^1 + C(n,2)x^2 + \cdots + C(n,n)x^n$$
>
> 通过二项式展开可以化简为$(1+x)^n$，则有：
>
> $$P(n,r) = C(n,r)P(r,r) = C(n,r)r!$$
>
> 根据上式可重写为
>
> $$P(n,0)\frac{x^0}{0!} + P(n,1)\frac{x^1}{1!} + P(n,2)\frac{x^2}{2!} + \cdots + P(n,n)\frac{x^n}{n!} = (1+x)^n$$
>
> 其中$P(n,k)$是$(1+x)^n$的展开式中$\frac{x^k}{k!}$的系数。
> 如果(a_k)是任意序列，则一序列的指数生成函数是
>
> $$H(x) = a_0\frac{x^0}{0!} + a_1\frac{x^1}{1!} + a_2\frac{x^2}{2!} + \cdots + a_k\frac{x^k}{k!} = \sum_k a_k \frac{x^k}{k!}$$

证明 $S(n,k)$ 的公式及其指数增长

步骤 1: $S(n,k)$ 的定义

首先，根据定义，$S(n,k)$ 表示将 n 个可区分的样本分配到 k 个非空簇的所有可能的方法数。这个数量可以通过以下公式计算：

$$S(n,k) = \frac{1}{k!}\sum_{l=0}^{k}(-1)^l \binom{k}{l}(k-l)^n$$

这个公式表示将 n 个样本分到 k 个簇的所有可能分法，去除了每个簇为空的情况。

步骤 2: $T(n,k)$ 的定义及其生成函数

为了更好地理解 $S(n,k)$，我们需要先理解 $T(n,k)$，它表示将 n 个可区分的样本分配到 k 个有标签的可区分盒子（每个盒子至少有一个样本）的分配数。根据组合数学中的定理，有：

$$T(n,k) = k!S(n,k)$$

$T(n,k)$ 的生成函数为：

$$H(x) = (e^x - 1)^k$$

> $$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$
>
> $$e^x - 1 = x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$
>
> $e^x - 1$ 中的每一项 $\frac{x^i}{i!}$ 表示将至少一个样本分配到 i 个盒子的情况。这就是为什么我们说 $e^x - 1$ 表示至少一个样本的盒子数目的情况。
> 接下来，我们将 $e^x - 1$ 的 k 次幂考虑到 k 个盒子的情况。根据乘法原理，我们可以将每个盒子的情况相乘，这样就得到了 k 个盒子的情况。因此，$(e^x - 1)^k$ 表示将 n 个可区分的样本分配到 k 个有标签的可区分盒子的所有可能的方法数。
> 综上所述，$(e^x - 1)^k$ 的生成函数中的每一项 $\frac{x^n}{n!}$ 的系数就是 $T(n,k)$ 的值，因此 $(e^x - 1)^k$ 是 $T(n,k)$ 的生成函数。
> 我们将其展开，可以得到：
>
> $$H(x) = \left(x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots\right)^k = \sum_{n=0}^{\infty} T(n,k)\frac{x^n}{n!}$$

> 这里，$T(n,k)$ 是 $H(x)$ 展开式中 $\frac{x^n}{n!}$ 的系数。

步骤 3: $T(n,k)$ 的计算

通过将 $H(x)$ 的展开式与组合数学中的幂级数展开结合，可以得到：

$$H(x)=\sum_{i=0}^{k}\binom{k}{i}(-1)^i e^{(k-i)x}$$

将 $e^{(k-i)x}$ 展开并代入上式，可以得到：

$$H(x)=\sum_{i=0}^{k}\binom{k}{i}(-1)^i\sum_{n=0}^{\infty}\frac{(k-i)^n x^n}{n!}$$

$$H(x)=\sum_{n=0}^{\infty}\frac{x^n}{n!}\sum_{i=0}^{k}\binom{k}{i}(-1)^i(k-i)^n$$

步骤 4: $T(n,k)$ 的系数

从上式中可以看出，$T(n,k)$ 是 $H(x)$ 的展开式中 $\frac{x^n}{n!}$ 的系数：

$$T(n,k)=\sum_{i=0}^{k}\binom{k}{i}(-1)^i(k-i)^n$$

步骤 5: 结合 $T(n,k)=k!S(n,k)$

将 $T(n,k)$ 与 $k!S(n,k)$ 结合，得到：

$$S(n,k)=\frac{1}{k!}\sum_{i=0}^{k}\binom{k}{i}(-1)^i(k-i)^n$$

步骤 6: 指数级增长

为了证明 $S(n,k)$ 是指数级的，我们可以观察其公式中的主要项：

$$S(n,k)=\frac{1}{k!}\left[k^n-\binom{k}{1}(k-1)^n+\binom{k}{2}(k-2)^n-\cdots+(-1)^k\right]$$

在这个公式中，最大的一项是 $\frac{k^n}{k!}$。忽略较小的项，可以得到：

$$S(n,k)\approx\frac{k^n}{k!}$$

因为 $k!$ 是一个常数，对 n 而言是一个常数，因此 $S(n,k)$ 主要由 k^n 决定，是指数级的。即：

$$S(n,k)=O\left(\frac{k^n}{k!}\right)=O(k^n)$$

14.4 比较k均值聚类与高斯混合模型加EM算法的异同。

相同点

1. 迭代求解方式：k均值聚类算法和高斯混合模型的EM算法都采用迭代求解的方式。
2. 可能收敛到局部最优解：由于这两种算法都是基于迭代优化的方法，它们都可能在局部最优解处停止，而无法保证得到全局最优解。

不同点

1. 软聚类 vs. 硬聚类：k均值聚类是一种硬聚类算法，即每个样本只能被分配到一个类别中；而高斯混合模型是一种软聚类算法，每个样本可以以一定的概率分布属于不同的类别
2. 待估计的参数不同：在k均值聚类中，需要更新的参数是各个聚类的中心（均值）；而在高斯混合模型中，需要更新的参数是每个高斯分布的均值、方差以及每个分布的权重。
3. 损失函数不同：k均值聚类算法通常采用欧式距离作为损失函数，而高斯混合模型的EM算法通常采用对数似然函数作为损失函数。