Cluster analysis

Yusen Ye

School of Computer Science and Technology

Xidian University

Roadmap

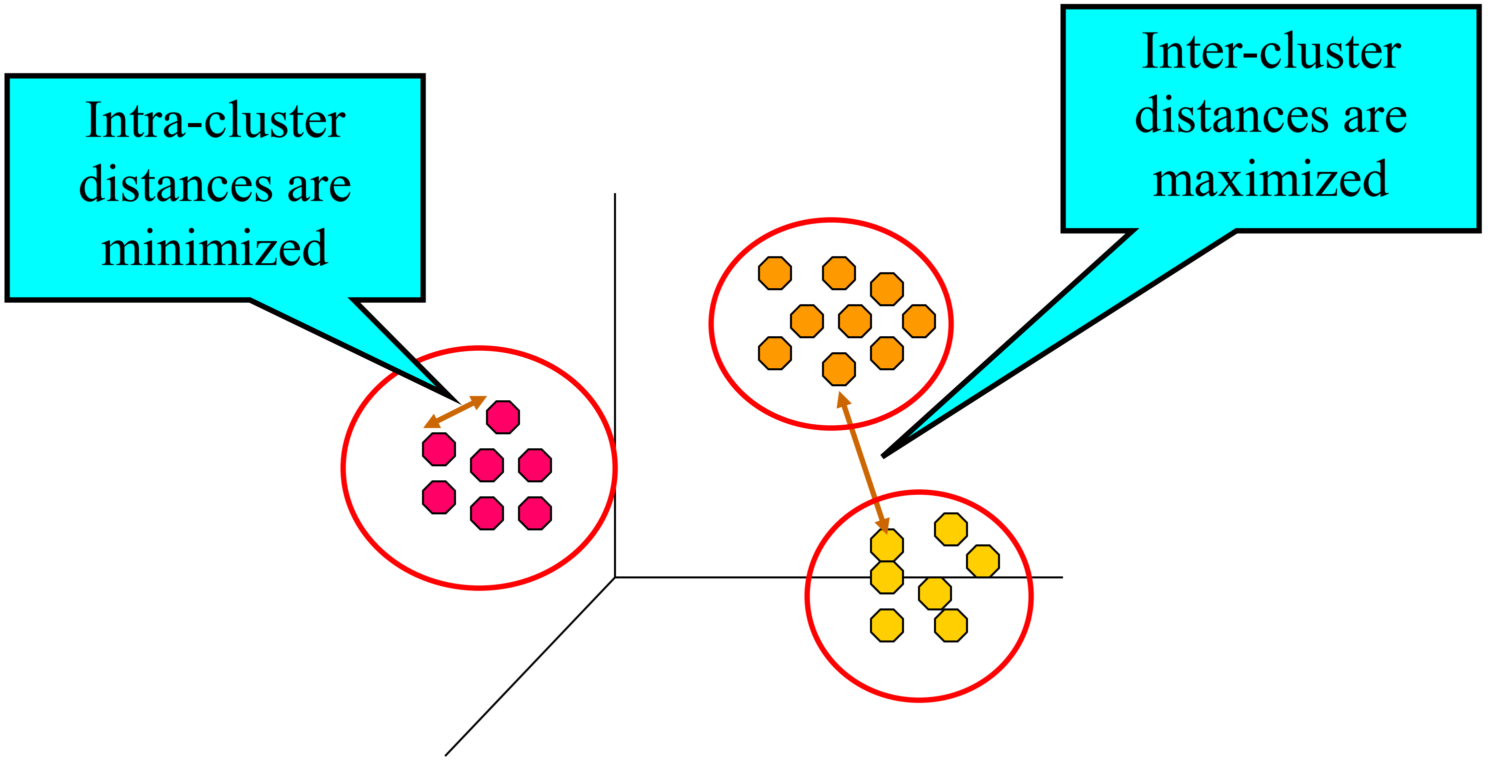
- What is Cluster Analysis? (Sec 8.1)
- Partitioning Methods (Sec 8.2)
- Hierarchical Methods (Sec 8.3)
- Density-Based Methods (Sec 8.4)
- Cluster Evaluation (Sec 8.5)

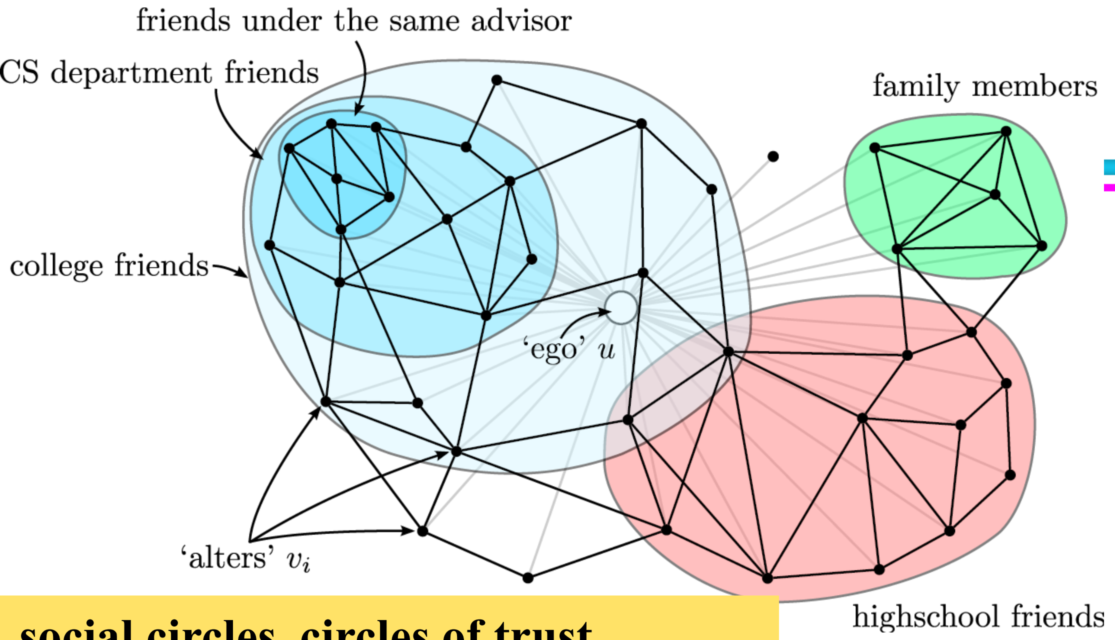
What is Cluster Analysis?

- **Cluster: a collection of data objects**
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters
- **Cluster analysis**
  - Grouping a set of data objects into clusters
- **Clustering is unsupervised classification**
  - No predefined classes
- **Typical applications**
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

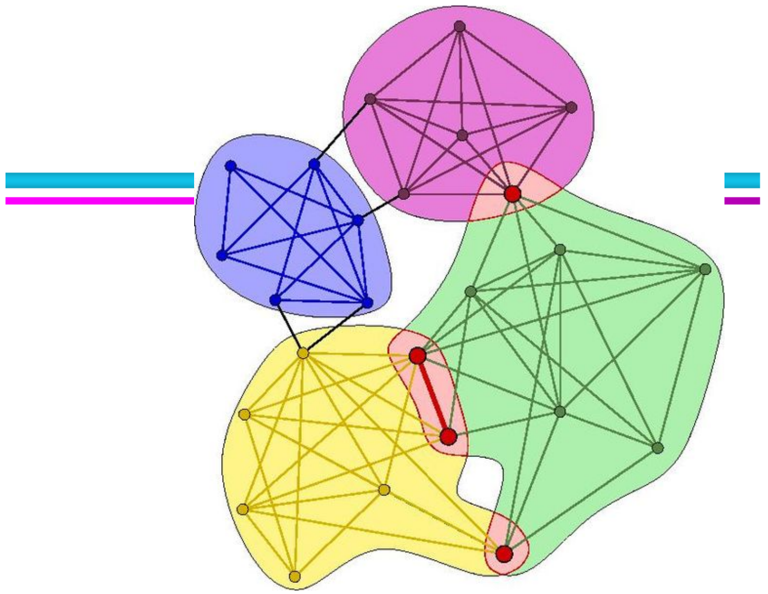
What is Cluster Analysis?

- Finding groups of objects such that the objects in a group will be ***similar*** (or related) to one another and ***different*** from (or unrelated to) the objects in other groups

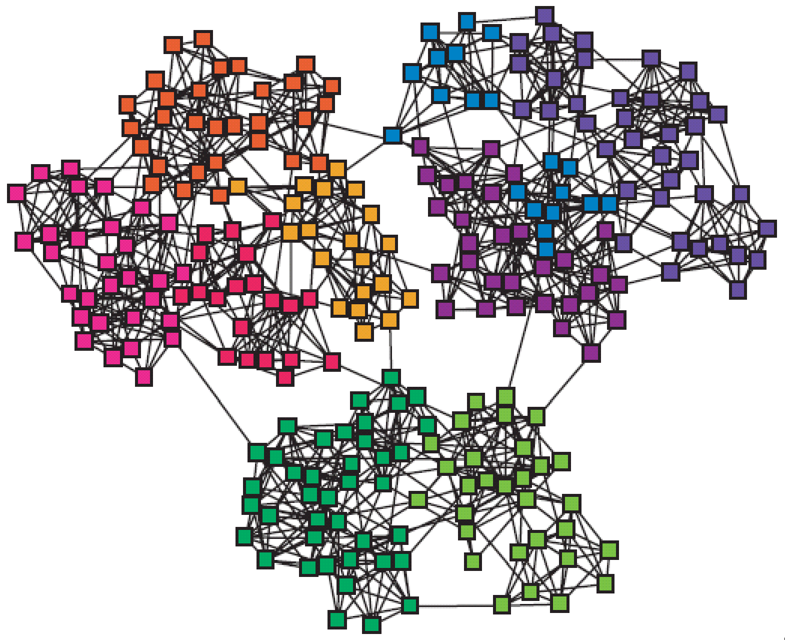

Intra-cluster distances are minimized
Inter-cluster distances are maximized

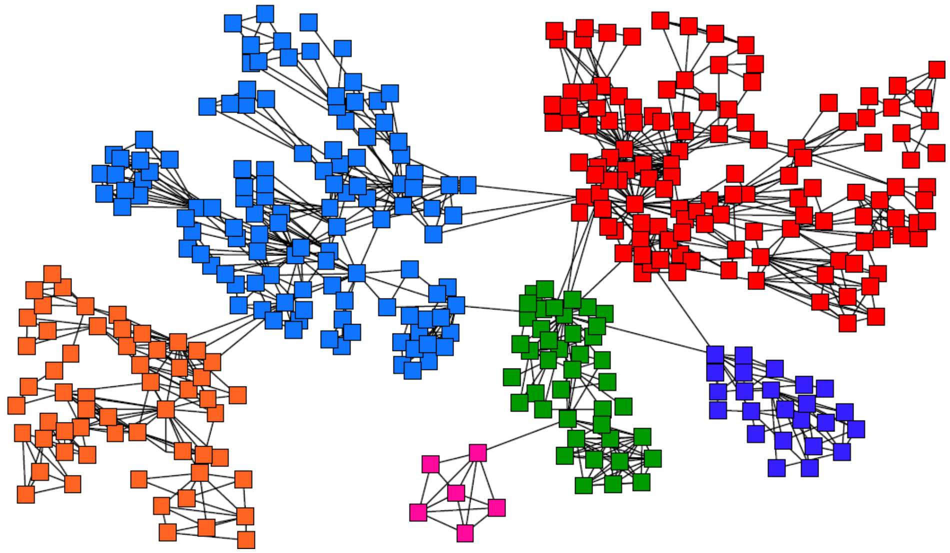


social circles, circles of trust



Overlapped communities





modules, cluster, communities

Densely Linked Clusters

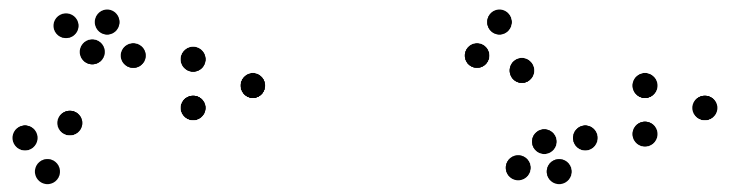
What Is Good Clustering?

- A **good clustering** method will produce high quality clusters with
  - high intra-class similarity : **cohesive** within clusters
  - low inter-class similarity : **distinctive** between clusters
- The **quality** of a clustering result depends on both the similarity measure used by the method and its implementation.
- The **quality** of a clustering method is also measured by its ability to discover some or all of the **hidden** patterns

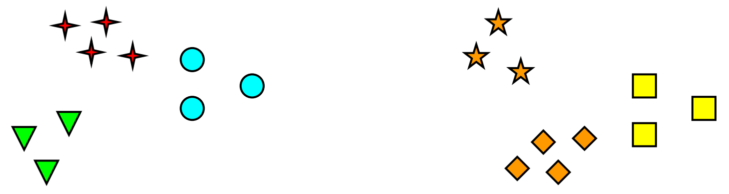
Requirements of Clustering in Data Mining

- Scalability
- Interpretability and usability
- Able to deal with ***noise*** and ***outliers***
- Ability to deal with different ***types of attributes***
- Discovery of clusters with ***arbitrary shape***
- Insensitive to order of input records
- Incorporation of user-specified constraints
- Minimal requirements for domain knowledge to determine input parameters

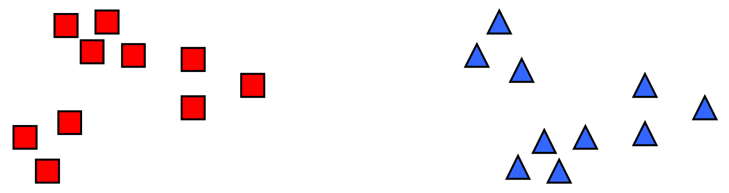
Notion of a Cluster can be Ambiguous



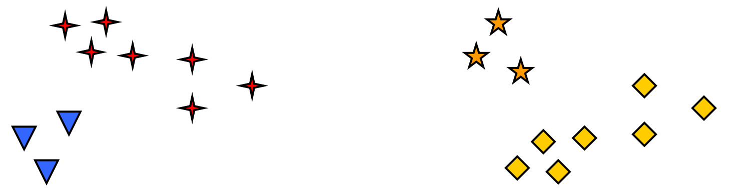
How many clusters?



Six Clusters



Two Clusters

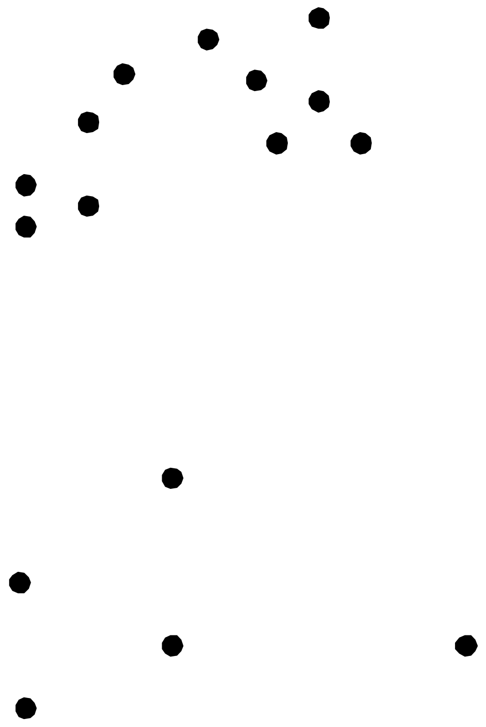


Four Clusters

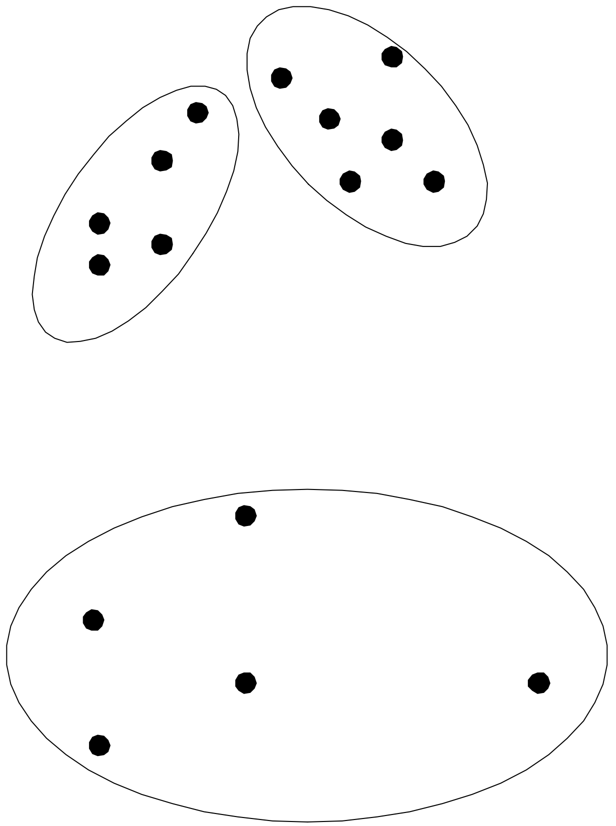
Types of Clusterings 8.1.2

- A **clustering** is a set of clusters
- **Partitional clustering**
  - A division data objects into ***non-overlapping subsets*** (clusters) such that each data object is in exactly one subset
- **Hierarchical clustering**
  - A set of nested clusters organized as a ***hierarchical tree***

Partitional Clustering

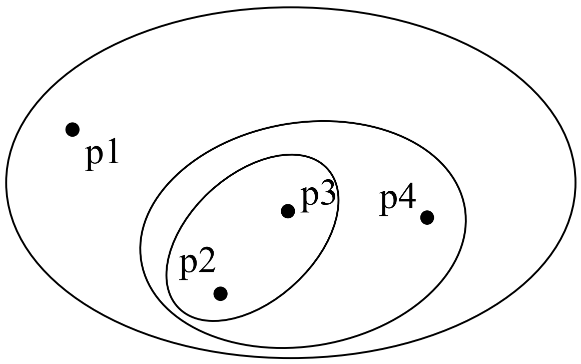


Original Points

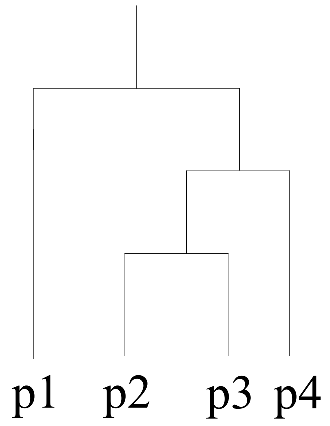


A Partitional Clustering

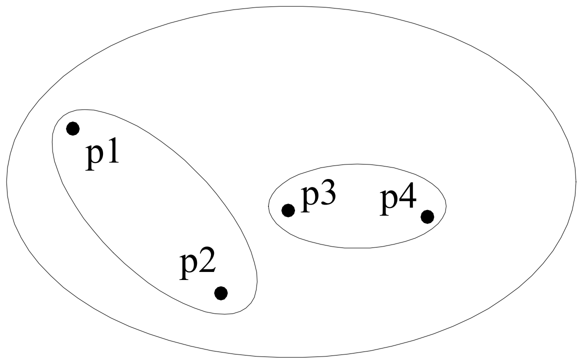
Hierarchical Clustering



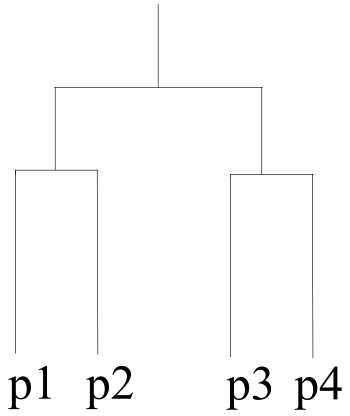
Traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Hierarchical Clustering



Non-traditional Dendrogram

Other distinctions between sets of clusters

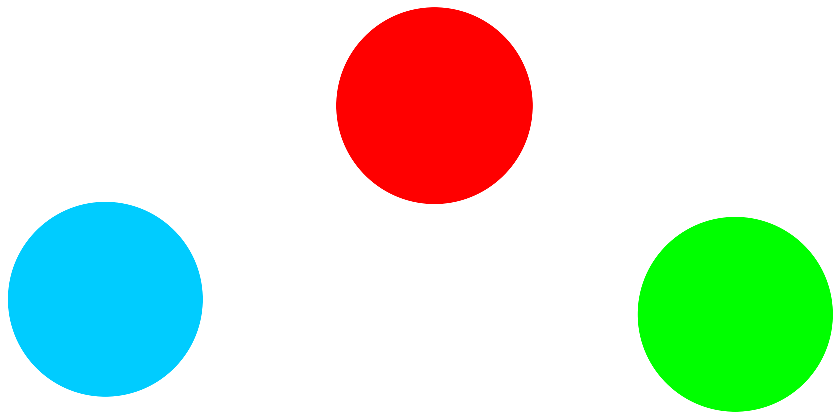
- **Exclusive (互斥) versus non-exclusive**
  - In non-exclusive clusterings, points may belong to multiple clusters.
  - Can represent multiple classes or ‘border’ points
- **Fuzzy (模糊) versus non-fuzzy**
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1,weights must sum to 1
  - Probabilistic clustering has similar characteristics
- **Partial versus complete**
  - In some cases, we only want to cluster some of the data
- **Heterogeneous (异质) versus homogeneous**
  - Cluster of widely different sizes, shapes, and densities

Types of Clusters 8.1.3

- Well-separated clusters
- Center-based clusters
- Contiguous clusters
- Density-based clusters
- Property or Conceptual
- Described by an Objective Function

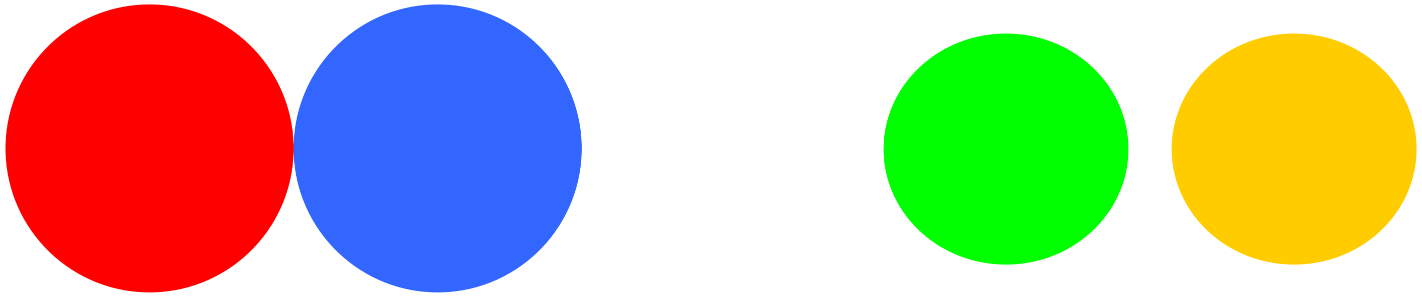
Types of Clusters: Well-Separated

- **Well-Separated Clusters**
  - A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster



Types of Clusters: Center-Based

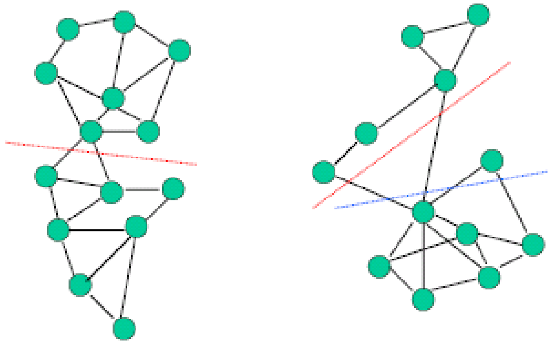
- **Center-based**
  - A cluster is a set of objects such that an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster
  - The center of a cluster is often a **centroid (质心)**, the average of all the points in the cluster, or a **medoid(中心点)**, the most "representative" point of a cluster



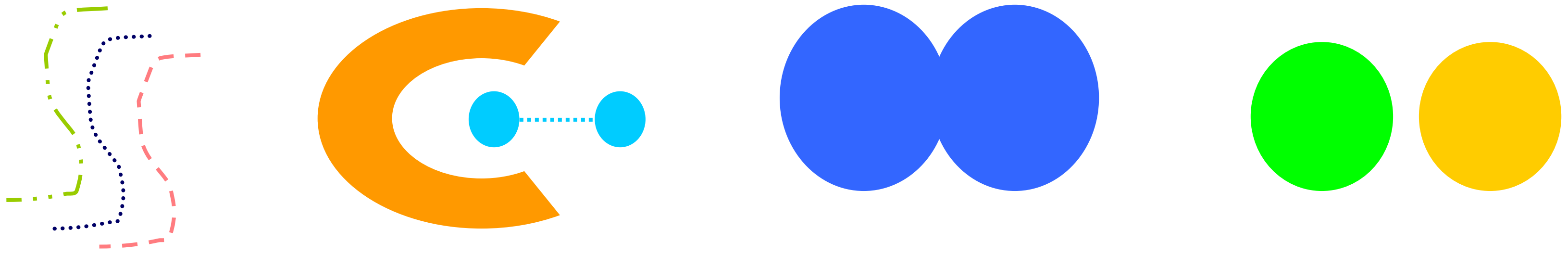
4 center-based clusters

Types of Clusters: Graph-Based

- **Graph based cluster**
  - Connected component



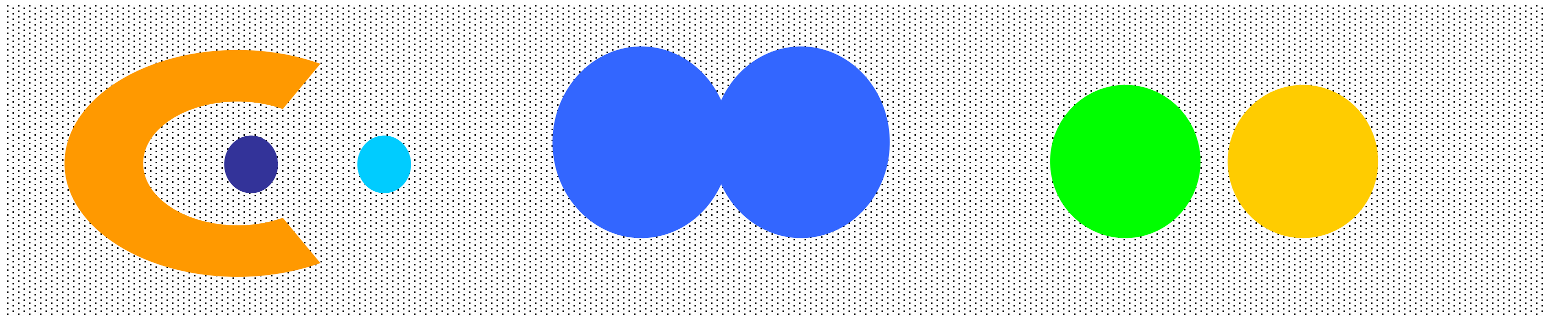
- **Contiguous Cluster (Nearest neighbor or Transitive)**
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster



8 contiguous clusters

Types of Clusters: Density-Based

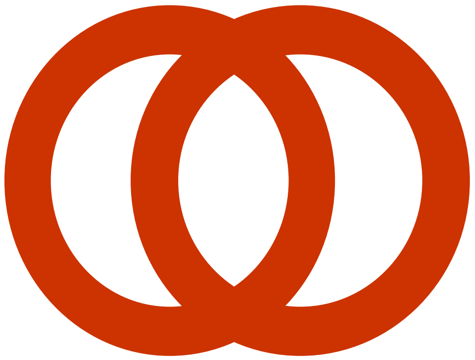
- **Density-based**
  - A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
  - Used when the clusters are irregular or intertwined, and when noise and outliers are present.



6 density-based clusters

Types of Clusters: Conceptual Clusters

- **Shared Property or Conceptual Clusters**
  - Finds clusters that share some common property or represent a particular concept



2 Overlapping Circles

Types of Clusters: Objective Function

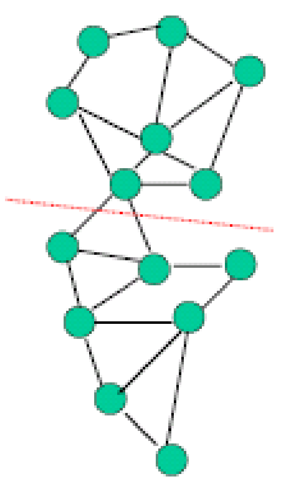
- **Clusters Defined by an Objective Function**
  - Finds clusters that **minimize\maximize** an objective function.
  - ***Enumerate all possible ways*** of dividing the points into clusters and evaluate the `goodness' of each potential set of clusters by objective function (NP-Hard)
  - Global or local objectives
    - ✓ **Hierarchical algorithms** typically have local objectives
    - ✓ **Partitional algorithms** typically have global objectives
  - A variation of the global objective function approach is to fit the data to a ***parameterized model***
    - ✓ Parameters for the model are determined from the data.
    - ✓ Mixture models assume that the data is a ‘mixture' of a number of statistical distributions

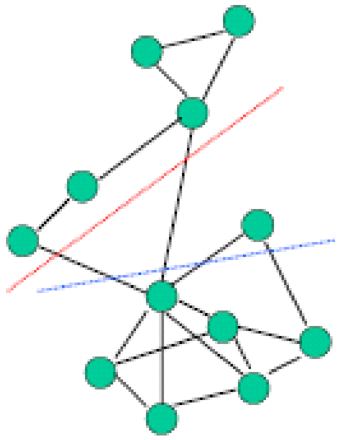
Types of Clusters: Objective Function

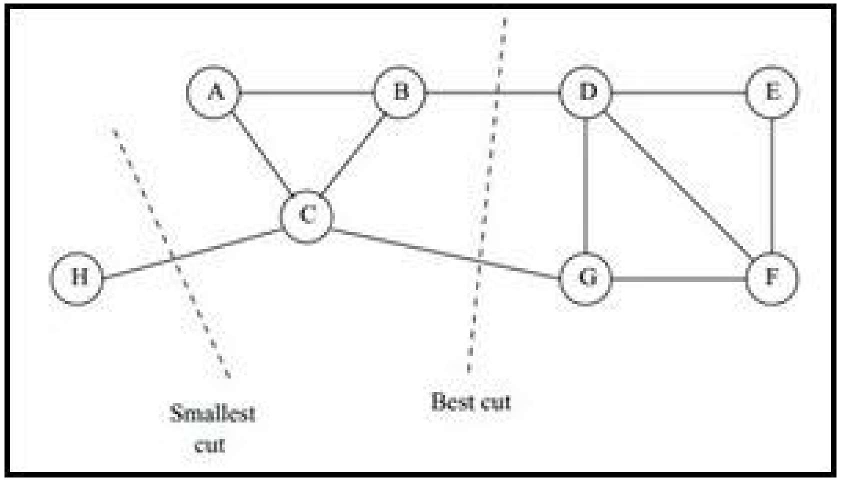
- Map the clustering problem to a different domain and solve a related problem in that domain
  - **Proximity matrix** defines a weighted graph, where the nodes are the points being clustered, and the weighted edges represent the proximities between points
  - Clustering is equivalent to breaking the **graph into connected components**, one for each cluster.
  - Want to **minimize the edge weight** between clusters and maximize the edge weight within clusters

Example: Min-cut problem

- Min-Cut: min cutsize, NP-hard problem
- Cutsize = No. of cut edges







Characteristics of the Input Data

- Type of proximity or density measure
- Sparseness
- Attribute type
- Type of Data
- Type of Distribution
- Dimensionality
- Noise and Outliers

Major Clustering Approaches

- **Partitioning algorithms**: Construct various partitions and then evaluate them by some criterion
- **Hierarchy algorithms:** Create a hierarchical decomposition of the set of data (or objects) using some criterion
- **Density-based:** based on connectivity and density functions
- **Model-based:** A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other
- **Grid-based:** based on a multiple-level granularity structure

Clustering Algorithms

- K-means and its variants
- Hierarchical clustering
- Density-based clustering

K-means 8.2

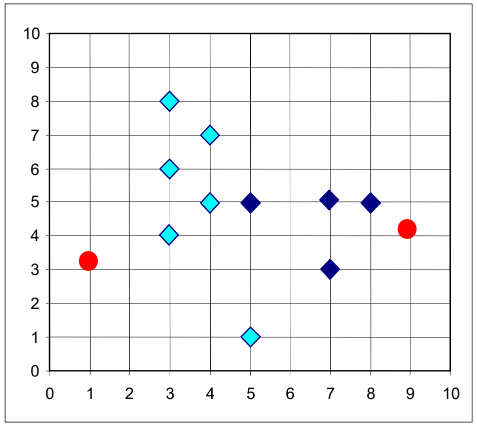
Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database ***D*** of ***n*** objects into a set of ***k*** clusters
- Given a *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - ***k-means*** (MacQueen'67): Each cluster is represented by the center of the cluster
  - *k-medoids* or PAM (**P**artition **A**round **M**edoids) (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects in the cluster

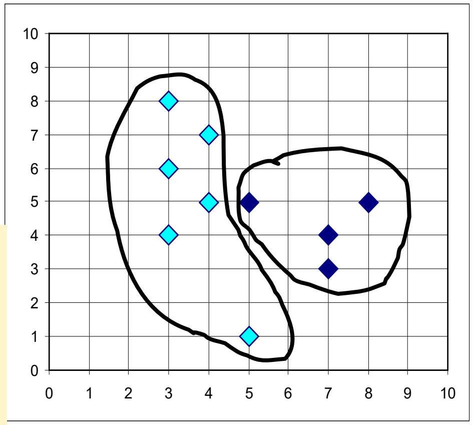
The *k-Means* Clustering Method

- Given *k*, the *k-means* algorithm is implemented in four steps
  - Partition objects into *k* nonempty subsets
  - Compute seed points as the ***centroids*** of the clusters of the current partition (the centroid is the center, i.e., *mean point*, of the cluster)
  - Assign each object to the cluster with the nearest seed point
  - Go back to Step 2, stop when no more new assignment

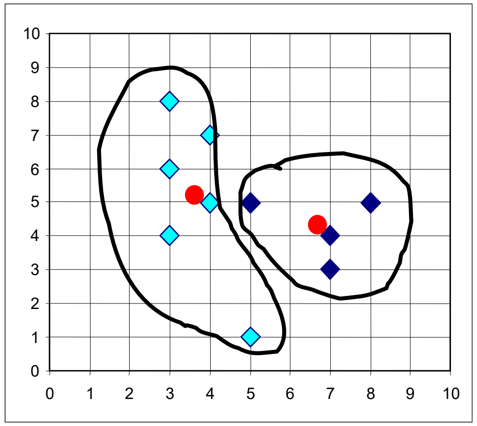
The *k-Means* Clustering Method:Example



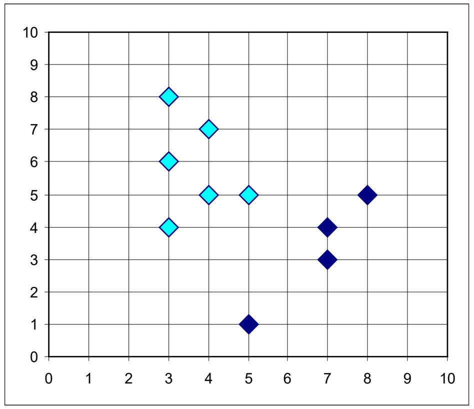
Assign each objects to most similar center



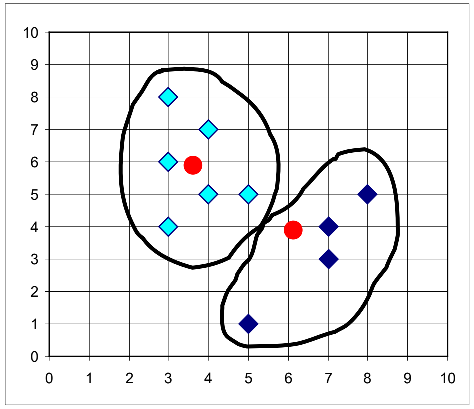
Update the cluster means



reassign



Update the cluster means



reassign

K=2

Arbitrarily choose K object as initial cluster center

k-means Clustering – Details

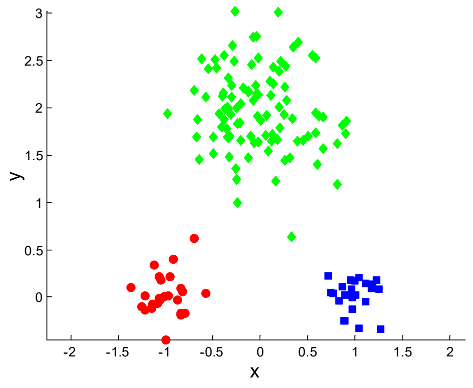
- Initial centroids are often **chosen randomly**
  - Clusters produced vary from one run to another
- The centroid is the **mean** of the points in the cluster
- '**Closeness**' is measured by **Euclidean distance**
- *k*-means will converge for common similarity measures
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to '**Until relatively few points change clusters'**
- Complexity is $O(n \times k \times I \times d)$

  n = number of points, k = number of clusters,
  I = number of iterations, d = number of attributes

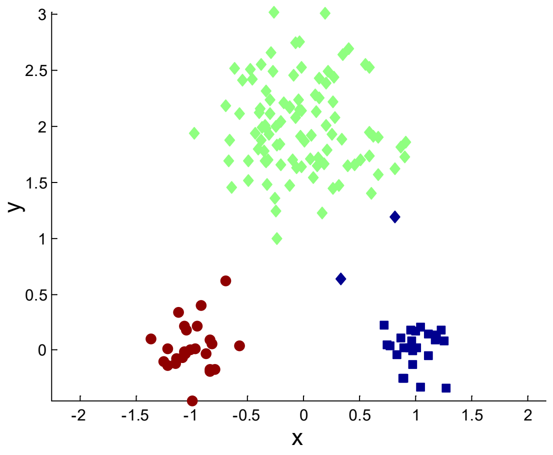
Variations of the *k-Means* Method

- A few variants of the *k*-means which differ in
  - Selection of the initial *k* means
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- Handling **categorical data**: ***k-modes*** (Huang'98)
  - Replacing means of clusters with modes
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters
  - A mixture of categorical and numerical data: *k-prototype* (原型）method

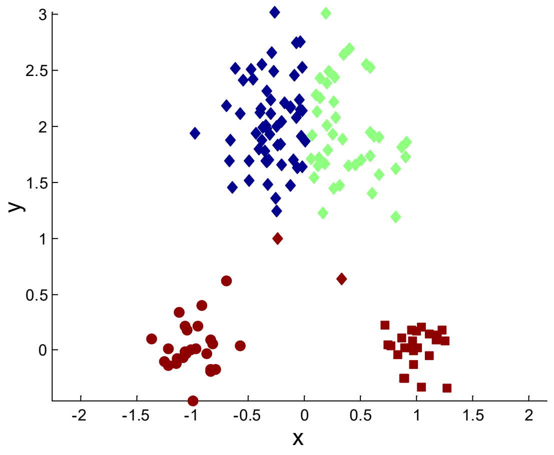
Two different *k*-means Clusterings



Original Points

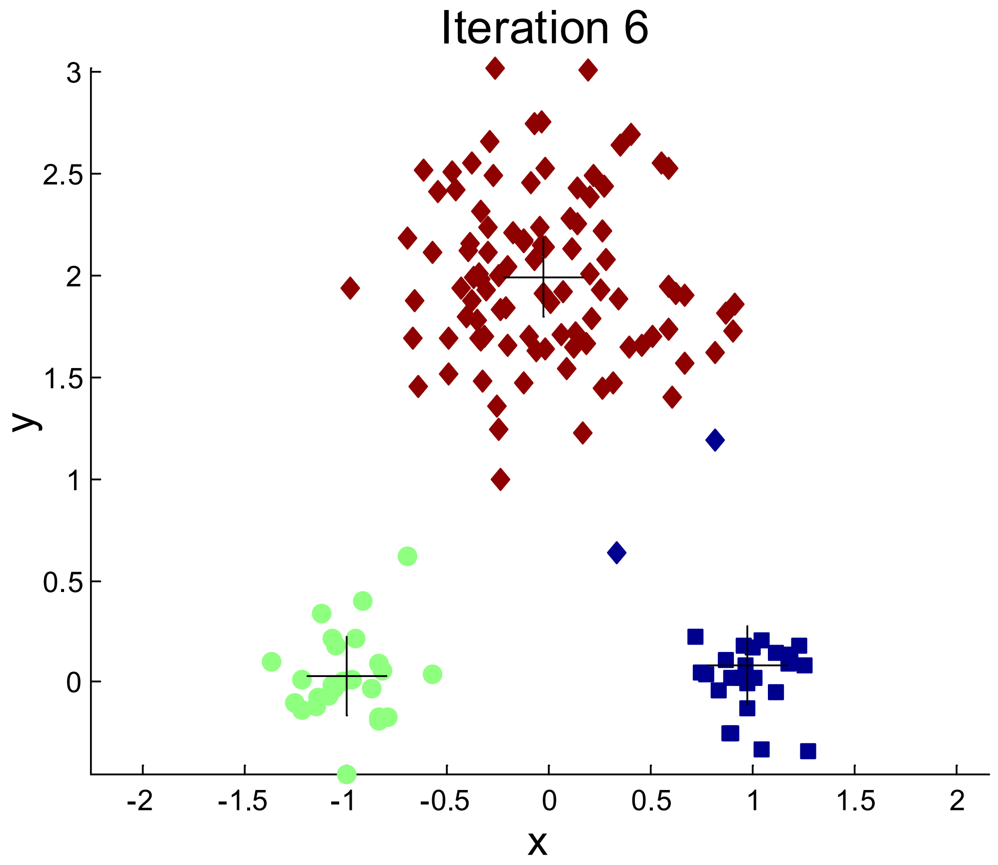


Optimal Clustering

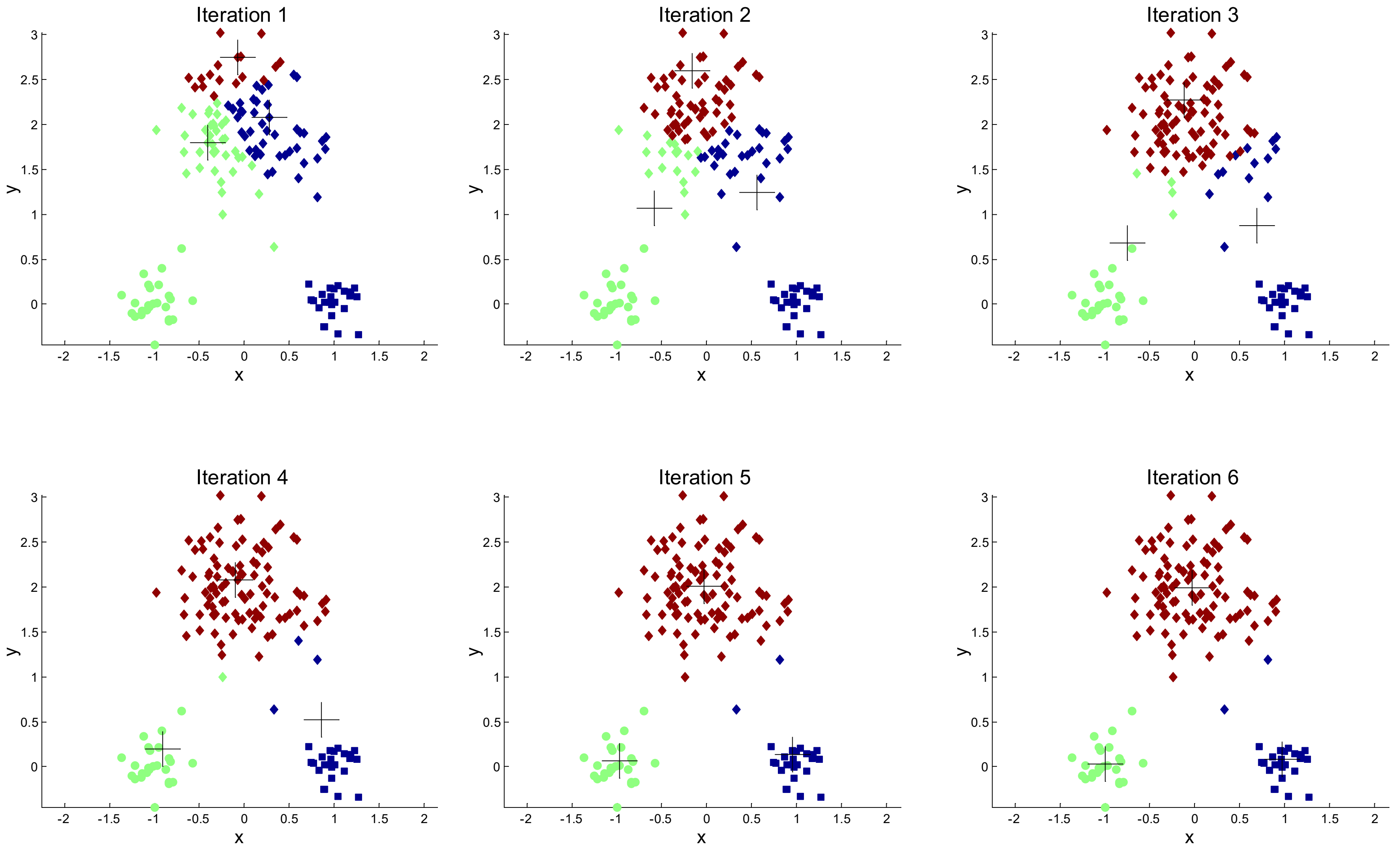


Sub-optimal Clustering

Importance of Choosing Initial Centroids



Importance of Choosing Initial Centroids



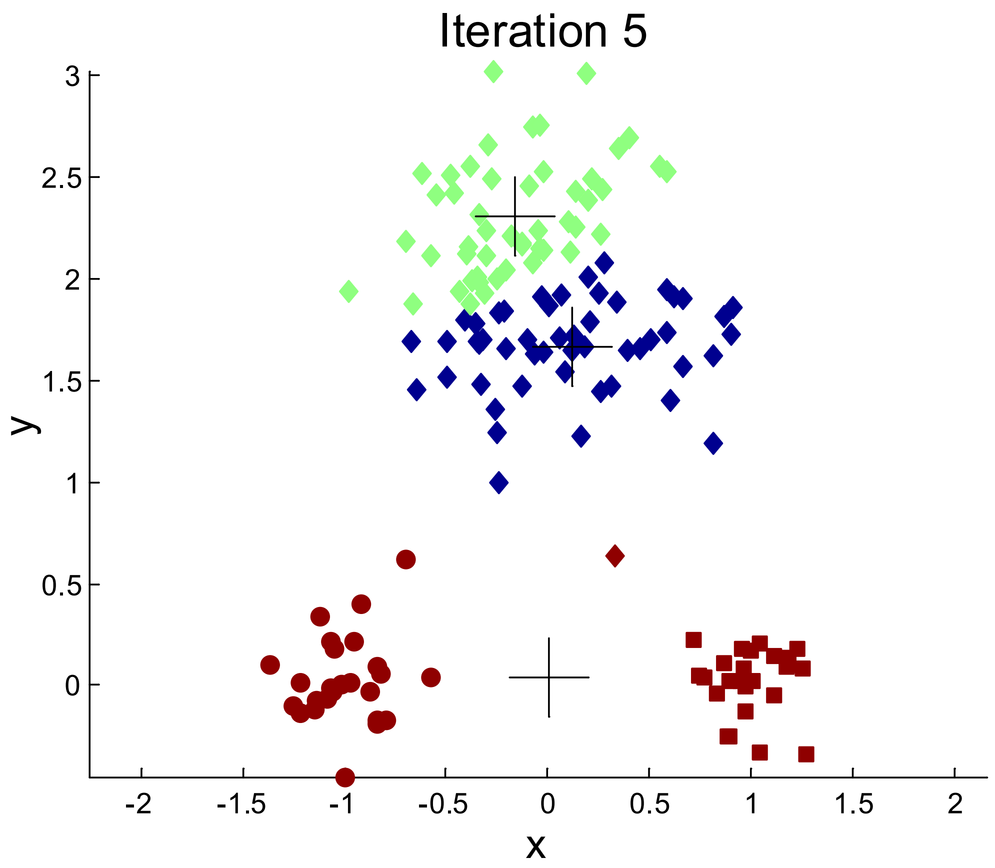
Evaluating *k*-means Clusters

- Most common measure is **Sum of Squared Error (SSE)**
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, square these errors and sum them

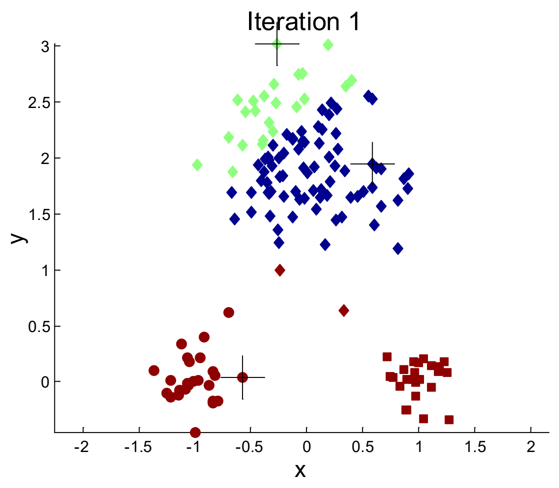
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(c_i, x) \qquad c_i = \frac{1}{m_i} \sum_{x \in C_i} x$$

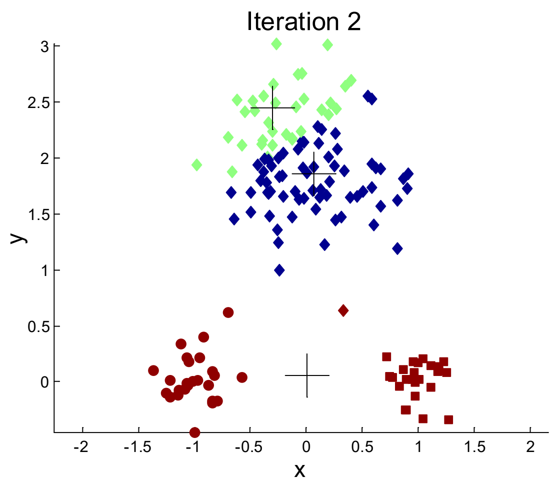
  - x is a data point in cluster C_i and c_i is the representative point for cluster C_i ， which m_i is the number of the cluster
  - One easy way to reduce SSE is to increase *k*
    - ✓ A good clustering with smaller *k* can have a lower SSE than a poor clustering with higher *k*

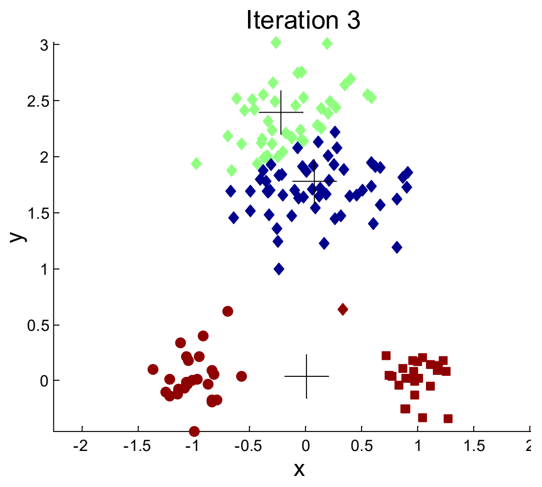
Importance of Choosing Initial Centroids

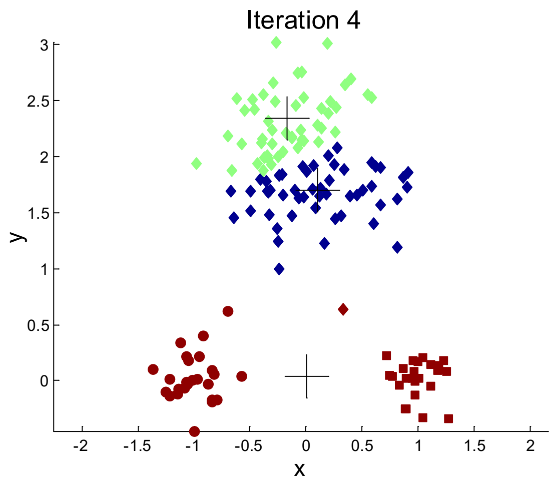


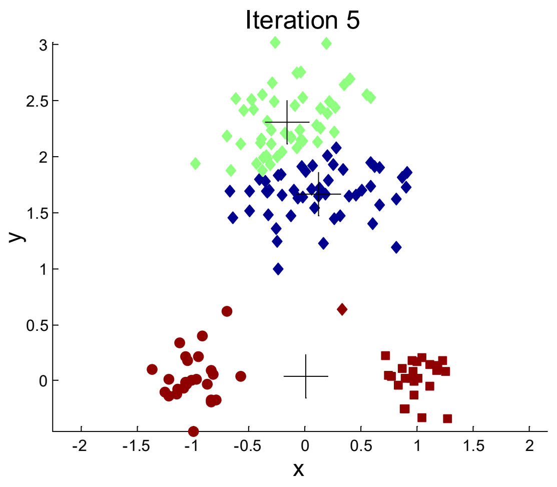
Importance of Choosing Initial Centroids











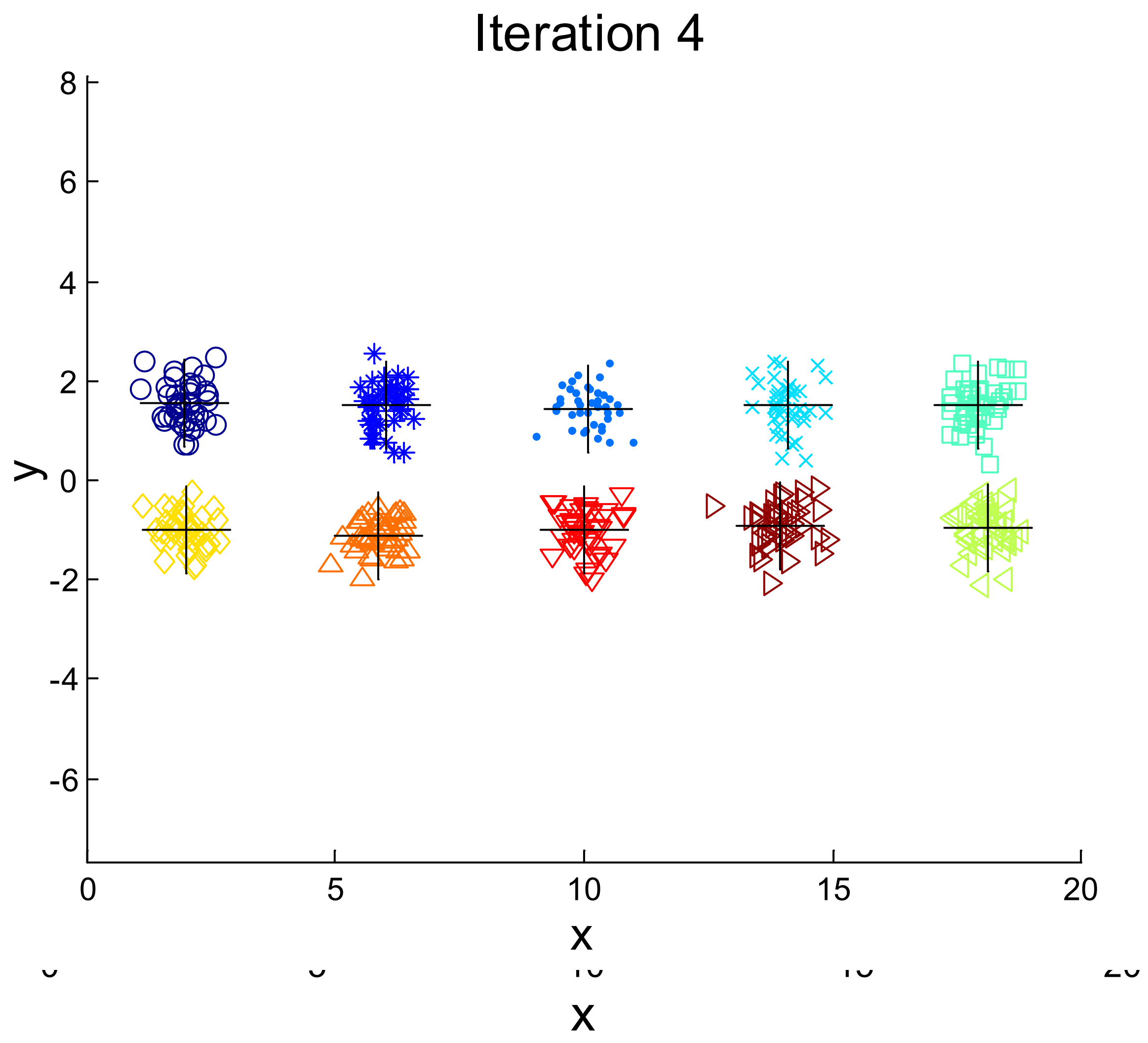
Problems with Selecting Initial Points

- If there are *k* 'real' clusters then the chance of selecting one centroid from each cluster is small
  - Chance is relatively small when *k* is large
  - If clusters are the same size, *n*, then

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

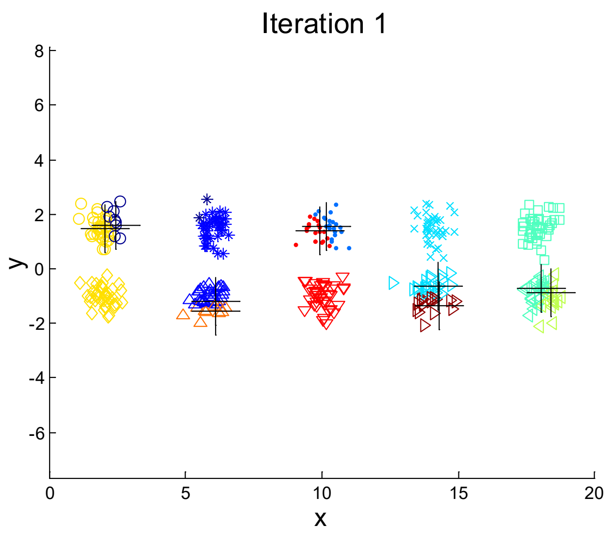
  - For example, if *k* = 10, then probability = $10!/10^{10} = 0.00036$
  - Sometimes the initial centroids will readjust (微调)themselves in 'right' way, and sometimes they don't

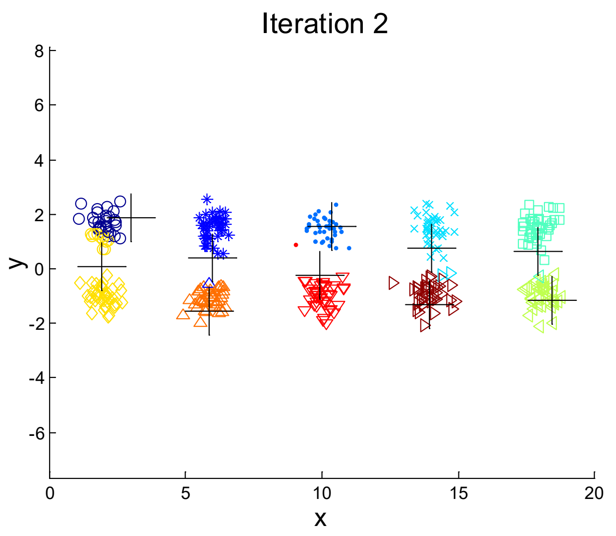
10 Clusters Example

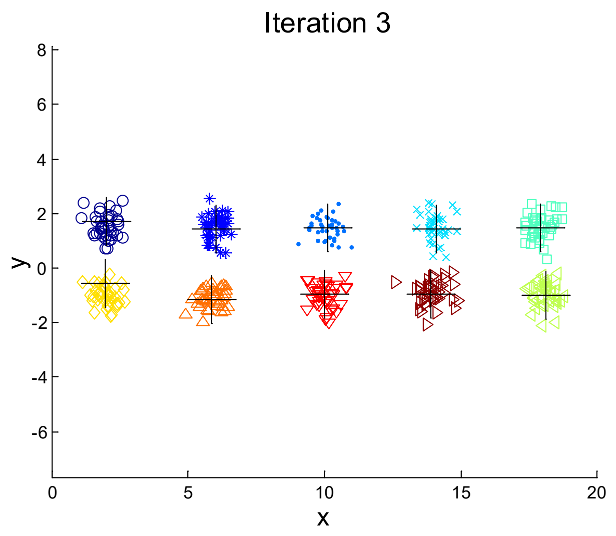


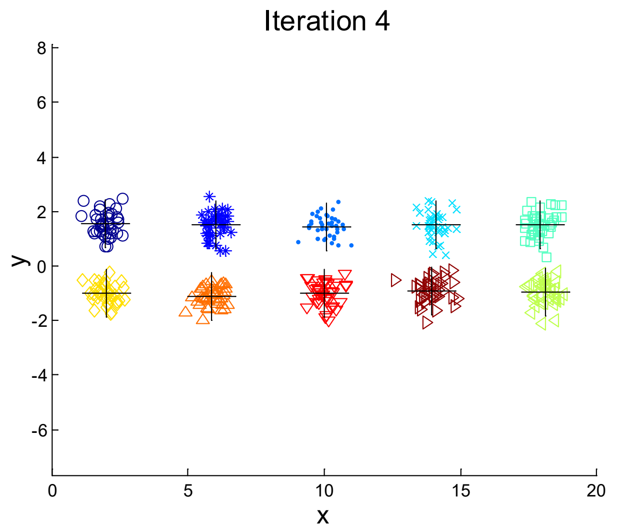
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example









Starting with two initial centroids in one cluster of each pair of clusters

Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side
- Sample and use hierarchical clustering to determine initial centroids
- Select more than *k* initial centroids and then select among these initial centroids
  - Select most widely separated
- Postprocessing
- Bisecting *k*-means
  - Not as susceptible(易受影响) to initialization issues

Handling Empty Clusters

- Basic *k*-means algorithm can yield empty clusters
- Several strategies
  - Choose the point that contributes most to SSE
  - Choose a point from the cluster with the highest SSE
  - If there are several empty clusters, the above can be repeated several times

Updating Centers Incrementally

- In the basic *k*-means algorithm, centroids are updated after all points are assigned to a centroid
- An alternative is to update the centroids after each assignment (incremental approach)
  - Each assignment updates zero or two centroids
  - Advantage: never get an empty cluster
  - Disadvantage: More expensive, introduces an order dependency

Pre-processing and Post-processing

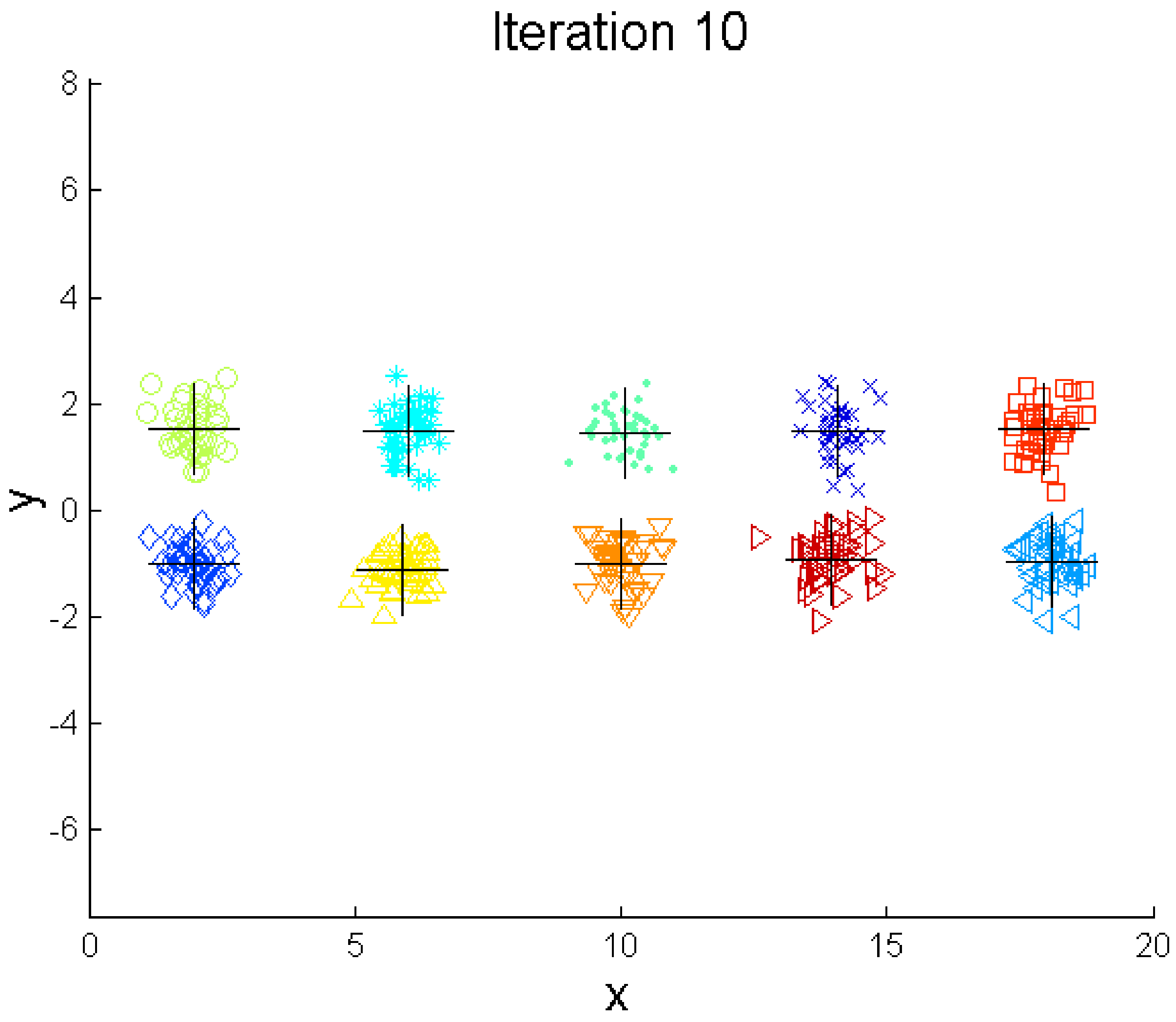
- **Pre-processing**
  - Normalize the data
  - Eliminate outliers
- **Post-processing**
  - Eliminate small clusters that may represent outliers
  - Split ‘loose’ clusters, i.e., clusters with relatively high SSE
  - Merge clusters that are ‘close’ and that have relatively low SSE
  - Can use these steps during the clustering process

Bisecting *k*-means

- Bisecting *k*-means algorithm
  - Variant of *k*-means that can produce a partitional or a hierarchical clustering

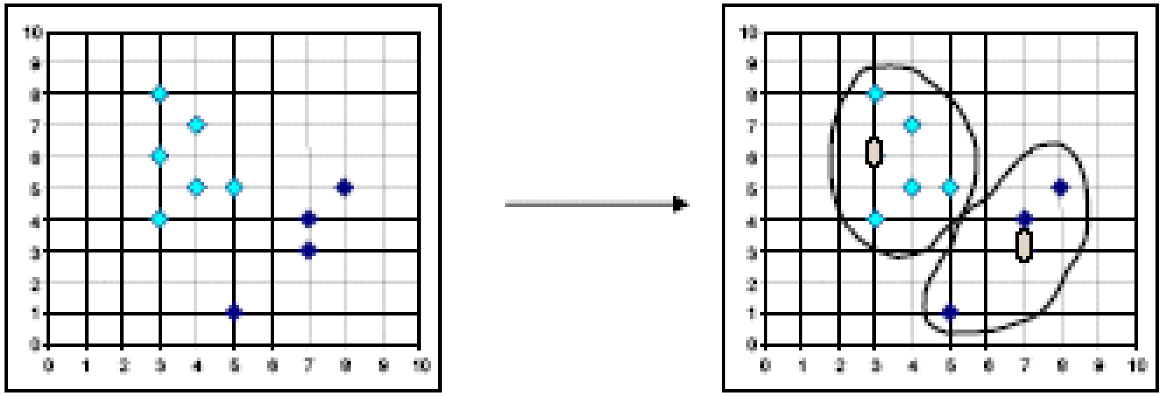
```
1: Initialize the list of clusters to contain the cluster containing all points.
2: repeat
3:   Select a cluster from the list of clusters
4:   for $i = 1$ to $number\_of\_iterations$ do
5:     Bisect the selected cluster using basic K-means
6:   end for
7:   Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: until Until the list of clusters contains $K$ clusters
```

Bisecting *k*-means Example

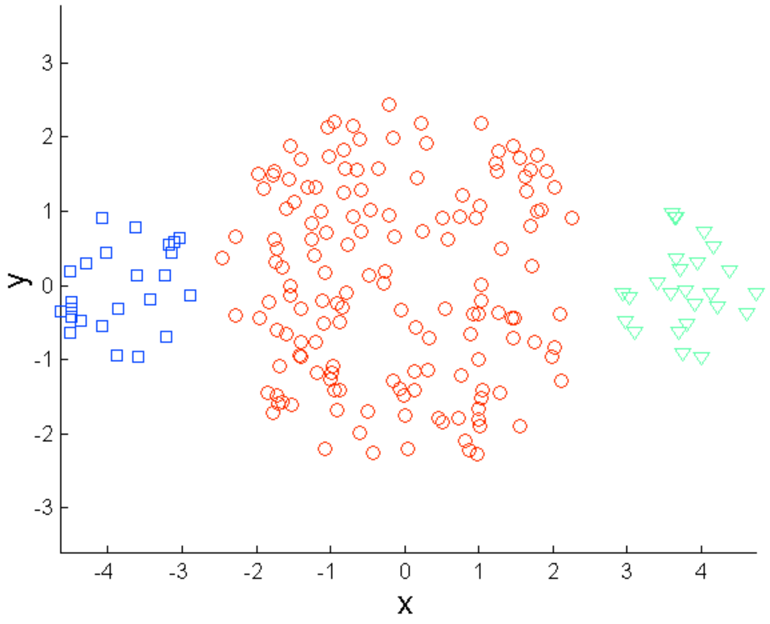


Limitations of *k*-means

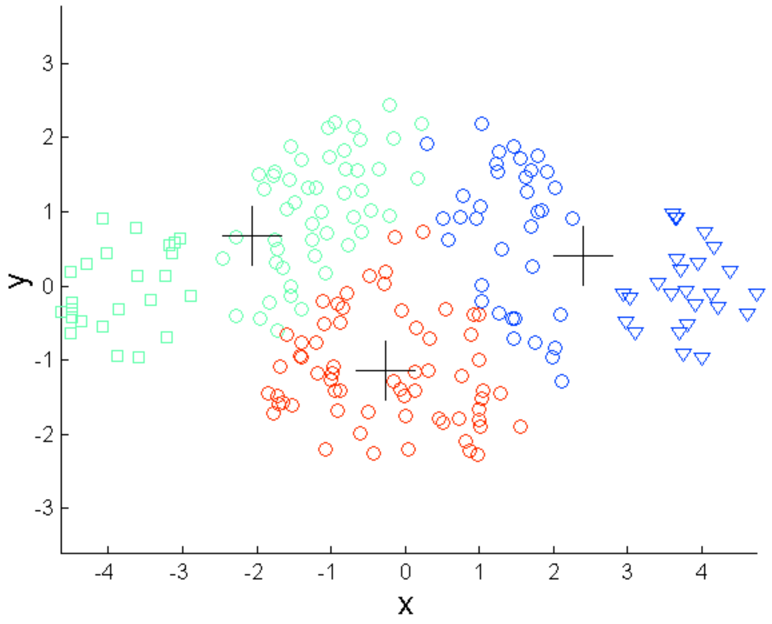
- K-means has problems when clusters are of differing
  - Sizes， Densities， Non-globular shapes
- The k-means algorithm is sensitive to outliers !
- k-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.



Limitations of K-means: Differing Sizes

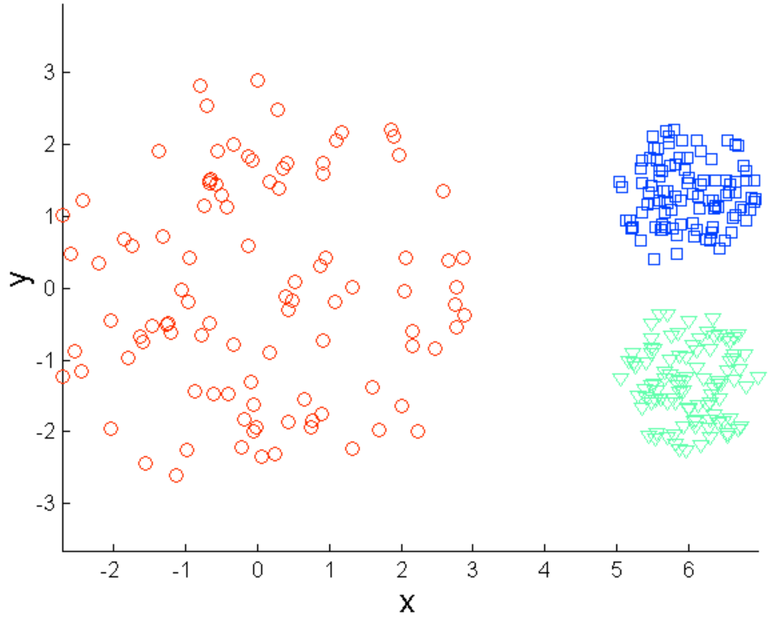


Original Points

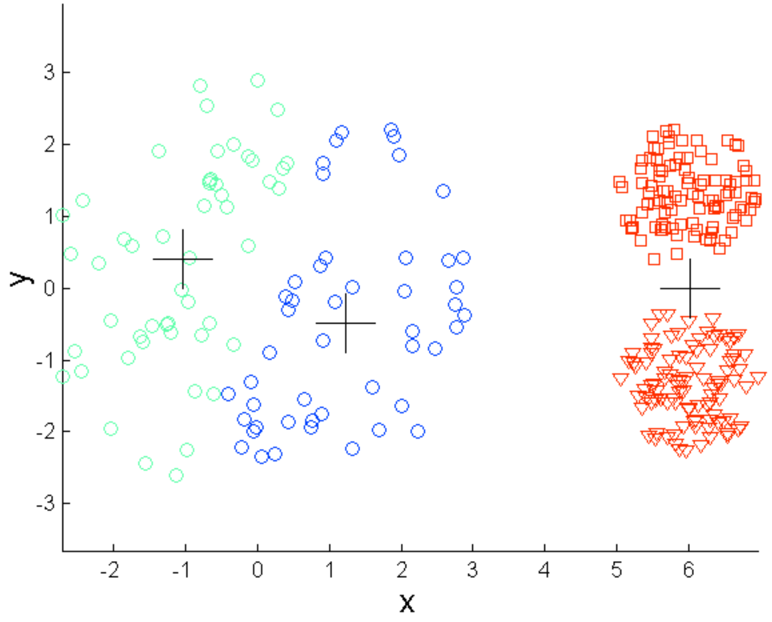


K-means (3 Clusters)

Limitations of K-means: Differing Density

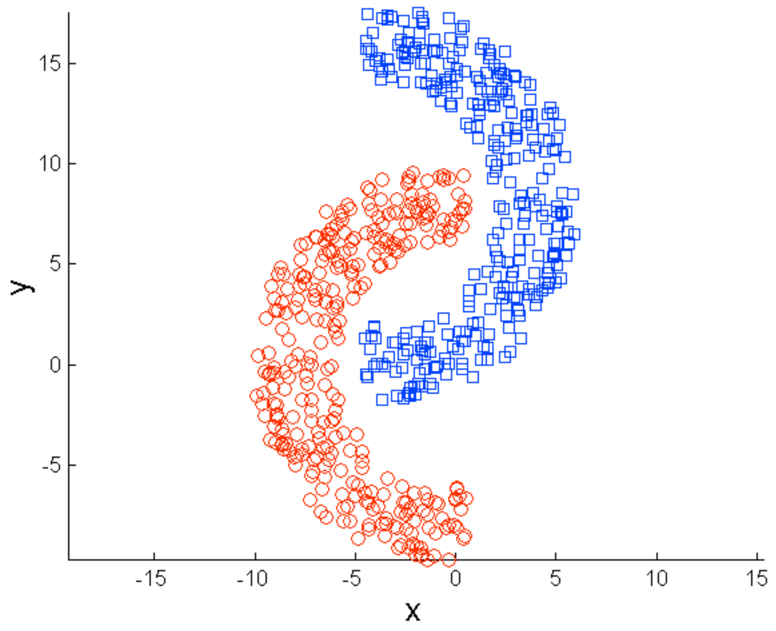


Original Points

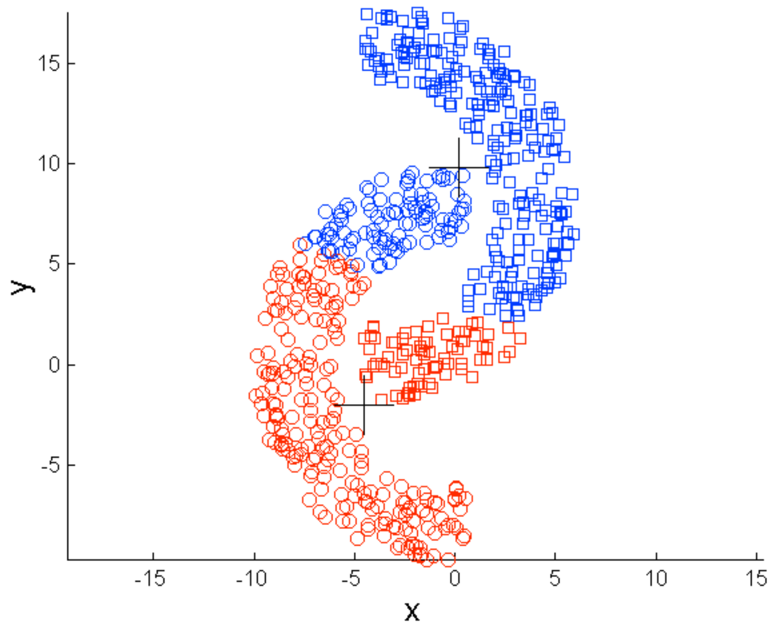


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

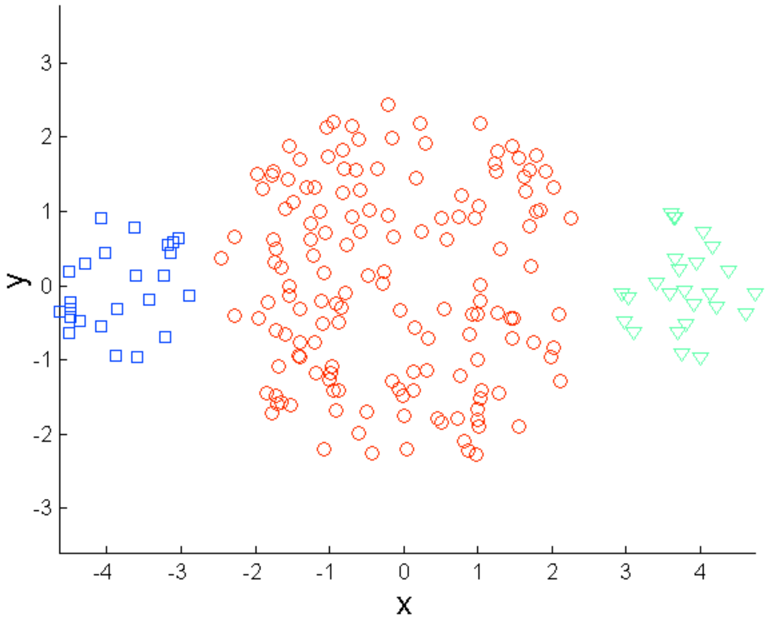


Original Points

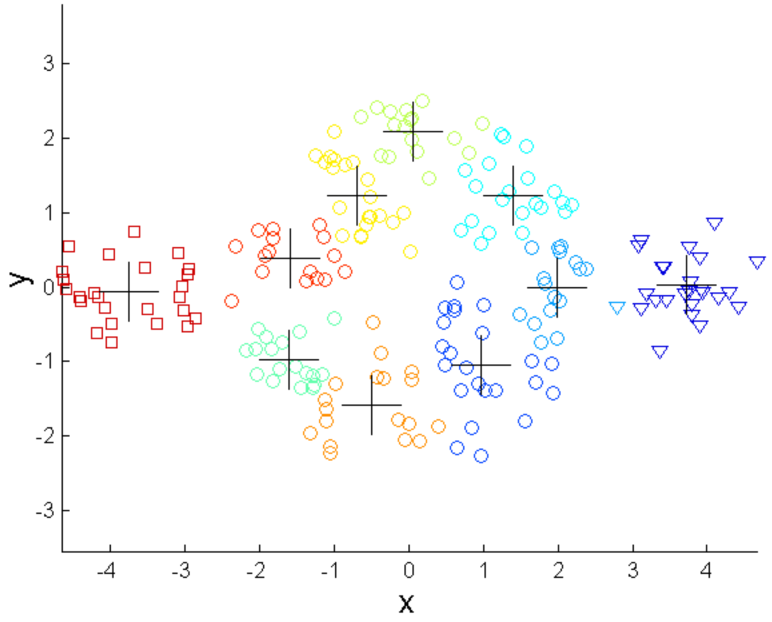


K-means (2 Clusters)

Overcoming K-means Limitations



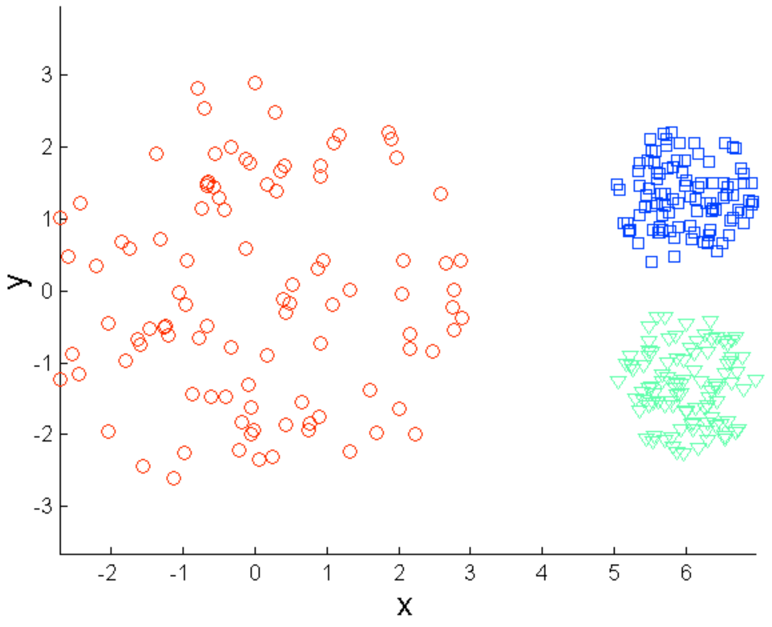
Original Points



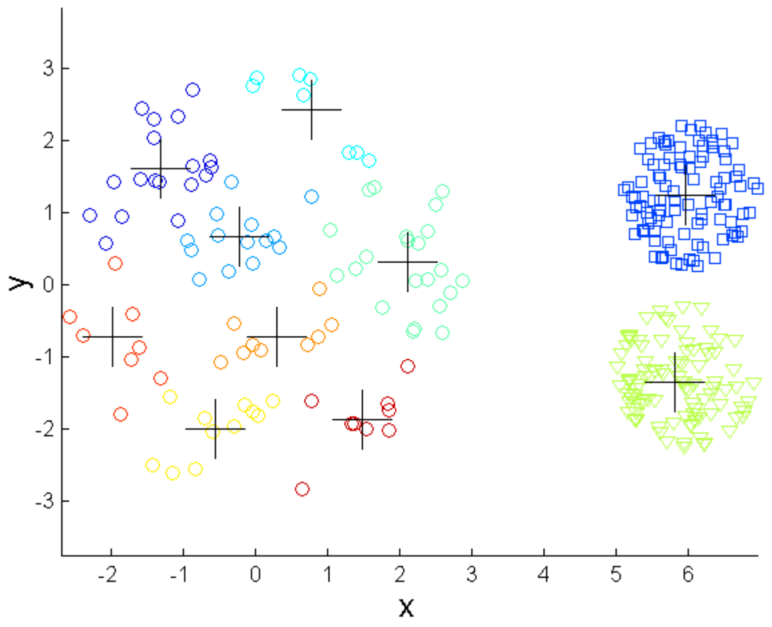
K-means Clusters

One solution is to use many clusters.
Find parts of clusters, but need to put together.

Overcoming K-means Limitations

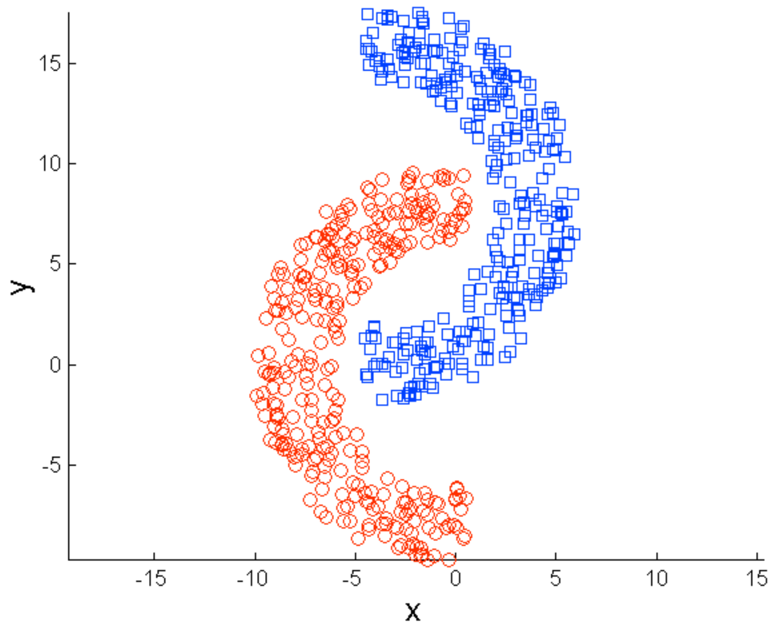


Original Points

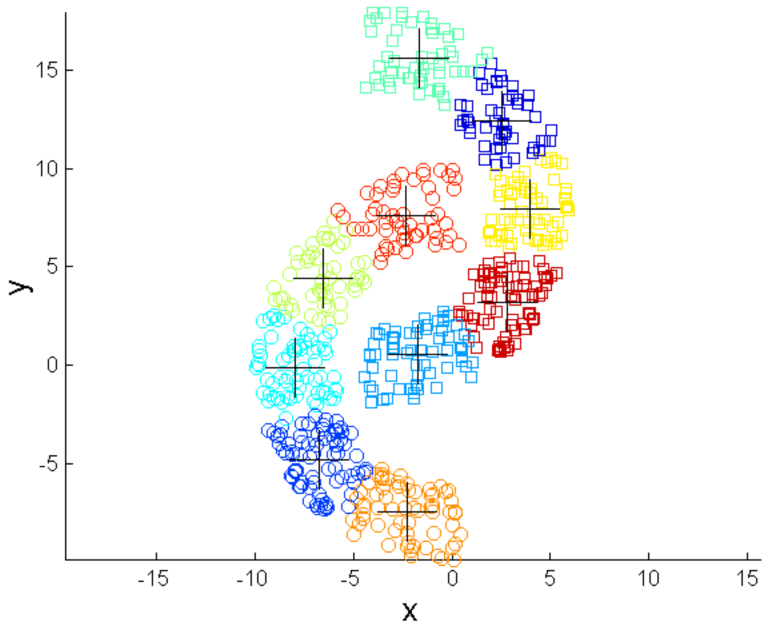


K-means Clusters

Overcoming K-means Limitations



Original Points

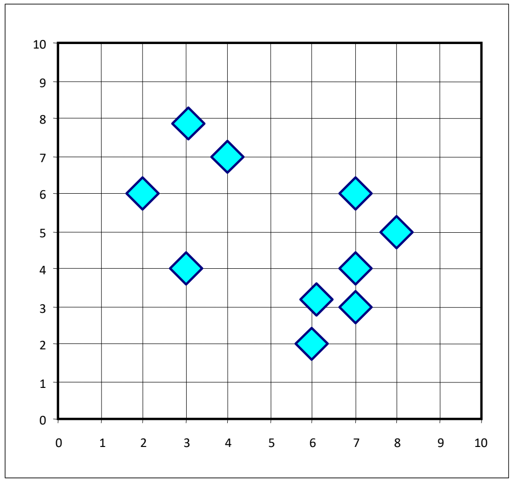


K-means Clusters

The *K-Medoids* Clustering Method

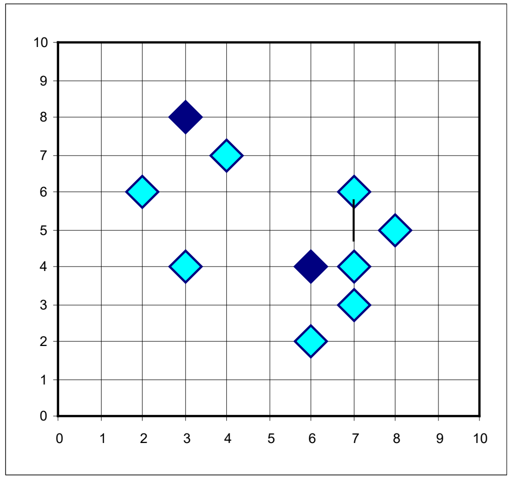
- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - *PAM* works effectively for small data sets, but does not scale well for large data sets
- *CLARA (Clustering LARge Applications)* (1990)
- *CLARANS (Clustering LARge Applications based upon RANdomized Search)* (1994): Randomized sampling

A Typical K-Medoids Algorithm (PAM)



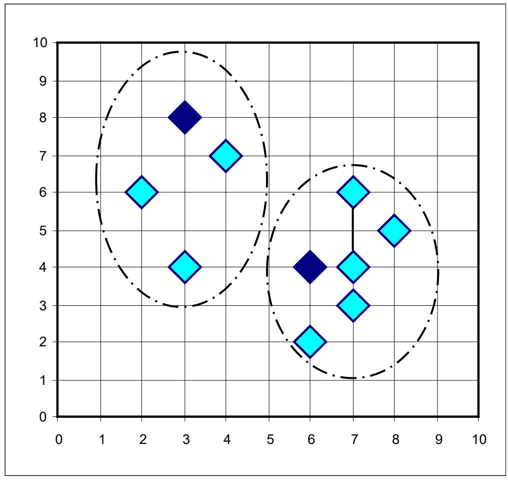
K=2

Arbitrary choose k object as initial medoids

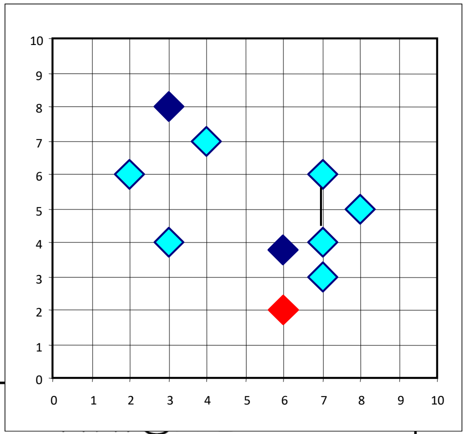


Assign each remaining object to nearest medoids

Total Cost = 20

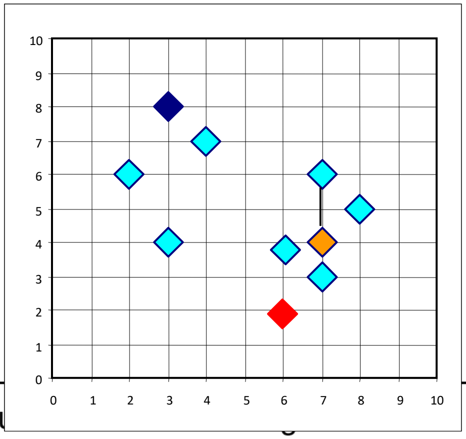


Randomly select a nonmedoid object,O_{ramdom}



Compute total cost of swapping

Total Cost = 26



Swapping O and O_{ramdom}

If quality is improved.

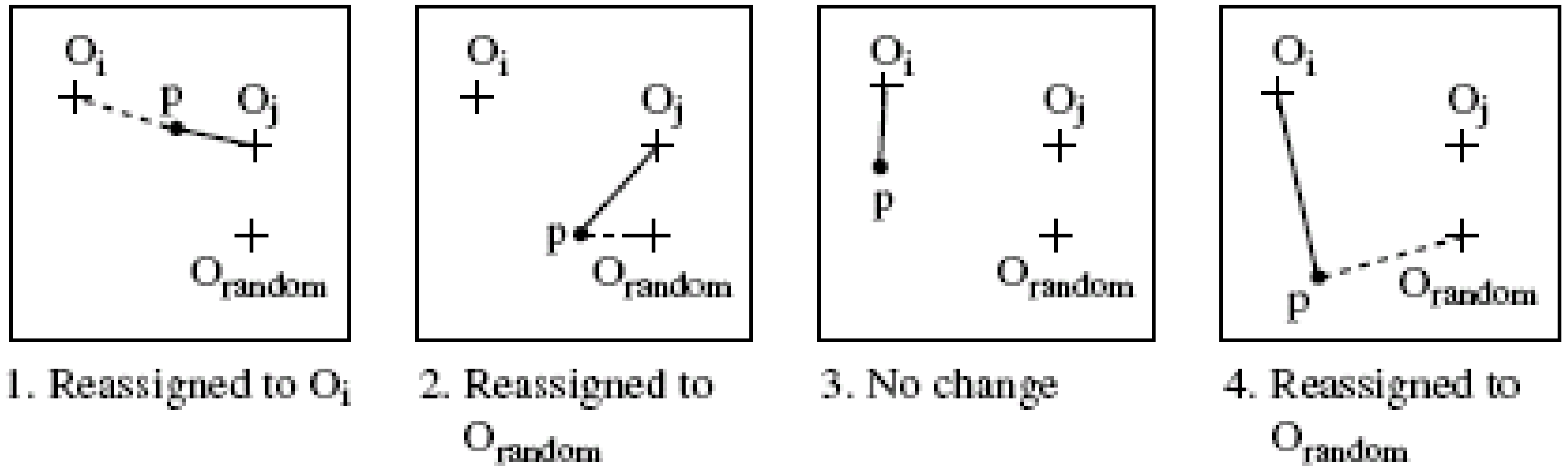
Do loop

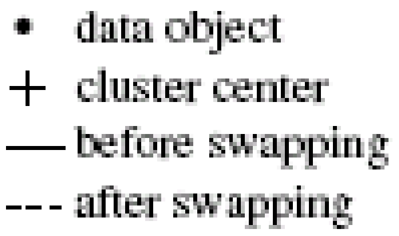
Until no change

Cost function

- Case 1: $p \in O_j$,If O_j is replaced by O_{random} and ***p*** is closest to one of the other representative objects, ***oi***, $i \neq j$, then $p \in O_i$
- Case 2: $p \in O_j$, If O_j is replaced by O_{random} and ***p*** is closest to O_{random} , then $p \in O_{random}$
- Case 3: $p \in O_i, i \neq j$, If O_j is replaced by O_{random} and ***p*** is still closest to O_i, then the assignment does not change
- Case 4: $p \in O_i, i \neq j$, If O_j is replaced by O_{random} and ***p*** is closest to O_{random}, then $p \in O_{random}$

Four cases of the cost function for *k*-medoids clustering





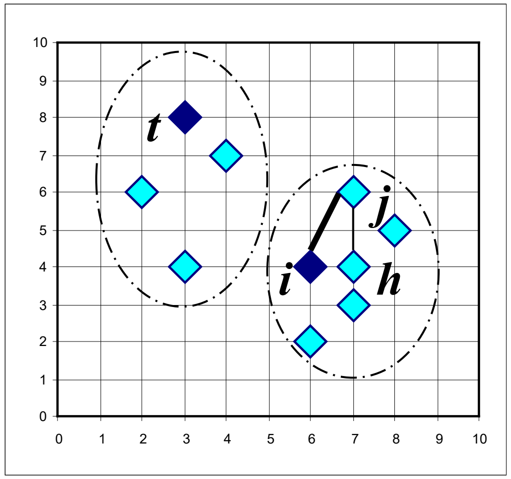
PAM (Partitioning Around Medoids) algorithm

- **Use real object to represent the cluster**
  - 1. Select ***k*** representative objects arbitrarily
  - 2. For each pair of non-selected object ***h*** and selected object ***i***, calculate the total swapping cost $\boldsymbol{TC_{ih}}$
  - 3. For each pair of ***i*** and ***h***,
    - ✓ If $TC_{ih} < 0$, ***i*** is replaced by ***h***
    - ✓ Then assign each non-selected object to the most similar representative object
  - repeat steps 2-3 until there is no change

PAM Clustering: Total swapping

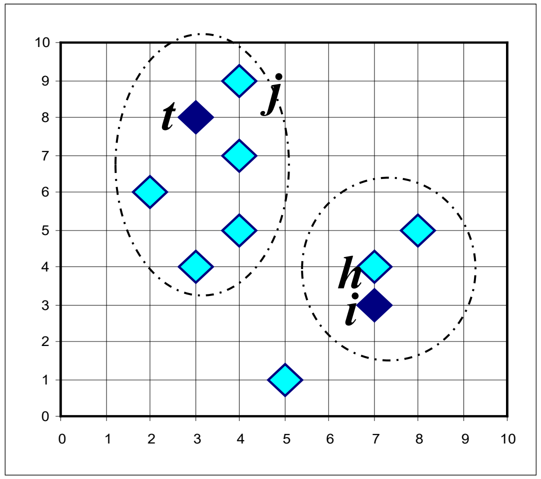
cost $TC_{ih}=\sum_j C_{jih}$

Case 1



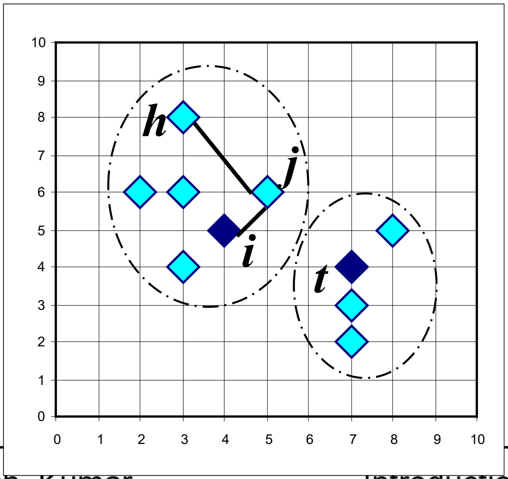
$$C_{jih} = d(j, h) - d(j, i)$$

Case 3



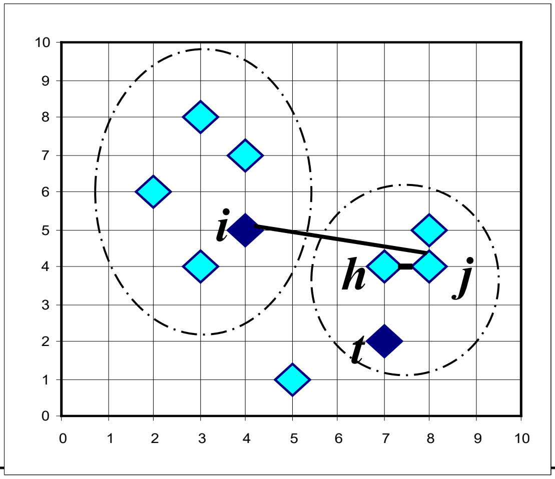
$$C_{jih} = 0$$

Case 2



$$C_{jih} = d(j, t) - d(j, i)$$

Case 4



$$C_{jih} = d(j, h) - d(j, t)$$

What is the problem with PAM?

- PAM is more robust than k-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than a mean
- PAM works efficiently for small data sets but does not **scale well** for large data sets.
  - $O(k(n-k)^2)$ for each iteration, where n is number of data, k is number of clusters

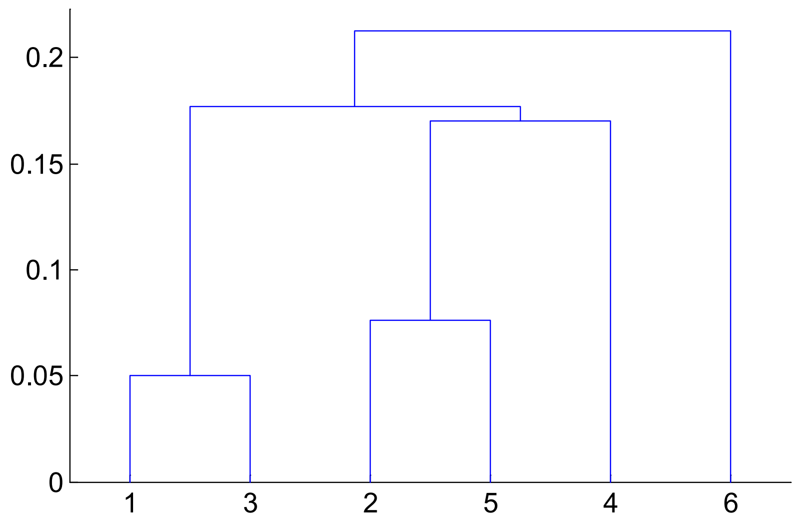
Assignment pp.347~352,

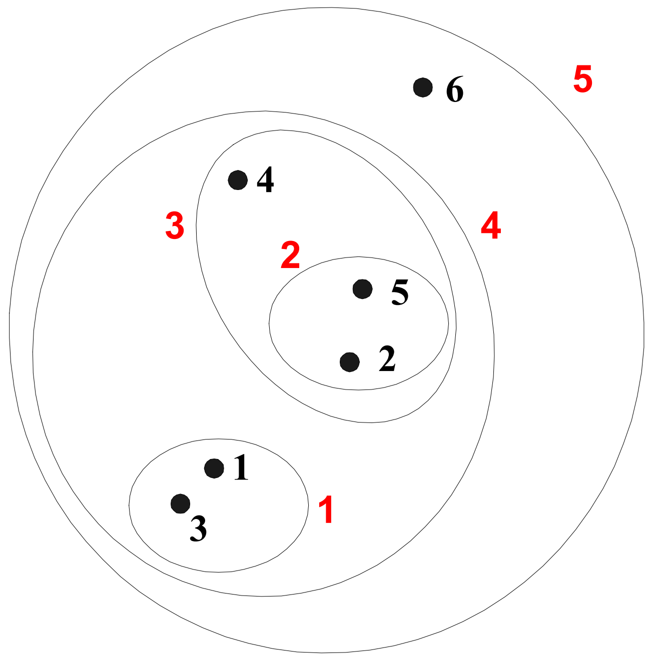
6,7,11

Hierarchical Clustering 8.3

Hierarchical Clustering

- Produces a set of ***nested clusters*** organized as a hierarchical tree
- Can be visualized as a dendrogram（生物系统树）
  - ➢ A tree like diagram that records the sequences of merges or splits





© Tan,Steinbach, Kumar Introduction to Data Mining

Strengths of Hierarchical Clustering

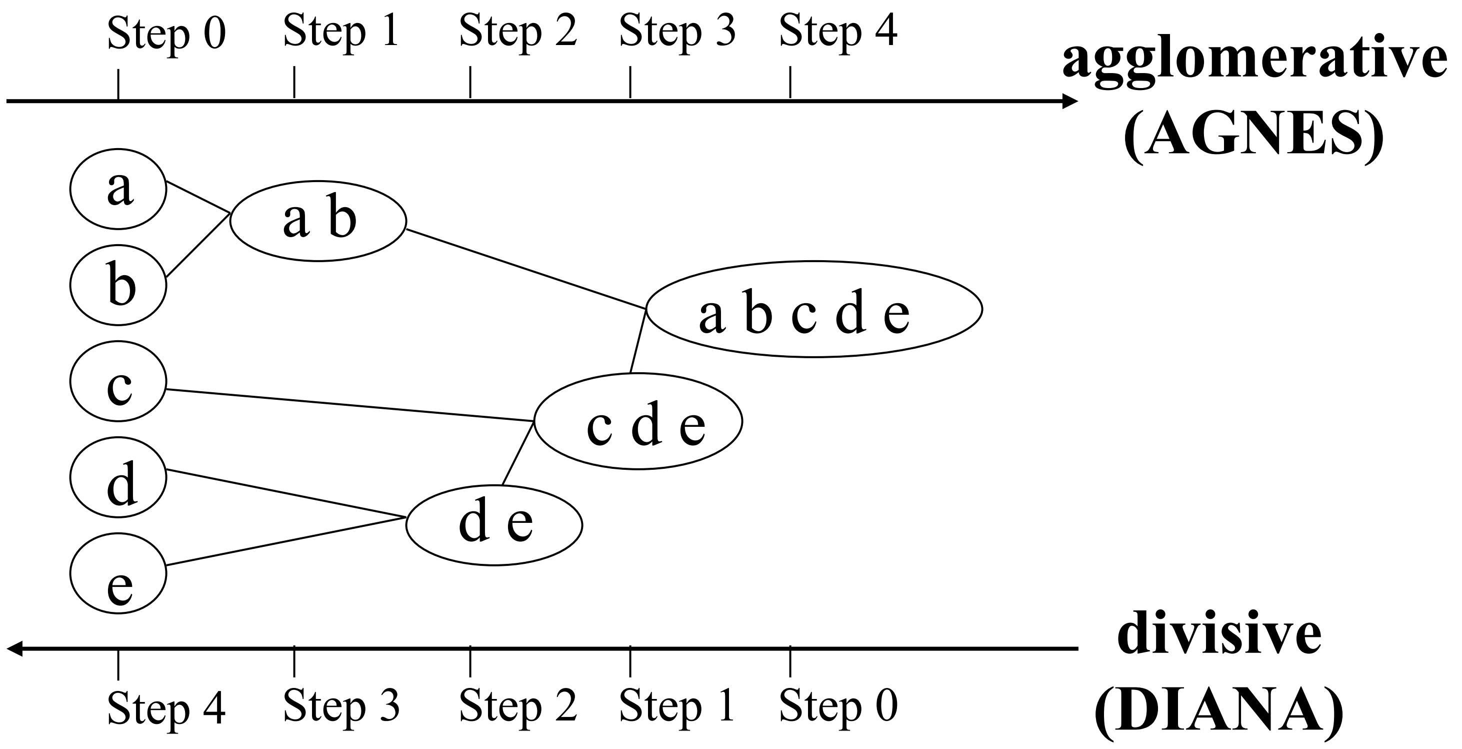
- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by ‘cutting’ the dendogram at the proper level
- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

© Tan,Steinbach, Kumar Introduction to Data Mining

Hierarchical Clustering

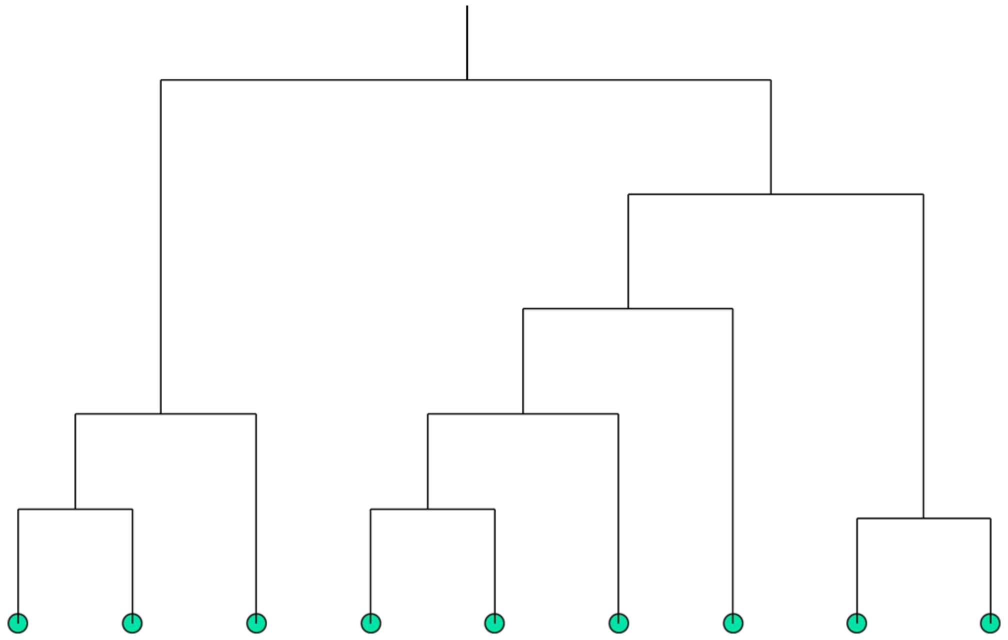
- Two main types of hierarchical clustering
  - **Agglomerative**
    - ✓ Start with the points as individual clusters
    - ✓ At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - **Divisive**
    - ✓ Start with one, all-inclusive cluster
    - ✓ At each step, split a cluster until each cluster contains a point (or there are k clusters)
- Use distance matrix as clustering criteria. This method does not require the number of clusters k as an input, but needs a termination condition

Hierarchical clustering



Dendrogram shows how the clusters are merged hierarchically

- Decompose data objects into a several levels of nested partitioning (tree of clusters), called a **dendrogram**.
- A **clustering** of the data objects is obtained by **cutting** the dendrogram at the desired level, then each connected component forms a cluster.



Agglomerative algorithm

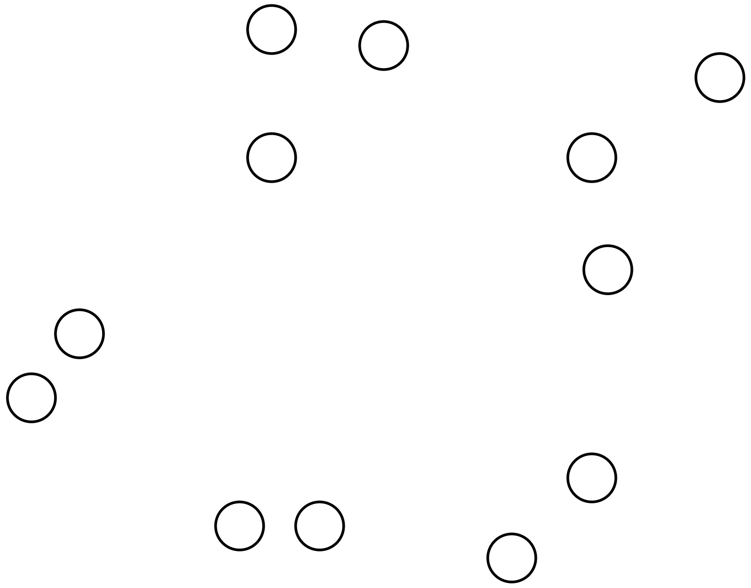
- More popular hierarchical clustering technique
- Basic algorithm is straightforward

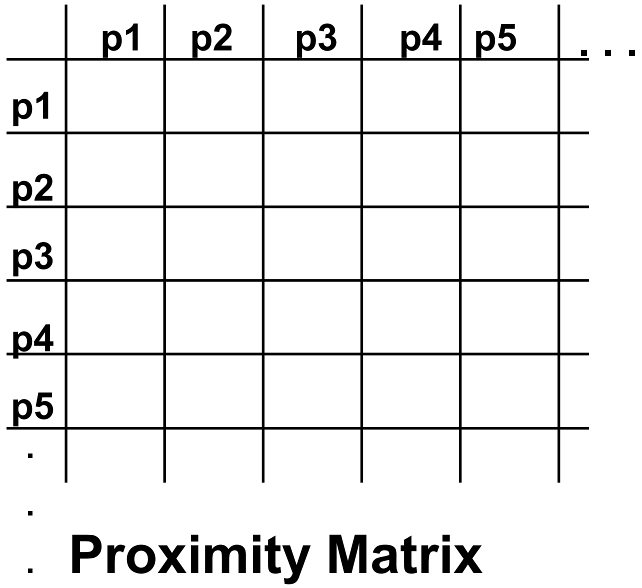
```
1. Compute the proximity matrix
2. Let each data point be a cluster
3. Repeat
4.         Merge the two closest clusters
5.         Update the proximity matrix
6. Until only a single cluster remains
```

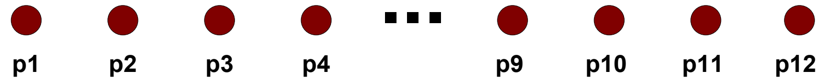
- Key operation is the computation of the proximity of two clusters
  - Different approaches to **defining the distance** between clusters distinguish the different algorithms

Starting situation

- Start with clusters of individual points and a proximity matrix





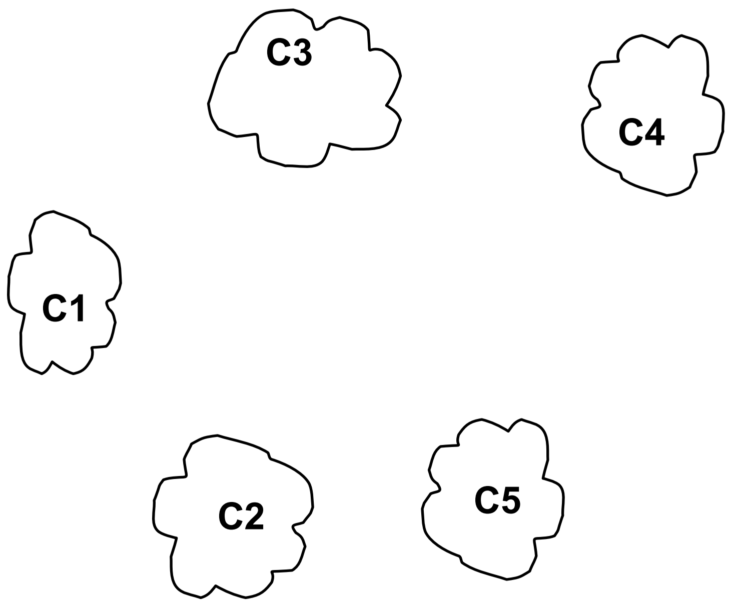


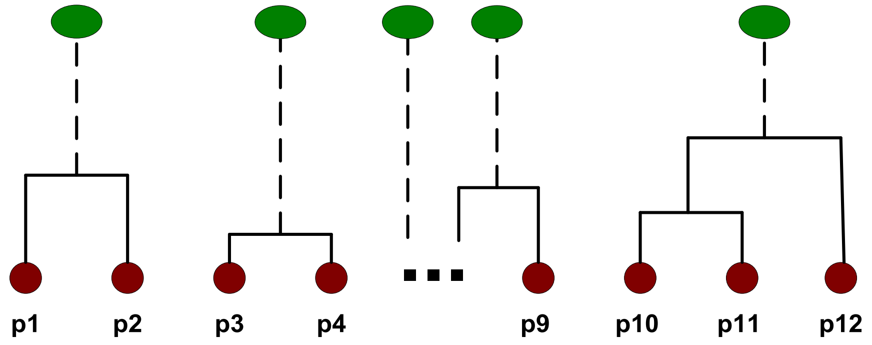
Intermediate situation

- After some merging steps, we have some clusters

	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

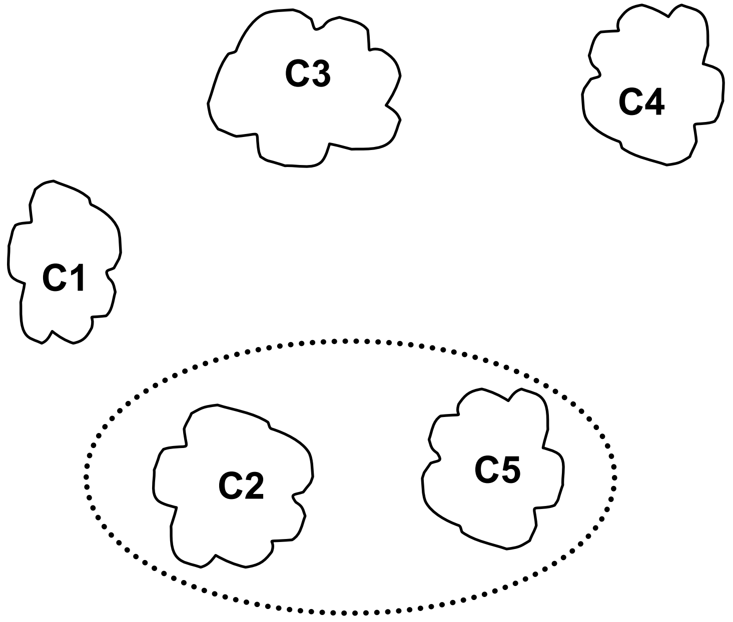
Proximity Matrix

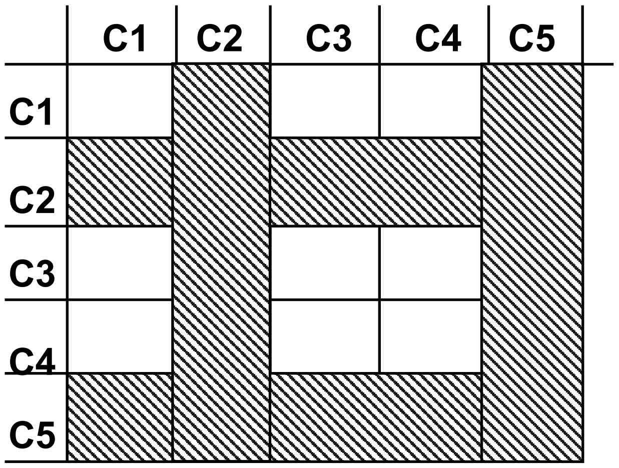




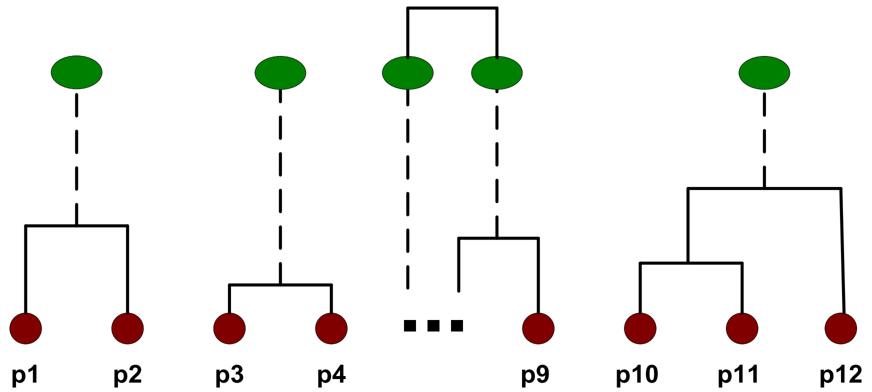
Intermediate situation

- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.





Proximity Matrix

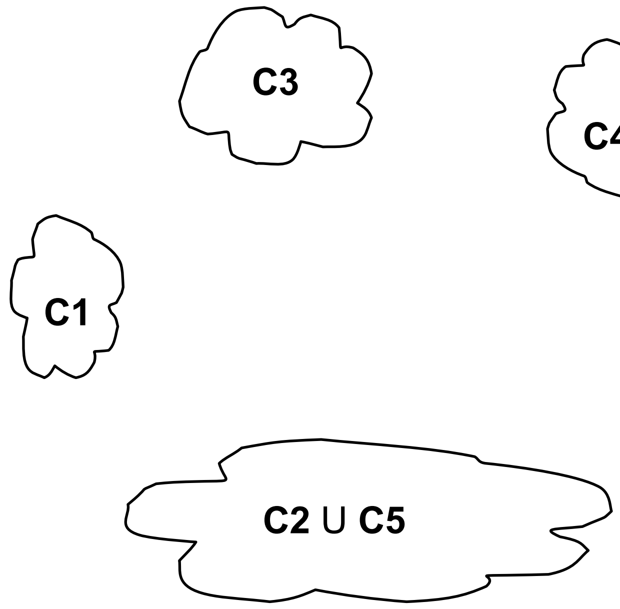


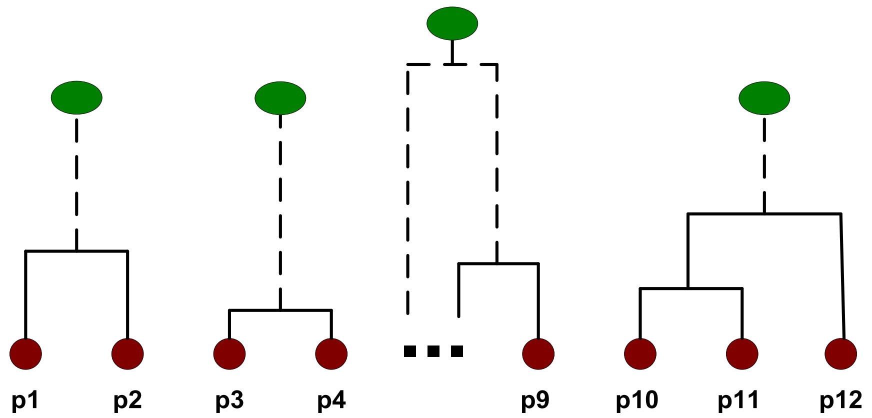
After merging

- The question is "How do we update the proximity matrix?"

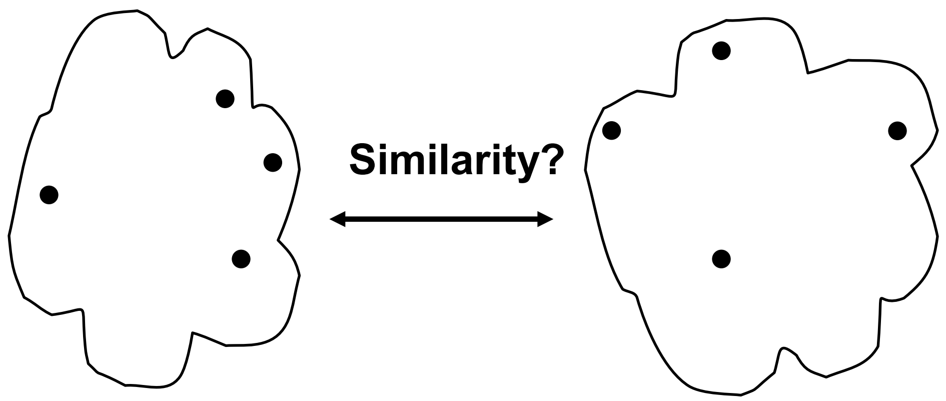
	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix





How to define inter-cluster similarity

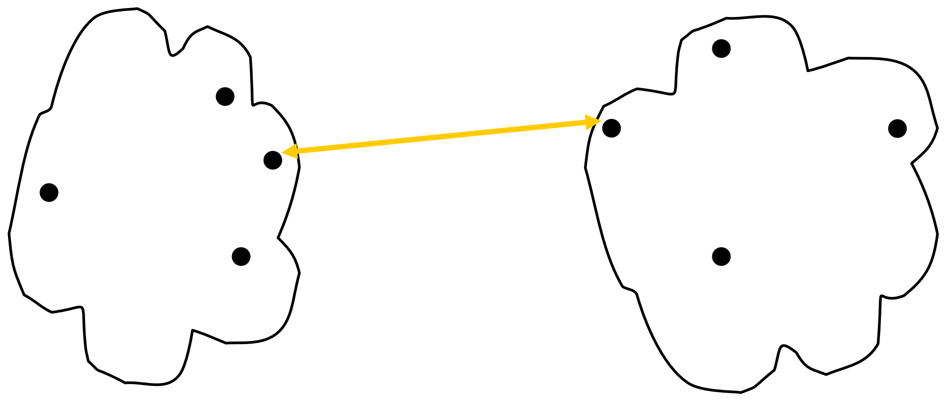


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

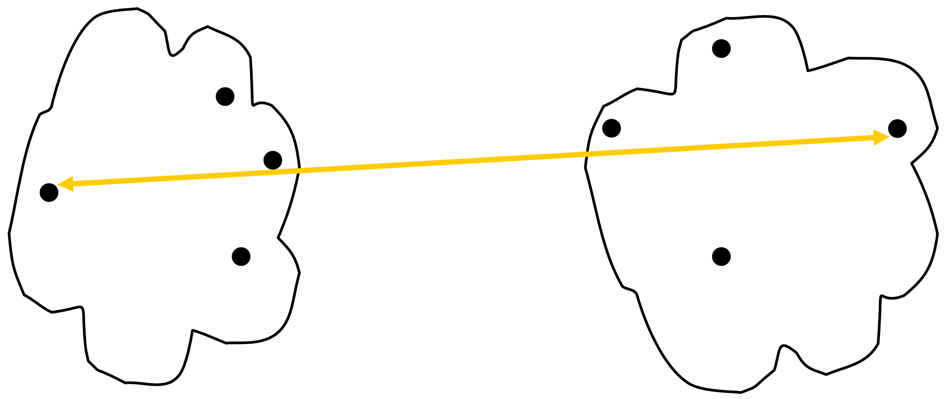


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - ✓ Ward's Method uses squared error

	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

How to Define Inter-Cluster Similarity

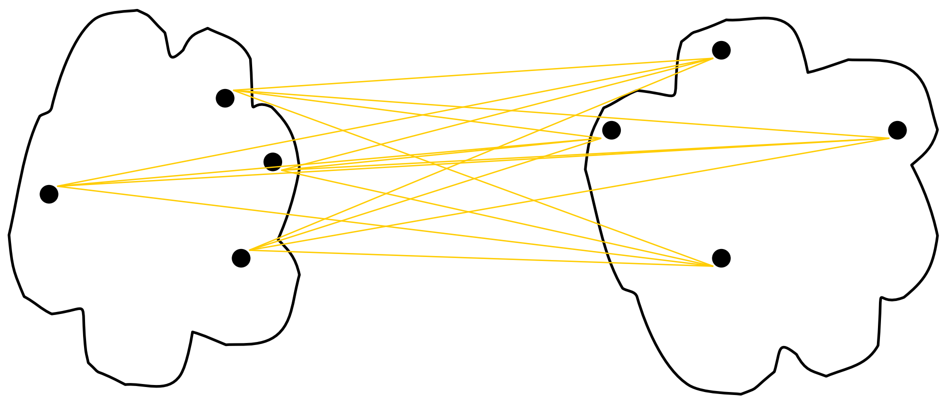


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - ✓ Ward's Method uses squared error

How to Define Inter-Cluster Similarity

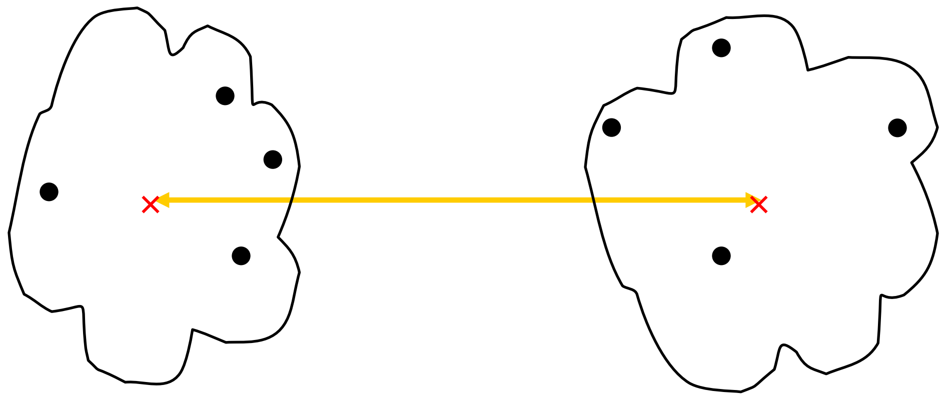


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- **Group Average**
- Distance Between Centroids
- Other methods driven by an objective function
  - ✓ Ward's Method uses squared error

How to Define Inter-Cluster Similarity



	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

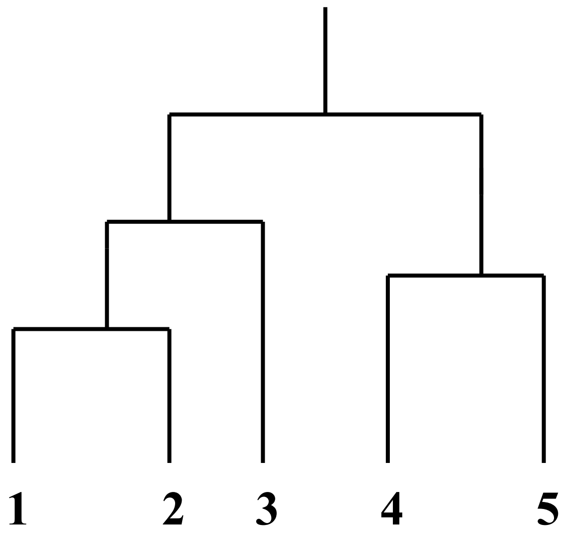
Proximity Matrix

- MIN
- MAX
- Group Average
- **Distance Between Centroids**
- Other methods driven by an objective function
  - ✓ Ward's Method uses squared error

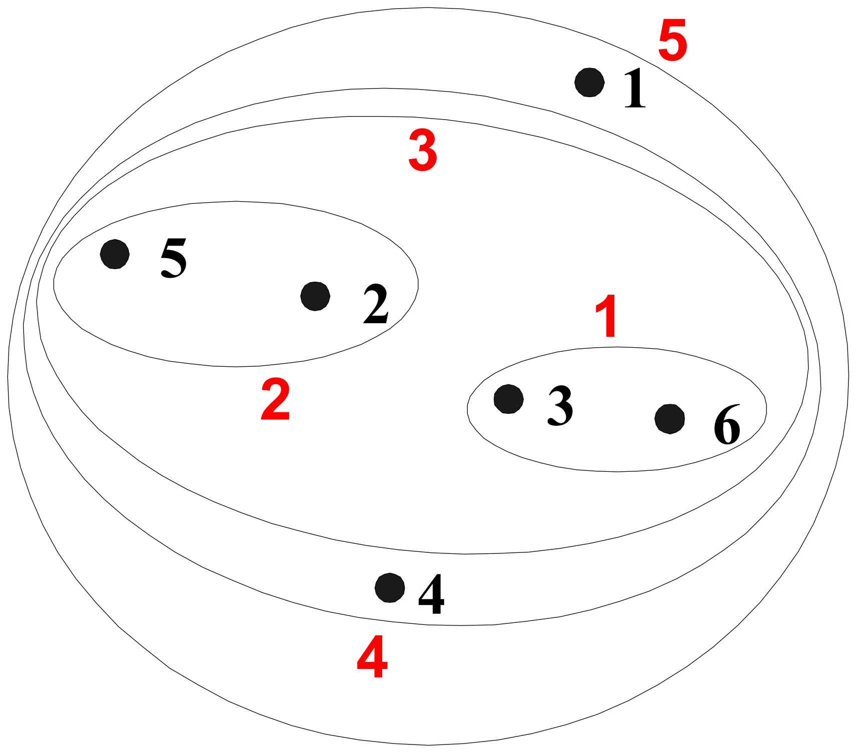
Cluster Similarity: MIN or Single Link

- Similarity of two clusters is based on the two most similar (closest) points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph.

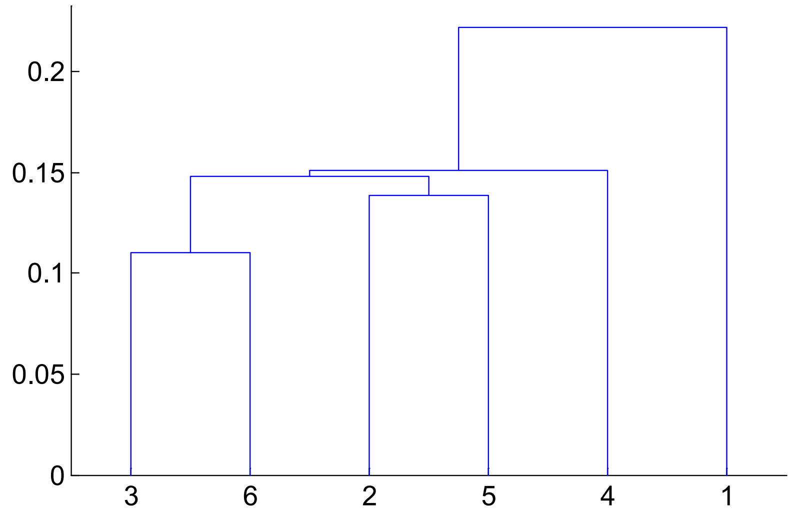
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



Hierarchical Clustering: MIN



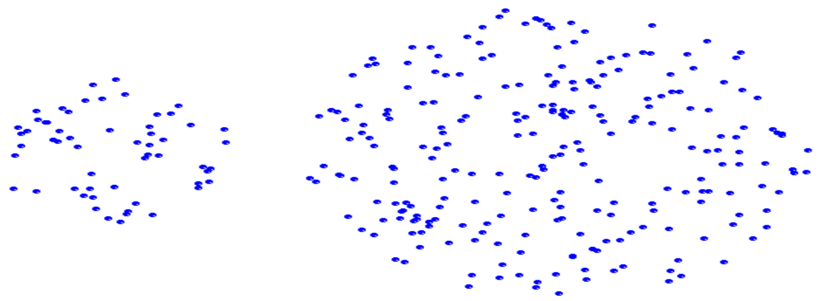
Nested Clusters

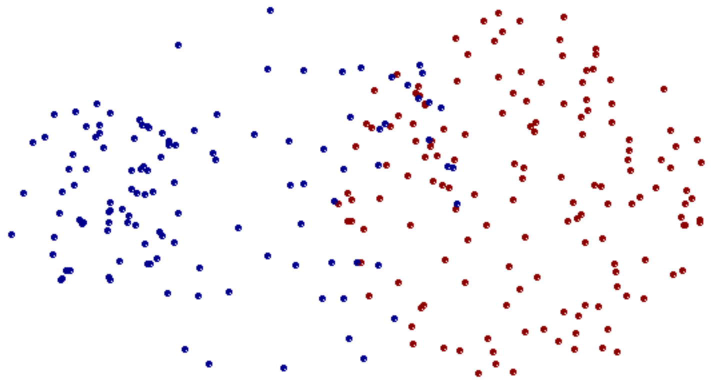


Dendrogram

Strength of MIN

- Can handle non-elliptical shapes

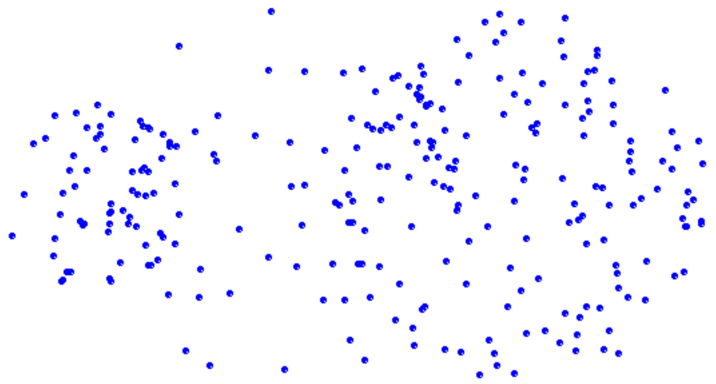

Original Points


Two Clusters

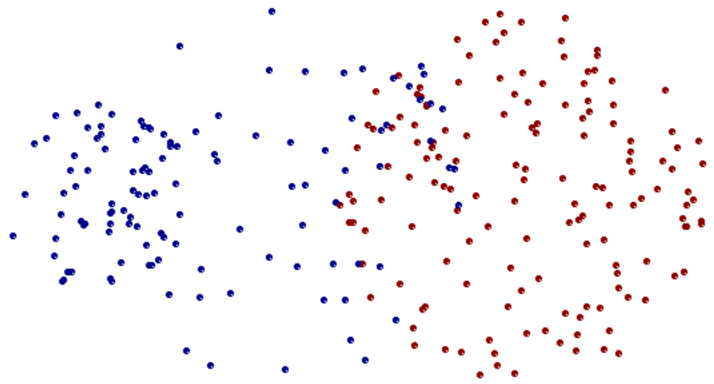
© Tan,Steinbach, Kumar Introduction to Data Mining 叶育森@西电

Limitations of MIN

- Sensitive to noise and outliers



Original Points



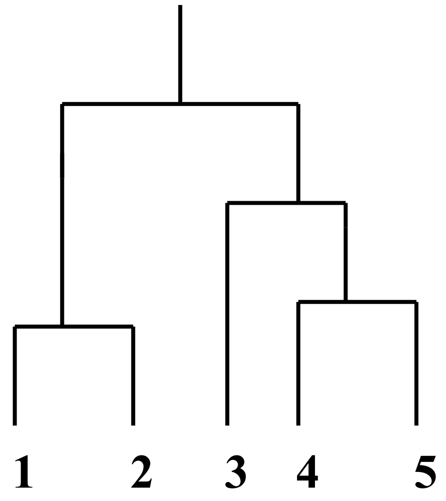
Two Clusters

© Tan,Steinbach, Kumar Introduction to Data Mining 叶育森@西电

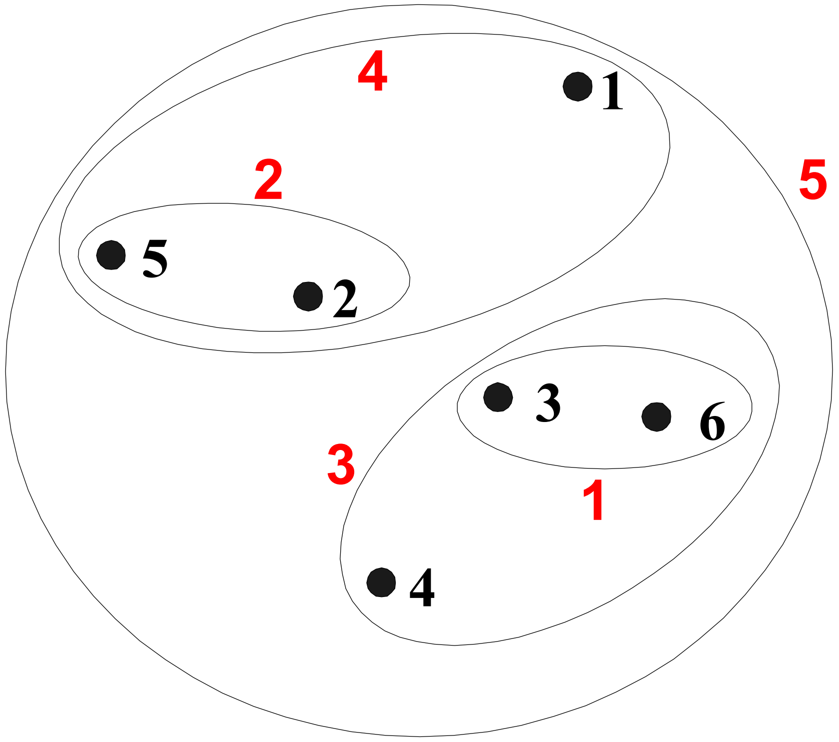
Cluster Similarity: MAX or Complete Linkage

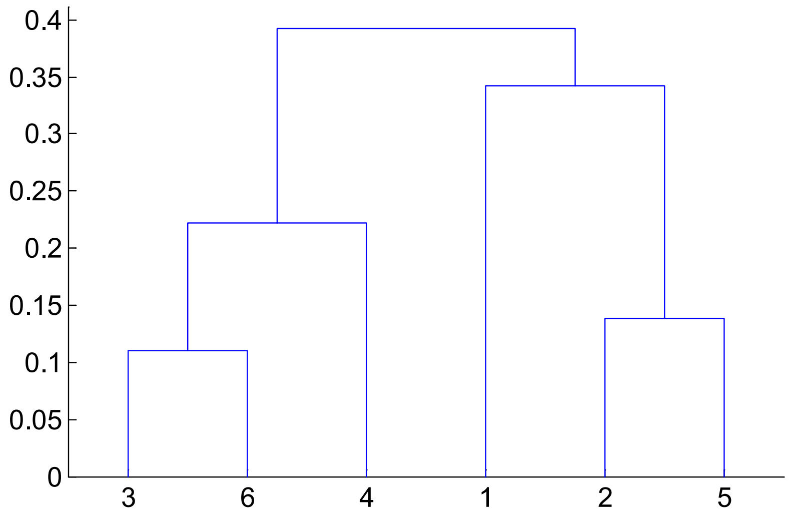
- Similarity of two clusters is based on the two least similar (most distant) points in the different clusters
  - Determined by all pairs of points in the two clusters

	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



Hierarchical Clustering: MAX

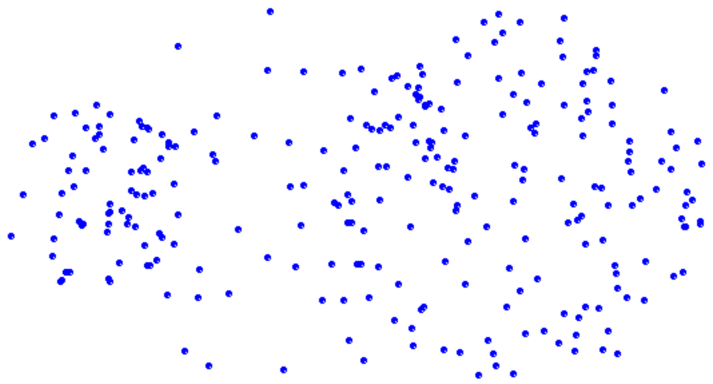




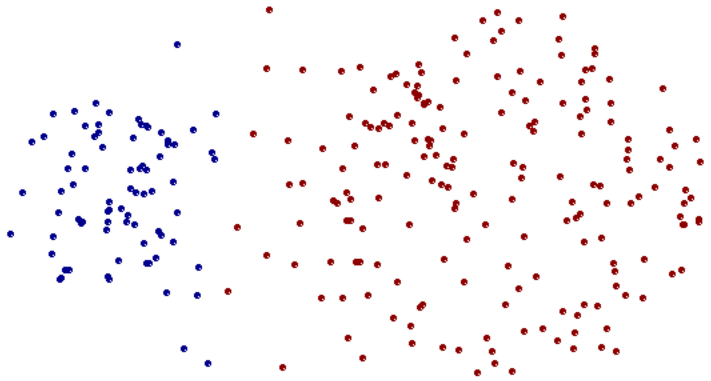
Nested Clusters **Dendrogram**

Strength of MAX

- Less susceptible to noise and outliers

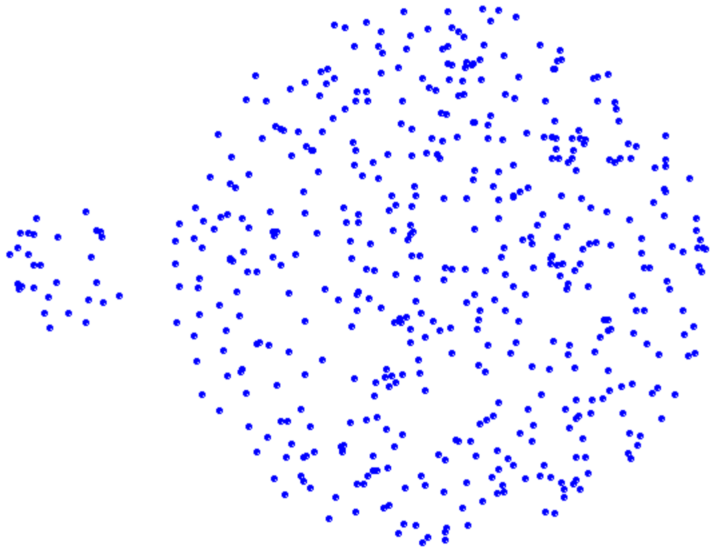


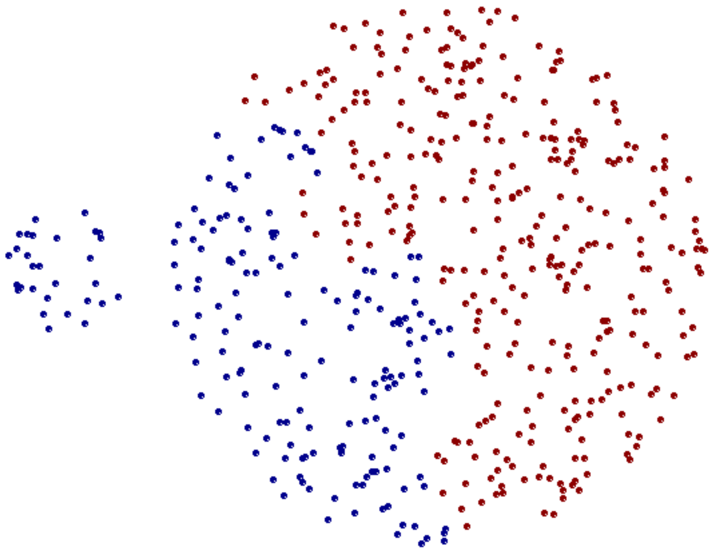
Original Points



Two Clusters

Limitations of MAX


Original Points


Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

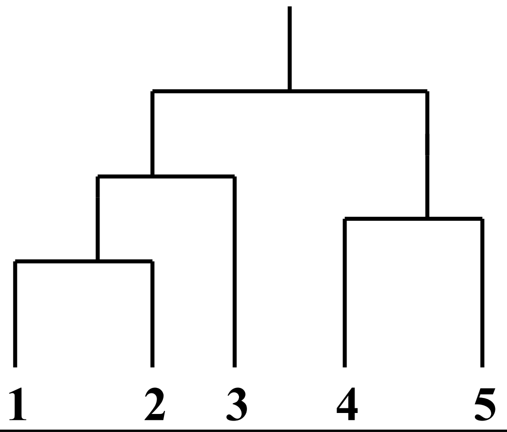
Cluster Similarity: Group Average

- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

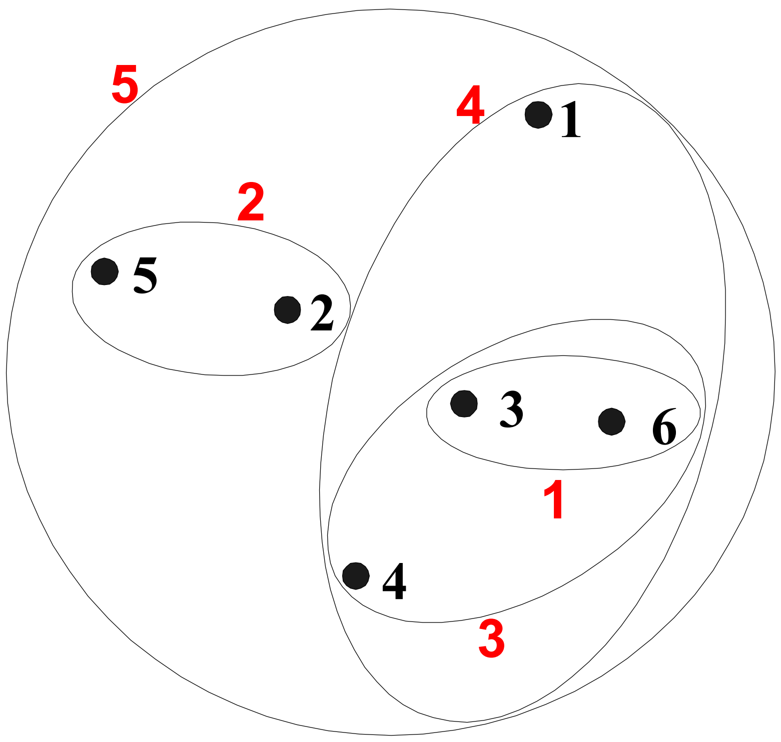
$$proximity(Cluster_i, Cluster_j) = \frac{\sum_{\substack{p_i \in Cluster_i \\ p_j \in Cluster_j}} proximity(p_i, p_j)}{|Cluster_i| \times |Cluster_j|}$$

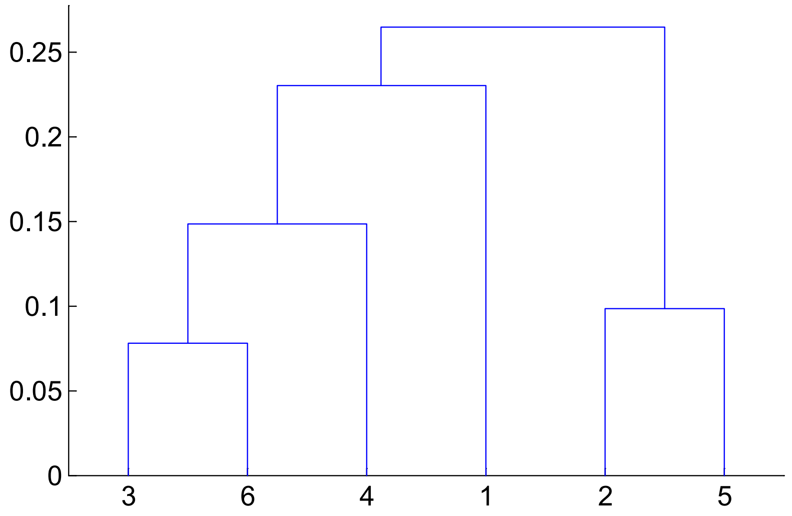
- Need to use average connectivity for scalability since total proximity favors large clusters

	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



Hierarchical Clustering: Group Average





Nested Clusters

Dendrogram

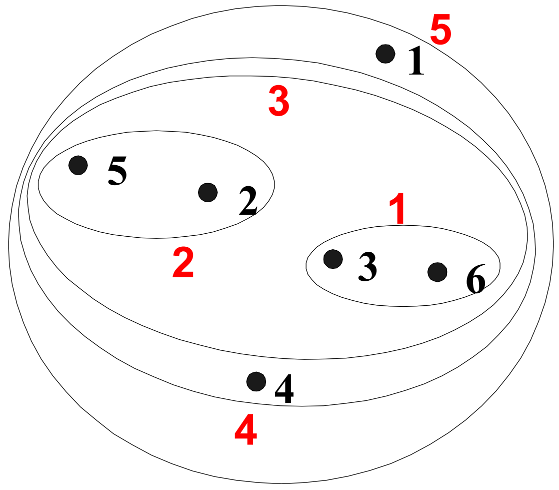
Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise and outliers
- Limitations
  - Biased towards globular(球状的) clusters

Cluster Similarity: Ward's Method

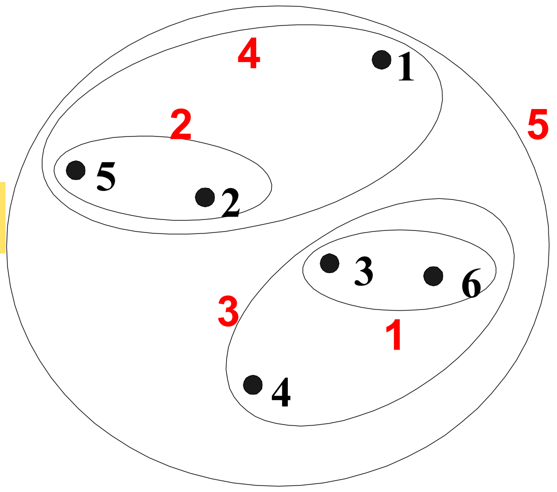
- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of *k*-means
  - Can be used to initialize *k*-means

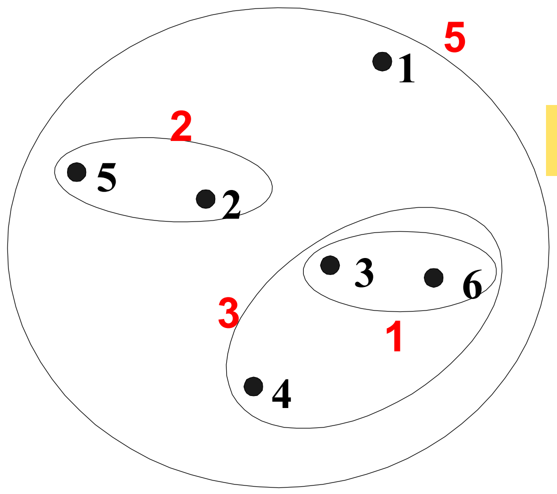
Hierarchical Clustering: Comparison



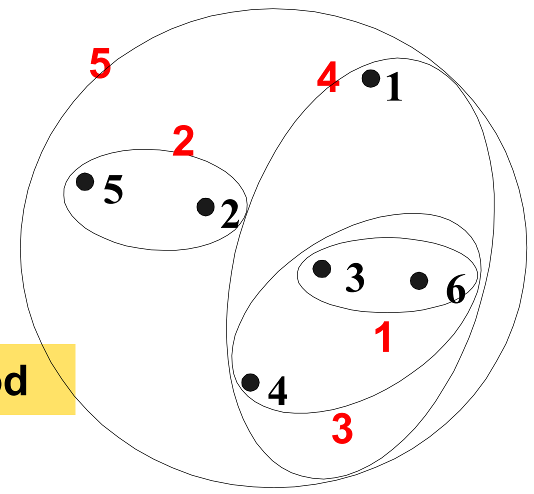
MIN

MAX





Group Average



Ward's Method

Hierarchical Clustering: Time &Space

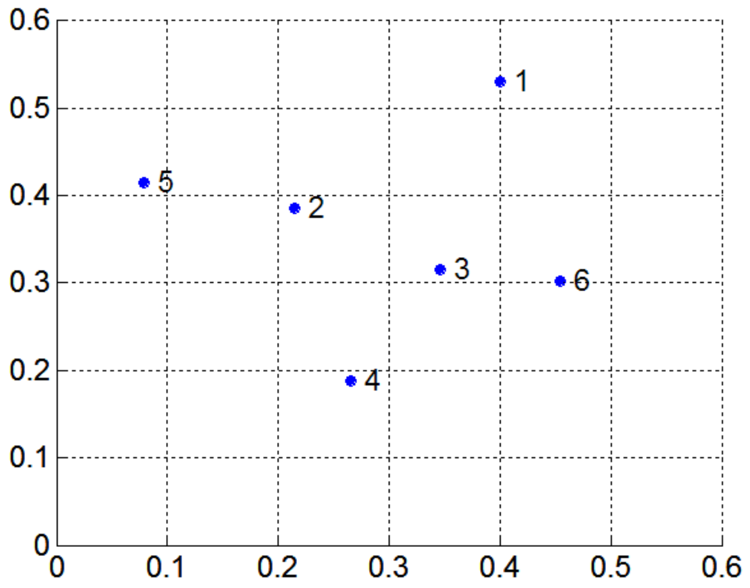
- O(n^2) space since it uses the proximity matrix
  - N is the number of points.
- O(n^3) time in many cases
  - There are N steps and at each step the size, n^2, proximity matrix must be updated and searched
  - Complexity can be reduced to O(n^2 log(n)) time for some approaches

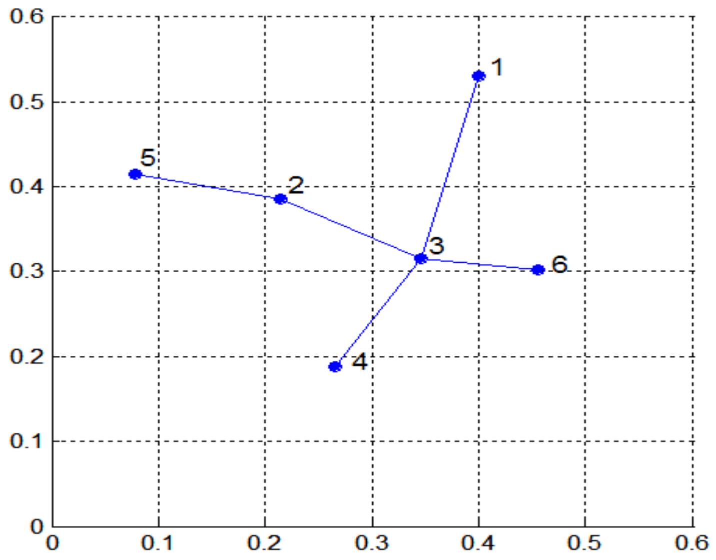
Hierarchical Clustering: Problems &Limitations

- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers
  - Difficulty handling different sized clusters and convex shapes
  - Breaking large clusters

MST: Divisive Hierarchical Clustering

- Build MST (Minimum Spanning Tree)
  - Start with a tree that consists of any point
  - In successive steps, look for the closest pair of points (p, q) such that one point (p) is in the current tree but the other (q) is not
  - Add q to the tree and put an edge between p and q

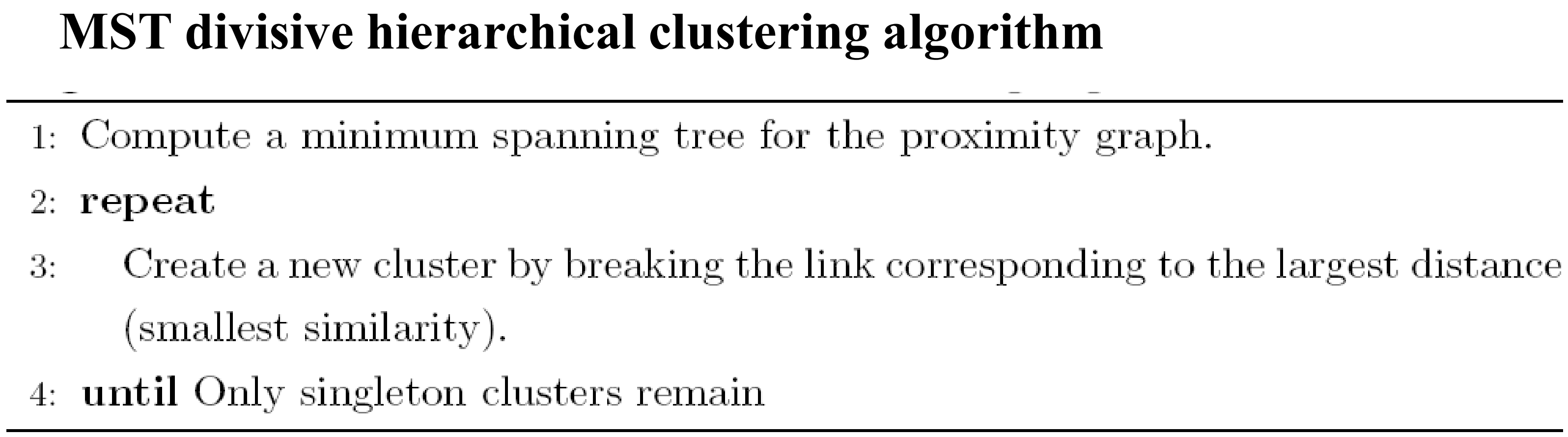




Kruskal算法，Prim算法

MST: Divisive Hierarchical Clustering

- Use MST for constructing hierarchy of clusters



MST divisive hierarchical clustering algorithm

```
1: Compute a minimum spanning tree for the proximity graph.
2: repeat
3:     Create a new cluster by breaking the link corresponding to the largest distance
       (smallest similarity).
4: until Only singleton clusters remain
```

Clustering in large database

- Major weakness of agglomerative clustering methods
  - do not scale well: time complexity of at least $O(n^2)$ where n is the number of total objects
  - can never undo what was done previously
- Integration of hierarchical with distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - ROCK (1999): clustering categorical data by neighbor and link analysis
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

BIRCH (SIGMOD'1996)

- BIRCH: Balanced Iterative Reducing and Clustering using Hierarchies
- BIRCH introduces two concepts, **clustering feature** and **clustering feature tree** (CF tree)
- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering
  - Phase 1: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)
  - Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree

BIRCH: an efficient data clustering method for very large databases

BIRCH (SIGMOD'1996)

- Achieve good speed and scalability in ***large databases***
- Effective for incremental and dynamic clustering of incoming objects
- Scales linearly: finds a good clustering with a single scan and improves the quality with a few additional scans
- Weakness: handles **only numeric data**, and sensitive to the order of the data record

ROCK: Clustering Categorical Data

- ROCK: RObust Clustering using linKs
  - S. Guha, R. Rastogi & K. Shim, ICDE'99
- Major ideas
  - Use links to measure similarity/proximity
  - Not distance-based
- Algorithm: sampling-based clustering
  - Draw random sample
  - Cluster with links
  - Label data in disk

ROCK: A Robust Clustering Algorithm for Categorical Attributes

CHAMELEON: Hierarchical clustering using dynamic modeling (1999)

- CHAMELEON: by G. Karypis, E. H. Han, &V. Kumar, 1999
- Measures the similarity based on a dynamic model
  - Two clusters are merged only if the *interconnectivity* and *closeness (proximity)* between two clusters are high *relative to* the internal interconnectivity of the clusters and closeness of items within the clusters
  - **Cure** (Hierarchical clustering with multiple representative objects) ignores information about **interconnectivity** of the objects, **Rock** ignores information about the **closeness** of two clusters
- A two-phase algorithm
  - Use a graph partitioning algorithm: cluster objects into a large number of relatively small sub-clusters
  - Use an agglomerative hierarchical clustering algorithm: find the genuine clusters by repeatedly combining these sub-clusters

Assignment pp.347~352,

6,7,11

15,16,17

Density-Based Clustering 8.4

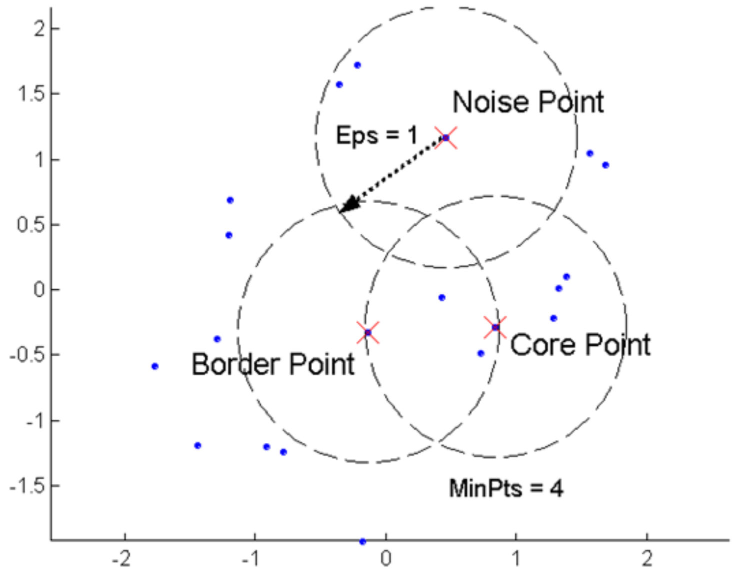
Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as density-connected points
- Major features
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters as termination condition
- Several interesting studies:
  - **DBSCAN**: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - DENCLUE: Hinneburg & D. Keim (KDD'98)
  - CLIQUE: Agrawal, et al. (SIGMOD'98)

Ester M, Kriegel HP, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Second Int Conf Knowl Discov Data Min. 1996;226–231

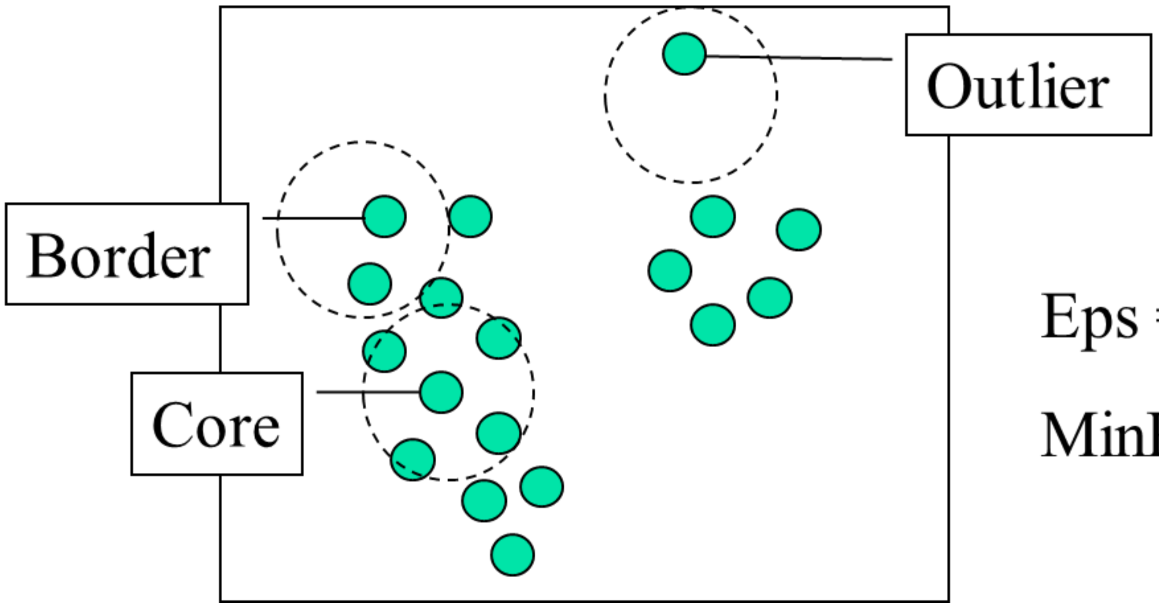
DBSCAN: Density based spatial clustering of applications with noise

- Density = number of points within a specified radius (Eps)
- A point is a **core point** if it has more than a specified number of points (MinPts) within Eps
  - These are points that are at the interior of a cluster
- A **border point** has fewer than MinPts within Eps, but is in the neighborhood of a core point
- A **noise point** is any point that is not a core point or a border point.



© Tan,Steinbach, Kumar Introduction to Data Mining 叶育森@西电

DBSCAN: Core, Border, and Noise Points



Eps = 1cm

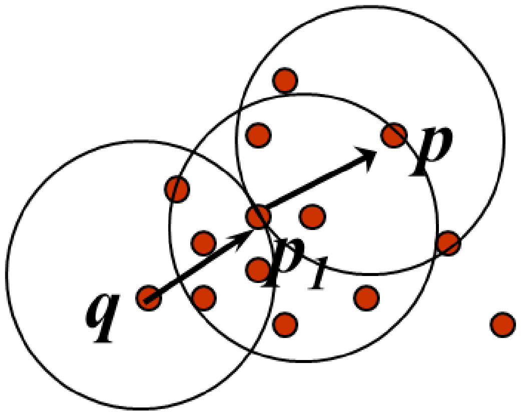
MinPts = 5

Density Concepts

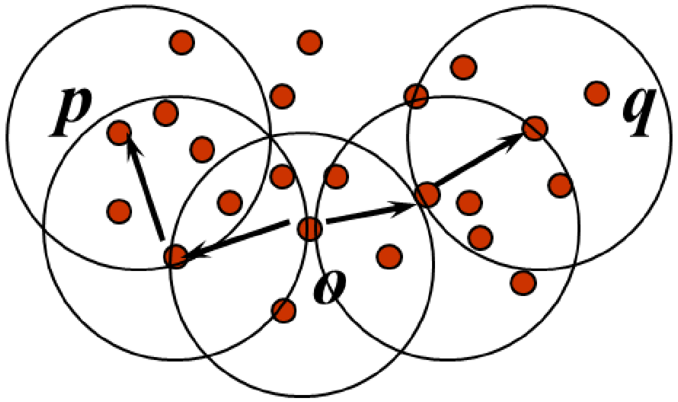
- **Eps-neighborhood**: the neighborhood within a radius Eps of a given object is called the of the object.
- **Core object (CO)** : If the Eps-neighborhood of an object contains at least a minimum number, *MinPts*, of objects
- **Directly density reachable (DDR):** Given a set of objects, *D*, an object ***p*** is directly density-reachable from object ***q*** if ***p*** is within the Eps-neighborhood of ***q***, and ***q*** is a core object.

Density Concepts (II)

- **Density-reachable (DR)**
  - A point ***p*** is density-reachable from a point ***q*** wrt. *Eps*, *MinPts* if there is a chain of points ***p*₁, …, *p*ₙ**, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i

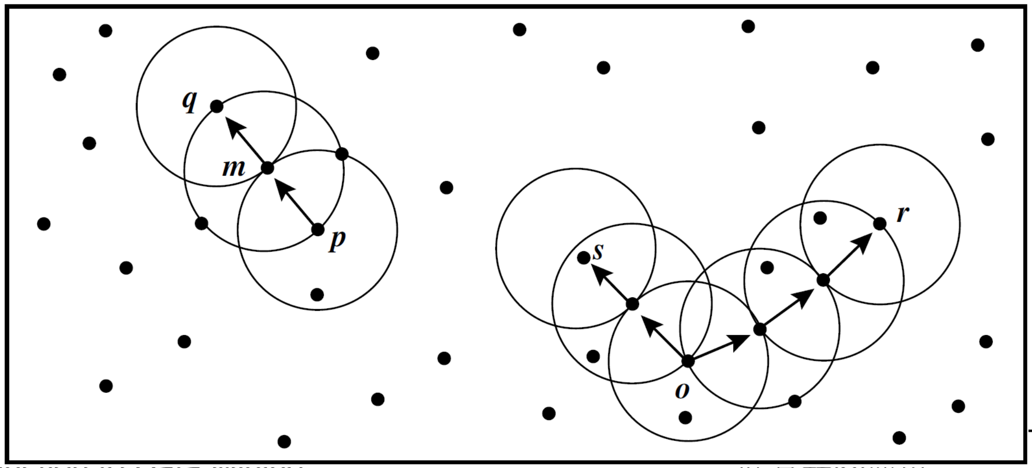


- **Density-connected (DC)**
  - A point ***p*** is density-connected to a point ***q*** wrt. Eps, MinPts if there is a point ***o*** such that both, ***p*** and ***q*** are density-reachable from ***o*** wrt. Eps and MinPts



Density-reachability and density connectivity

- let *MinPts* = 3, the labeled points, ***m***, ***p***, ***o***, and ***r*** are **CO**
- ***q*** is directly density-reachable from ***m***. ***m*** is directly density-reachable from ***p*** and vice versa.
- ***q*** is (indirectly) density-reachable from ***p*** because ***q*** is directly density-reachable from ***m*** and ***m*** is directly density-reachable from ***p***.
- ***p*** is not density-reachable from ***q*** because ***q*** is not a core object. Similarly, ***r*** and ***s*** are density-reachable from ***o***, and ***o*** is density-reachable from ***r***.
- ***o***, ***r***, and ***s*** are all density-connected



Cluster and noise

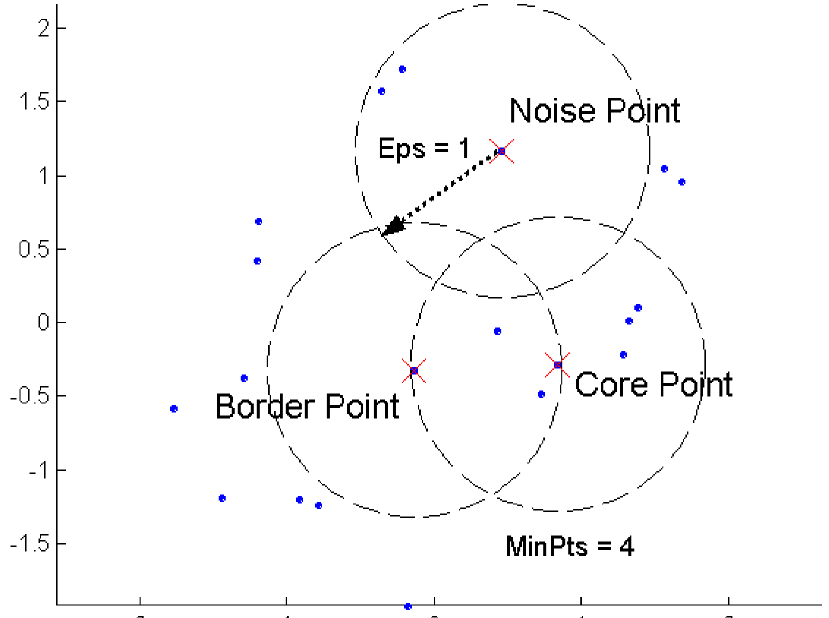
- **Cluster**: Let *D* be a database of points. A ***cluster C*** w.r.t. Eps and MinPts is a non-empty subset of *D* satisfying the following conditions
  - ∀ p, q: if p ∈ C and q is density-reachable from p wrt. Eps and MinPts, then q ∈ C. (**Maximality**)
  - ∀ p, q ∈ C: p is density-connected to q wrt. EPS and MinPts. (**Connectivity**)
- **Noise**: Let $C_1, C_2, \cdots C_k$ be the clusters of the database *D* wrt. parameters Eps_i and $MinPts_i$, $i = 1, \cdots, k$. Then we define the ***noise*** as the set of points in the database *D* not belonging to any cluster C_i , i.e. noise = $\{p \in D \mid \forall$ i: p $\notin C_i\}$

Density-Based Clustering

- A density-based cluster is a set of **density-connected (DC)** objects that is maximal with respect to density-reachability. Every object not contained in any cluster is considered to be ***noise***.
- Two parameters
  - ***Eps***: Maximum radius of the neighbourhood
  - ***MinPts***: Minimum number of points in an Eps-neighbourhood of that point
- $\mathbf{N}_{\boldsymbol{Eps}}(\boldsymbol{p})$: { q belongs to D | $dist(p,q) <= Eps$}

DBSCAN: Density Based Spatial Clustering of Applications with Noise

- Relies on a ***density-based*** notion of cluster: A ***cluster*** is defined as a maximal set of **density-connected** points
- Discovers clusters of arbitrary shape in spatial databases with noise



DBSCAN: The Algorithm

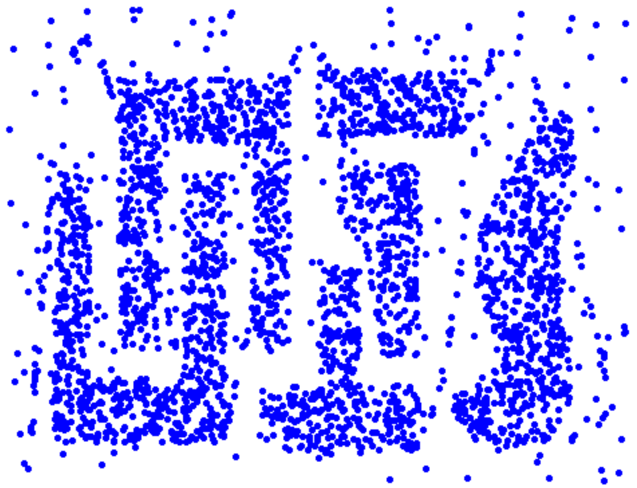
- Arbitrary select a point ***p***
- Retrieve all points **density-reachable** from ***p*** wrt ***Eps*** and ***MinPts***
- If ***p*** is a **core point**, a cluster is formed
- If ***p*** is a **border point**, no points are density-reachable from ***p*** and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed

DBSCAN: The Algorithm

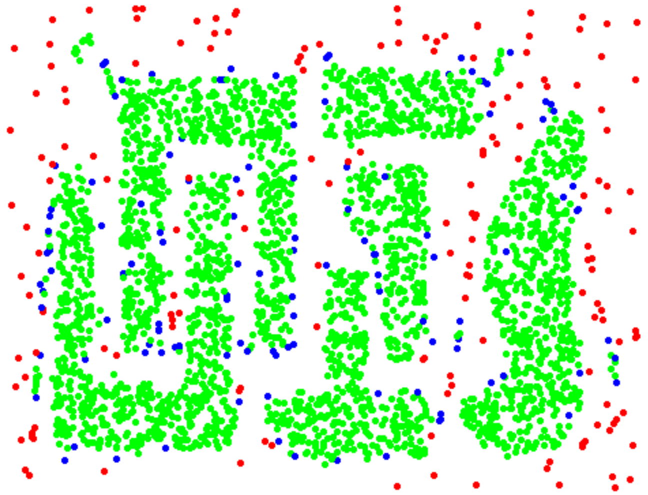
- Eliminate noise points
- Perform clustering on the remaining points

```
$current\_cluster\_label \leftarrow 1$
for all core points do
  if the core point has no cluster label then
    $current\_cluster\_label \leftarrow current\_cluster\_label + 1$
    Label the current core point with cluster label $current\_cluster\_label$
  end if
  for all points in the $Eps$-neighborhood, except $i^{th}$ the point itself do
    if the point does not have a cluster label then
      Label the point with cluster label $current\_cluster\_label$
    end if
  end for
end for
```

DBSCAN: Core, Border and Noise Points



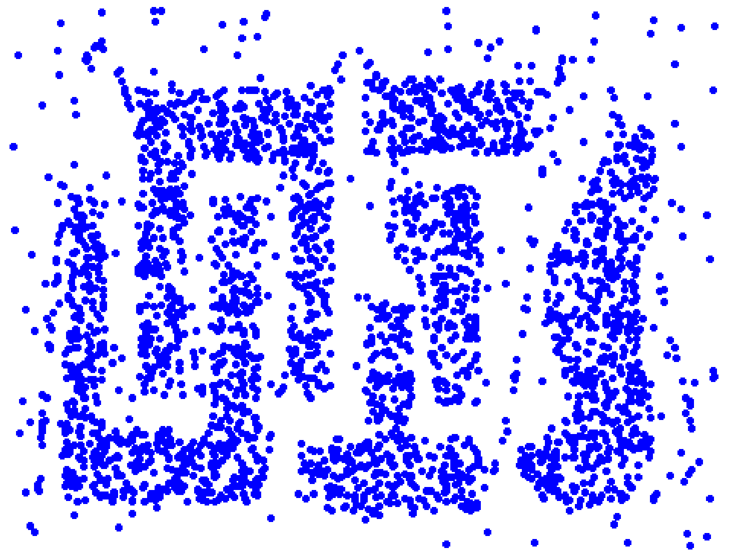
Original Points

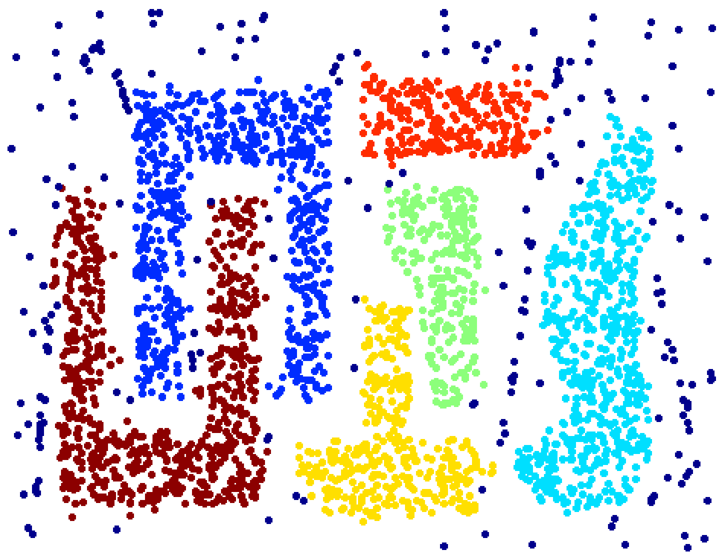


Point types: **core**, **border** **and noise**

Eps = 10, MinPts = 4

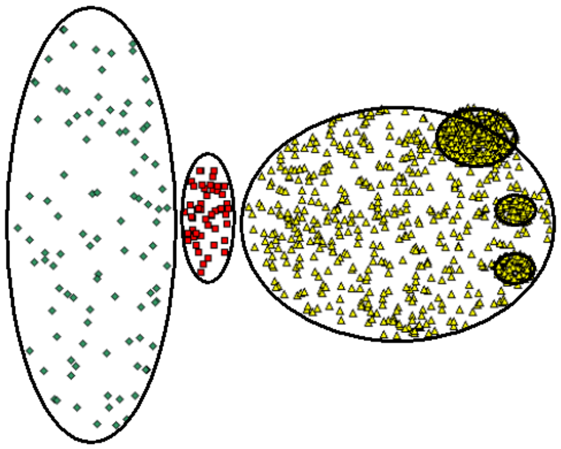
Strength: When DBSCAN Works Well


Original Points


Clusters

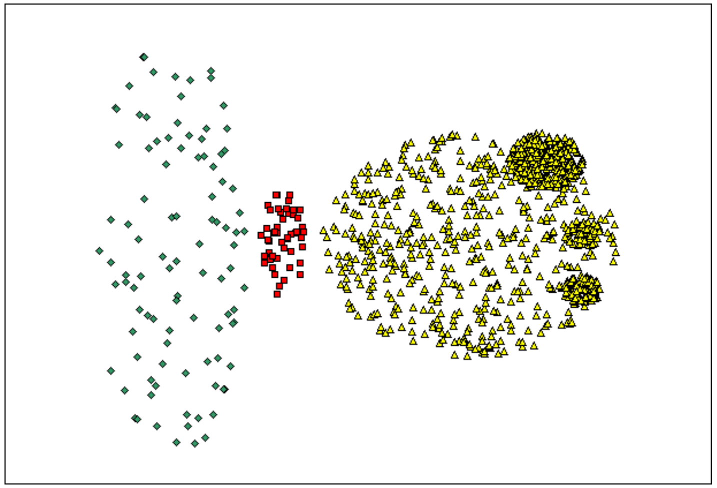
- **Resistant to Noise**
- **Can handle clusters of different shapes and sizes**

Weaknesses:When DBSCAN Does NOT Work Well

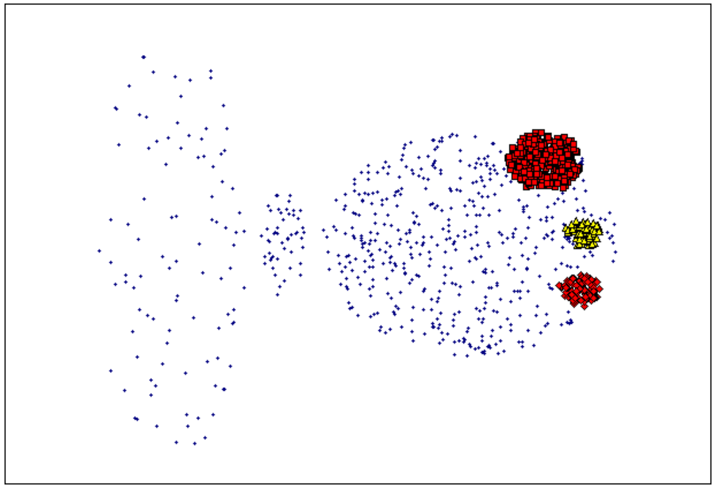


Original Points

(MinPts=4, Eps=9.75)



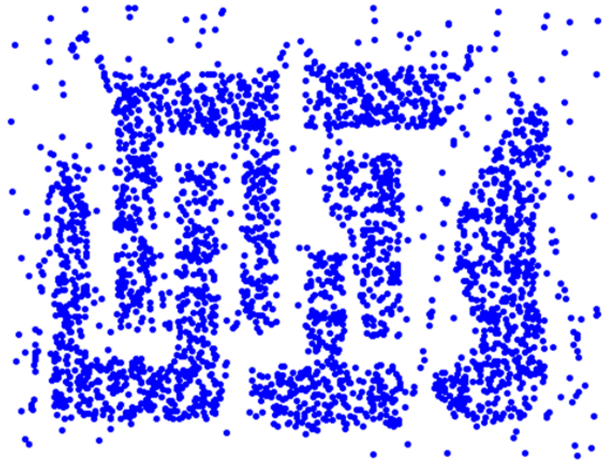
(MinPts=4, Eps=9.92)

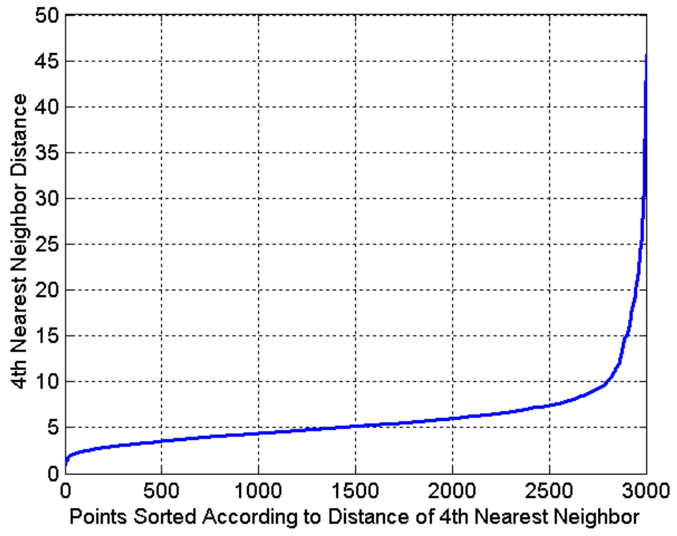


- Widely varying densities
- High-dimensional data
- High complexity

DBSCAN: Determining EPS and MinPts

- Idea is to look at the distance from a point to its ***k*^th^** nearest neighbors , call ***k*-dist**
  - For points in a cluster, the value of *k*-dist will be small
  - For points not in a cluster, noise points, *k*-dist will be relatively large
- Compute the ***k*-dist** for all points, plot sorted distance of every point in increasing order, sharp change corresponds to Eps

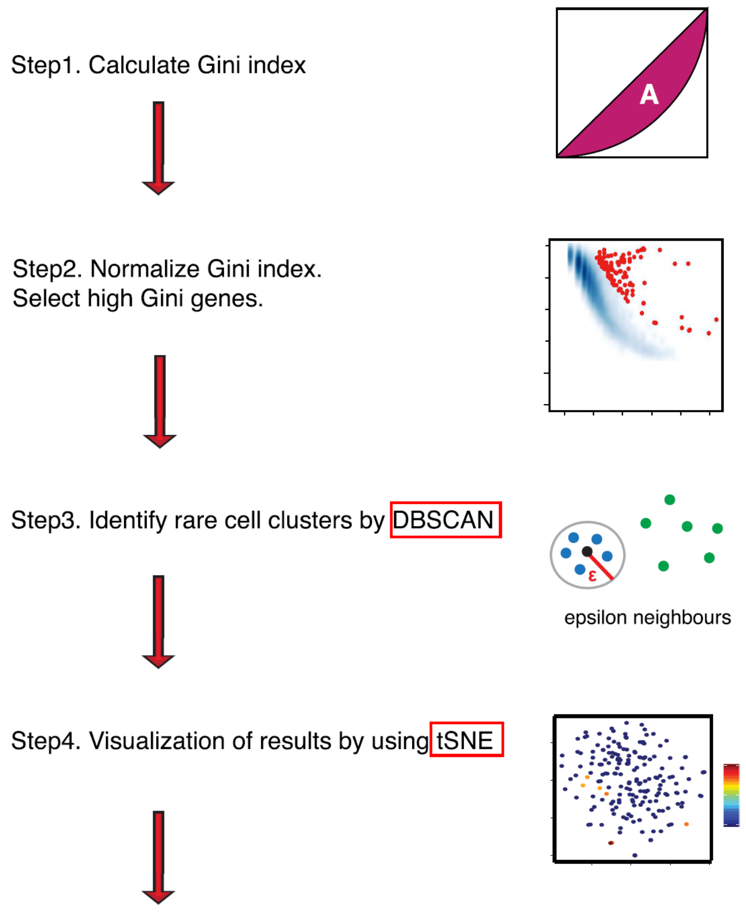




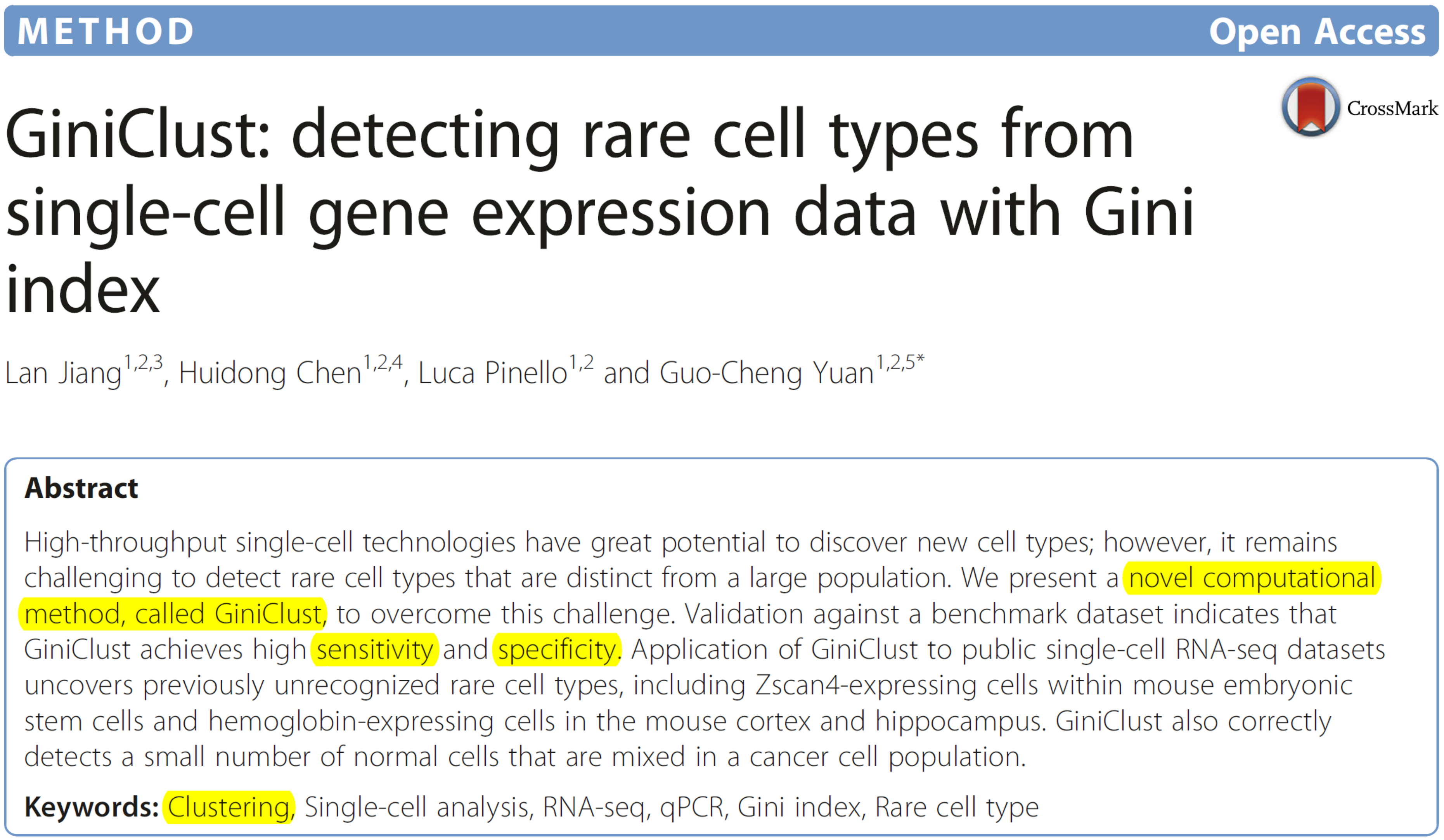
Application: Detecting rare cell types from single-cell gene expression data with Gini index

Identify rare cell clusters by ***DBSCAN***

- A number of distance metrics may be used for clustering, depending on the statistical property of the gene expression data.
- Using a standardized parameter to analyze real and simulated RNA-seq datasets with MinPts = 3, eps = 0.5
- To test robustness, varied the parameters over a range of values and found that the results are not significantly affected


Step1. Calculate Gini index
A
Step2. Normalize Gini index.
Select high Gini genes.
Step3. Identify rare cell clusters by DBSCAN
ε
epsilon neighbours
Step4. Visualization of results by using tSNE

The **Gini index** is the most commonly used measure of inequality. It was developed by Italian statistician Corrado Gini (1884–1965) and is named after him.



METHOD Open Access

GiniClust: detecting rare cell types from single-cell gene expression data with Gini index

Lan Jiang[1,2,3], Huidong Chen[1,2,4], Luca Pinello[1,2] and Guo-Cheng Yuan[1,2,5*]

Abstract

High-throughput single-cell technologies have great potential to discover new cell types; however, it remains challenging to detect rare cell types that are distinct from a large population. We present a novel computational method, called GiniClust, to overcome this challenge. Validation against a benchmark dataset indicates that GiniClust achieves high sensitivity and specificity. Application of GiniClust to public single-cell RNA-seq datasets uncovers previously unrecognized rare cell types, including Zscan4-expressing cells within mouse embryonic stem cells and hemoglobin-expressing cells in the mouse cortex and hippocampus. GiniClust also correctly detects a small number of normal cells that are mixed in a cancer cell population.

Keywords: Clustering, Single-cell analysis, RNA-seq, qPCR, Gini index, Rare cell type

Cluster Evaluation 8.5

Cluster Validation 5.1

- For classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall

$$\Pr e\,cision(p) = \frac{TP}{TP+FP}$$

$$\mathrm{Re}\,c\,all(r) = \frac{TP}{TP+FN}$$

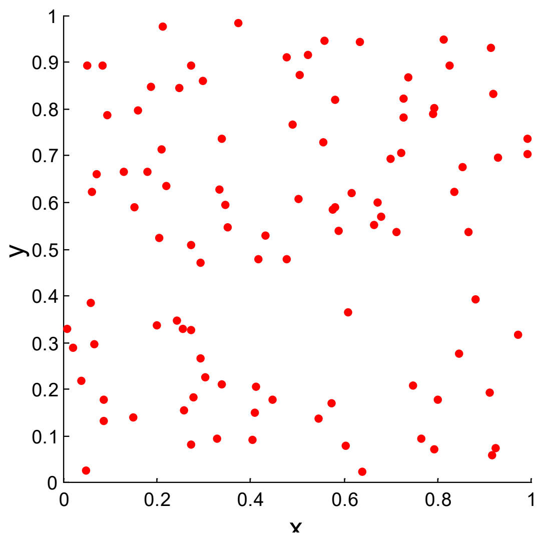
$$Acc(M) = \frac{TP+TN}{TP+TN+FP+FN}$$

- For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?
- Why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

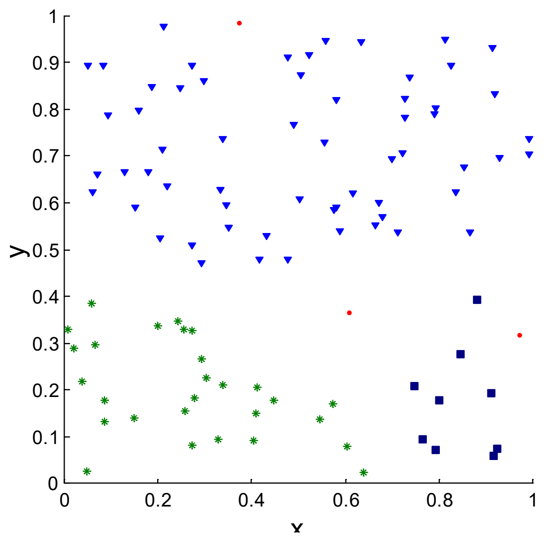
© Tan,Steinbach, Kumar Introduction to Data Mining

Clusters found in Random Data

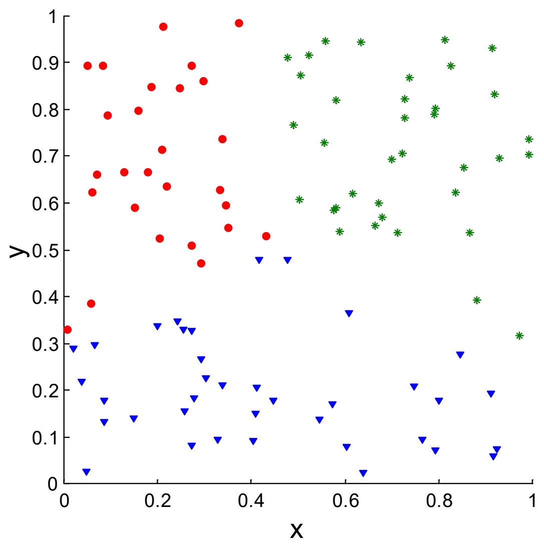
Random Points



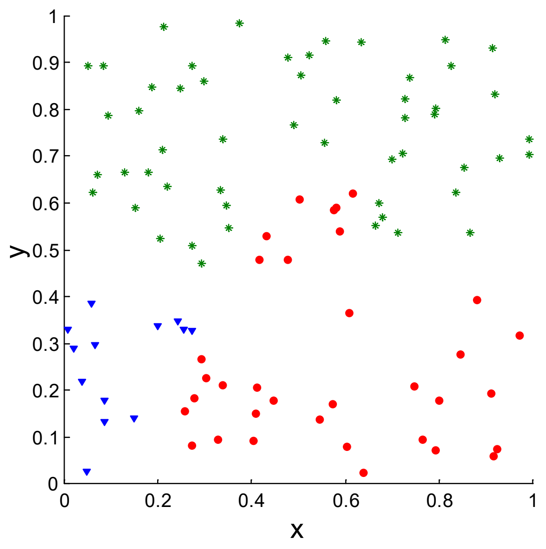
DBSCAN



K-means



Complete Link



Different Aspects of Cluster Validation

Step1: Determining the **clustering tendency** of a set of data, i.e., distinguishing whether ***non-random structure*** actually exists in the data

Step2: Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.

Step3: Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.

- ✓ Use only the data

Step4: Comparing the results of two different sets of cluster to determine which is better.

Step5: Determining the‘correct’ number of clusters.

For 2, 3, and 4, further distinguish whether evaluating the entire clustering /individual clusters

Measures of Cluster Validity

- **Numerical measures are applied to judge various aspects**
  - **Internal Index**: to measure the goodness of a clustering structure *without* respect to external information
    - ✓ Sum of Squared Error (SSE)
  - **External Index**: to measure the extent to which cluster labels match externally supplied class labels
    - ✓ Entropy
  - **Relative Index**: to compare two different clusterings or clusters.
    - ✓ Often an external or internal index is used for this function, e.g., SSE or entropy
- **Sometimes these are referred to as criteria instead of indices**
  - However, sometimes criterion is the **general strategy** and index is the **numerical measure** that implements the criterion.

Unsupervised Cluster Evaluation
Cohesion(凝聚度) & Separation(分离度) 5.2

Internal Measures: SSE

- Clusters in more complicated figures aren't well separated
- **Internal Index**: Used to measure the goodness of a clustering structure without respect to external information
  - SSE

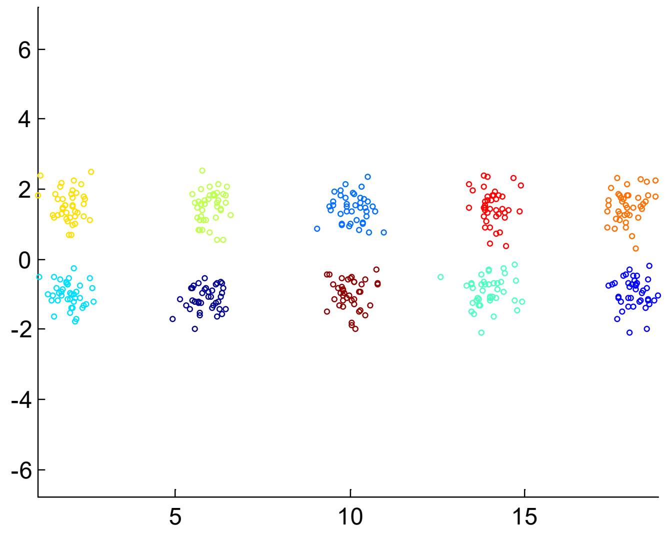
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(c_i, x)$$

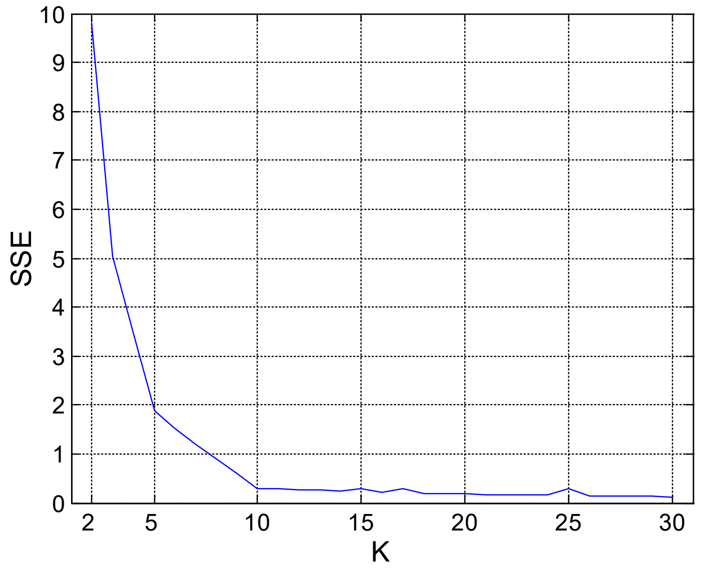
$$c_i = \frac{1}{m_i} \sum_{x \in C_i} x$$

- **SSE** is good for comparing two clusterings or two clusters (average SSE).

Internal Measures: SSE

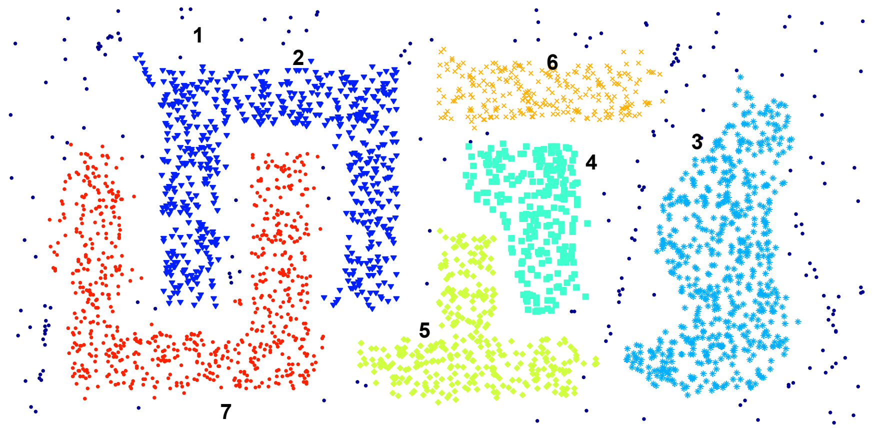
- Can also be used to estimate the number of clusters

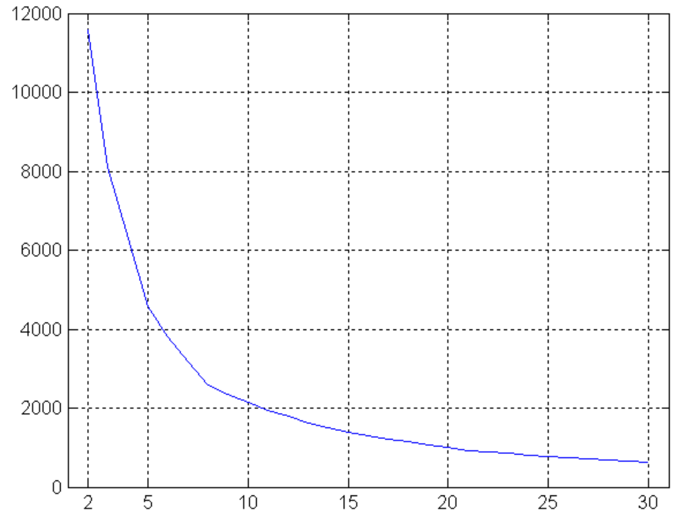




Internal Measures: SSE

- SSE curve for a more complicated data set





SSE of clusters found using K-means

Internal Measures: Cohesion & Separation

- **Cohesion(凝聚度)**: Measures how closely related are objects in a cluster

$$cohesion(C_i) = \sum_{\substack{x \in C_i \\ y \in C_i}} proximity(x, y)$$

- **Separation(分离度)**: Measure how distinct or well-separated a cluster is from other clusters

$$separation(C_i, C_j) = \sum_{\substack{x \in C_i \\ y \in C_j}} proximity(x, y)$$

- Example: Squared Error
  - Cohesion is measured by the within cluster **Sum of Squares Error (SSE)**

$$SSE = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

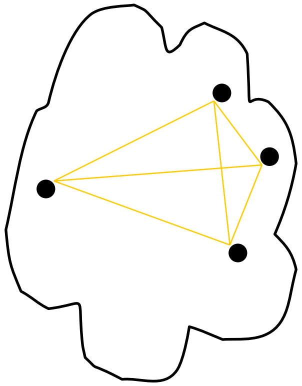
  - Separation is measured by the **Between cluster Sum of Squares(SSB)**

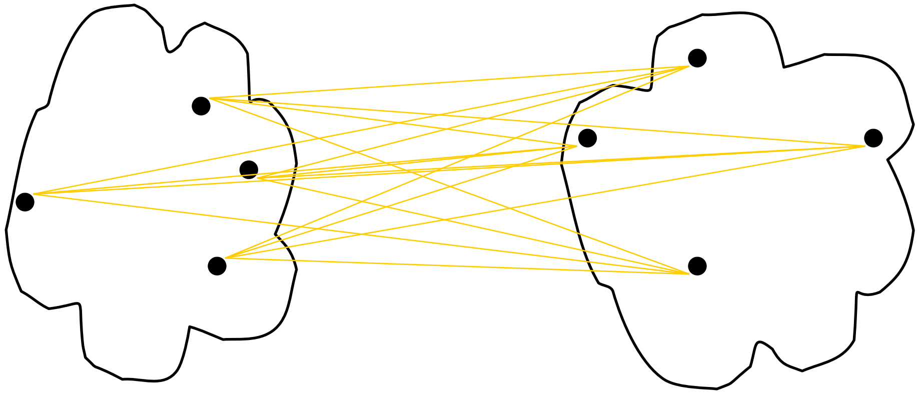
$$SSB = \sum_i |C_i| (m - m_i)^2$$

Where $|C_i|$ is the size of cluster i , m is the average of m_i

Internal Measures: Cohesion & Separation

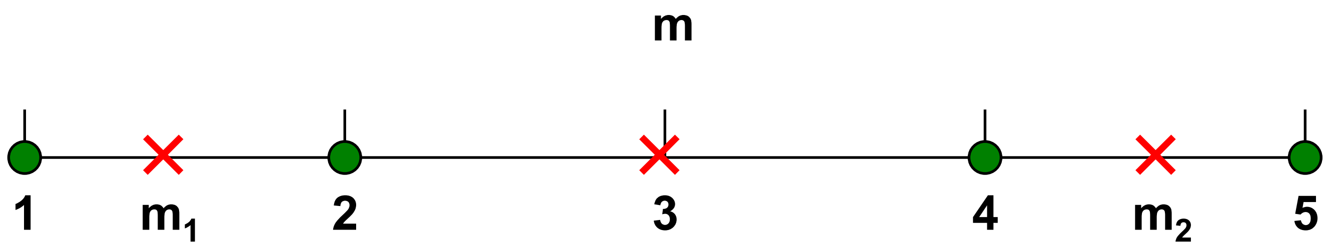
- A proximity **graph based approach** can also be used for cohesion and separation.
  - Cluster cohesion is the sum of the weight of all links within a cluster.
  - Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.


cohesion


separation

Internal Measures: Cohesion and Separation

- Relationship between cohesion and separation
  - Example: SSE
    - ✓ SSB + SSE = TSS (constant)   (Total of Sum of Squares)



K=1 cluster:

$$SSE = (1-3)^2 + (2-3)^2 + (4-3)^2 + (5-3)^2 = 10$$
$$SSB = 4 \times (3-3)^2 = 0$$
$$Total = 10 + 0 = 10$$

K=2 clusters:

$$SSE = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$
$$SSB = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$
$$Total = 1 + 9 = 10$$

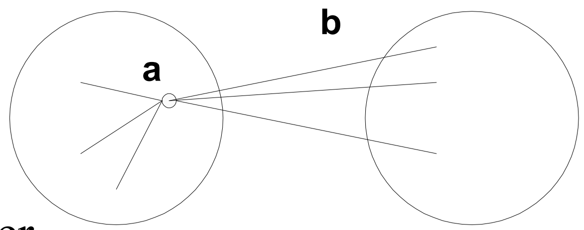
Minimizing SSE (cohesion) is equivalent to maximizing SSB (separation)

Internal Measures: Silhouette Coefficient

- Silhouette Coefficient combing ideas of both cohesion and separation
- For an individual point, *i*
  - Calculate ***a*** = average distance of *i* to the points in its cluster
  - Calculate ***b*** = min (average distance of *i* to points in another cluster)
  - The silhouette coefficient for a ***point***

    $$s_i = (b_i - a_i) / \max(a_i, b_i)$$

  - Typically between -1 and 1, a_i closer to 0 the better



- Calculate the **Average Silhouette** width for a ***cluster*** or a ***clustering***

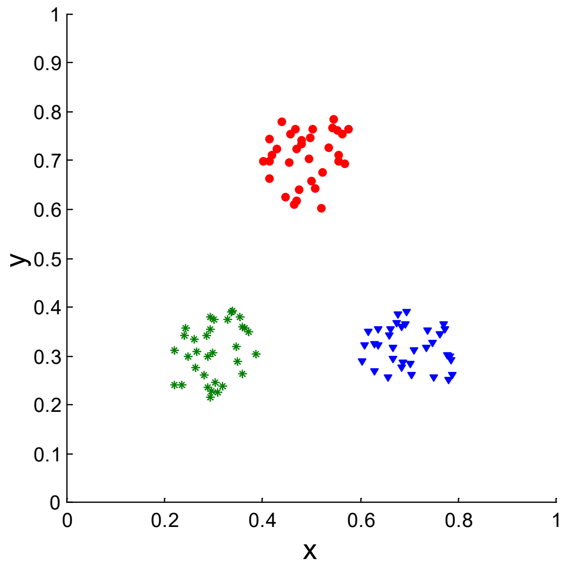
Unsupervised Cluster Evaluation
Proximity Matrix 5.3

Measuring Cluster Validity Via Correlation

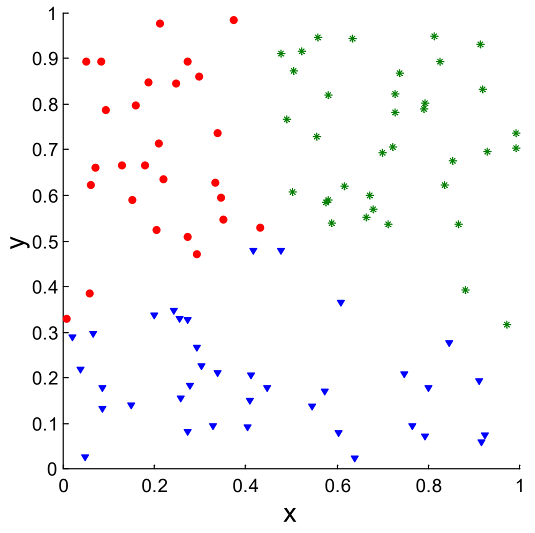
- Proximity/ incidence matrix
  - One row and one column for each data point
  - An entry is 1 if the associated pair of points belong to the same cluster
  - An entry is 0 if the associated pair of points belongs to different clusters
- Compute the ***correlation*** between the two matrices
  - Matrices are symmetric
- High correlation indicates that points that belong to the same cluster are close to each other
- Not a good measure for some **density** or **contiguity** based clusters

Internal Measures: Correlation

- Correlation of incidence and proximity matrices for the *k*-means clusterings of the following two data sets.



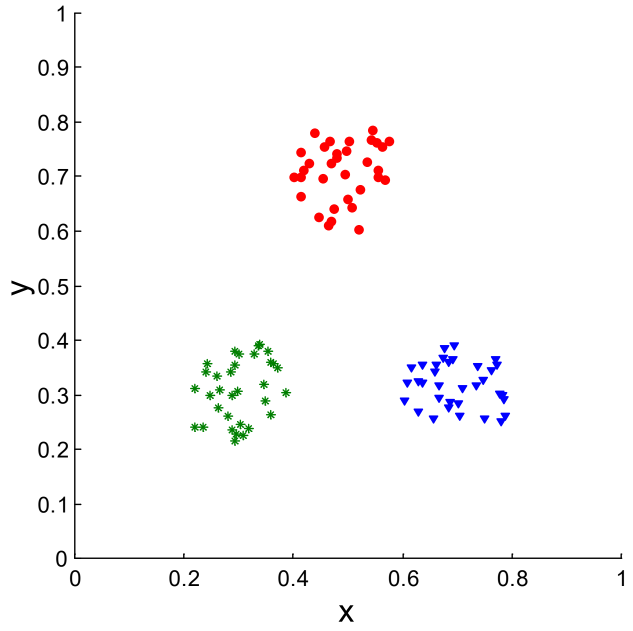
Corr =0.9235

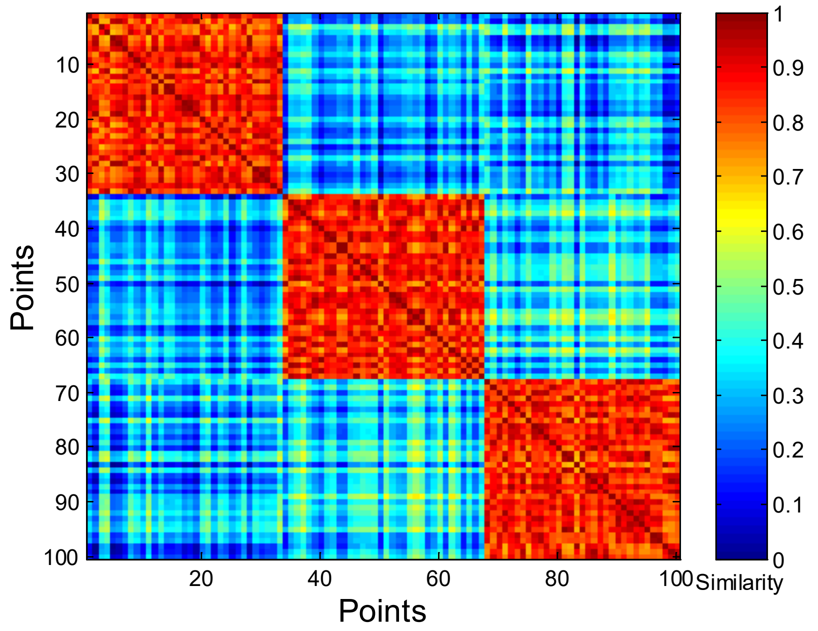


Corr = 0.5810

Internal Measure: Similarity Matrix

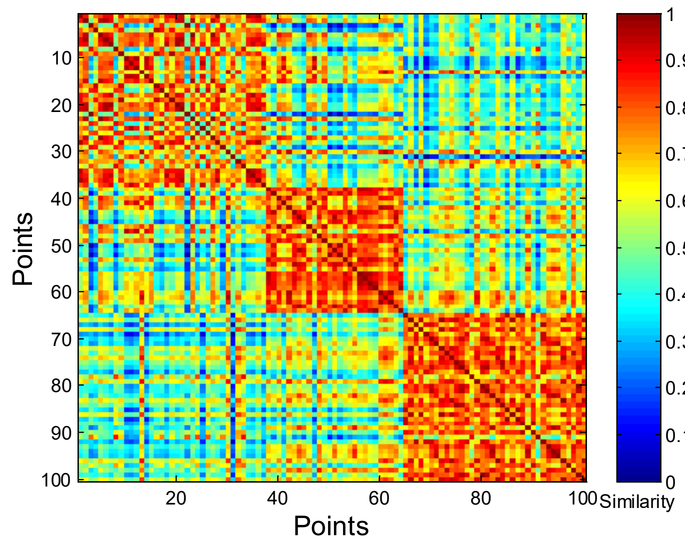
- Order the similarity matrix with respect to cluster labels and inspect visually

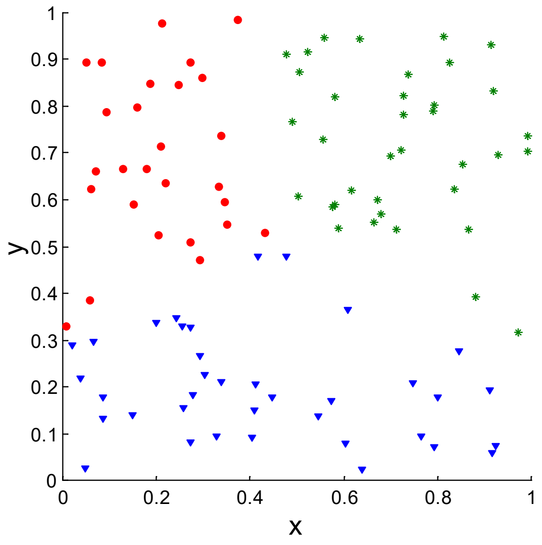




Internal Measure: Similarity Matrix

- Clusters in random data are not so crisp

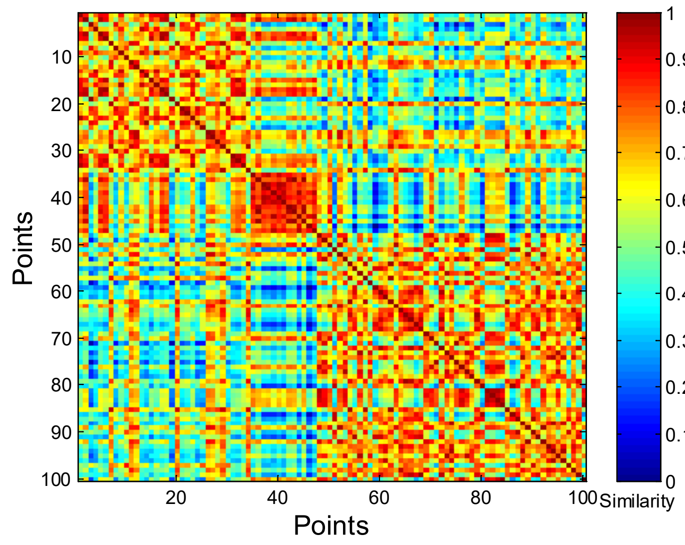


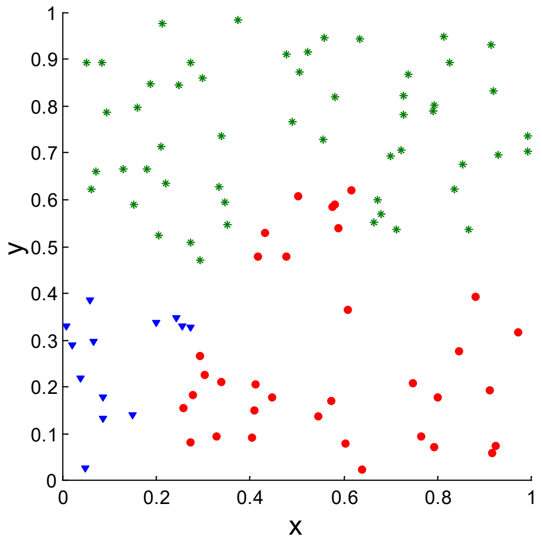


K-means

Internal Measure: Similarity Matrix

- Clusters in random data are not so crisp

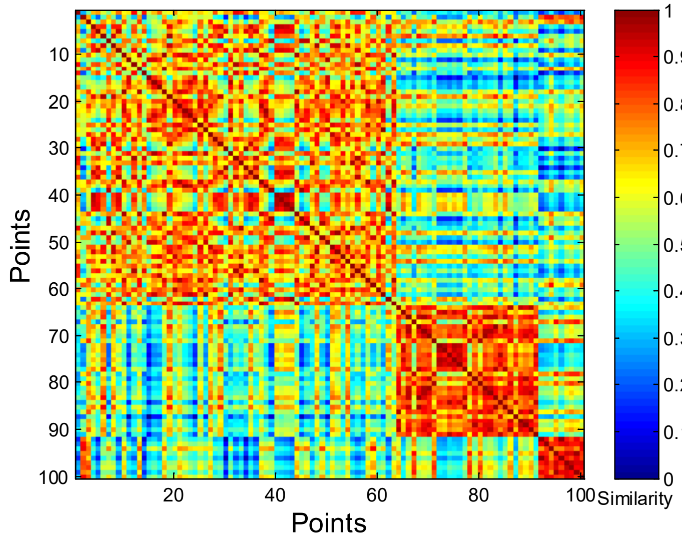


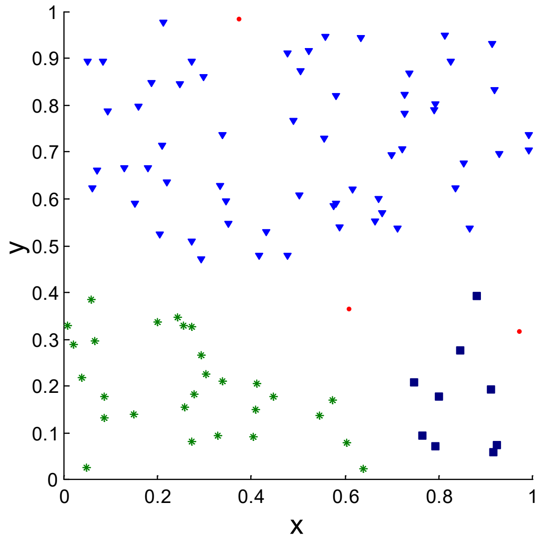


Complete Link

Internal Measure: Similarity Matrix

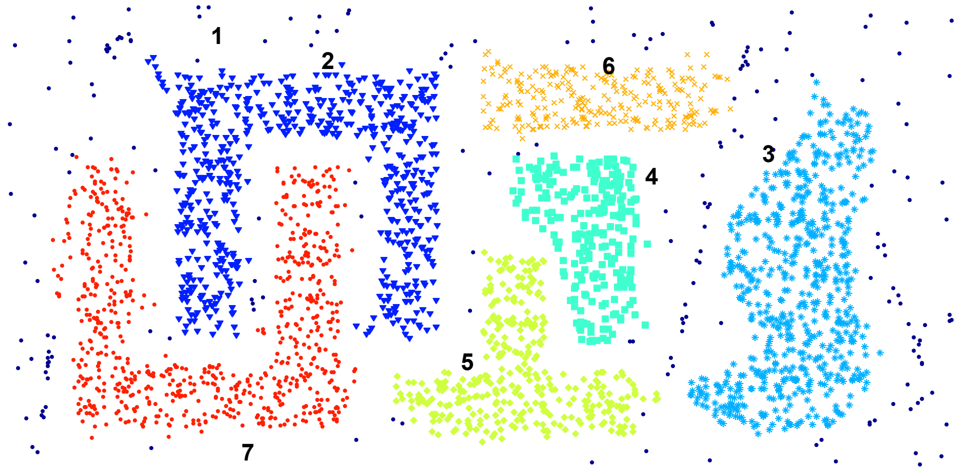
- Clusters in random data are not so crisp

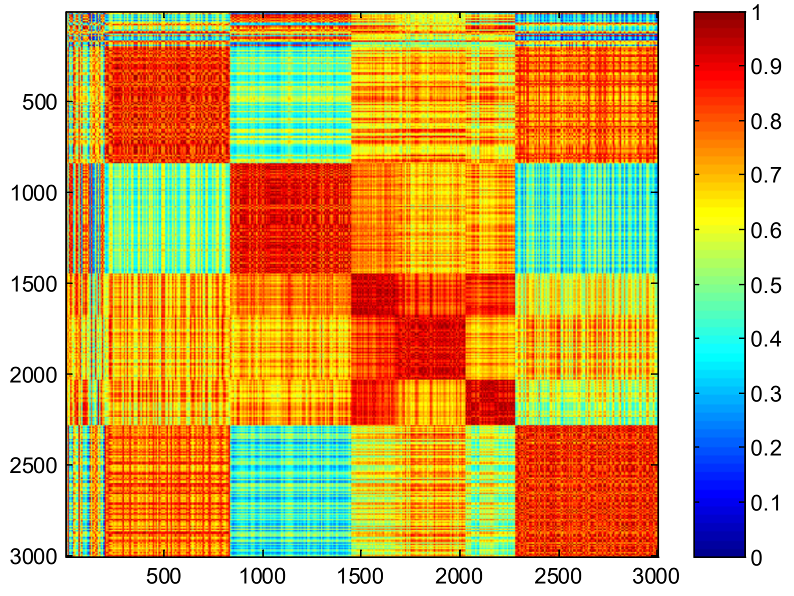




DBSCAN

Internal Measure: Similarity Matrix





DBSCAN

Supervised Cluster Evaluation 5.7

Supervised measures of cluster validity

- Two different kinds of approaches
  - **Classification oriented**: entropy, purity, and the F-measure
  - **Similarity oriented**: measure for binary data, Jaccard
- **Entropy**: The degree to which each cluster consists of objects of a single class

$$e = \sum_{i=1}^{k} \frac{m_i}{i} e_i \qquad e_i = -\sum_{j=1}^{L} p_{ij} \log_2(p_{ij}) \qquad \mathrm{p}_{ij} = m_{ij}/m_i$$

- **Purity**: Another measure of the extent to which a cluster contains objects of a single class

$$purity = \sum_{i=1}^{k} \frac{m_i}{i} p_i$$

Supervised measures of cluster validity

- **Precision(精度):** The fraction of a cluster that consists of objects of a specified class $precision(i,j) = p_{ij}$
- **Recall(召回率):** cluster *i* with respect to class *j* is

$$recall(i,j) = m_{ij}/m_i$$

where m_j is the number of objects in class *j*.

- **F-measure:** A combination of both **precision** and **recall** that measures the extent to which a cluster contains only objects of a particular class and all objects of that class

$$F(i,j) = \frac{2 \times precision(i,j) \times recall(i,j)}{precall(i,j) + recall(i,j)}$$

Supervised measures of cluster validity- Similarity oriented

- **Jaccard coefficient**:

$$Jaccard = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}$$

- **Rand statistic**:

$$Rand\ statistic = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}$$

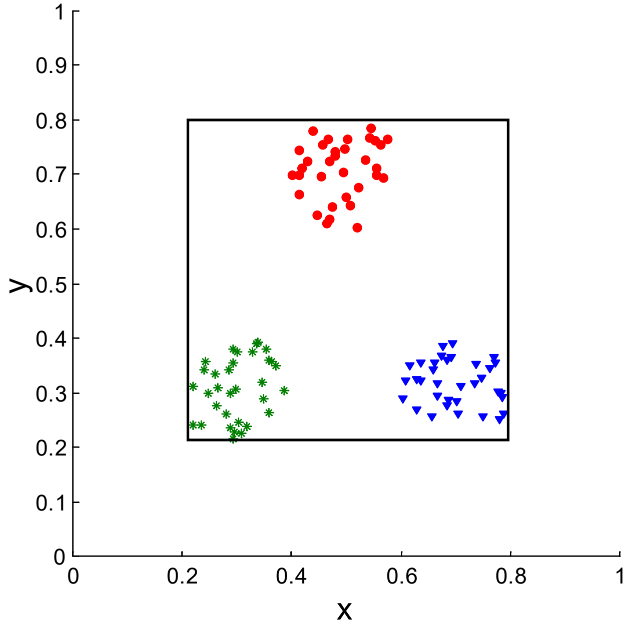
Significance of Cluster Validity Measures 5.8

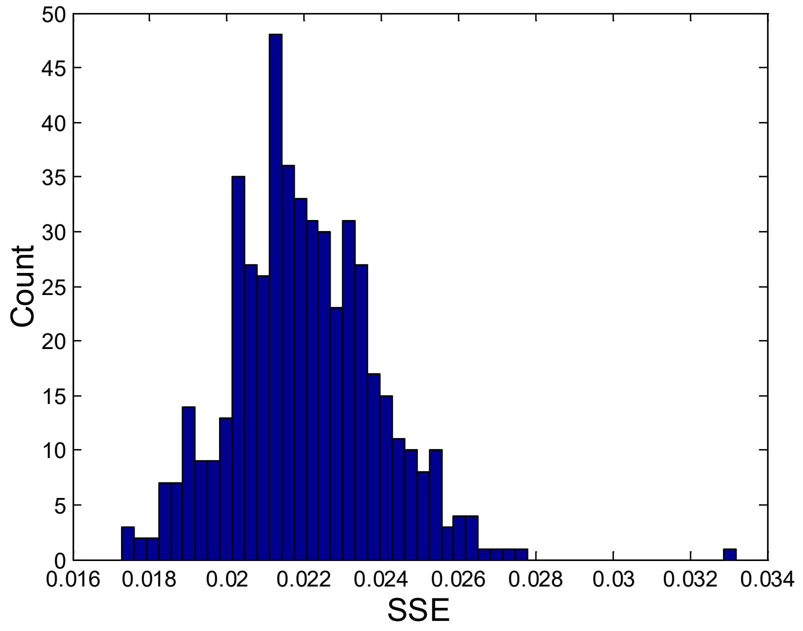
Statistics for Cluster Validity

- **Need a framework to interpret any measure**
  - For example, if our measure of evaluation has the value, 10, is that good, fair, or poor?
- **Statistics provide a framework for cluster validity**
  - The more "atypical" a clustering result is, the more likely it represents valid structure in the data
  - Can compare the values of an index that result from random data or clusterings to those of a clustering result.
  - These approaches are more complicated and harder to understand.
- **For comparing the results of two different sets of cluster analyses, a framework is less necessary**
  - However, there is the question of whether the difference between two index values is significant

Statistical Framework for SSE

- Example
  - Compare SSE of 0.005 against three clusters in random data
  - Histogram shows SSE of three clusters in 500 sets of random data points of size 100 distributed over the range 0.2 – 0.8 for x and y values

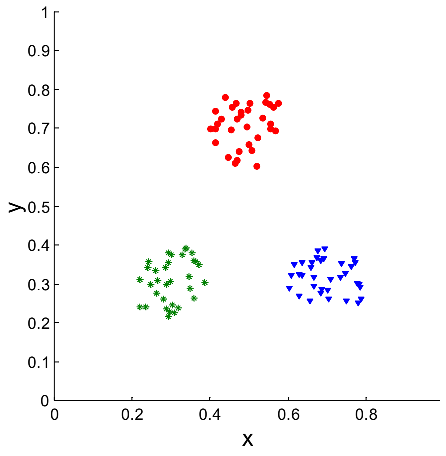




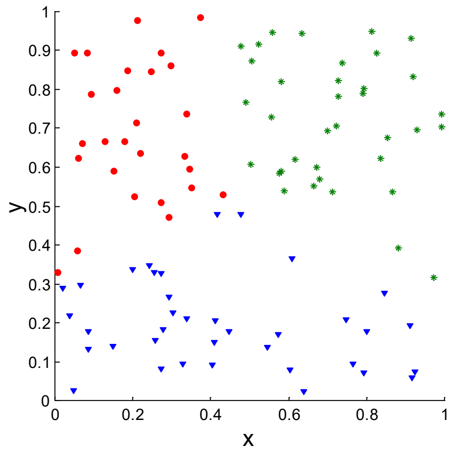
SSE from 500 random runs

Statistical Framework for Correlation

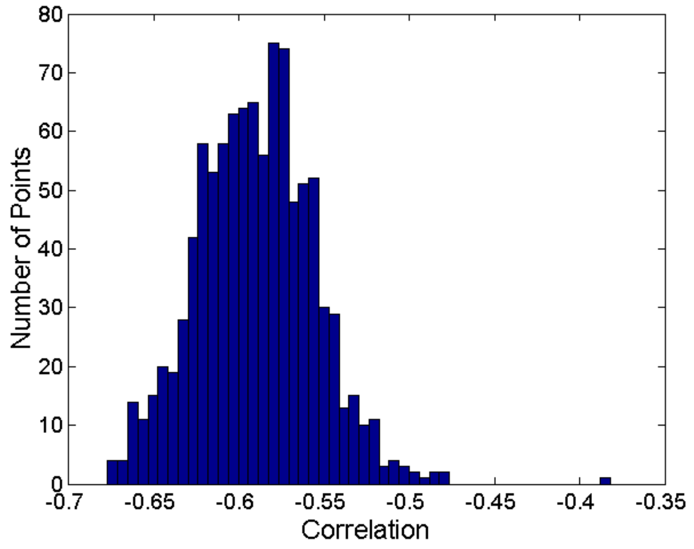
- Correlation of incidence and proximity matrices for the K-means clusterings of the following two data sets.



Corr =0.9235



Corr = 0.5810



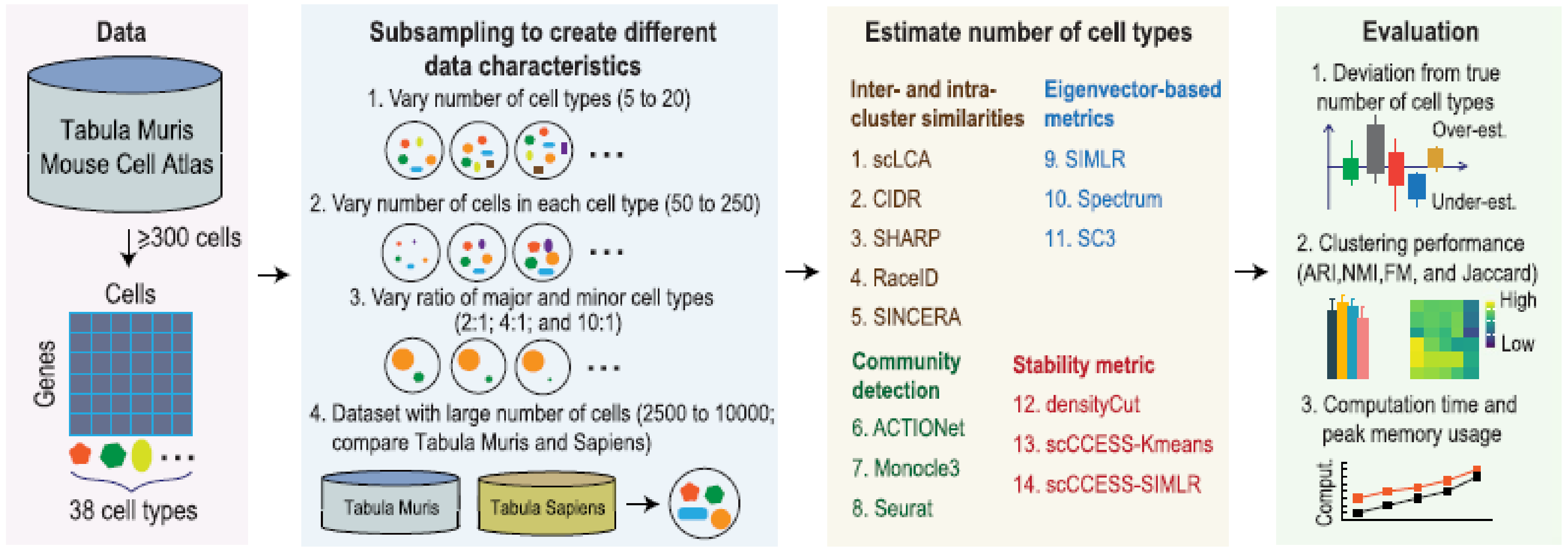
Problems and Challenges

- Considerable progress has been made in scalable clustering methods
  - **Partitioning: k-means, k-medoids**
  - **Hierarchical: Agglomerative**
  - **Density-based: DBSCAN**
- Current clustering techniques do not address all the requirements adequately

RESEARCH

Benchmarking clustering algorithms on estimating the number of cell types from single-cell RNA-sequencing data

Lijia Yu[1,2,3], Yue Cao[1,3], Jean Y. H. Yang[1,3] and Pengyi Yang[1,2,3*]


Data
Tabula Muris
Mouse Cell Atlas
≥300 cells
Cells
Genes
38 cell types
Subsampling to create different data characteristics
1. Vary number of cell types (5 to 20)
2. Vary number of cells in each cell type (50 to 250)
3. Vary ratio of major and minor cell types (2:1; 4:1; and 10:1)
4. Dataset with large number of cells (2500 to 10000; compare Tabula Muris and Sapiens)
Tabula Muris
Tabula Sapiens
Estimate number of cell types
Inter- and intra-cluster similarities
1. scLCA
2. CIDR
3. SHARP
4. RaceID
5. SINCERA
Eigenvector-based metrics
9. SIMLR
10. Spectrum
11. SC3
Community detection
6. ACTIONet
7. Monocle3
8. Seurat
Stability metric
12. densityCut
13. scCCESS-Kmeans
14. scCCESS-SIMLR
Evaluation
1. Deviation from true number of cell types
Over-est.
Under-est.
2. Clustering performance (ARI,NMI,FM, and Jaccard)
High
Low
3. Computation time and peak memory usage
Comput.

Summary

- **Cluster analysis** groups objects based on their **similarity** and has wide applications
- Measure of similarity can be computed for **various types of data**
- Clustering algorithms can be **categorized** into partitioning methods, hierarchical methods, and density-based methods,
- **Outlier detection** and analysis are very useful for fraud detection, etc. and can be performed by statistical, distance-based or deviation-based approaches
- There are still lots of research issues on cluster analysis, such as **constraint-based clustering**

- Assignment pp.347~352,
  - 5,6,7,11,14
  - 15,16,17
  - 22,24,25

References

- Jiawei Han and Micheline Kamber.数据挖掘：概念与技术. 机械工业出版社，2005. Chapter 8
- M Ester, HP Kriegel, J Sander, X Xu A density-based algorithm for discovering clusters in large spatial databases with noise Proc. KDD, 1996
- Rodriguez A, Laio A. Clustering by fast search and find of density peaks. Science, 2014, 344(6191): 1492-1496.

Thanks for your patience!

Questions and Comments?