8. 聚类

是非监督学习任务，它并没有领域相关的学习目标。聚类任务的学习目标：将数据划分成不同的“簇” clusters ，并且簇应该满足：簇内样本相似，簇间样本相异。

原型聚类

假设聚类结果能够通过一组原型刻画。通常情形下，算法先对原型进行初始化，然后对原型进行迭代更新求解。

K-means

输入: $x_1, \ldots, x_n,\ x \in \mathbb{R}^d$

输出: 簇标记向量 $\boldsymbol{c}$ 以及 K个簇对应的均值向量 $\boldsymbol{\mu}$

$\boldsymbol{c} = (c_1, \ldots, c_n), \quad c_i \in \{1, \ldots, K\}$

$c_i = c_i = k$ 表明 x_i 和 x_i 同属于标记为 k的簇

$\boldsymbol{\mu} = (\mu_1, \ldots, \mu_K), \mu_k \in \mathbb{R}^d$(与 x_i所在的空间相同)

每一个均值 μ_k (也被称作类心 *centroid*) 能够作为一个簇的代表

如同讲分类或回归算法那样，我们定义一个目标函数来刻画我们对参数$\boldsymbol{c}$和$\boldsymbol{\mu}$的偏好，当然它最好便于优化求解。

cost function:

$$\mu^*, c^* = \arg\min_{\mu,c} \sum_{i=1}^{n} \sum_{k=1}^{K} I\{c_i = k\} ||x_i - \mu_k||^2 = \sum_{k=1}^{K} \sum_{i:c_i=k} ||x_i - \mu_k||^2$$

我们将变量分为c和μ 两类. 我们很难去同时优化它们，但是我们可以观察到: 固定μ 则很容易求得最优的c；固定c 则很容易求得最优的μ，使用坐标下降法

输入: $x_1, \ldots, x_n,\ x \in \mathbb{R}^d$

目标: Minimize $\mathcal{L} = \sum_{i=1}^{n} \sum_{k=1}^{K} \mathbb{1}\{c_i = k\} \|x_i - \mu_k\|^2$

随机初始化 $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_K)$.

进行下面的迭代直到收敛 ($\boldsymbol{c}$ 和 $\boldsymbol{\mu}$ 不再发生变化)

1. 更新c_i :

$$c_i = \arg\min_k \ \|x_i - \mu_k\|^2$$

2. 更新 μ_k :

$$n_k = \sum_{i=1}^{n} \mathbb{1}\{c_i = k\},\ \mu_k = \frac{1}{n_k} \sum_{i=1}^{n} x_i \mathbb{1}\{c_i = k\}$$

K-means算法不一定能够收敛到全局最优解。当c不再改变时，表示其收敛到一个局部最优解 local optimal solution. 这说明不同的随机初始化可能导致不同的聚类结果，解决这个问题的方法是...多做几次随机初始化，但是效果也不一定很好，应该采取层次聚类的方法。

K-medoids 是K-means 的一个扩展，其允许定义其他的距离替代K-means里使用的欧式距离：

邻近度函数	质　心	目标函数
曼哈顿距离（L_1）	中位数	最小化对象到其簇质心的 L_1 距离和
平方欧几里得距离（L_2^2）	均值	最小化对象到其簇质心的 L_2 距离的平方和
余弦	均值	最大化对象与其簇质心的余弦相似度和
Bregman 散度	均值	最小化对象到其簇质心的 Bregman 散度和

二分K均值

为了得到K个簇，将所有点的集合分裂成两个簇，从这些簇中选取一个继续分裂，直到产生K个簇。

算法 8.2　二分 K 均值算法

1：初始化簇表，使之包含由所有的点组成的簇。
2：**repeat**
3：　从簇表中取出一个簇。
4：　{对选定的簇进行多次二分“试验”。}
5：　**for** $i = 1$ to 试验次数 **do**
6：　　使用基本 K 均值，二分选定的簇。
7：　**end for**
8：　从二分试验中选择具有最小总 SSE 的两个簇。
9：　将这两个簇添加到簇表中。
10：**until** 簇表中包含 K 个簇。

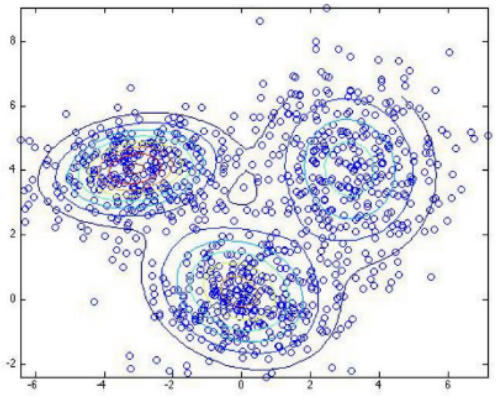
不受初始化的影响，但是一旦簇有着非球型的形状或者不同尺寸，不同密度，分类效果不好。

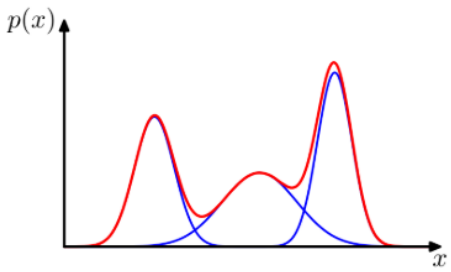
基于高斯混合模型

好了这里我意识到没学好概率论是多么窒息，这一块儿西瓜书上面讲的很明白

k-means 的结果是每个数据点被 assign 到其中某一个 cluster 了，而 GMM 则给出这些数据点被 assign 到每个 cluster 的概率（通过引入后验概率的方式，其实很像那个MLE与MAP），又称作 soft assignment 。通过将更基本的概率分布（例如高斯分布）用**线性组合**的方式进行叠加，可以被形式化为概率模型，被称为混合模型 mixture models。

例如高斯混合模型：





$$p(X) = \sum_{l=1} \alpha_l \mathcal{N}(X|\mu_l, \Sigma_l) \qquad \sum_{l=1} \alpha_l = 1$$

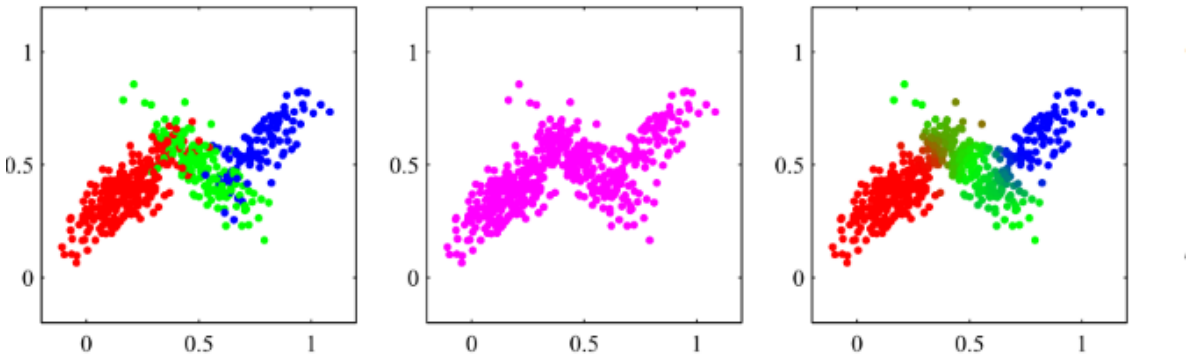
cost function

$$\theta_{ML} = \arg\max_{\theta} p(x_1, \dots x_n|\theta) = \arg\max_{\theta} \Pi_{i=1}^{n} p(x_i|\theta) = \arg\max_{\theta} \sum_{i=1}^{n} \ln p(x_i|\theta) \nabla_\theta \ln \Pi_{i=1}^{n} p(x_i|\theta) = 0$$

参考上面的图，这里通过EM算法r的角度来理解高斯混合模型。如果我们已知αk，可以依ddz照αk选择第k个高斯分模型，然后根据其概率密度函数进行采样生成相应的样本。但是现在我们拿到手的只有采样得来的样本，却不知道αk即每个样本来自于哪个分模型。随机变量zj={1，2，...，k}表示生成样本xj的高斯混合成份，zj的后验概率分布对应于：

$$\gamma_{ji} = p_m(z_j = i|x_j) = \frac{p(z_j = i) * p_m(x_j|z_j = i)}{p_m(x_j)}, \quad p(z_j = i) = \alpha_i$$
$$= \frac{\alpha_i * p(x_j|\mu_i, \Sigma_i)}{\sum_{l=1}^{k} \alpha_l * p(x_j|\mu_l, \Sigma_l)}$$

引入后验概率的直观作用：1是没有加的后验概率的，2是没有加z作为隐变量的。



求解模型参数α μ ∑：使用极大似然估计,D为给定样本集

$$LL(D) = \ln \Pi_{j=1}^{m} p_m(x_j) = \sum_{j=1}^{m} \ln \sum_{i=1}^{k} \alpha_i * p(x_j|\mu_i, \Sigma_i)$$

这里面需要对log进行求导，但是又加法最大化不好求，所以使用EM算法。

首先进行一个推导，对三个参数分别求导结果：

$$\mu_i = \frac{\sum_{j=1}^{m} \gamma_{ji} x_j}{\sum_{j=1}^{m} \gamma_{ji}}, \Sigma_i = \frac{\sum_{j=1}^{m} \gamma_{ji}(x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^{m} \gamma_{ji}}, \alpha_i = \frac{1}{m} \sum_{j=1}^{m} \gamma_{ji}$$

在每步迭代中，现根据当前参数计算每个样本属于高斯分布的后验概率（E），在根据这三个式子更新模型参数（M）。

具体如下：

输入: 样本集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$;
高斯混合成分个数 k.

过程:

初始化高斯混合分布的模型参数 $\{(\alpha_i,\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\mid 1\leqslant i\leqslant k\}$
repeat
  for $j=1,2,\ldots,m$ **do**
    根据式(9.30)计算 $\boldsymbol{x}_j$ 由各混合成分生成的后验概率, 即
    $\gamma_{ji}=p_{\mathcal{M}}(z_j=i\mid\boldsymbol{x}_j)\ (1\leqslant i\leqslant k)$
  end for
  for $i=1,2,\ldots,k$ **do**
    计算新均值向量: $\boldsymbol{\mu}_i'=\frac{\sum_{j=1}^m\gamma_{ji}\boldsymbol{x}_j}{\sum_{j=1}^m\gamma_{ji}}$;
    计算新协方差矩阵: $\boldsymbol{\Sigma}_i'=\frac{\sum_{j=1}^m\gamma_{ji}(\boldsymbol{x}_j-\boldsymbol{\mu}_i')(\boldsymbol{x}_j-\boldsymbol{\mu}_i')^{\mathrm{T}}}{\sum_{j=1}^m\gamma_{ji}}$;
    计算新混合系数: $\alpha_i'=\frac{\sum_{j=1}^m\gamma_{ji}}{m}$;
  end for
  将模型参数 $\{(\alpha_i,\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\mid 1\leqslant i\leqslant k\}$ 更新为 $\{(\alpha_i',\boldsymbol{\mu}_i',\boldsymbol{\Sigma}_i')\mid 1\leqslant i\leqslant k\}$
until 满足停止条件
$C_i=\varnothing\ (1\leqslant i\leqslant k)$
for $j=1,2,\ldots,m$ **do**
  根据式(9.31)确定 $\boldsymbol{x}_j$ 的簇标记 λ_j;
  将 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j}=C_{\lambda_j}\bigcup\{\boldsymbol{x}_j\}$
end for

输出: 簇划分 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$

K-means 是一种硬聚类 hard clustering ，即它每次只是将一个样本划分到一个 簇里，软聚类 soft clustering则能给出这个样本属于每个簇的可能性这种信息。 高斯混合模型中的“责任”可以看做是一种软聚类的簇标记。

EM算法

既然讲到了EM算法就好好说一说，EM算法是一种迭代算法，每次迭代由两步组成：E步，求期望 ，即利用当前估计的参数值来计算对数似然函数的期望值；M步，求极大，即求参数θ 来极大化E步中的期望值，而求出的参数θ将继续用于下一个E步中期望值的估计。

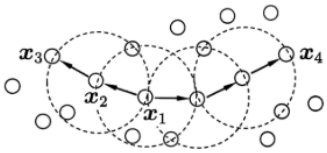
密度聚类

此类算法假设聚类结构能通过样本分布的紧密程度确定。 通常情形下，密度聚类算法从样本密度的角度来考察样本之间的可连接性，并基于可连接样本不断扩展聚类簇以获得最终的聚类结果。

DBSCAN

基本定义：

- 核心对象(core object): 若 $\boldsymbol{x}_j$ 的 ϵ-邻域至少包含 $MinPts$ 个样本, 即 $|N_\epsilon(\boldsymbol{x}_j)| \geqslant MinPts$, 则 $\boldsymbol{x}_j$ 是一个核心对象;
- ϵ-邻域: 对 $\boldsymbol{x}_j \in D$, 其 ϵ-邻域包含样本集 D 中与 $\boldsymbol{x}_j$ 的距离不大于 ϵ 的样本, 即 $N_\epsilon(\boldsymbol{x}_j) = \{\boldsymbol{x}_i \in D \mid \text{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) \leqslant \epsilon\}$;



虚线显示出 ϵ-邻域
$\boldsymbol{x}_1$ 是核心对象
$\boldsymbol{x}_2$ 由 $\boldsymbol{x}_1$ 密度直达
$\boldsymbol{x}_3$ 由 $\boldsymbol{x}_1$ 密度可达
$\boldsymbol{x}_3$ 与 $\boldsymbol{x}_4$ 密度相连

样本间的紧密程度:

- 密度直达(directly density-reachable): 若 $\boldsymbol{x}_j$ 位于 $\boldsymbol{x}_i$ 的 ϵ-邻域中, 且 $\boldsymbol{x}_i$ 是核心对象, 则称 $\boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度直达;
- 密度可达(density-reachable): 对 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$, 若存在样本序列 $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n$, 其中 $\boldsymbol{p}_1 = \boldsymbol{x}_i$, $\boldsymbol{p}_n = \boldsymbol{x}_j$ 且 $\boldsymbol{p}_{i+1}$ 由 $\boldsymbol{p}_i$ 密度直达, 则称 $\boldsymbol{x}_j$ 由 $\boldsymbol{x}_i$ 密度可达;
- 密度相连(density-connected): 对 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$, 若存在 $\boldsymbol{x}_k$ 使得 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 均由 $\boldsymbol{x}_k$ 密度可达, 则称 $\boldsymbol{x}_i$ 与 $\boldsymbol{x}_j$ 密度相连.

算法:

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
邻域参数 $(\epsilon, MinPts)$.

过程:

初始化核心对象集合: $\Omega = \varnothing$
for $j = 1, 2, \ldots, m$ **do**
  确定样本 $\boldsymbol{x}_j$ 的 ϵ-邻域 $N_\epsilon(\boldsymbol{x}_j)$;
  if $|N_\epsilon(\boldsymbol{x}_j)| \geqslant MinPts$ **then**
    将样本 $\boldsymbol{x}_j$ 加入核心对象集合: $\Omega = \Omega \bigcup \{\boldsymbol{x}_j\}$
  end if
end for
初始化聚类簇数: $k = 0$
初始化未访问样本集合: $\Gamma = D$
while $\Omega \neq \varnothing$ **do**
  记录当前未访问样本集合: $\Gamma_{\text{old}} = \Gamma$;
  随机选取一个核心对象 $\boldsymbol{o} \in \Omega$, 初始化队列 $Q = < \boldsymbol{o} >$;
  $\Gamma = \Gamma \setminus \{\boldsymbol{o}\}$;
  while $Q \neq \varnothing$ **do**
    取出队列 Q 中的首个样本 $\boldsymbol{q}$;
    if $|N_\epsilon(\boldsymbol{q})| \geqslant MinPts$ **then**
      令 $\Delta = N_\epsilon(\boldsymbol{q}) \bigcap \Gamma$;
      将 Δ 中的样本加入队列 Q;
      $\Gamma = \Gamma \setminus \Delta$;
    end if
  end while
  $k = k + 1$, 生成聚类簇 $C_k = \Gamma_{\text{old}} \setminus \Gamma$;
  $\Omega = \Omega \setminus C_k$
end while

输出: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

DBSCAN通过检查数据集中每点的Eps邻域来搜索簇，如果点p的Eps邻域包含的点多于MinPts个，则创建一个以p为核心对象的簇；然后，DBSCAN将p直接密度可达的对象加入这个簇；迭代上述过程（可能涉及一些密度可达簇的合并）；当没有新的点添加到任何簇时，该过程结束.；结果把数据集分为离群点、中心点、聚集点三类

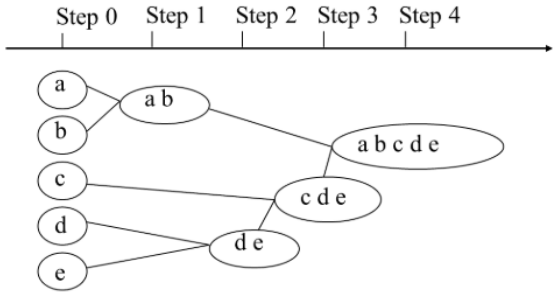
层次聚类

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。优点是不用初始确定特定数目的簇，而且可以与一些分类的含义相联系。

两种聚类方法：凝聚层次聚类（个体簇开始，每一步合并两个最近的簇），分裂层次聚类（从某个簇开始，每一步分裂一个簇，最后剩下单点簇）

凝聚：AGNES

AGNES 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。



agglomerative
(AGNES)

两个聚类簇之间的距离可以根据：

最小距离：

$$d_{\min}(C_i, C_j) = \min_{\boldsymbol{x} \in C_i, \boldsymbol{z} \in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z})$$

最大距离：

$$d_{\max}(C_i, C_j) = \max_{\boldsymbol{x} \in C_i, \boldsymbol{z} \in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z})$$

平均距离：

$$d_{\text{avg}}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{\boldsymbol{x} \in C_i} \sum_{\boldsymbol{z} \in C_j} \operatorname{dist}(\boldsymbol{x}, \boldsymbol{z})$$

最小距离也叫单链，最大距离也叫全链，平均距离也叫组平均

举个例子：

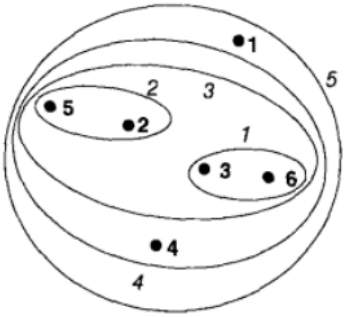
表 8-4　6 个点的欧几里得距离矩阵

	p1	p2	p3	p4	p5	p6
p1	0.0000	0.2357	0.2218	0.3688	0.3421	0.2347
p2	0.2357	0.0000	0.1483	0.2042	0.1388	0.2540
p3	0.2218	0.1483	0.0000	0.1513	0.2843	0.1100
p4	0.3688	0.2042	0.1513	0.0000	0.2932	0.2216
p5	0.3421	0.1388	0.2843	0.2932	0.0000	0.3921
p6	0.2347	0.2540	0.1100	0.2216	0.3921	0.0000

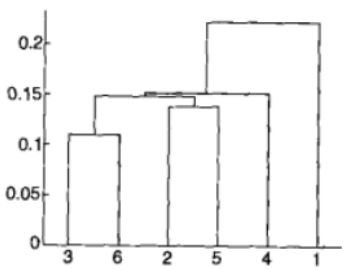
min法：

$$\begin{aligned} dist(\{3,6\},\{2,5\}) &= \min(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\ &= \min(0.15, 0.25, 0.28, 0.39) \\ &= 0.15 \end{aligned}$$

□



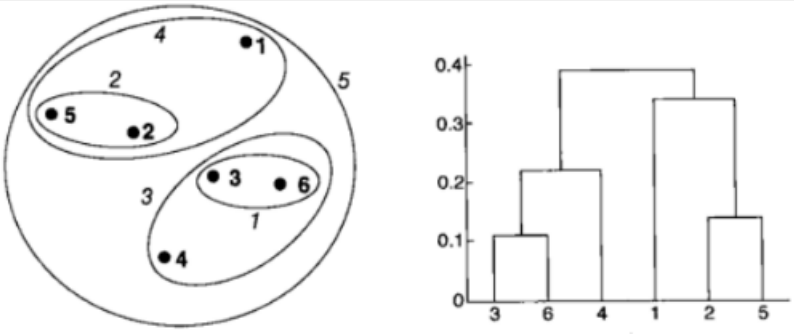
(a) 单链聚类



(b) 单链树状图

max法：

$$
\begin{aligned}
dist(\{3,6\},\{4\}) &= \max(dist(3,4), dist(6,4)) \\
&= \max(0.15, 0.22) \\
&= 0.22 \\
dist(\{3,6\},\{2,5\}) &= \max(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\
&= \max(0.15, 0.25, 0.28, 0.39) \\
&= 0.39 \\
dist(\{3,6\},\{1\}) &= \max(dist(3,1), dist(6,1)) \\
&= \max(0.22, 0.23) \\
&= 0.23
\end{aligned}
$$

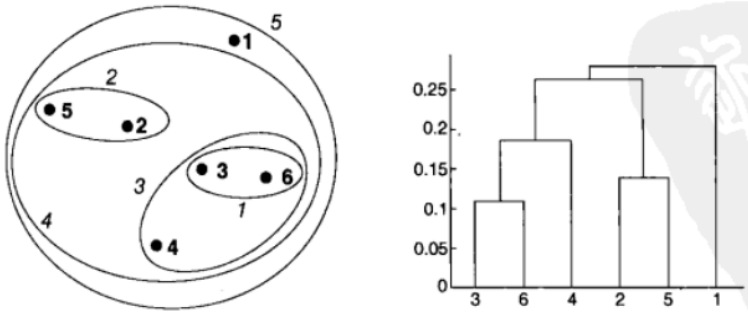


(a) 全链聚类　　(b) 全链树状图

组平均：

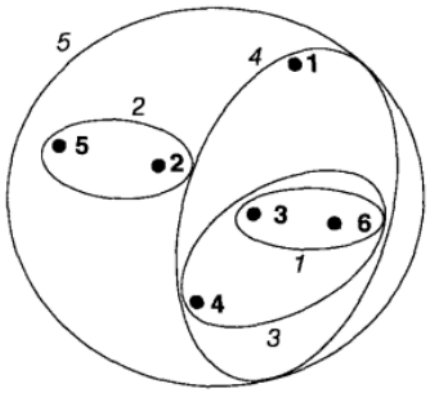
$$
\begin{aligned}
dist(\{3,6,4\},\{1\}) &= (0.22 + 0.37 + 0.23)/(3*1) \\
&= 0.28 \\
dist(\{2,5\},\{1\}) &= (0.2357 + 0.3421)/(2*1) \\
&= 0.2889 \\
dist(\{3,6,4\},\{2,5\}) &= (0.15 + 0.28 + 0.25 + 0.39 + 0.20 + 0.29)/(3*2) \\
&= 0.26
\end{aligned}
$$

因为 $dist(\{3,6,4\},\{2,5\})$比 $dist(\{3,6,4\},\{1\})$和 $dist(\{2,5\},\{1\})$小，簇{3, 6, 4}和{2, 5}在第 4 阶段合并。 □

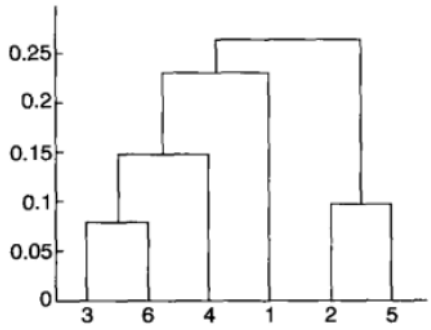


(a) 组平均聚类　　(b) 组平均树状图

质心：ward方法：两个簇合并时导致的平方误差的增量。



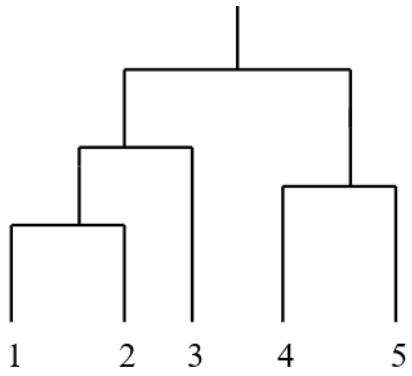
(a) Ward 聚类



(b) Ward 树状图

举例2：用单列法进行层次聚类(矩阵为相似度矩阵，使用方法为单链法)

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



聚类算法评估

外部指标

“外部指标”（external index) 将聚类结果与某个“参考模型”（reference model)进行比较。
假设聚类模型输出为 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$，参考模型的输出为 $\mathcal{C}^*=\{C_1^*,C_2^*,\ldots,C_s^*\}$，记$\boldsymbol{\lambda}$, $\boldsymbol{\lambda}^*$
为两个输出对应的簇标记。计算以下数量：

$$a=|SS|,\quad SS=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i=\lambda_j,\lambda_i^*=\lambda_j^*,i<j)\},$$

$$b=|SD|,\quad SD=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i=\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j)\},$$

$$c=|DS|,\quad DS=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i\neq\lambda_j,\lambda_i^*=\lambda_j^*,i<j)\},$$

$$d=|DD|,\quad DD=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i\neq\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j)\},$$

由此得出以下指标：

Rand Index　　Jaccard Coefficient

$$\mathrm{RI}=\frac{2(a+d)}{m(m-1)}\qquad \mathrm{JC}=\frac{a}{a+b+c}$$

内部指标

“内部指标”(internal index)直接考察聚类结果而不利用任何参考模：
假设聚类模型输出为 $\mathcal{C}=\{C_1,C_2,\ldots,C_k\}$
通常以“簇内相似，簇间相异”的出发点来评估聚类输出，可定义

$$\mathrm{avg}(C)=\frac{2}{|C|(|C|-1)}\sum_{1\leqslant i<j\leqslant|C|}\mathrm{dist}(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

$$d_{\mathrm{cen}}(C_i,C_j)=\mathrm{dist}(\boldsymbol{\mu}_i,\boldsymbol{\mu}_j)$$

然后设计指标 DB Index

$$\mathrm{DBI}=\frac{1}{k}\sum_{i=1}^{k}\max_{j\neq i}\left(\frac{\mathrm{avg}(C_i)+\mathrm{avg}(C_j)}{d_{\mathrm{cen}}(\boldsymbol{\mu}_i,\boldsymbol{\mu}_j)}\right)$$