

Hierarchical Clustering

层次聚类

基于数据之间距离, 自下而上聚合, 或自上而下分裂



如果不能简单地解释某个理论,说明你并没有真正理解它。

If you can't explain it simply, you don't understand it well enough.

—— 阿尔伯特·爱因斯坦 (Albert Einstein) | 理论物理学家 | 1879 ~ 1955



- ◀ numpy.triu() 提取上三角矩阵
- scipy.cluster.hierarchy.dendrogram() 绘制树形图
- ◀ seaborn.clustermap() 绘制树形图和热图
- ◀ seaborn.heatmap() 绘制热图
- ◀ sklearn.cluster.AgglomerativeClustering() 层次聚类函数

14.1 层次聚类

层次聚类 (hierarchical clustering) 算法是一种聚类分析算法。层次聚类依据数据之间的距离远近,或者亲近度大小,将样本数据划分为簇。层次聚类可以通过**自下而上** (agglomerative) 合并,或者**自上而下** (divisive) 分割来构造分层结构聚类。

图 1 所示为根据鸢尾花样本数据前两个特征——花萼长度和宽度——获得的层次聚类<mark>树形图</mark> (dendrogram)。

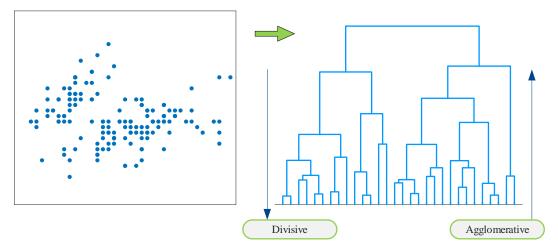


图 1. 区分"自上而下"和"自下而上"层次聚类

自下而上合并

图 2 所示为自下而上合并原理。整个过程有点像"搭积木",首先以每个数据点本身作为一簇,每次迭代合并"距离"较近或亲近度大的类别,直到最后只剩一簇为止。这个过程可以使用的距离度量或亲和度也是多种多样的。请大家回顾本书第 3 章有关距离度量和亲近度内容。

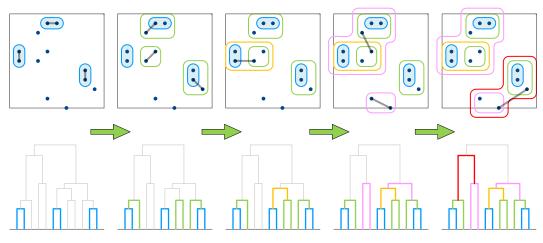


图 2. 层次聚类原理

本章下面首先介绍如何一步步通过自下而上合并获得树形图。大家可能已经注意到,图 2 中不仅仅要考虑,"点"与"点"之间距离,还需要考虑"簇"与"簇"之间的距离。"簇"与"簇之间的距离,也是本章要探讨的核心内容之一。

14.2 树形图

图 3 给出 12 个样本数据在平面上的位置。相信大家还记得成对距离矩阵 (pairwise distance matrix) 这个概念。图 4 所示 12 个样本数据成对欧氏距离矩阵的热图。

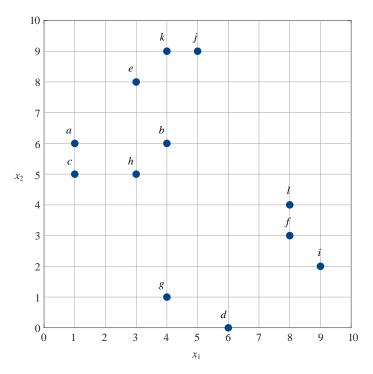


图 3.12 个样本数据

有了图 4 所示成对欧氏距离矩阵,便可以得到如图 5 所示树形图。树形图横轴对应样本数据编号,纵轴对应数据点间距离和簇间欧氏距离。

观察图 5 树形图,在距离值 2.5 处切一刀,可以将图 3 所示数据分成 3 簇;如果在距离值为 4 处切一刀,可以将图 3 所示数据分成 2 簇。下面,我们一步步介绍如何自下而上构造如图 5 所示树形图。

а	0	3	1	7.81	2.828	7.616	5.831	2.236	8.944	5	4.243	7.28
b	3	0	3.162	6.325	2.236	5	5	1.414	6.403	3.162	3	4.472
c	1	3.162	0	7.071	3.606	7.28	5	2	8.544	5.657	5	7.071
d	7.81	6.325	7.071	0	8.544	3.606	2.236	5.831	3.606	9.055	9.22	4.472
e	2.828	2.236	3.606	8.544	0	7.071	7.071		8.485	2.236	1.414	6.403
f	7.616	5	7.28	3.606	7.071	0	4.472	5.385	1.414	6.708	7.211	1
g	5.831	5	5	2.236	7.071	4.472	0	4.123	5.099	8.062	8	5
h	2.236	1.414	2	5.831	3	5.385	4.123	0	6.708	4.472	4.123	5.099
i	8.944	6.403	8.544	3.606	8.485	1.414	5.099	6.708	0	8.062	8.602	2.236
j	5	3.162	5.657	9.055	2.236	6.708	8.062	4.472	8.062	0	1	5.831
k	4.243		5	9.22	1.414	7.211	8	4.123	8.602	1	0	6.403
l	7.28	4.472	7.071	4.472	6.403	1	5	5.099	2.236	5.831	6.403	0
	а	b	c	d	e	f	g	h	i	j	k	l

图 4.12 个样本数据成对距离构成的方阵热图

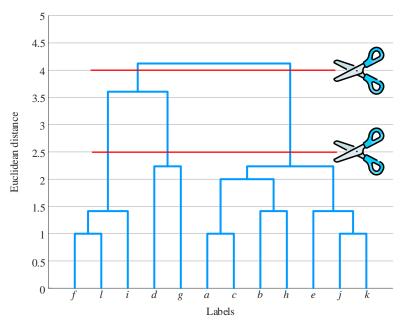


图 5. 数据树形图

第一层

图 6 所示,首先发现 a 和 c、k 和 j、f 和 l 成对距离最短,均为 1; 这样我们便构造树形图最底层。这三个成对距离在热图位置如图 8 (a) 所示。

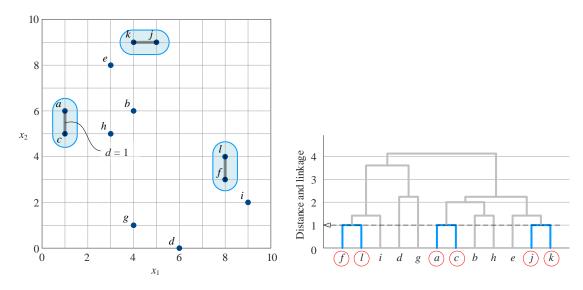


图 6. 构建树形图, 第一层

第二层

构造树形图第二层时,遇到一个麻烦——簇间距离如何定义。这里,我们首先采用最简单的最近点距离 (single linkage 或 nearest neighbor)。最近点距离指的是两个簇样本数据成对距离最近值。

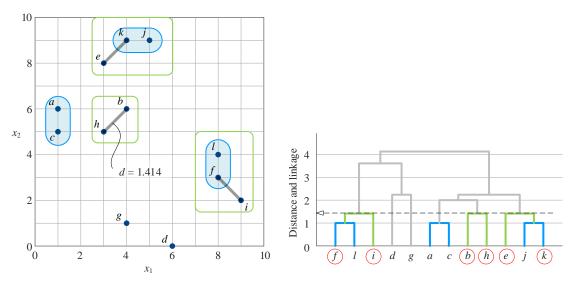


图 7. 构建树形图, 第二层

k 和 j、f 和 l 已经分别"成团"; e 距离 k 更近,而 i 距离 f 更近。因此树形图第二层的距离值定为 sqrt(2),也就是约 1.414。同样,b 和 h 的距离也是 1.414。这样我们便构造得到了如图 7 所示的树形图第二层。这三个"簇间"/"点间"距离在热图位置如图 8 (b) 所示。

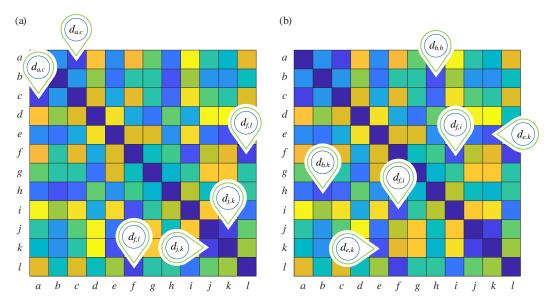


图 8. 成对距离矩阵热图,第一层和第二层距离位置

第三层

再向上一层,利用簇 a 和 c (第一层)、簇 b 和 h (第二层)之间簇间距离 2,从而得到树形图第三层。这个距离在热图上的位置如图 11 (a) 所示。

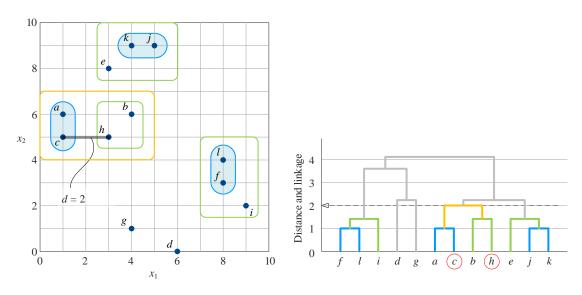


图 9. 构建树形图,第三步

第四层

树形图的第四层采用的距离值为 sqrt(5), 约 2.236。图 10 所示为树形图第四层位置。可以发现此时,所有的数据点均参与聚类,形成 3 簇。

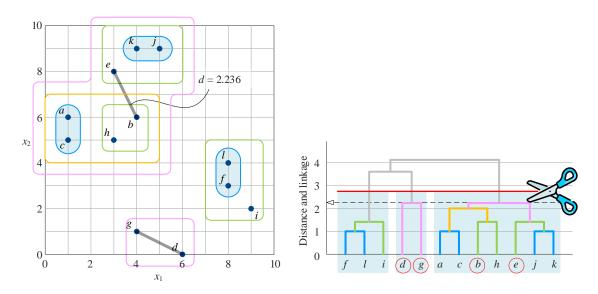


图 10. 构建树形图, 第四层

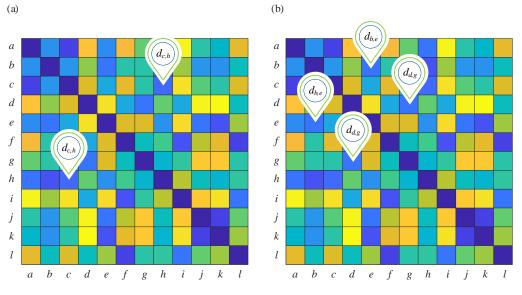


图 11. 成对距离矩阵热图,第三层和第四层距离位置

第五层

图 12 所示为树形图第五层位置。在第五层,样本数据被划分为两簇;再加一层,整个树形图便封顶。

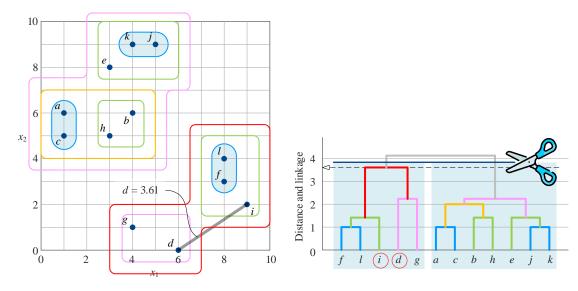


图 12. 构建树形图, 第五层

重新排序

树形结构把数据序号重新排列。根据这个顺序,可以得到一个全新的热图,如图 13 所示。根据颜色,图 13 所示热图很容易分为两个区域,对应数据划分为两簇。这便是层次聚类的思路。

		_										
					Ц.						٦.	
		5										
f	0	1	1.414	3.606	4.472	7.616	7.28	5	5.385	7.071	6.708	7.211
l	1	0	2.236	4.472	5	7.28	7.071	4.472	5.099	6.403	5.831	6.403
i	1.414	2.236	0	3.606	5.099	8.944	8.544	6.403	6.708	8.485	8.062	8.602
d	3.606	4.472	3.606	0	2.236	7.81	7.071	6.325	5.831	8.544	9.055	9.22
g	4.472	5	5.099	2.236	0	5.831	5	5	4.123	7.071	8.062	8
а	7.616	7.28	8.944	7.81	5.831	0	1		2.236	2.828	5	4.243
С	7.28	7.071	8.544	7.071	5	1	0	3.162	2	3.606	5.657	5
b	5	4.472	6.403	6.325	5		3.162	0	1.414	2.236	3.162	3
h	5.385	5.099	6.708	5.831	4.123	2.236	2	1.414	0		4.472	4.123
e	7.071	6.403	8.485	8.544	7.071	2.828	3.606	2.236	3	0	2.236	1.414
j	6.708	5.831	8.062	9.055	8.062	5	5.657	3.162	4.472	2.236	0	1
k	7.211	6.403	8.602	9.22	8	4.243	5	3	4.123	1.414	1	0
	f	l	i	d	g	а	с	b	h	e	j	k

图 13. 按树形图重组数据树形图

14.3 簇间距离

上一节提到,两簇之间的距离可以采用最近点距离;当然,簇间距离也有其他定义。本节介绍常用的几种簇间距离。

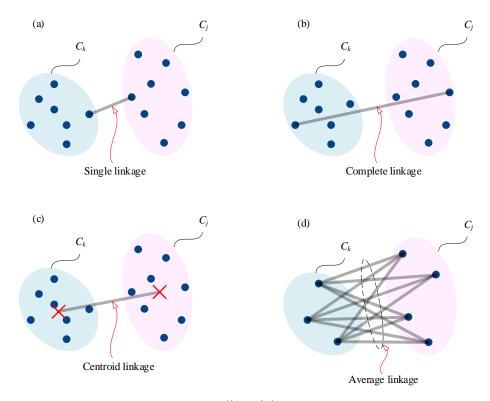


图 14. 簇间距离定义

最近点距离

簇间距离,也叫做**距离值** (linkage distance 或者 linkage)。如图 14 (a) 所示,**最近点距离** (single linkage 或 nearest neighbor),代号为 'single',指的是两个簇样本数据成对距离最近值:

$$d(C_k, C_j) = \min_{\mathbf{x} \in C_k, \ \mathbf{z} \in C_j} (\operatorname{dist}(\mathbf{x}, \mathbf{z}))$$
 (1)

图 15 所示为,采用 'single' 层次聚类得到的树形图和鸢尾花数据聚类结果。可以发现,树形图分支并不均衡,聚类结果不理想。

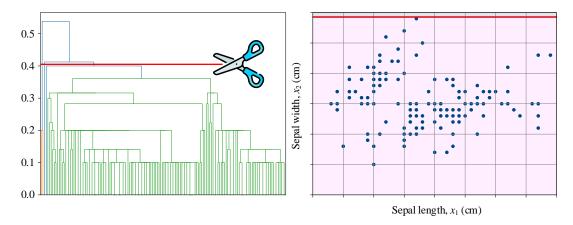


图 15. 鸢尾花聚类结果, 'single' 层次聚类

最远点距离

如图 14 (b) 所示,**最远点距离** (complete linkage 或 farthest neighbor) 定义为,两簇样本数据成对距离最远值:

$$d(C_k, C_j) = \max_{x \in C_k, \ z \in C_j} (\operatorname{dist}(x, z))$$
 (2)

最远点距离代号为 'complete'。图 16 所示为,采用 'complete' 层次聚类得到的树形图和鸢尾花数据聚类结果。最远点距离对于离群点/噪音点敏感。

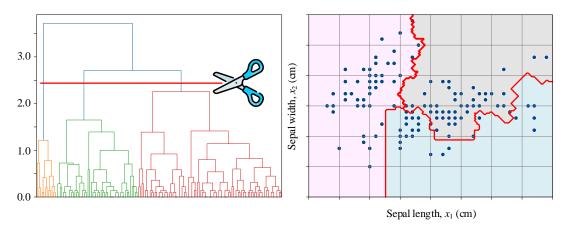


图 16. 鸢尾花聚类结果, 'complete' 层次聚类

均值点距离

如图 14 (c) 所示,均值点距离 (centroid linkage) 采用两个簇样本数据均值点之间的距离:

$$d\left(C_{i},C_{j}\right)=d\left(\boldsymbol{\mu}_{i},\boldsymbol{\mu}_{j}\right) \tag{3}$$

其中, μ_i 和 μ_j 分别为 C_i 和 C_j 的质心点。目前 Scikit-learn 中的层次聚类函数并不支持均值点距离; 但是 scipy.cluster.hierarchy.linkage 支持均值点距离,代号为 'centroid'。

平均距离

如图 14 (d) 所示,平均距离 (average linkage) 采用两簇样本数据成对点之间距离取平均值:

$$d\left(C_{k}, C_{j}\right) = \underset{\boldsymbol{x} \in C_{k}, \ \boldsymbol{z} \in C_{j}}{\operatorname{mean}} \left(\operatorname{dist}\left(\boldsymbol{x}, \boldsymbol{z}\right)\right) = \frac{\sum_{\boldsymbol{x} \in C_{k}, \ \boldsymbol{z} \in C_{j}} \operatorname{dist}\left(\boldsymbol{x}, \boldsymbol{z}\right)}{\operatorname{count}\left(C_{k}\right) \cdot \operatorname{count}\left(C_{j}\right)}$$
(4)

平均距离代号为 'average'。图 17 所示为,采用 'average' 层次聚类得到的树形图和鸢尾花数据聚类结果。

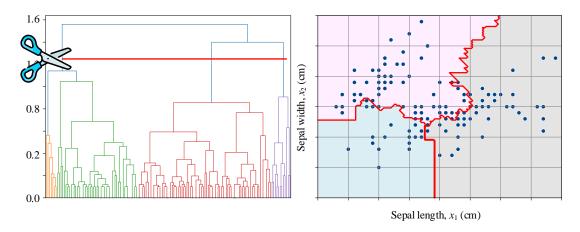


图 17. 鸢尾花聚类结果,'average' 层次聚类

平均距离

Ward's 簇间距离的定义如下:

$$d(C_{k}, C_{j}) = \sqrt{2 \times \left(\underbrace{\sum_{\substack{\mathbf{x} \in C_{k} \cup C_{j} \\ \text{After merge}}}}_{\text{After merge}} \operatorname{dist}(\mathbf{x}, \boldsymbol{\mu}_{C_{k} \cup C_{j}})^{2} - \left(\underbrace{\sum_{\substack{\mathbf{x} \in C_{k} \\ \text{Before merge}}}}_{\text{Before merge}} \operatorname{dist}(\mathbf{x}, \boldsymbol{\mu}_{C_{j}})^{2} \right) \right)}_{\text{Before merge}}$$

$$= \sqrt{\frac{2 \cdot \operatorname{count}(C_{k}) \cdot \operatorname{count}(C_{j})}{\operatorname{count}(C_{k}) + \operatorname{count}(C_{j})}} \cdot \operatorname{dist}(\boldsymbol{\mu}_{C_{k}}, \boldsymbol{\mu}_{C_{j}})}$$
(5)

观察(5), 可以发现它等价于:

$$d\left(C_{k}, C_{j}\right) = \sqrt{2 \times \left(\underbrace{\mathbf{SST}\left(C_{k} \cup C_{j}\right)}_{\text{After merge}} - \underbrace{\left(\mathbf{SST}\left(C_{k}\right) + \mathbf{SST}\left(C_{j}\right)\right)}_{\text{Before merge}}\right)}$$
 (6)

其中,SSE 为丛书前文介绍的总离差平方和 (Sum of Squares for Total, SST)。

SSE 便是本书前文介绍的"簇惯性 (cluster inertia)", 也就是:

$$\begin{aligned}
&\left\{ \operatorname{SST}\left(C_{k} \cup C_{j}\right) = \sum_{\boldsymbol{x} \in C_{k} \cup C_{j}} \operatorname{dist}\left(\boldsymbol{x}, \boldsymbol{\mu}_{C_{k} \cup C_{j}}\right)^{2} = \sum_{\boldsymbol{x} \in C_{k} \cup C_{j}} \left\|\boldsymbol{x} - \boldsymbol{\mu}_{C_{k} \cup C_{j}}\right\|^{2} \\
&\left\{ \operatorname{SST}\left(C_{j}\right) = \sum_{\boldsymbol{x} \in C_{j}} \operatorname{dist}\left(\boldsymbol{x}, \boldsymbol{\mu}_{C_{j}}\right)^{2} = \sum_{\boldsymbol{x} \in C_{j}} \left\|\boldsymbol{x} - \boldsymbol{\mu}_{C_{j}}\right\|^{2} \\
&\left\{ \operatorname{SST}\left(C_{k}\right) = \sum_{\boldsymbol{x} \in C_{k}} \operatorname{dist}\left(\boldsymbol{x}, \boldsymbol{\mu}_{C_{k}}\right)^{2} = \sum_{\boldsymbol{x} \in C_{k}} \left\|\boldsymbol{x} - \boldsymbol{\mu}_{C_{k}}\right\|^{2}
\end{aligned} \right. \tag{7}$$

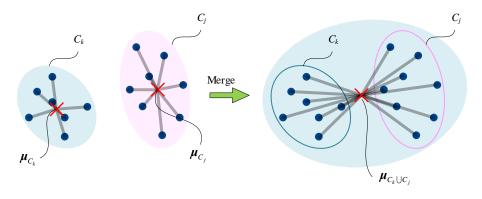


图 18. 鸢尾花聚类结果, Ward's 簇间距离

Ward's 簇间距离定义看着复杂,实际上背后的思想很简单——计算**合并后** (after merge)、**合并前** (before merge) 残差平方和 SSE 的差值。原理如图 18 所示。这个差值,也就是一种簇数据"合并"的"代价"。

平均距离代号为 <mark>'ward'</mark>。 <mark>'ward'</mark>为 Scikit-learn 默认簇间距离。图 19 所示为,采用 <mark>'ward'</mark> 层次聚类得到的树形图和鸢尾花数据聚类结果。

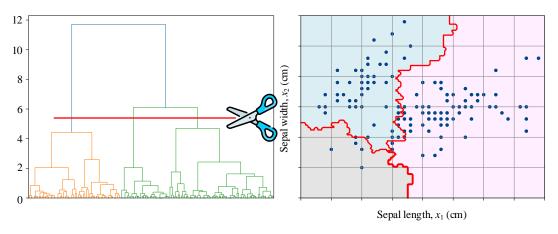


图 19. 鸢尾花聚类结果, 'ward' 层次聚类



代码 Bk7_Ch14_01.py 可以绘制图 15、图 16、图 17 和图 19。

14.4 亲近度层次聚类

本章前文介绍的是采用欧氏距离构造树形图,以便进行层次聚类;其实,亲近度也可以用来构造树形图,从而聚类。回顾**高斯核** (Gaussian kernel) 亲近度定义:

$$\kappa(\mathbf{x}, \mathbf{q}) = \exp(-\gamma \|\mathbf{x} - \mathbf{q}\|^2)$$
(8)

图 20 左图是鸢尾花数据高斯核亲近度成对矩阵热图。利用 seaborn.clustermap() 函数可以绘制基于亲近度矩阵的树形图,以及相应热图,具体如图 20 右图所示。

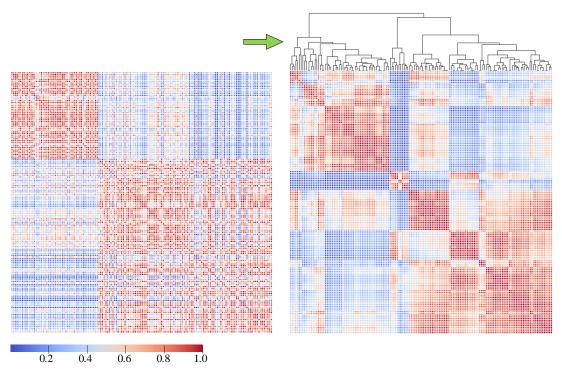


图 20. 鸢尾花花萼两个特征构造的亲近度矩阵,以及树形结构和重排额的亲近度矩阵



代码 Bk7_Ch14_02.py 可以绘制图 20 两幅子图。