



Chapter 5: Clustering

崔金华

电子邮箱: jhcui@hust.edu.cn

个人主页: <https://csjhcui.github.io/>

Contents

提纲

5.1

聚类技术介绍

Introduction to Clustering Techniques

5.2

层次聚类

Hierarchical Clustering

5.3

K-均值算法

K-means Algorithms

5.4

BFR算法

BFR Algorithm

5.5

CURE算法

CURE Algorithm

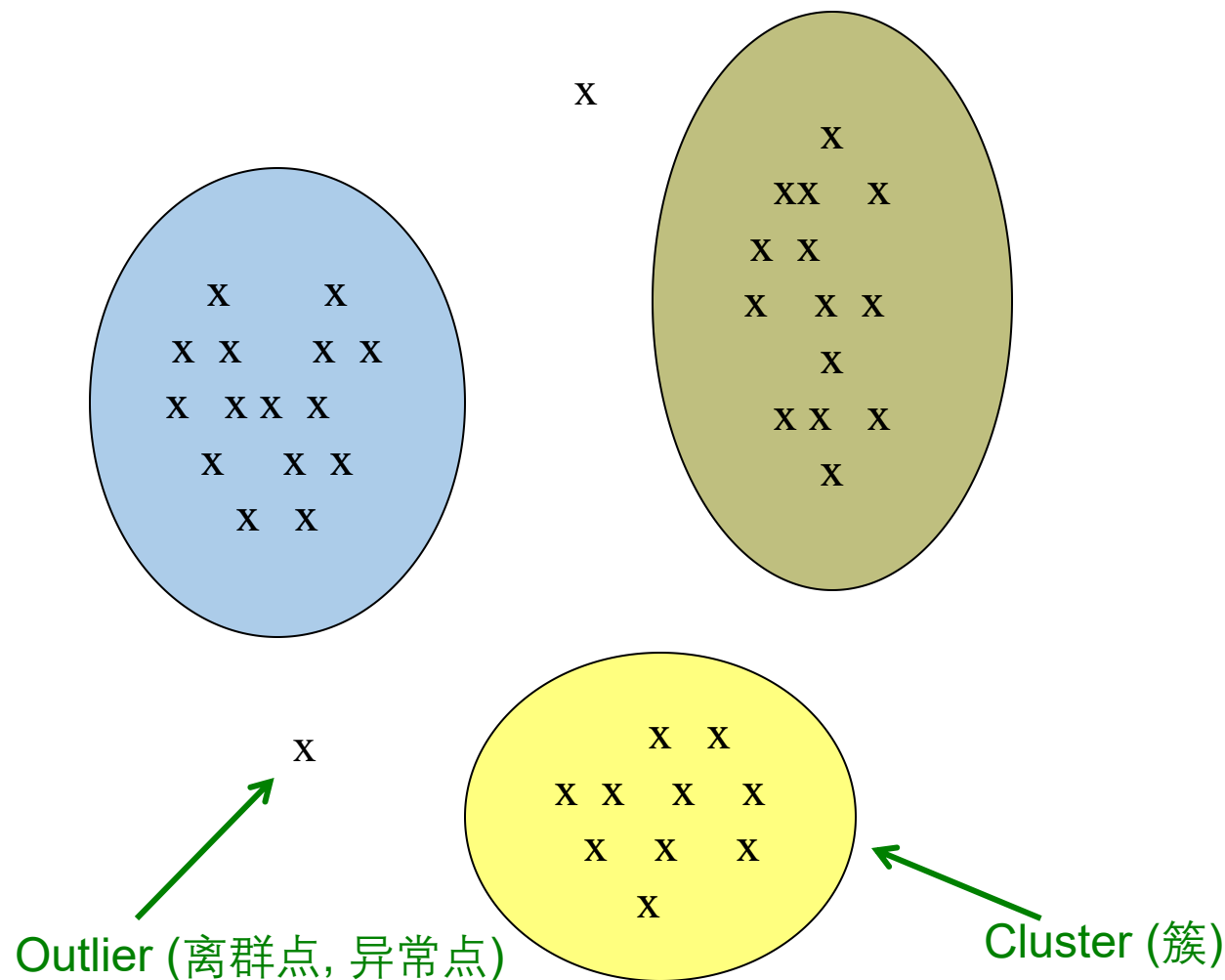
High Dimensional Data

□ Given a cloud of data points we want to understand its structure

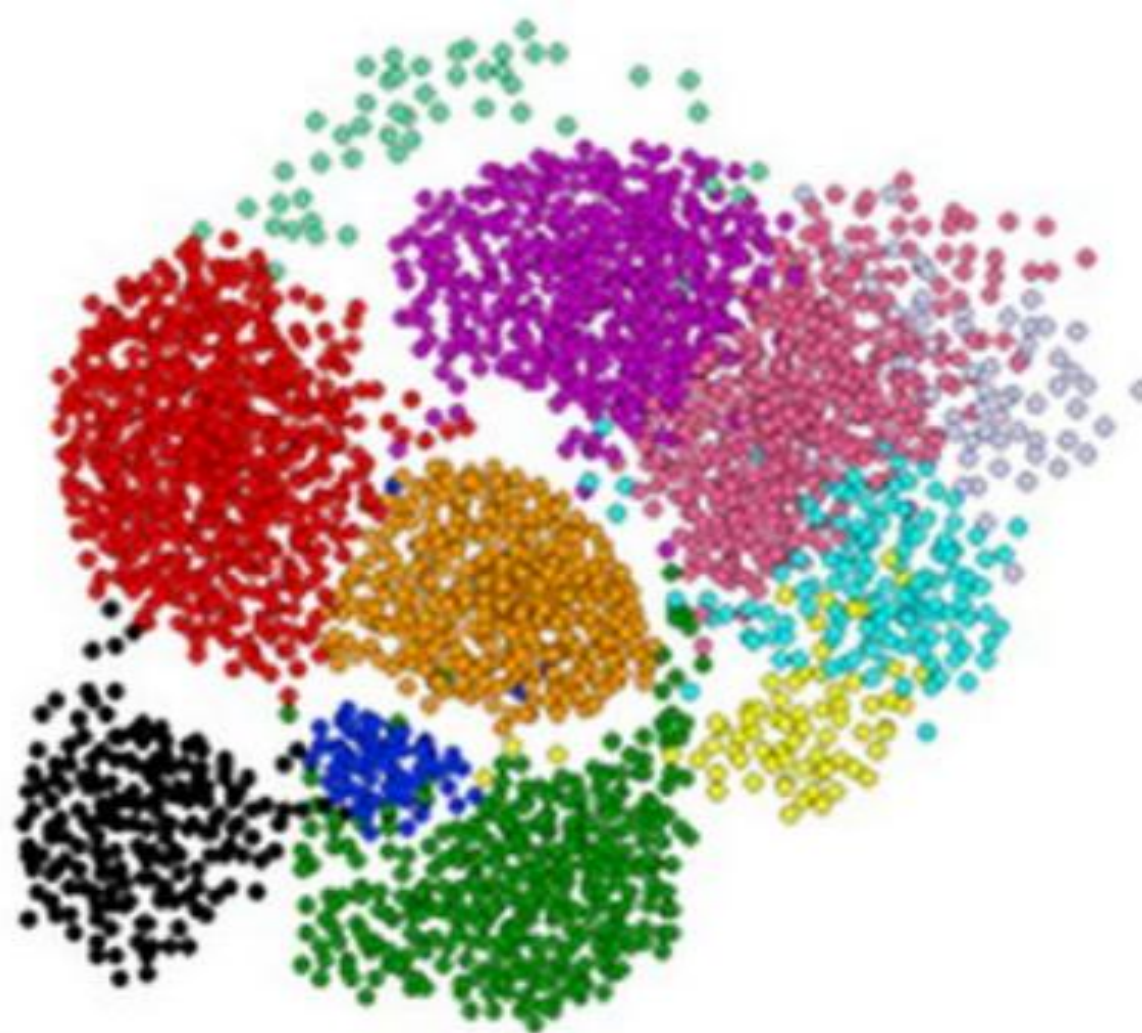


- Given a **set of points(点集)**, with a notion of **distance** between points, **group the points** into some number of ***clusters(簇)***, so that
 - Members of a cluster are close/similar to each other
 - Members of different clusters are dissimilar
- **Usually:**
 - Points are in a high-dimensional space
 - Similarity is defined using a distance measure
 - Euclidean, Cosine, Jaccard, edit distance, ...

Example: Clusters & Outliers



Clustering is a hard problem!



Why is it hard?

- ❑ Clustering in two dimensions looks easy
- ❑ Clustering small amounts of data looks easy
- ❑ And in most cases, looks are **not** deceiving (欺骗性)
- ❑ However, many applications involve not 2, but **10 or 10,000 dimensions**
- ❑ **High-dimensional spaces look different:** Almost all pairs of points are at about the same distance

Clustering Problem: Galaxies

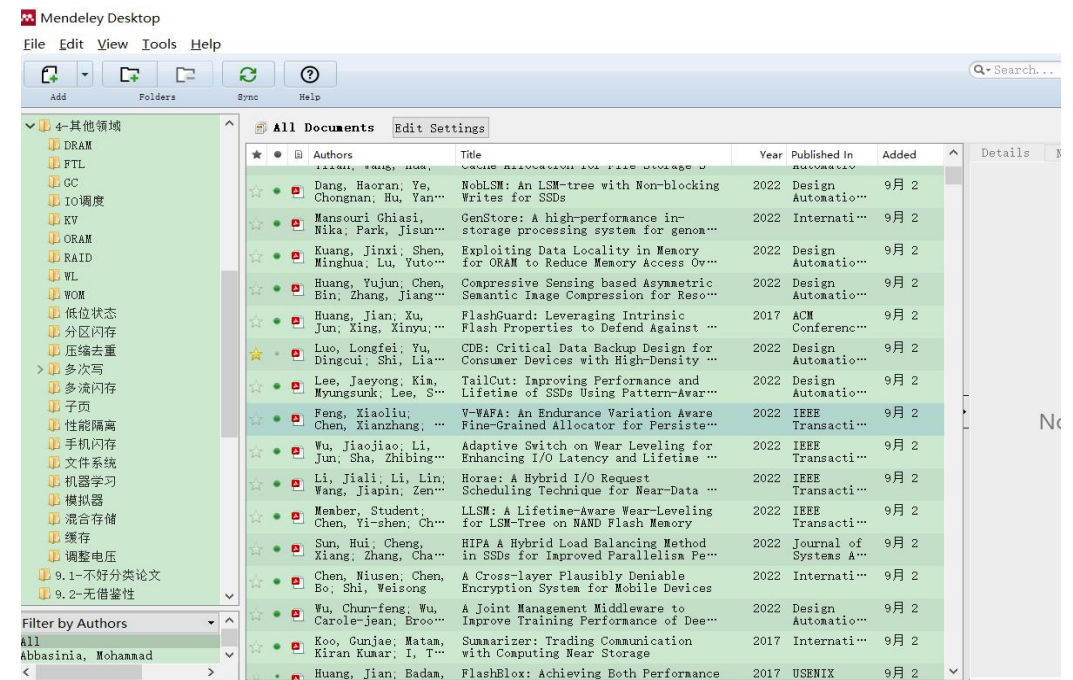
- ❑ **SkyCat: A catalog of 2 billion “sky objects” represents objects by their radiation in 7 dimensions (frequency bands)**
- ❑ **Problem:** Cluster into similar objects, e.g., galaxies (星系), nearby stars (近恒星), quasars (类星体), etc.
- ❑ **Sloan Digital Sky Survey (斯隆数字化巡天项目)**



Clustering Problem: Documents

- Finding topics: Represent a document by a vector (x_1, x_2, \dots, x_k) , where $x_i = 1$ iff the i^{th} word (in some order) appears in the document
 - It actually doesn't matter if k is infinite; i.e., we don't limit the set of words

- Documents with similar sets of words may be about the same topic



Authors	Title	Year	Published In	Added	Details
Dang, Haoran, Ye, Chongnan, Hu, Yan...	NobLSM: An LSM-tree with Non-blocking Writes for SSDs	2022	Design Automation...	9月 2	
Mansouri Chiasi, Nika, Park, Jisun...	GenStore: A high-performance in-storage processing system for genom...	2022	Internati...	9月 2	
Kuang, Jinxi, Shen, Minghua, Lu, Yuto...	Exploiting Data Locality in Memory for ORAM to Reduce Memory Access Ov...	2022	Design Automation...	9月 2	
Huang, Yujun, Chen, Bin, Zhang, Jjiang...	Compressive Sensing based Asymmetric Semantic Image Compression for Reso...	2022	Design Automation...	9月 2	
Huang, Jian, Xu, Jun, Xing, Xinyu...	FlashGuard: Leveraging Intrinsic Flash Properties to Defend Against ...	2017	ACM Conferenc...	9月 2	
Luo, Longfei, Yu, Dingcui, Shi, Lia...	CDE: Critical Data Backup Design for Consumer Devices with High-Density ...	2022	Design Automation...	9月 2	
Lee, Jaeyong, Kim, Myungseok, Lee, S...	TailOut: Improving Performance and Lifetime of SSDs Using Pattern-Awar...	2022	Design Automation...	9月 2	
Feng, Xiaolu, Chen, Xianzhong...	V-WAFA: An Endurance Variation Aware Fine-Grained Allocator for Persiste...	2022	IEEE Transacti...	9月 2	
Wu, Jiaojiao, Li, Jun, Sha, Zhibing...	Adaptive Switch on Wear Leveling for Enhancing I/O Latency and Lifetime ...	2022	IEEE Transacti...	9月 2	
Li, Jialin, Li, Lin, Wang, Jiajin, Zen...	Horae: A Hybrid I/O Request Scheduling Technique for Near-Data ...	2022	IEEE Transacti...	9月 2	
Member, Student, Chen, Yi-shen, Ch...	LLSM: A Lifetime-Aware Wear-Leveling for LSM-Tree on NAND Flash Memory	2022	IEEE Transacti...	9月 2	
Sun, Hui, Cheng, Xiang, Zhang, Cha...	HIPA: A Hybrid Load Balancing Method in SSDs for Improved Parallelism Fe...	2022	Journal of Systems A...	9月 2	
Chen, Niusep, Chen, Bo, Shi, Weisong	A Cross-layer Plausibly Deniable Encryption System for Mobile Devices	2022	Internati...	9月 2	
Wu, Chun-feng, Wu, Carole-jean, Broo...	A Joint Management Middleware to Improve Training Performance of Dee...	2022	Design Automation...	9月 2	
Koo, Gunjae, Matan, Kiran Kumar, I, T...	Summarizer: Trading Communication with Computing Near Storage	2017	Internati...	9月 2	
Huang, Jian, Badan...	FlashBlox: Achieving Both Performance	2017	USENIX	9月 2	

Clustering Problem: Music CDs

- **Intuitively:** Music divides into categories, and customers prefer a few categories
 - But what are categories really?
- Represent a CD by a set of customers who bought it:
 - Similar CDs have similar sets of customers, and vice-versa

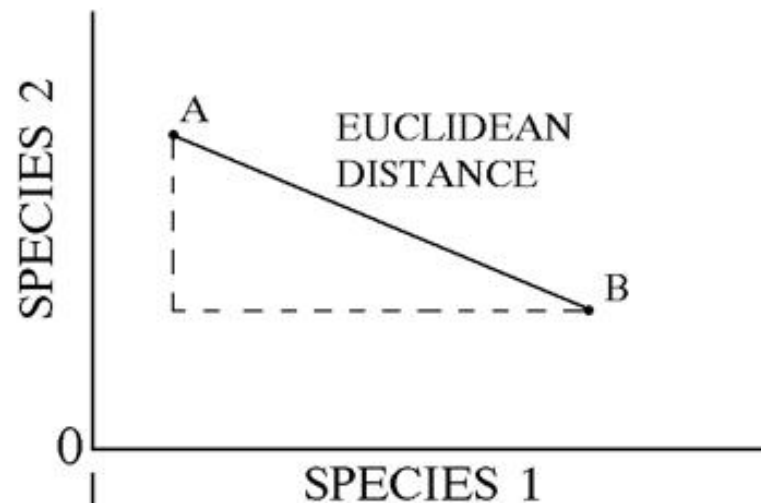
Clustering Problem: Music CDs

- ❑ **Space of all CDs:** Think of a space with one dim. for each customer
 - Values in a dimension may be 0 or 1 only
 - A CD is a point in this space (x_1, x_2, \dots, x_k) , where $x_i = 1$ iff the i^{th} customer bought the CD
- ❑ For Amazon, the dimension is tens of millions
- ❑ **Task:** Find clusters of similar CDs

- **As with CDs we have a choice when we think of documents as sets of words or shingles:**
 - **Sets as points:** Measure similarity by **Euclidean distance**
 - **Sets as sets:** Measure similarity by the **Jaccard distance**
 - **Sets as vectors:** Measure similarity by the **Cosine distance**

□ Euclidean distance(欧氏距离)

- 也就是我们通常想象的距离. 在n维欧氏空间下, 每个点是一个n维实数向量. 在该空间下的传统距离测度, 即我们常说的L2范式(L2-norm).



$$d(p, q) = \sqrt{\sum_{j=1}^d (p_j - q_j)^2}$$

□ Euclidean similarity

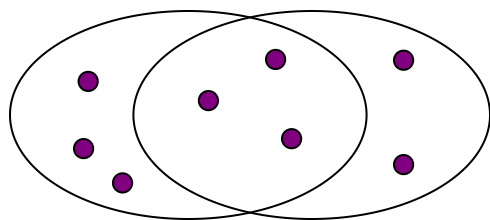
- $\frac{1}{1+d(p,q)}$, $(0, 1]$. closer to 1, the more similar

□ Goal: define what “distance” means in high-dim. space

- The **Jaccard similarity** of two **sets** is the size of their intersection(交集) divided by the size of their union(并集):

$$\text{sim}(\mathbf{C}_1, \mathbf{C}_2) = |\mathbf{C}_1 \cap \mathbf{C}_2| / |\mathbf{C}_1 \cup \mathbf{C}_2|$$

- **Jaccard distance:** $d(\mathbf{C}_1, \mathbf{C}_2) = 1 - |\mathbf{C}_1 \cap \mathbf{C}_2| / |\mathbf{C}_1 \cup \mathbf{C}_2|$



3 in intersection
8 in union
Jaccard similarity = 3/8
Jaccard distance = 5/8

Example: Jaccard Distance

□ Example: The Jaccard distance for the documents?

apple
releases
new ipod

D1

apple
releases
new ipad

D2

new
apple pie
recipe

D3

Vefa rereases
new book with
apple pie
recipes

D4

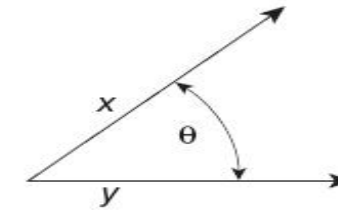
□ Ans:

- $\text{sim}(D1, D2) = 3/5$, $\text{sim}(D1, D3) = \text{sim}(D2, D3) = 2/6$, $\text{sim}(D1, D4) = \text{sim}(D2, D4) = 3/9$
- $d(D1, D2) = 2/5$, $d(D1, D3) = d(D2, D3) = 4/6$, $d(D1, D4) = d(D2, D4) = 6/9$

Cosine Similarity

□ $\text{Sim}(X, Y) = \cos(X, Y)$

➤ The cosine of the angle between X and Y



Geometric illustration of the cosine measure.

□ If the vectors are **aligned (correlated)** angle is **zero degrees** and $\cos(X, Y) = 1$

□ If the vectors are **orthogonal** (no common coordinates) angle is **90 degrees** and $\cos(X, Y) = 0$

□ Cosine is commonly used for comparing **documents**, where we assume that the vectors are **normalized** by the document length.

□ **Cosine similarity:** $\cos(d_1, d_2) = \frac{d_1 \bullet d_2}{\|d_1\| \|d_2\|}$

- d_1 and d_2 are two vectors. \bullet indicates vector dot product (向量的点积, 或称内积); $\|d\|$ is the length of vector d (向量的大小).
- $(-1, 1)$. closer to 1, the more similar

□ Example: $d_1 = 3\ 2\ 0\ 5\ 0\ 0\ 0\ 2\ 0\ 0$, $d_2 = 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 2$

□ **Ans:**

- First, $d_1 \bullet d_2 = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5$;
- $\|d_1\| = \sqrt{3*3 + 2*2 + 0*0 + 5*5 + 0*0 + 0*0 + 0*0 + 2*2 + 0*0 + 0*0} = 6.481$;
- $\|d_2\| = \sqrt{1*1 + 0*0 + 0*0 + 0*0 + 0*0 + 0*0 + 0*0 + 1*1 + 0*0 + 2*2} = 2.245$.
- So, $\cos(d_1, d_2) = 0.3150$

□ **Cosine Distance(余弦距离):** $dis(d_1, d_2) = 1 - \cos(d_1, d_2)$

➤ (0,2)

➤ closer to 0, the more similar

Example: Cosine Distance

□ Example: The cosine distance for the documents?

document	Apple	Microsoft	Obama	Election
D1	10	20	0	0
D2	30	60	0	0
D3	60	30	0	0
D4	0	0	10	20

□ Ans:

$\text{dis}(D1, D2) = 0$  D1 and D2 similarity

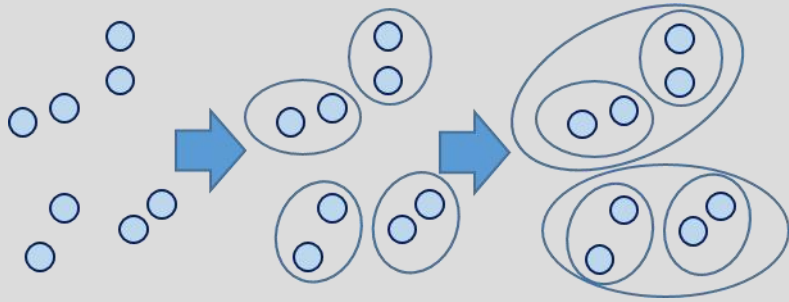
$\text{dis}(D3, D1) = \text{dis}(D3, D2) = 1/5$

$\text{dis}(D4, D1) = \text{dis}(D4, D2)$  D4 and D1 dissimilarity;
 $= \text{dis}(D4, D3) = 1$

- **Edit Distance(编辑距离):** The edit distance between two strings $x = x_1x_2...x_n$ and $y = y_1y_2...y_m$ is the smallest number of insertions and deletions of single characters that will convert x to y .
 - This distance makes sense when points are strings (编辑距离适用于字符串比较).
- **Example:** What is the edit distance between the strings $x = abcde$ and $y = acfdeg$?
- **Ans:** $d(x, y) = 3$. To convert x to y : Step 1. Delete b ; Step 2: Insert f after c ; Step 3: Insert g after e . No sequence of fewer than three insertions and/or deletions will convert x to y .

- **Longest common subsequence (LCS, 最长公共子序列)** of x and y is a string that is constructed by deleting positions from x and y , and that is as long as any string that can be constructed that way.
- Then, edit distance $d(x, y) = \text{the length of } x + \text{the length of } y - 2 * \text{the length of their LCS}$.
- **Example:** What is the edit distance between the strings $x = \text{abcde}$ and $y = \text{acfdeg}$?
- **Ans:** $d(x, y) = 3$. LCS is "acde". Then $5 + 6 - 2 * 4 = 3$.

- **Hamming Distance(海明距离):** Given a space of vectors, we define the Hamming distance between two vectors to be the number of components in which they differ.
- **Example:** What is the Hamming distance between the vectors 10101 and 11110
- **Ans:** Hamming distance is 3. That is, these vectors differ in the second, fourth, and fifth components (1**0**1**0**1 and 1**1**1**0**), while they agree in the first and third components.



Section 5.2: Hierarchical clustering

Clustering Overview

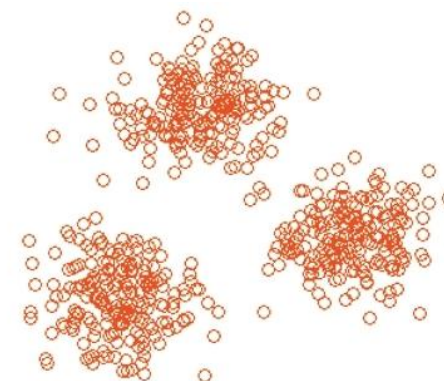
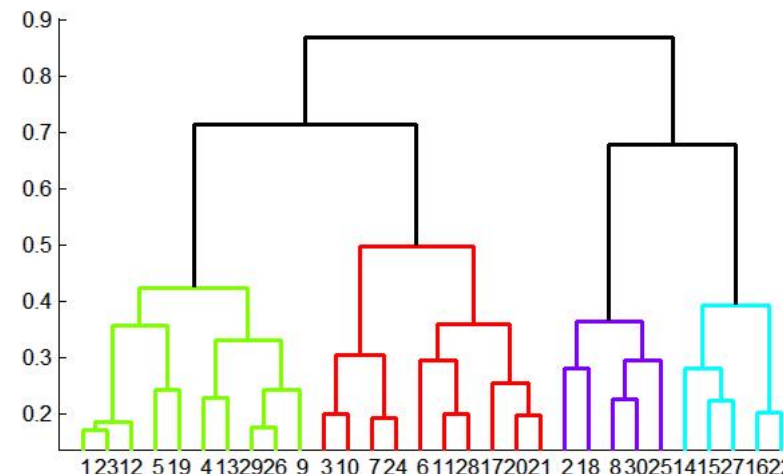
□ Methods of clustering:

□ 1、Hierarchical(层次聚类, 分级聚类, 凝聚式算法):

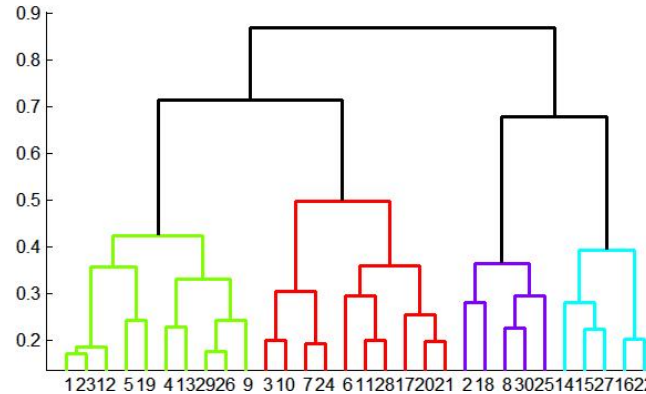
- **Agglomerative** (bottom up):
 - Initially, each point is a cluster
 - Repeatedly combine the two "nearest" clusters into one
- **Divisive** (top down):
 - Start with one cluster and recursively split it

□ 2、Point assignment(点分配):

- Maintain a set of clusters
- Points belong to "nearest" cluster
- e.g. K-mean, BFR, CURE...



□ **Key operation:** Repeatedly combine two nearest clusters



□ **Three important questions:**

- 1) How do you represent a cluster of more than one point?
- 2) How do you determine the “nearness” of clusters?
- 3) When to stop combining clusters?