Clustering

Ying-Chao Hung
Department of Statistics
National Chengchi University
E-mail: *hungy@nccu.edu.tw*

References

- "The Lecture Note of Applied Multivariate Analysis" by George Michailidis, Dept. of Statistics, Univ. of Michigan.
- Struyf, A., Hubert, M. and Rousseeuw, P. J. (1996) "Clustering in an Object-Oriented Environment", *Journal of Statistical Software*, **1**.
- Kaufman, L. and Rousseeuw, P.J. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York.
- Struyf, A., Hubert, M. and Rousseeuw, P.J. (1997). Integrating Robust Clustering Techniques in S-PLUS, *Computational Statistics and Data Analysis*, **26**, 17–37.

Introduction

- Also known as unsupervised pattern recognition, grouping, clumping, classification, numerical taxonomy, Q-analysis.

Ideas:

- Investigate the relationships between objects in order to establish if the data can be summarized by a small number of groups (clusters) of similar objects.
- Assign the objects into a small number of groups such that objects in the same group are as "similar" as possible.

Formulization

Let $O = \{o_1, o_2, \cdots, o_N\}$ be a set of N objects.

Let $C = \{C_1, C_2, \cdots, C_K\}$ be a partition of O into subsets.

That is, $C_i \cap C_j = \phi,\ i \neq j$ and $\bigcup_k C_k = O$.

➔ Each subset is called a cluster.

Differences from Classification

- The classes (groups) are not predetermined as in classification.
- The goal is to find a good grouping of the objects, Not to design a good decision rule for "prediction".
- In most clustering problems, few assumptions are made.
- Applications and connections can be found in the fields of natural, social and life sciences, pattern recognition, computer vision, computational geometry, and operations research.

Some Issues

- Homogeneity vs Separation – what is the criterion used ?
- What type of clustering should be considered ?
- Issue of complexity – how difficult to perform the clustering ?
- Issue of algorithm design – how should the clustering be done ?
- Interpretation of the resulting clustering

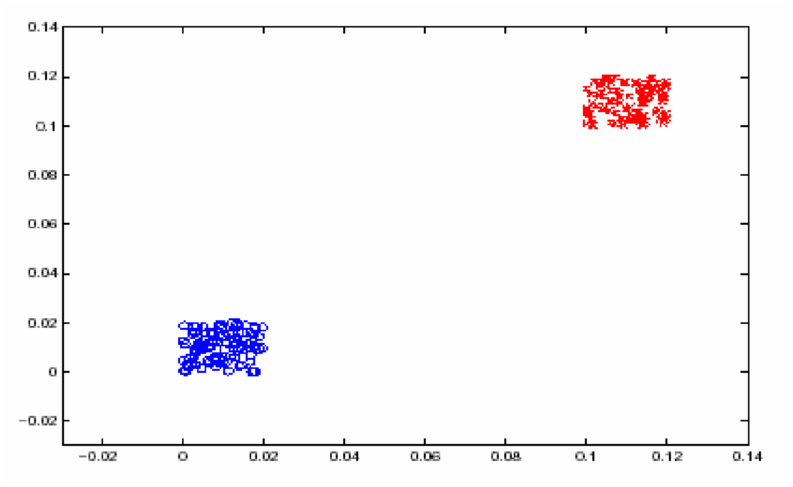
Data and Dissimilarity Matrix

- Profile data matrix: $X_{N\times m}$, N objects with m variables for each.
- Dissimilarity matrix: $\Delta_{N\times N}$, that contains the pairwise dissimilarities (or similarities) computed from the profile data matrix.

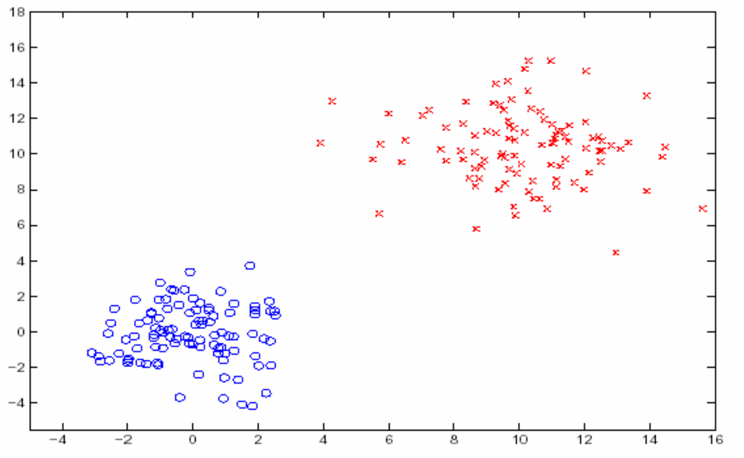
  For example, you can let $\Delta(i, j)$ be the Euclidean distance between object i and object j.
- The variables should be (nearly) continuous so that the Euclidean distance between objects are meaningful.

  (For variables that are categorical (or mixture), see later for MDS and MCA)

Some Visualization

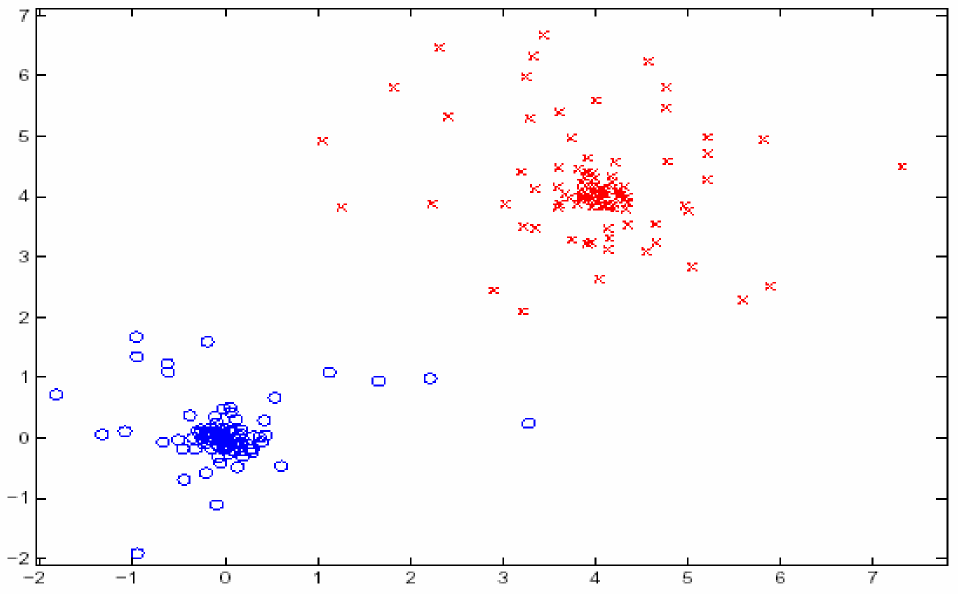


Ideal situation



Well separated but not particularly homogeneous

Some Visualization



Homogeneous but not well separated

Clustering Methods

<u>Heuristic Approaches</u>

- PCA: might be successful in "summarizing" the data using 2 or 3 PCs ➔ then find clusters
- MDS (Multi-dimensional Scaling)
- MCA (Multivariate Correspondence Analysis)

- No guarantee that the groupings will be along the directions of largest variation (problem of PCA)
- MDS usually works well.

Formal Approaches

Mostly based on a measure of the "similarity" or "dissimilarity" between objects.

- **Partition Methods**: create a family of clusters where each object belongs to just a single partition of the data space.
  - Requirement: If object *i* and *j* belong to cluster *A*, while object *k* belongs to cluster *B*, then $d_{ij} < d_{ik}$ and $d_{ij} < d_{jk}$.
- **Tree-type Methods**: the distance (dissimilarity) is ultrametric, i.e. $d_{ij} \le \max\{d_{ik}, d_{jk}\}$.

Note: "metric" means $d_{ij} \le d_{ik} + d_{jk}$ (triangular inequality)

Thus, ultrametric $\Rightarrow$ metric

Other Approaches

- Model Based Approaches
- Self-Organizing Maps (SOM): A neural network approach.
- Generalized Association Plots (GAP)

Specific Algorithms

- **Sorting**: Popular in computer science and most useful for categorical data.

  ➔ Choose an important variable and objects are partitioned according to its category.

- **Switching**: An initial partition is given, and new partitions are obtained by switching an object from one cluster to another, with the algorithm stopping when no further switches improves some criterion.

  ➔ *K*-means, model based approaches.

Specific Algorithms

- **Joining**: Begin with as many clusters as the number of objects in the data set (i.e. one cluster contains one object). Then find the "closest" pair of clusters and join them together to form a larger cluster.

  ➔ Some Hierarchical trees.

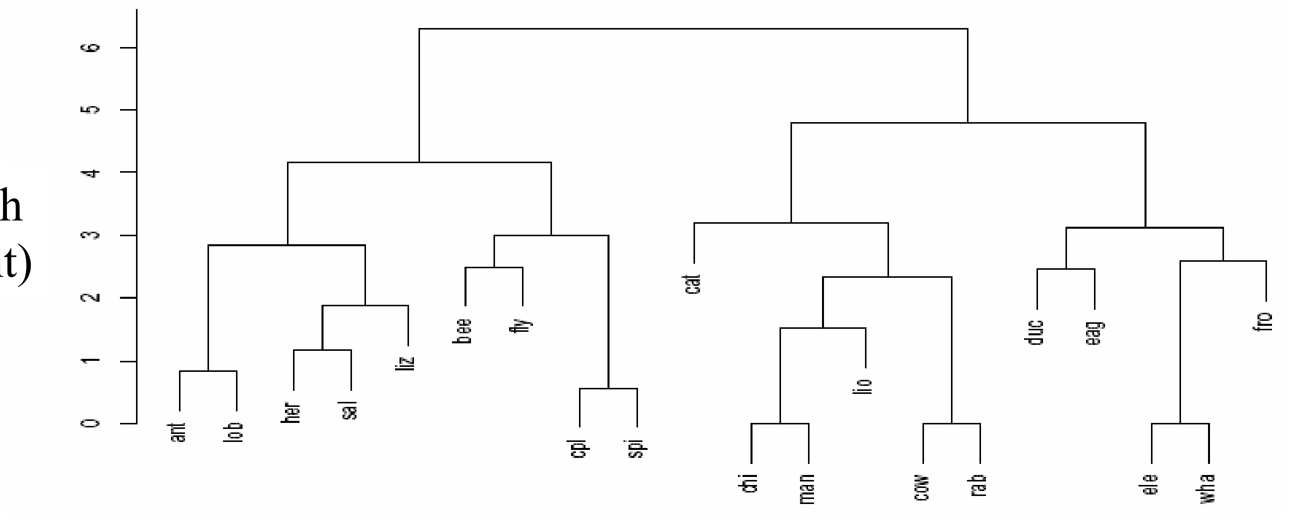
- **Splitting**: The inverse of "Joining Algorithm".

  ➔ Some Hierarchical trees (e.g. the idea of classification tree)

Agglomerative Hierarchical Clustering

- Tree-type methods.
- The clustering result is presented by a dendrogram, which is a rooted weighted tree *T*, with the leaves corresponding to the objects.

Length
(Height)



Some Notes for the Tree Methods

- For each pair of objects (*i*, *j*), $l_{ij} = l_{ji}$ is defined to be the length of the smallest subset containing both objects *i* and *j*.

- l_{ij} small ➔ high degree of similarity between objects *i* and *j*.

- l_{ij} satisfies the ultrametric inequality:

$$l_{ij} \le \max\{l_{ik}, l_{jk}\} \text{ for all objects } i, j, k.$$

Linkages for Joining Methods

Since groups are merged to form larger groups, we need a definition of distance between groups.

- Single Linkage: The dissimilarity between two groups is defined by

$$d(C_i, C_j) = \min_{s,t} \left\{ d(s,t) \mid o_s \in C_i,\ o_t \in C_j \right\}$$

- Complete Linkage:

$$d(C_i, C_j) = \max_{s,t} \left\{ d(s,t) \mid o_s \in C_i,\ o_t \in C_j \right\}$$

- Average Linkage:

$$d(C_i, C_j) = \frac{1}{n_i n_j} \sum_{s \in C_i} \sum_{t \in C_j} d(s,t)$$

Linkages for Joining Methods

- Wald's Method: The dissimilarity between two groups is defined by

$$d(C_i, C_j) = \sum_{s \in C_i \cup C_j} \| s - \mu \|^2$$

  where μ is the mean vector of all objects in $C_i \cup C_j$.

  Note: The Wald's method merges the two clusters with the minimum dissimilarity $d(C_i, C_j)$, which is equivalent to minimizing the within-cluster variation.

- Other Methods: Weighted average linkage.

An Illustrative Example

- Dissimilarity matrix
- N = 5, d_{ij} = euclidean distance.

$$\Delta = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & & & & \\ 2 & 9 & 0 & & & \\ 3 & 3 & 7 & 0 & & \\ 4 & 6 & 5 & 9 & 0 & \\ 5 & 11 & 10 & 2 & 8 & 0 \end{array}$$

Single Linkage (1)

- Step 1: Start with 5 clusters 1, 2, 3, 4, 5.

  First join cluster 3 and 5 ➔ (3, 5), 1, 2, 4

- Step 2: Calculate the distances between the cluster (3, 5) and the remaining clusters 1, 2, 4.

➔

$$d((35),1) = \min\{d(3,1), d(5,1)\} = \min\{3, 11\} = 3$$
$$d((35),2) = \min\{d(3,2), d(5,2)\} = \min\{7, 10\} = 7$$
$$d((35),4) = \min\{d(3,4), d(5,4)\} = \min\{9, 8\} = 8$$

The new dissimilarity matrix is then

$$D = \begin{array}{c} (35) \\ 1 \\ 2 \\ 4 \end{array} \begin{bmatrix} 0 & & & \\ 3 & 0 & & \\ 7 & 9 & 0 & \\ 8 & 6 & 5 & 0 \end{bmatrix}$$

Single Linkage (2)

- Step 3: Next, join cluster 1 and (3, 5) ➔ (1, 3, 5), 2, 4
- Step 4: Calculate the distances between the cluster (1, 3, 5) and the remaining clusters 2, 4.

➔ $d((1, 3, 5), 2) = 7$,
$d((1, 3, 5), 4) = 6$.

The new dissimilarity matrix is then

$$D = \begin{array}{c} (135) \\ 2 \\ 4 \end{array} \begin{bmatrix} 0 & & \\ 7 & 0 & \\ 6 & 5 & 0 \end{bmatrix}$$

Single Linkage (3)

- Step 5: Next, join cluster 2 and 4 ➔ (1, 3, 5), (2, 4)
- Step 6: Calculate the distances between the cluster (1, 3, 5) and (2, 4).

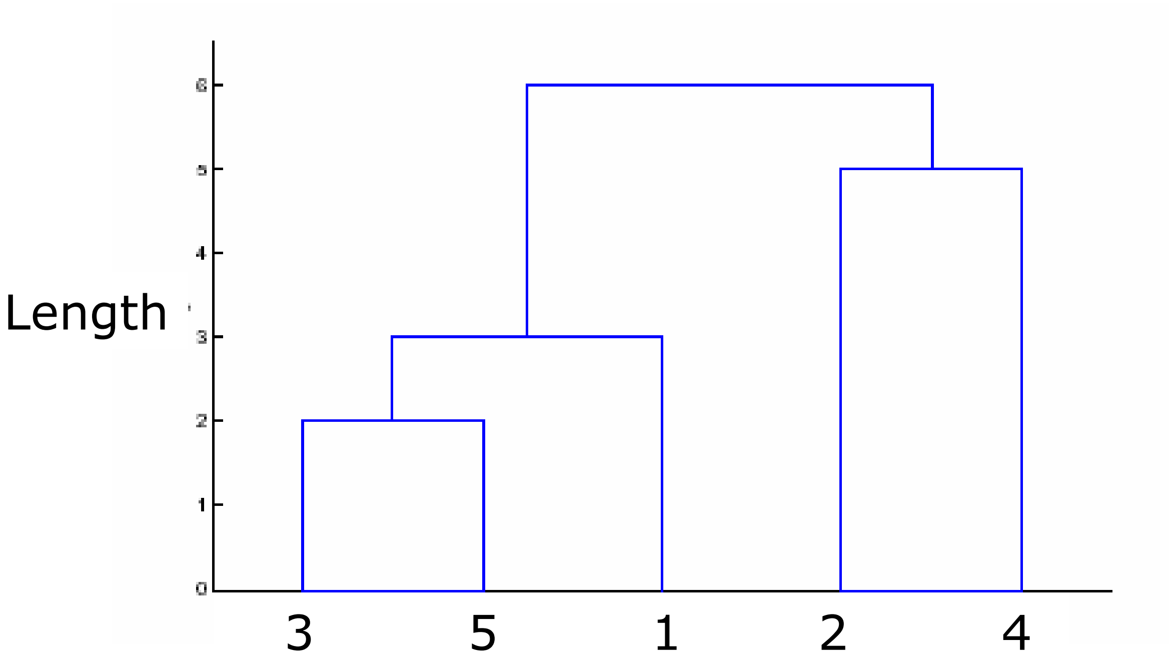
➔ $d((1, 3, 5), (2,4)) = 6$

The new dissimilarity matrix is then

$$D = \begin{array}{c} (135) \\ (24) \end{array} \begin{bmatrix} 0 & \\ 6 & 0 \end{bmatrix}$$

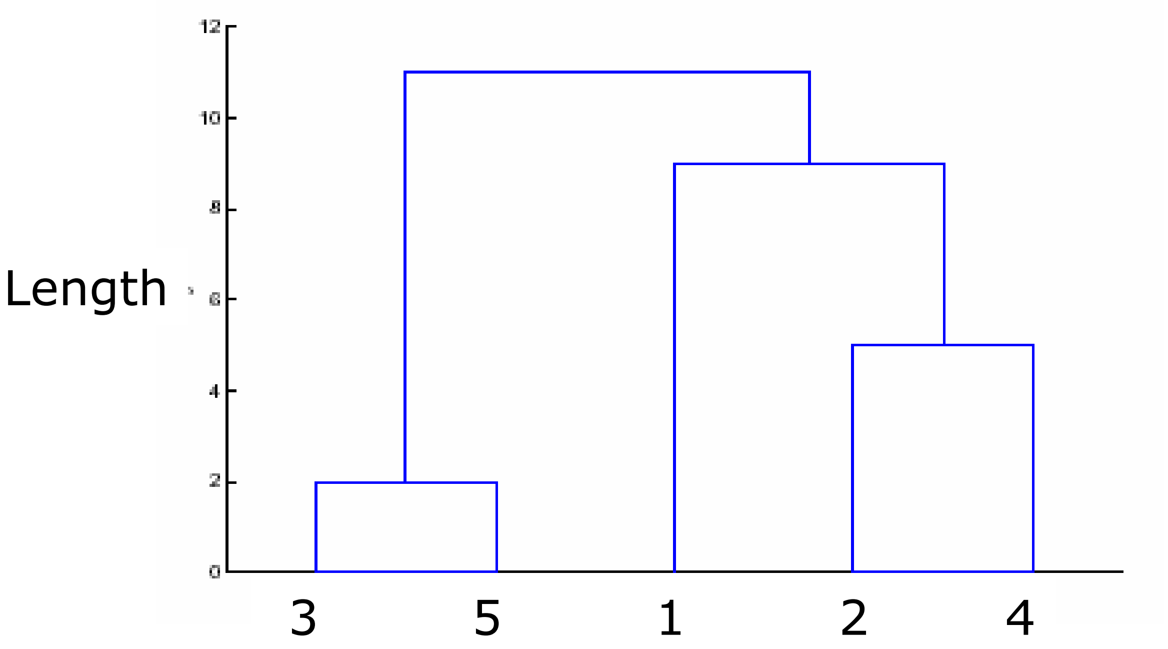
Single Linkage (4)

- The resulting tree:



Complete Linkage

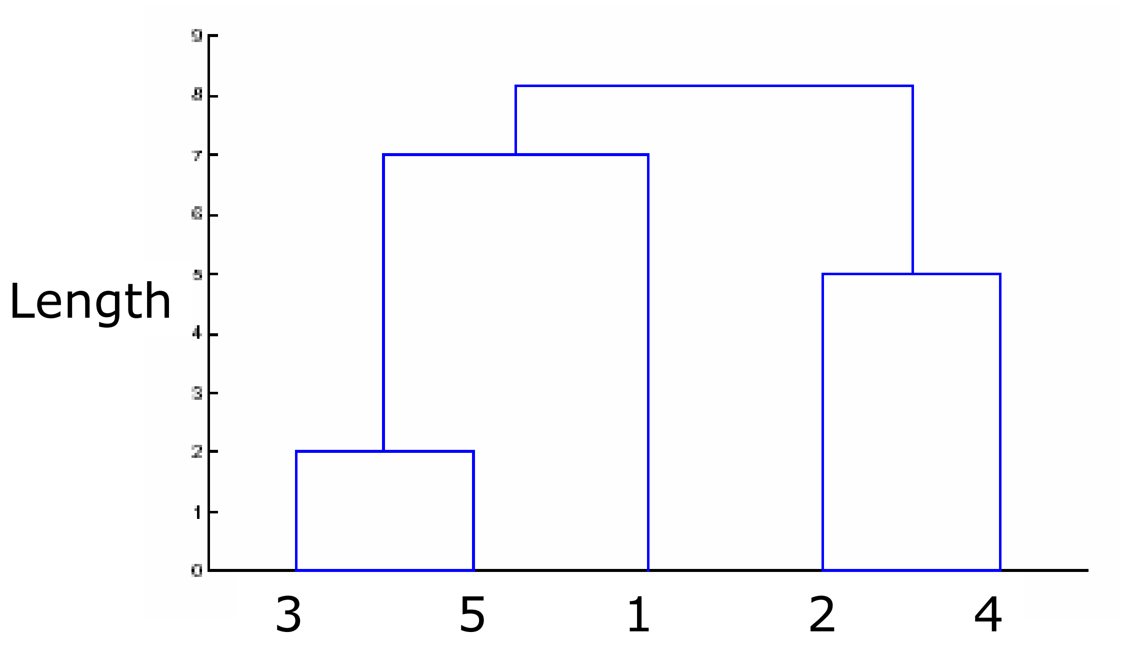
- The resulting tree:



- **Q**: How do you perform this picture?

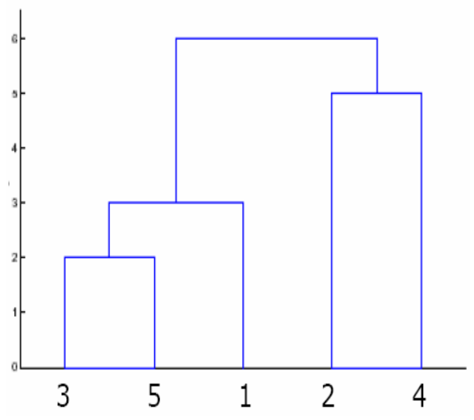
Average Linkage

- The resulting tree:

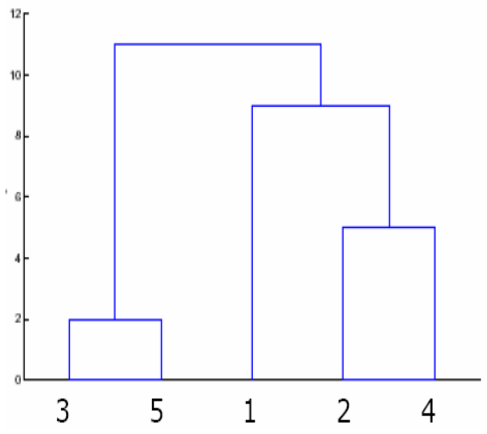


- **<u>Q</u>**: How do you perform this picture?

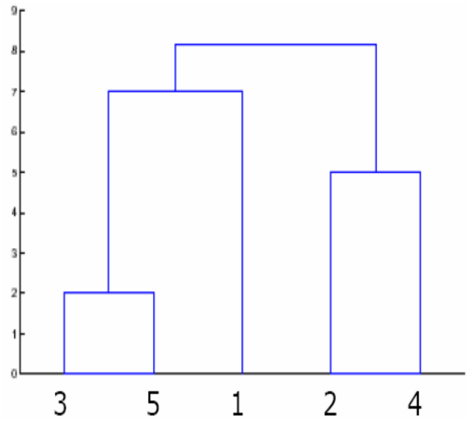
Comparison of Clustering Results



Single Linkage



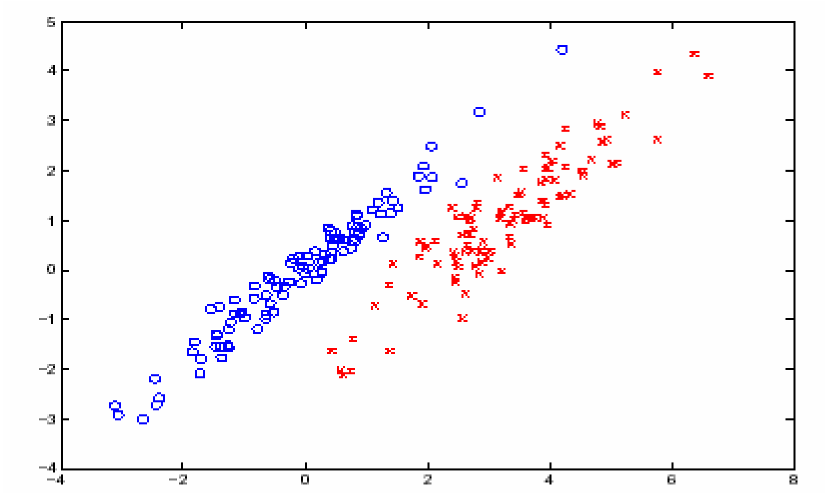
Complete Linkage



Average Linkage

Remarks

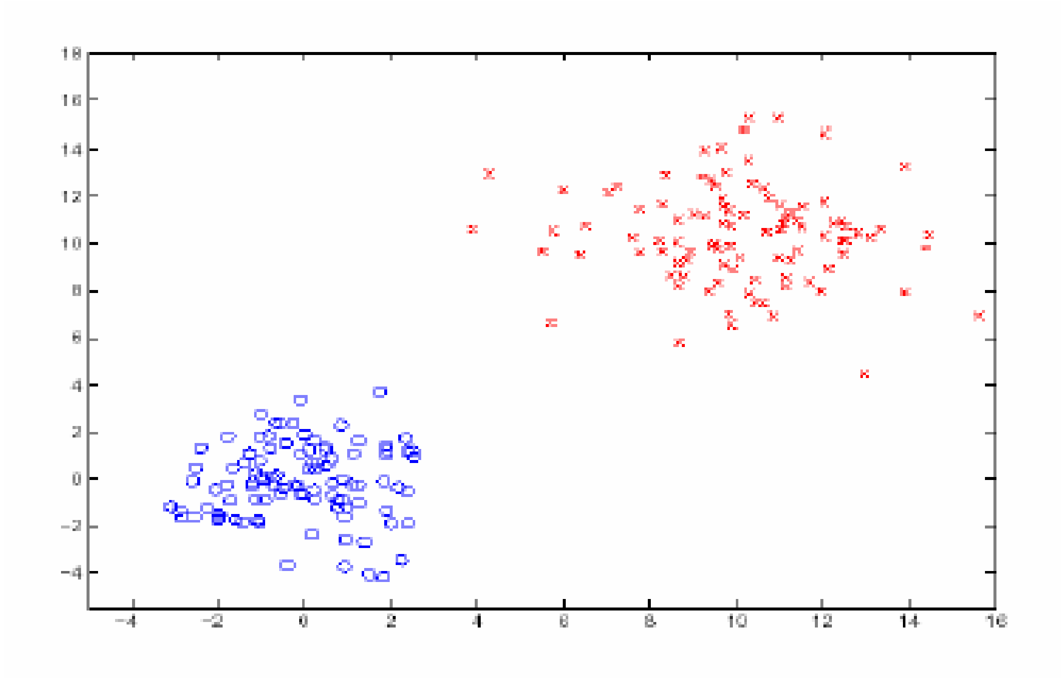
- Single linkage tends to produce long and loosely connected clusters.



(For example, two data sets from two normal populations.)

Remarks

- Complete linkage tends to produce compact clusters since it joins two clusters iff all objects of one cluster are close to the other cluster.



Remarks

- In Ward's method, clusters are formed at each step so that the resulting clustering has the smallest within-cluster sum of squared distances.

  ➔ Maximize the total within-cluster homogeneity.

- Single and Complete linkages are invariant to monotone transformations, while Average linkage is NOT invariant to monotone transformations.

  For example,

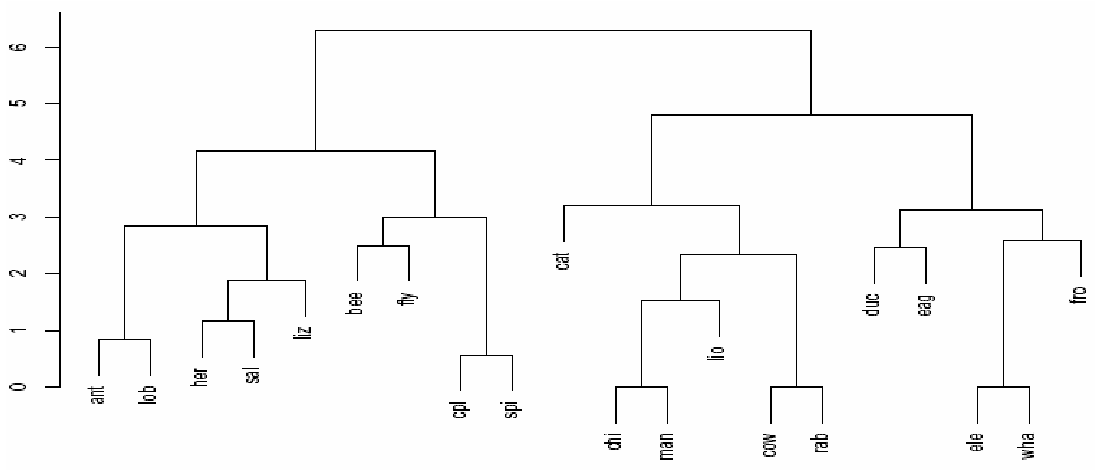
$$\frac{0.5+5}{2} < \frac{3+3}{2} \not\Rightarrow \frac{(0.5)^2+5^2}{2} < \frac{3^2+3^2}{2}$$

The Hierarchical (Tree) Methods in R

- **Agglomerative Nesting**: agnes( ), similar to hclust( ).
  ➔ use "single", "complete", "average", and other linkages.

Display:

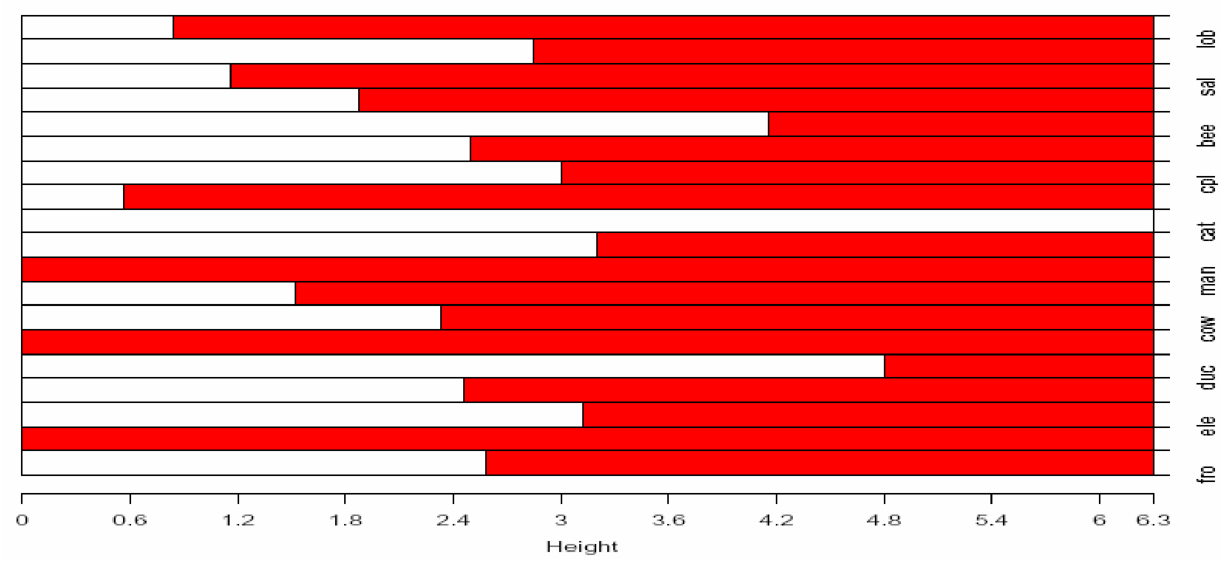
- Agglomerative Tree (or Dendragram)



Agglomerative Nesting

<u>Display</u>:

- Agglomerative Banner (using horizontal bars to display the dissimilarity)



Agglomerative Nesting

Display:

- Agglomerative Coefficient (AC)

Let $d(i)$ = dissimilarity of object i to the first cluster it is merged with.

Let L = the length of the dendrogram in the final merge.

Let $m(i) = d(i) / L$ ➔ $0 < m(i) \leq 1$

Ideally, a good clustering will result small $m(i)$ (or large $1 - m(i)$).

Let AC = average of all $1 - m(i)$.

If $\text{AC} \to 1$, the clustering is good.

Rule of Thumb (for AC)

$$\begin{cases} 0.71 \le \text{AC} \le 1 & \Rightarrow \text{ strong structure found} \\ 0.51 \le \text{AC} \le 0.7 & \Rightarrow \text{ reasonable structure} \\ 0.26 \le \text{AC} \le 0.5 & \Rightarrow \text{ weal structure} \\ \text{AC} \le 0.25 & \Rightarrow \text{ no substantial structure} \end{cases}$$

- **Notes**: AC grows with the number of objects *N*.

$$N \uparrow \Rightarrow L \uparrow \text{and } d(i) \downarrow \Rightarrow m(i) \downarrow \Rightarrow 1 - m(i) \uparrow \Rightarrow \text{AC} \uparrow$$

So, do not use AC to compare data sets with (very) different sizes.

The Hierarchical (Tree) Methods in R

- **Monothetic Analysis**: mona( ).
  - Operate the data matrix with binary variables
  - Splitting method – each split uses a “well-chosen” variable

  Display:
  - Agglomerative Tree
  - Agglomerative Banner
  - AC

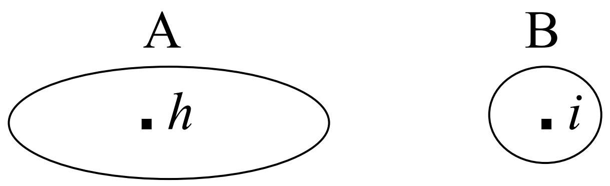
The Hierarchical (Tree) Methods in R

- **Divisive Analysis**: diana( ).
  - Splitting method
  - Using the algorithm by Macnaughton-Smith et al. (1964)

Step1: Start with one cluster A, let $a(i)$ = the average distance of object i to all other objects in A.

Step 2: If $a(i)$ is maximal, move object i to a new cluster B.

Step 3:



If $a(h) > \overbrace{d(h, \mathrm{B})}^{\text{Average dist of } h \text{ to all objects in B}}$ ➔ move object h to cluster B.

Otherwise, keep object h in A.

Divisive Analysis: diana( )

Display:

- Divisive tree (dendrogram)
- Divisive banner
- Divisive Coefficient (DC)

Let $\mathrm{diam}(\mathrm{A}) = \max_{i,j \in \mathrm{A}} \{d(i, j)\}$, and

$$d(i) = \frac{\mathrm{diam}(\text{the last cluster to which } i \text{ splits as a single cluster})}{\mathrm{diam}(\text{whole data})}$$

Then DC = Average (1- $d(i)$).

If DC → 1, the clustering is better.

Some Remarks

- Tree methods are sensitive to outliers.

  The stability of a tree solution is sometimes checked by applying the algorithm before and after small perturbations have been added to the data.

  → If the groups are well distinguished, the clustering before and after perturbation should agree.

  加入\移除資料後分群的結果沒有太大差異 -> 分的不錯

- Ties in the dissimilarity matrix can produce multiple solutions to a hierarchical clustering problem.

Optimization Methods

- Points are assigned to clusters with the objective of optimizing some criterion.
- Decompose the variation as $T = W + B$ where

$$T = \underbrace{\sum_{\text{all object } i} (x_i - \bar{x})(x_i - \bar{x})'}_{\text{Total variation}}, \quad W = \underbrace{\sum_{\text{all cluster } k} \sum_{i \le n_k} (x_i - \bar{x}_k)(x_i - \bar{x}_k)'}_{\text{Within-cluster variation}}.$$

- Given T, a good clustering algorithm seeks to minimize some measure of W and hence maximize some measure of B.

Some Criteria in Literature

- Minimize trace(W) – not invariant to changes in scale
- Minimize det(W)
- Maximize the sum of the eigenvalues of $W^{-1}B$

Problem of Computation

Question: How many ways to allocate 100 objects to 2 groups ?

(or, what is the best allocation to optimize the criterion ?)

Answer: There are $2^{100} \approx 10^{30}$ possible allocations.

➔ Exhaustive search is not a practical option !!

The K-means Procedure

Step 1: Start by fixing the number of clusters, say, *K*.

(this can be done by referencing the tree solution or PCA)

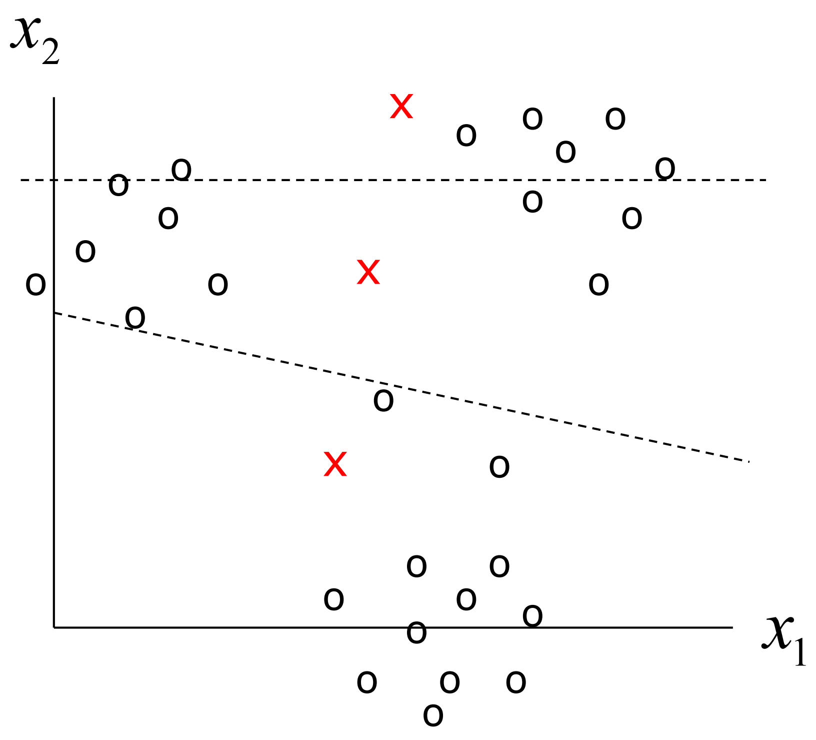
Step 2: Randomly choose *K* means and make an initial partition of *K* clusters using Criterion 1 (maximizing within-class sum of squared distances from those centroids).

Step 3: Move the *K* means to the centroids ($\bar{x}$) of the data in the current partition.

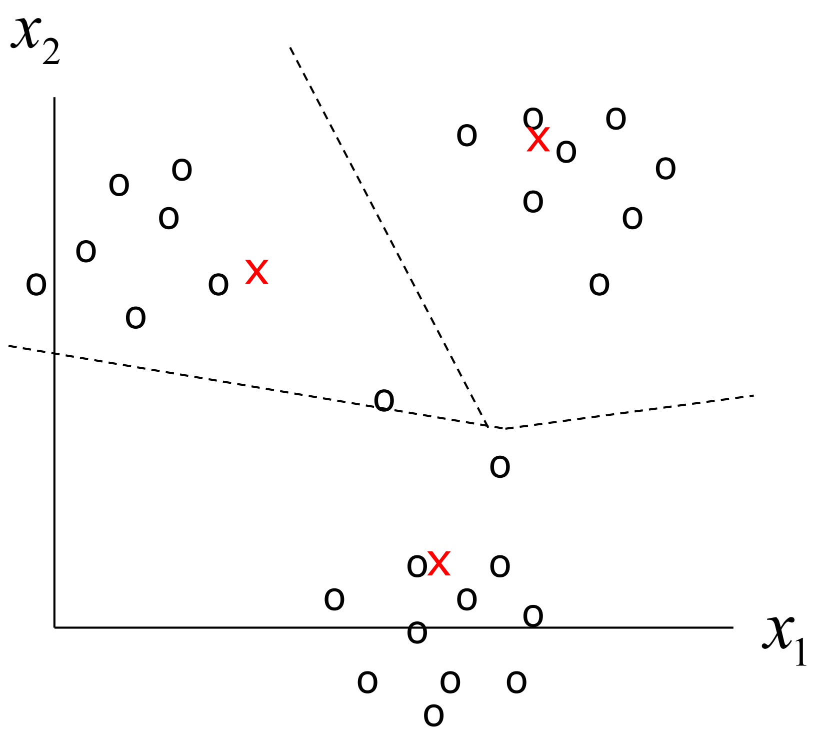
Step 4: Partition the data according to the current values of the means.

➔Repeat Step 3 and 4 until the partitions remain the same.

Example of K-means (K=3)

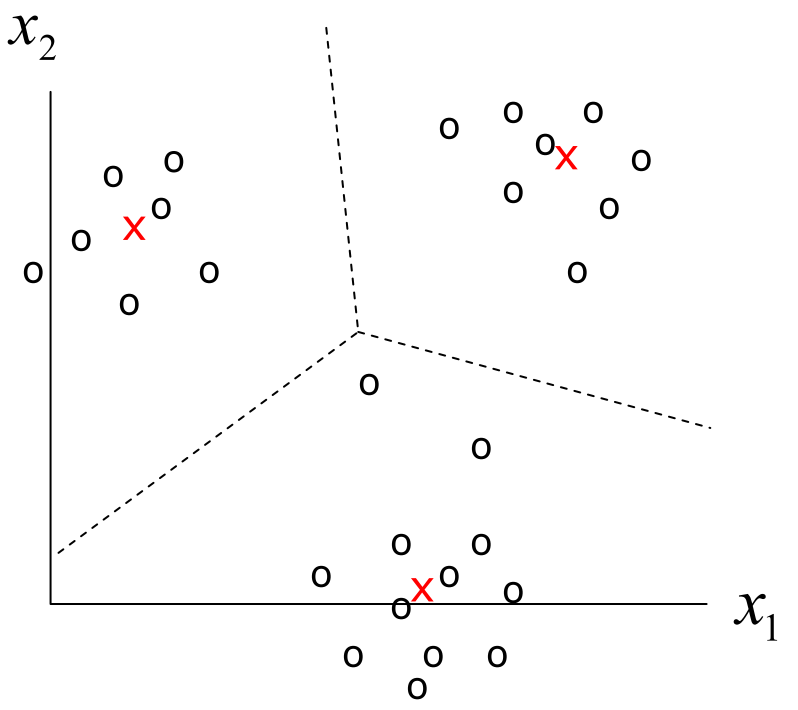


Initial centroids, initial partition



New centroids, new partition

Example of K-means (K=3)



Final centroids, final partition (stop)

Virtues of K-means

- Simplicity and generality

Remarks:

- Can perform poorly with overlapping clusters
- The K-means method is sensitive to outliers
- The algorithm can stop at a local minima
- ~~Usually $K \leq \sqrt{N}$, where N is the total number of objects.~~

Partitioning Methods in R

- **Partitioning Around Medoids**: pam( )
  - A more robust version of K-means by Kaufman and Rousseeuw (1987).
  - Use medoids instead of centroids in K-means.

The centroids are μ_0 that minimize $\sum_{i=1}^{n}(x_i - \mu_0)^2$

The medoids are μ_0 that minimizes $\sum_{i=1}^{n}|x_i - \mu_0|$ robust

➔ For univariate case, they correspond to mean and median, respectively.

Partitioning Around Medoids

Display:

- Show clusters on the PC1-PC2 space.
- Silhouette Plot (Rousseeuw, 1987)

The silhouette value $s(i)$ of object $i \in$ cluster A is defined as

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad \text{where}$$

正1 –> 最好

$a(i)$ = average distance of i to all other objects in A

$b(i)$ = average distance of i to all objects in the nearest neighbor cluster B

➔ $-1 \le s(i) \le 1$

Silhouette Coefficient (SC)

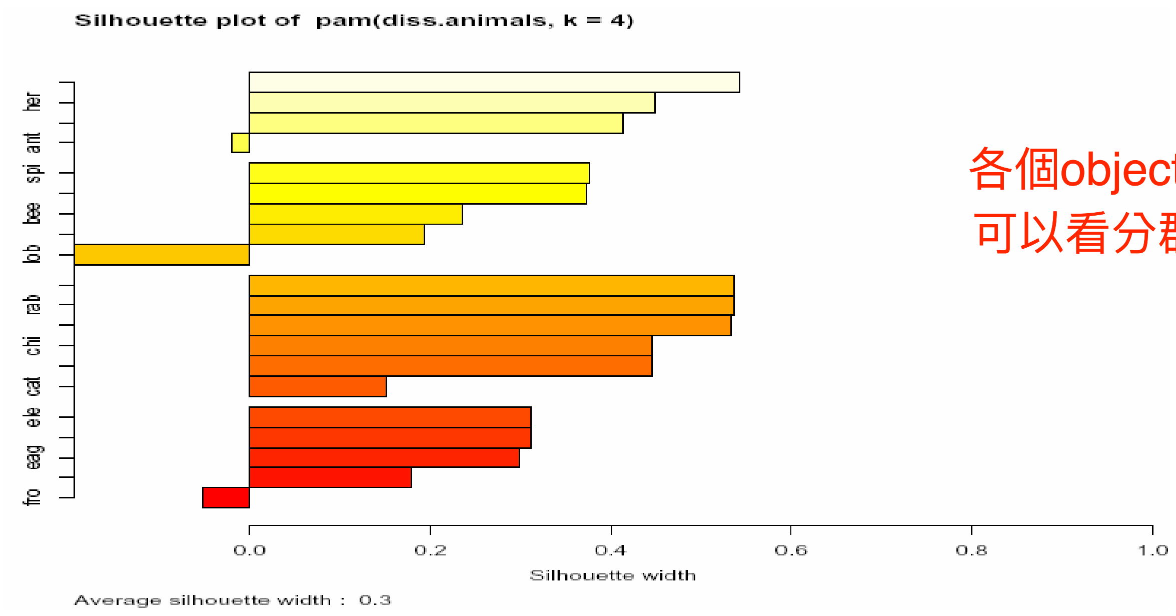
Note that

$$\begin{cases} s(i) \approx 1: \text{object } i \text{ is well classified in A} \\ s(i) \approx 0: \text{object } i \text{ lies between A and B} \\ s(i) \approx -1: \text{object } i \text{ is bad classified in A} \end{cases}$$

- Silhouette Coefficient: SC = average silhouette width

$$\begin{cases} 0.71 \le \text{SC} \le 1 \quad \Rightarrow \quad \text{strong structure found} \\ 0.51 \le \text{SC} \le 0.7 \quad \Rightarrow \quad \text{reasonable structure} \\ 0.26 \le \text{SC} \le 0.5 \quad \Rightarrow \quad \text{weal structure} \\ \text{SC} \le 0.25 \quad \Rightarrow \quad \text{no substantial structure} \end{cases}$$

Silhouette Coefficient (SC)



各個object的S(i)
可以看分群好壞

Q: How to choose the best number of clusters ?

- The best partition is to choose *k* with the highest SC.
- SC is not affected by the # of objects, which is unlike AC.

Other Approaches in R

- **Clustering Large Applications**: clara( )
  - By Kaufman and Rousseeuw (1986)
  - Computation time is $O(n)$ (better than pam( ), $O(n^2)$)
  - Best used for large data sets.
- **Fuzzy analysis**: fanny( )
  - Fuzzy version of K-means
  - Each object can be assigned to different clusters (with different probabilities)

Model Based Approach

- **Idea**: Assume objects are independent samples from a series of group populations, but group labels are lost.

  ➔ This turns the problem into a density-estimation problem.
- **Approach**: Given $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_K)$ to be the group labels, and each group k has pdf $f_{\alpha_k}(x_k;\theta)$.

  Thus, the likelihood function can be written as

  $$L(\theta, \alpha; x) = \prod_k f_{\alpha_k}(x_k;\theta)$$

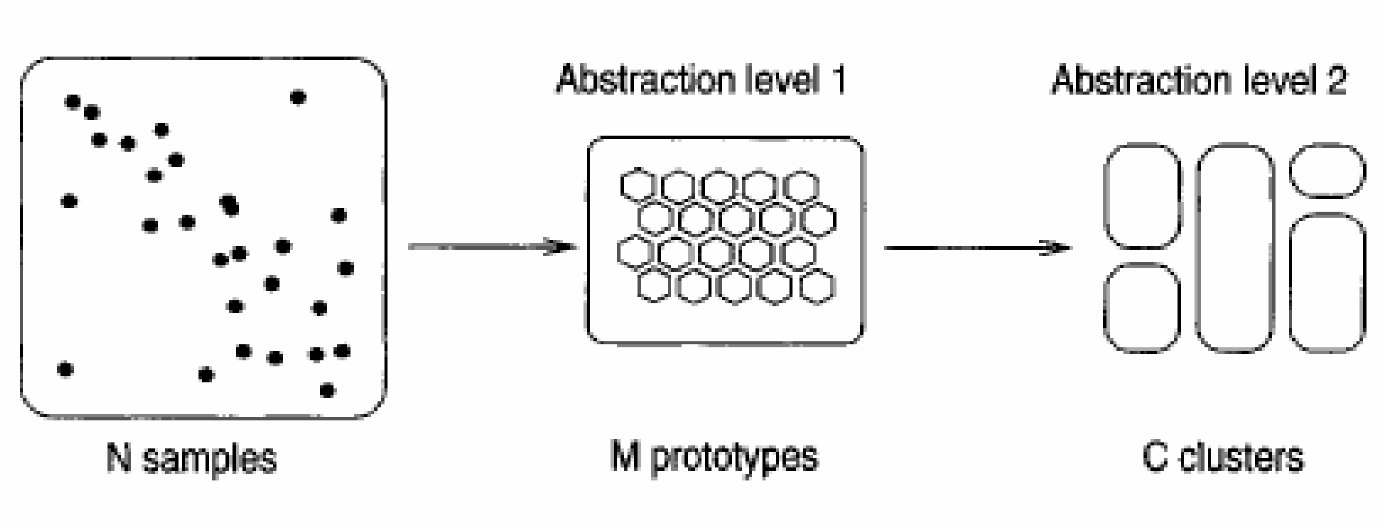
  The goal is to maximize $L(\theta, \alpha; x)$ over the parameters (θ, α).
- Assume $f_{\alpha_k}(x_k;\theta)$ are normal with common covariance matrix ➔ The solution = K-means method.

Other Clustering Approaches

- **Self-Organizing Maps (SOM)**
  - Originated in 1982 by Kohonen
  - SOM is a neural network algorithm
  - For micro-array data, see the paper by Golub et al. in 1999.
  - Reference book: Kohonen's book, 1995, 1997.
- **Generalized Association Plots (GAP)**
  - See Chen 2002, Ins. Of Statistics, Academia Sinica.

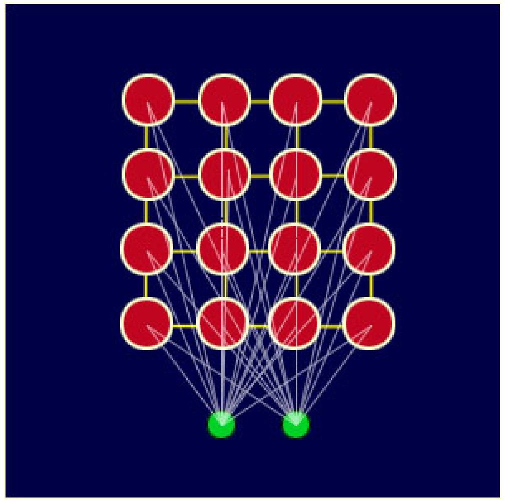
Picture of Self-Organizing Maps

- A large set of prototypes is placed in a 2-D lattice of nodes. (the # of prototypes is much larger than the expected # of clusters)
- Objects lying near each other in the input space are mapped onto nearby prototypes.



Kohonen's Self-Organizing Maps

- Network Architecture: The network is created from a 2D lattice of nodes (or neurons), each of which is fully connected to the input layer:



Kohonen network of 4×4 nodes connected to the input layer (shown in green) representing a two dimensional vector.

Kohonen's Self-Organizing Maps

- Each node has a specific topological position (an (*x*, *y*) coordinate in the lattice) and contains a vector of weights of the same dimension *m* as the input vectors.

  For example, suppose each node *i* has a weight vector

$$w_i = (w_{i1}, \ldots, w_{im}).$$

- Idea: To allocate similar objects to the nodes having similar weight vectors. Note that similar nodes are adjacent to each other and thus form a class of stable zones/clusters (of the same color) in the lattice.

Learning Algorithm Overview

Step 1: Each node's weights (i.e.w_{ij}) are initialized.

Step 2: An object x is randomly selected from the training data and its distances to all nodes (i.e., $\| x - w_i \|$) are calculated.

Step 3: Identify the Best Matching Unit (BMU), which is the node b such that $w_b = \min_i \| x - w_i \|$.

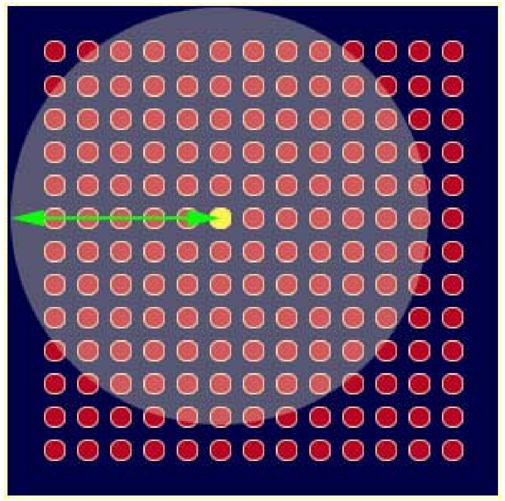
Step 4: Update the weights of BMU and its neighbors so that they are moved closer to the selected object x. The closer a node is to the BMU, the more its weights are adjusted.

Step 5: Go back to Step 2 for N iterations.

Learning Algorithm in Detail

Let’s explain how to perform Step 4 in detail:

- After identify the BMU, how to determine its neighbors?



- As shown in the picture, the neighbors are centered around the BMU (yellow color) for a specified radius α_0.

Shrinkage of Neighborhood

- A unique feature of the Kohonen learning algorithm is that the area of the neighborhood shrinks over time.

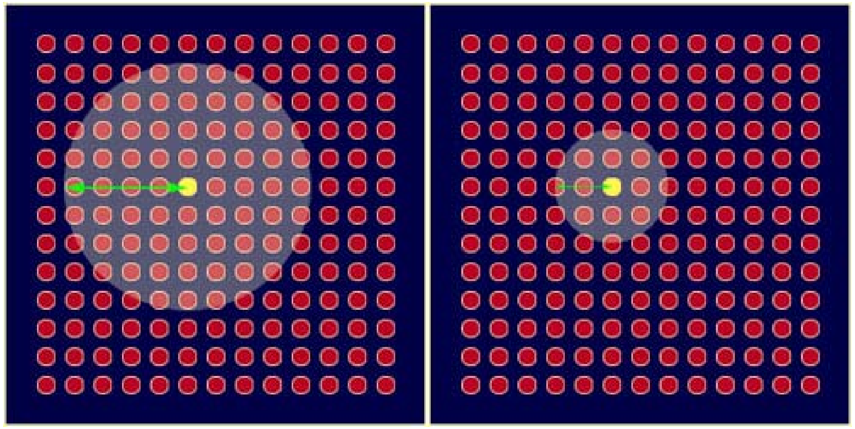
  For example, consider the exponential decay function for the radius:

$$\sigma(t) = \sigma_0 \exp\left\{-\frac{t}{\lambda}\right\}, \quad t = 1, 2, 3, \ldots$$

更新的範圍

  where t is the current time-step (iteration of the loop).

- The shrinkage of radius:



Updating the Weights

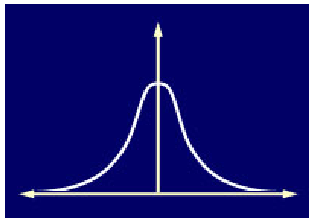
- Every node *i* within the BMU's neighborhood (including the BMU *b*) has its weight vector updated according to:

$$w_i(t+1) = w_i(t) + \alpha(t) h_{bi}(t)[x - w_i(t)],$$ 最後會希望收斂

where $\alpha(t) = \alpha_0 \exp\left\{-\frac{t}{\lambda}\right\}$ (called the learning rate) and

更新的範圍越來越小

$$h_{bi}(t) = \exp\left\{-\frac{\| r_i - r_b \|^2}{2\sigma^2(t)}\right\}$$



衡量node之間的相似度 越遠更新越少

is a neighborhood kernel centered on *b* (BMU), which represents the amount of influence based on the distance from node *i* to the BMU *b*.

Some Remarks

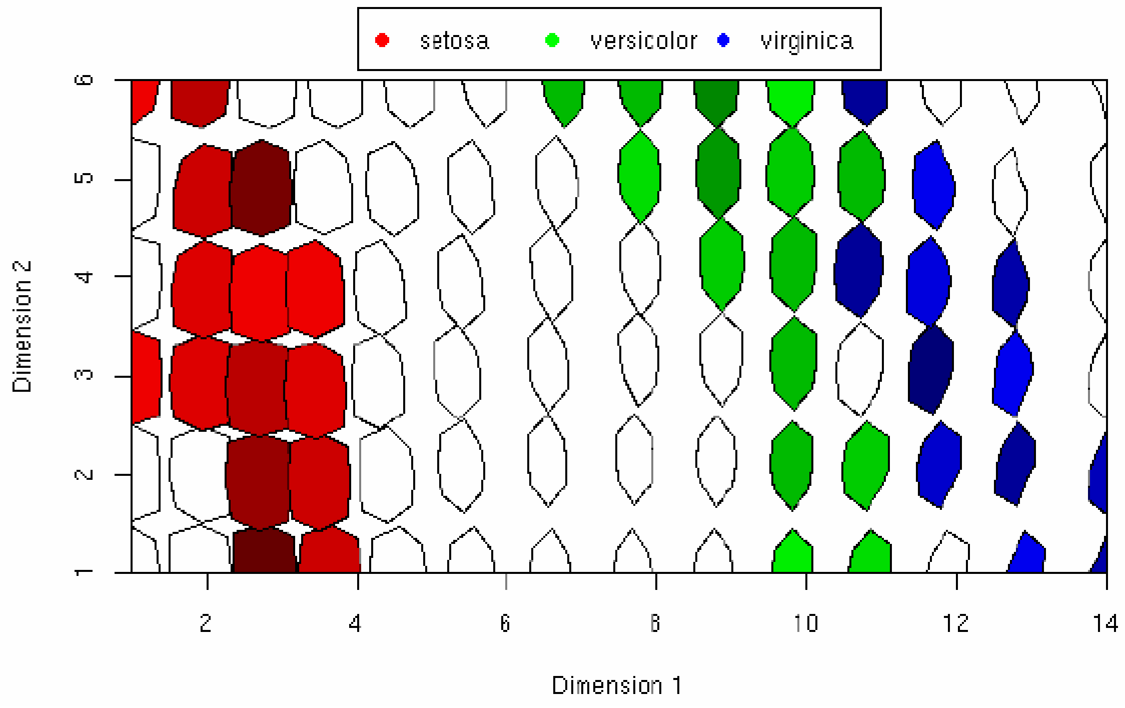
- Note that different choices of decay functions for the learning rate, neighborhood radius, and neighborhood kernel (Gaussian kernel is a common choice) can influence the clustering result.
- The resulting weight vector of each node can be replaced by colors (for visualization purpose).
- SOM may be considered a nonlinear generalization of PCA.
- If the kernel function is chosen to be $h_{bb}(t)=1$ and $h_{bi}(t)=0$ for $i \neq b$, then SOM reduces to adaptive K-means algorithm. 除了BMU以外都不更新 -> adaptive k means

會遇到data全部丟完都不收斂 -> 用resampling讓他收斂

data先做標準化

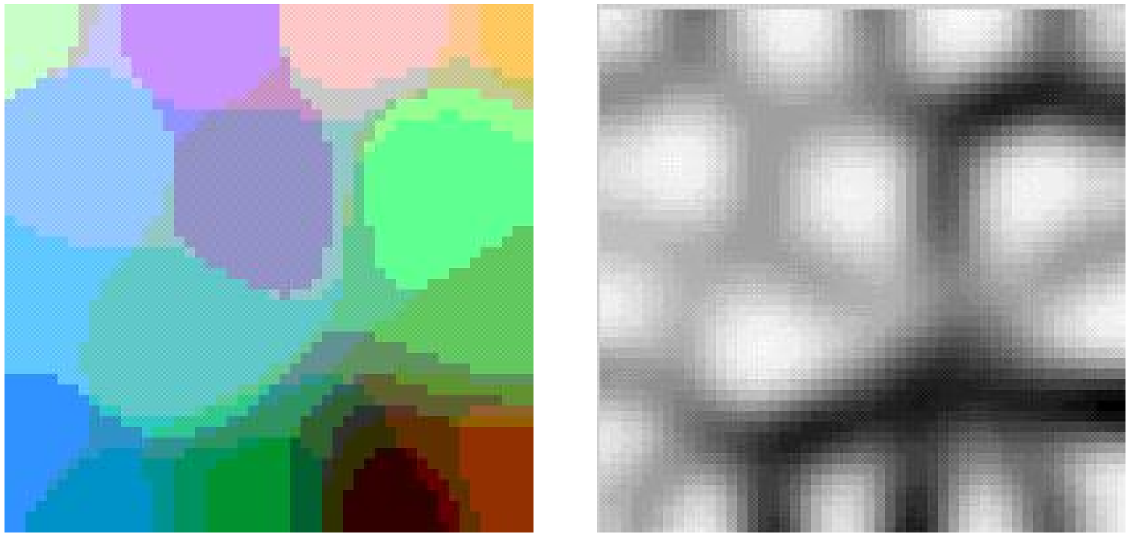
Example 1: Iris Data (plotted by R)

- Go through all weights of each node and determine how similar the neighbors are by calculating the distance between the weight vectors.
- With an average of all distances, a color is assigned to the node.



Example 2:

- The following is another example of SOM generated by the java program using 500 iterations.



Other Comments on SOM

<u>Advantages of SOM</u>:

- Easy to understand: If they are close together and there is grey connecting them, then they are similar. If there is a black ravine between them, then they are different.
- You can actually calculated how good a map is and how strong the similarities between objects are.

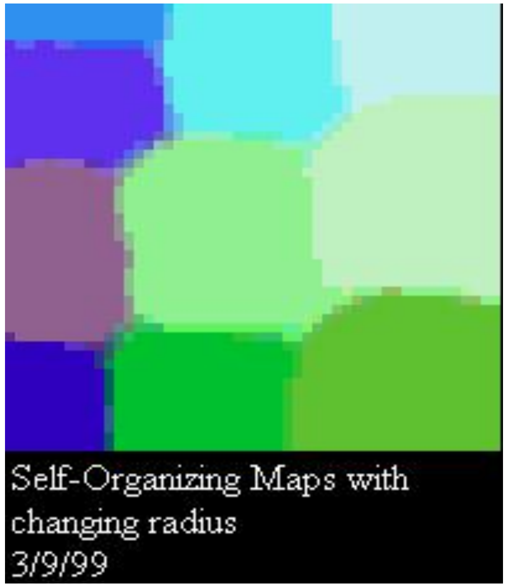
<u>Disadvantages of SOM</u>:

- Solutions are not unique (initial values, choice of parameters)

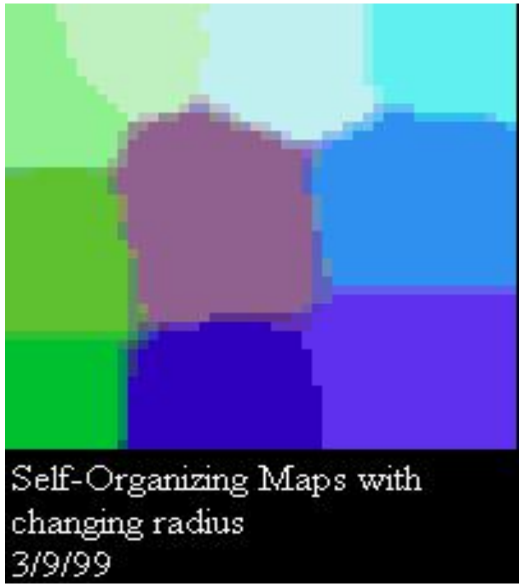
  ➔ Can produce many clustering results!!

Other Comments on SOM

(a)



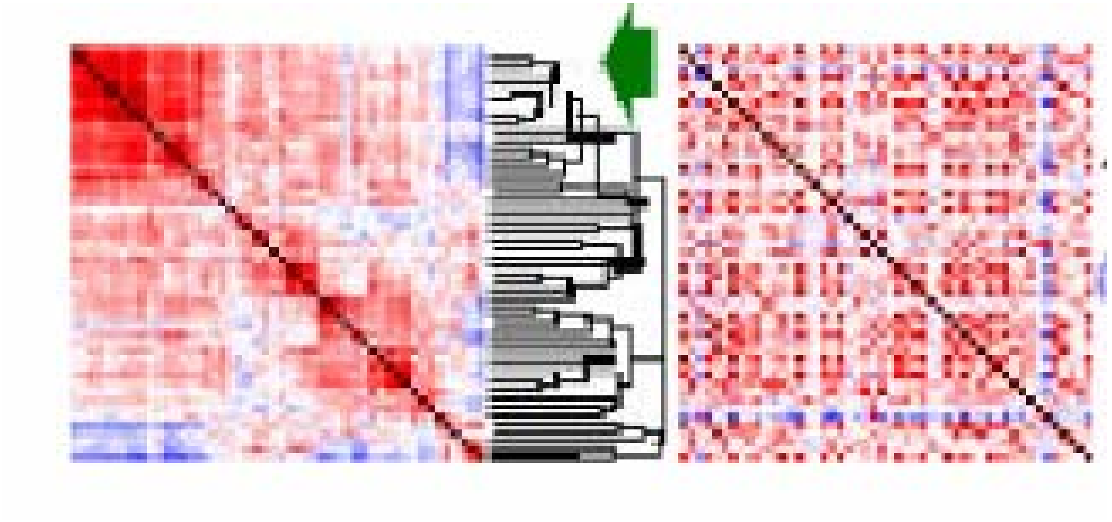
(b)



- Similar samples/clusters are not always near each other (can be splitted).
  For plot (a), the two deep purple areas are more similar than the light purple area.
  So, they should be near (but they are actually separated).
  To find a good clustering, we might have to produce a lot of SOMs so as to get a better explanation (e.g. plot (b))

GAP (An Example)

- Presented by data sorting.



- URL: http://gap.stat.sinica.edu.tw/
- Color presentation are particularly useful for some applications.

Some Important Issues

- Standardize or not ?
  - If all variables are considered equally important, standardize the variables is a good idea.
  - If some variables are considered more important

    → Use weight
  - Scaling might affect the grouping result.
- Variable Selection
  - No theoretical basis for the choice of variables
  - More variables do not necessarily lead to a better grouping.
  - Can apply PCA (or GGobi) to access the relationships between variables.

Number of Clusters

No standard procedure exists !!

One can reference the quality index such as SC (e.g. partitioning around medoids), B/W, or Davies-Bouldin index (DBI, see Davies, David L.; Bouldin, Donald W. (1979). "A Cluster Separation Measure". *IEEE Transactions on Pattern Analysis and Machine Intelligence*. PAMI-1 (2): 224 – 227).

Evaluation of Clustering

- **Clustering Quality**: Check quality index like AC, DC, SC, or B/W, etc.

- **Clustering Stability**: Check out the clustering result before and after adding small perturbations to the original data set.
  加入一些奇怪的data –> 若結果不怎影響 -> 分群方法不錯穩定

- **Computational Speed**: Especially for large data sets.