

# Data Mining Cluster Analysis: Basic Concepts and Algorithms

## Lecture Notes for Chapter 7

Introduction to Data Mining  
by  
Tan, Steinbach, Kumar  
+  
Other sources

## What is clustering

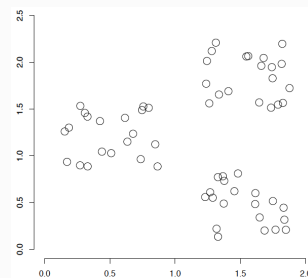
➤ **Clustering**: the process of grouping a set of objects into classes of similar objects

- ❖ Documents within a cluster should be similar.
- ❖ Documents from different clusters should be dissimilar.

➤ The commonest form of *unsupervised learning*

- ❖ Unsupervised learning = learning from raw data, as opposed to supervised data where a classification of examples/samples is given/known
- ❖ A common and important task that finds many applications in IR and other places

## A data set with clear cluster structure



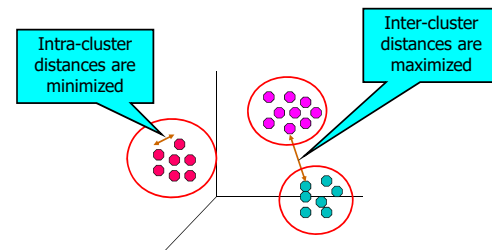
How would you design an algorithm for finding the three clusters in this case?

Heard of Gestalt Psychology?  
Has been applied to grouping in Object recognition!

Humans do it effortlessly

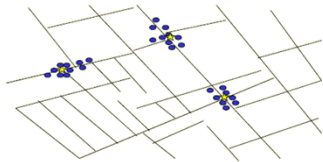
## Intra- and inter-cluster distance

- 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



## Historic application of clustering

- John Snow, a London physician plotted the location of cholera deaths on a map during an outbreak in the 1850s.
- The locations indicated that cases were clustered around certain intersections where there were polluted wells -- thus exposing both the problem and the solution.



From: Nina Mishra HP Labs

## Applications

### Location of new stores

- Pizza delivery locations
- Distribution centers (e.g., Amazon, ...)
- ATM machines
- Location of artilleries in combat

### Need to be careful about distance metric used

- If you end up picking a place on the other side of the river with only one bridge, it may not be a wise decision
- Placement of artillery. Hills and other obstacles are not taken care of in Euclidian distance!

6

## Applications of Cluster Analysis

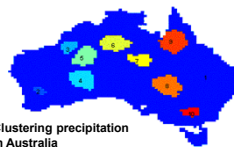
### Understanding

- Group related documents for browsing,
- group genes and proteins that have similar functionality, or
- group stocks with similar price fluctuations

### Summarization

- Reduce the size of large data sets

	Discovered Clusters	Industry Group
1	Applied-Mat-DOWN,IBM-Network-DOWN,3-COM-DOWN, Cabello-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Cams-DOWN,INTEL-DOWN,ISI-Logic-DOWN, Microm-Tech-DOWN,Texas-Instr-DOWN,Tellus-Instr-DOWN, Natl-Semiconductor-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN	Technology1-DOWN
2	Apple-Camp-DOWN,Autodesk-DOWN,DEC-DOWN, ADVC-Micro-Chassis-DOWN,Andrews-Corp-DOWN, Compu-Access-DOWN,Circuit-City-DOWN, Compug-DOWN,EMC-Corp-DOWN,Geo-Rail-DOWN, Motorola-DOWN,Microware-DOWN,Scientific-Air-DOWN	Technology2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loans-DOWN, MBSA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dynair-Tek-UP,Halliburton-RE-D-UP, Louisiana-Land-UP,Phillips-Petroleum-UP,Unocal-UP, Schlumberger-UP	Oil-UP



Clustering precipitation in Australia

## What is not Cluster Analysis?

### Supervised classification

- Have class label information

### Simple segmentation

- Dividing students into different registration groups alphabetically, by last name, zip code, age, ...

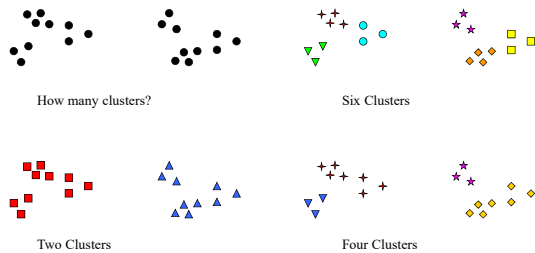
### Results of a query

- Groupings are a result of an external specification

### Graph partitioning

- Some mutual relevance and synergy, but areas are not identical

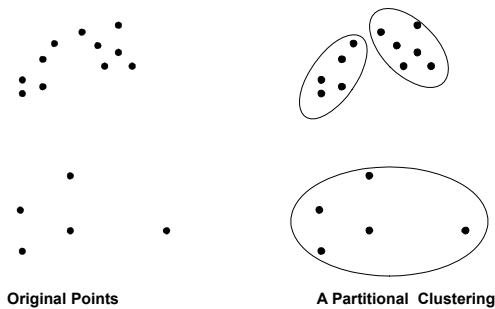
## Notion of a Cluster can be Ambiguous



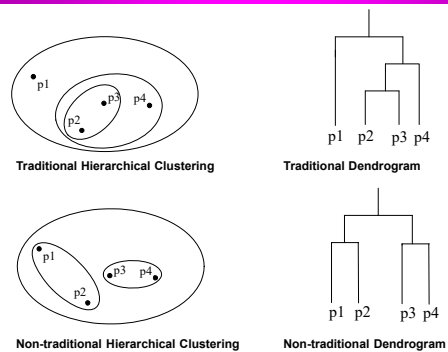
## Types of Clusterings

- A **clustering** is a set of clusters
- Important distinction between **hierarchical** and **partitional** sets of clusters
- **Partitional Clustering**
  - A division of 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



## Hierarchical Clustering



## Other Distinctions Between Sets of Clusters

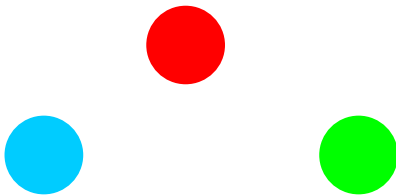
- Exclusive versus non-exclusive
  - In non-exclusive clusterings, points may belong to multiple clusters.
  - Can represent multiple classes or 'border' points
- Fuzzy versus non-fuzzy
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
  - Weights must sum to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

## Types of Clusters

- 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.



3 well-separated clusters

## 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**, the average of all the points in the cluster, or a **medoid**, the most "representative" point of a cluster
  - Medoid is an actual point where as centroid need not be.



4 center-based clusters

## Types of Clusters: Contiguity-Based

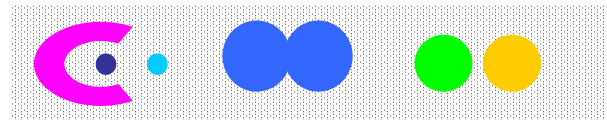
- Contiguous Cluster (Nearest neighbor or Transitive)
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.



8 contiguous clusters

## Types of Clusters: Density-Based

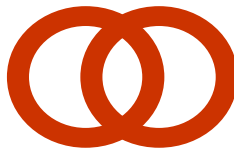
- Density-based
  - A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
  - Used when the clusters are irregular or intertwined, and when noise and outliers are present.



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 or maximize an objective function.
  - Enumerate all possible ways of dividing/assigning the points into clusters and evaluate the 'goodness' of each potential set of clusters by using the given objective function. (NP Hard)
  - Can have global or local objectives.
    - ♦ Hierarchical clustering 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

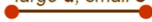
## Characteristics of the Input Data Are Important

- Type of proximity or density measure
  - This is a derived measure, but central to clustering
- Sparseness
  - Dictates type of similarity
  - Adds to efficiency
- Attribute type
  - Dictates type of similarity
- Type of Data
  - Dictates type of similarity
  - Other characteristics, e.g., autocorrelation
- Dimensionality
- Noise and Outliers
- Type of Distribution


## What do we need for clustering?

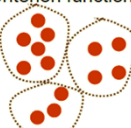
1. Proximity measure, *either*
  - similarity measure  $s(x_i, x_k)$ : large if  $x_i, x_k$  are similar
  - dissimilarity (or distance) measure  $d(x_i, x_k)$ : small if  $x_i, x_k$  are similar

large  $d$ , small  $s$




large  $s$ , small  $d$


2. Criterion function to evaluate a clustering
 





**good clustering**



**bad clustering**
3. Algorithm to compute clustering
  - For example, by optimizing the criterion function

## Distance (dissimilarity) measures

- Euclidean distance
 
$$d(x_i, x_j) = \sqrt{\sum_{k=1}^d (x_i^{(k)} - x_j^{(k)})^2}$$
  - translation invariant
- Manhattan (city block) distance
 
$$d(x_i, x_j) = \sum_{k=1}^d |x_i^{(k)} - x_j^{(k)}|$$
  - approximation to Euclidean distance, cheaper to compute
- They are special cases of **Minkowski distance**:
 
$$d_p(x_i, x_j) = \left( \sum_{k=1}^d |x_i^{(k)} - x_j^{(k)}|^p \right)^{\frac{1}{p}}$$

( $p$  is a positive integer)
- Cosine similarity

## Cluster evaluation (a hard problem)

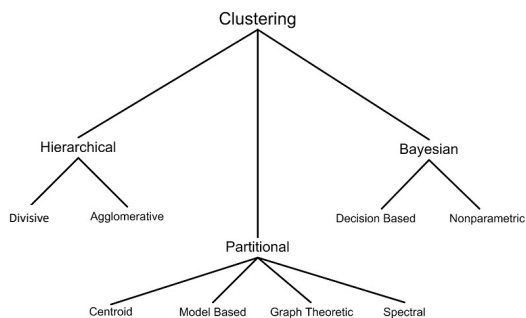
- **Intra-cluster cohesion** (compactness):
  - Cohesion measures how near the data points in a cluster are to the cluster centroid.
  - Sum of squared error (SSE) is a commonly used Measure (also called residual sum of squares (RSS) or sum of squared residuals (SSR))
- **Inter-cluster separation** (isolation):
  - Separation means that different cluster centroids should be far away from one another.
- In most applications, expert judgments are still the key

## How many clusters?

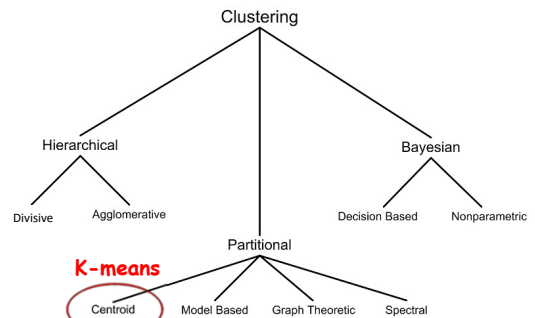


- Possible approaches
  1. fix the number of clusters to  $k$
  2. find the best clustering according to the criterion function (number of clusters may vary)

## Clustering techniques



## Clustering techniques



## Clustering techniques

- **Hierarchical** algorithms find successive clusters using previously established clusters. These algorithms can be either **agglomerative** ("bottom-up") or **divisive** ("top-down"):
  - **Agglomerative algorithms** begin with each element as a separate cluster and merge them into successively larger clusters;
  - **Divisive algorithms** begin with the whole set and proceed to divide it into successively smaller clusters.
- **Partitional** algorithms typically determine all clusters at once, but can also be used as divisive algorithms in the hierarchical clustering.
- **Bayesian** algorithms try to generate a *posteriori distribution* over the collection of all partitions of the data.

## Clustering Algorithms

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

## K-Means clustering

- K-means (MacQueen, 1967) is a **partitional clustering** algorithm
- Let the set of data points  $D$  be  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ , where  $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ir})$  is a **vector** in  $X \subseteq R^r$ , and  $r$  is the number of dimensions.
- The  $k$ -means algorithm partitions the given data into  $k$  clusters:
  - Each cluster has a cluster **center**, called **centroid**.
  - **$k$  has to be specified by the user**

## K-means algorithm

- Given  $k$ , the  $k$ -means algorithm works as follows:
  1. Choose  $k$  (random) data points (**seeds of clusters**) to be the initial **centroids**, cluster centers
  2. Assign each data point to the closest **centroid**
  3. Re-compute the **centroids** using the current cluster memberships
  4. If a convergence criterion is not met, repeat steps 2 and 3  
Steps 2 and 3 correspond to iteration



### K-means convergence (stopping) criterion

- no (or minimum) re-assignments of data points to different clusters, *or*
- no (or minimum) change of centroids, *or*
- minimum decrease in the **sum of squared error (SSE)**,

$$SSE = \sum_{j=1}^k \sum_{\mathbf{x} \in C_j} d(\mathbf{x}, \mathbf{m}_j)^2$$

- $C_j$  is the  $j$ th cluster
- $\mathbf{m}_j$  is the centroid of cluster  $C_j$  (the mean vector of all the data points in  $C_j$ ),
- $d(\mathbf{x}, \mathbf{m}_j)$  is the (Euclidean) distance between data point  $\mathbf{x}$  and centroid  $\mathbf{m}_j$ .
- There is also **Mean absolute error (MEA)** that uses absolute value.
  - Does not have good mathematical properties!

### K-means Clustering – Details

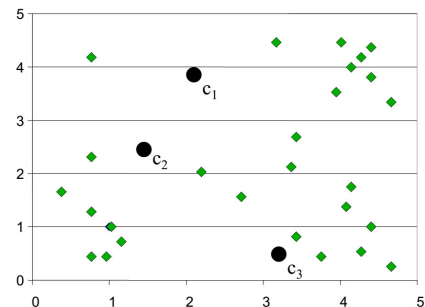
- Initial centroids are often chosen randomly.  $m_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$ 
  - Clusters produced vary from one run to another.
- The centroid is (typically) the mean of the points in the cluster.
- 'Closeness' is measured by Euclidean distance, cosine similarity, correlation, etc.
- K-means will converge for common similarity measures mentioned above.
- 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  $O(n * K * I * d)$  remember k and I are small!
  - $n$  = number of points,  $K$  = number of clusters,  $I$  = number of iterations,  $d$  = number of attributes (dimensions)

### K-means Clustering – Details

- Mean of cluster  $C_i$   $m_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$
- Mean of two vectors where  $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ir})$  and  $\mathbf{x}_j = (x_{j1}, x_{j2}, \dots, x_{jr})$  is  $M_{i,j}$
- $M_{i,j} = ((x_{i1} + x_{j1})/2, (x_{i2} + x_{j2})/2, \dots, (x_{ir} + x_{jr})/2)$
- Similarly, for more than 2 data points

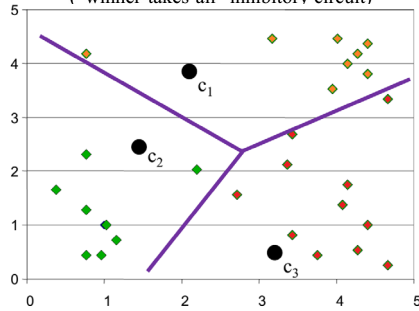
### K-means clustering example: step 1

Randomly initialize the cluster centers (synaptic weights)



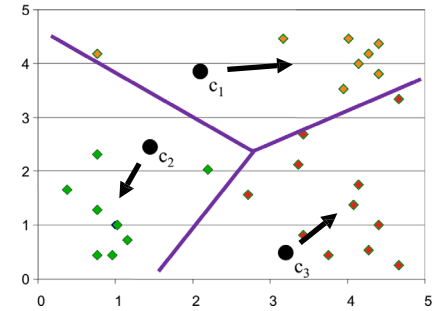
### K-means clustering example – step 2

Determine cluster membership for each input  
("winner-takes-all" inhibitory circuit)



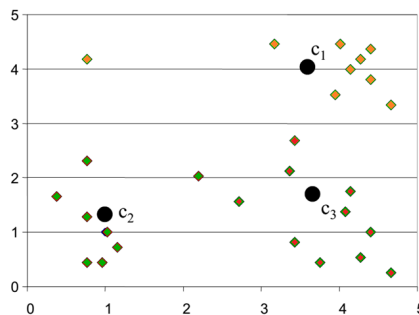
### K-means clustering example – step 3

Re-estimate cluster centers (adapt synaptic weights)



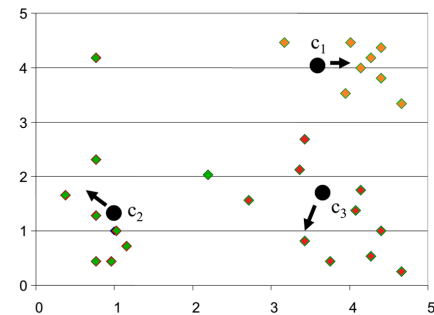
### K-means clustering example

Result of first iteration



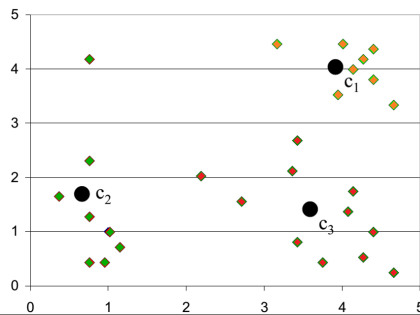
### K-means clustering example

Second iteration

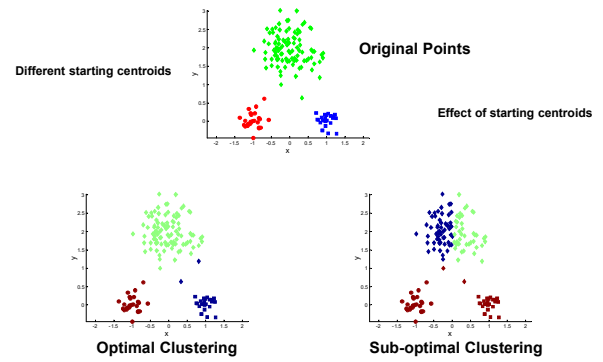


## K-means clustering example

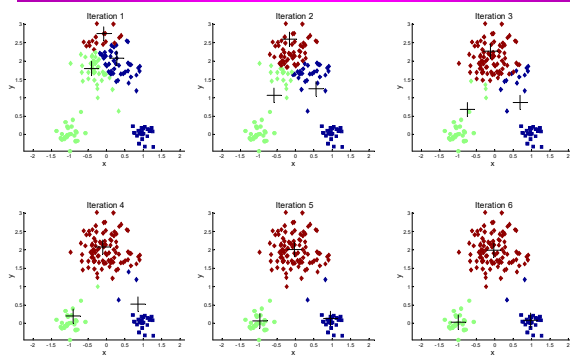
Result of second iteration



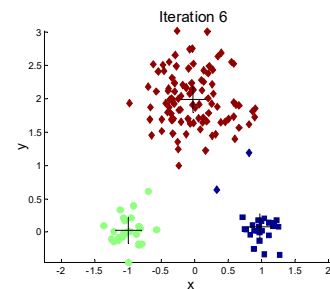
## Two different K-means Clusterings



## Importance of Choosing Initial Centroids



## Importance of Choosing Initial Centroids



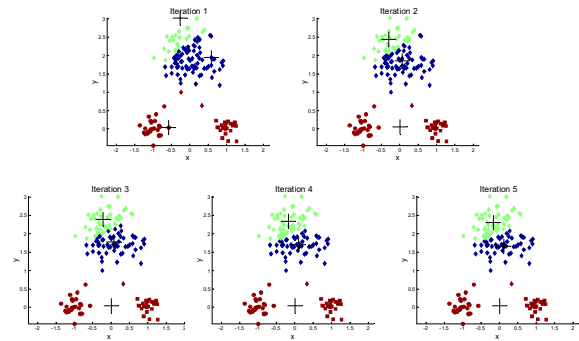
## Evaluating K-means Clusters

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

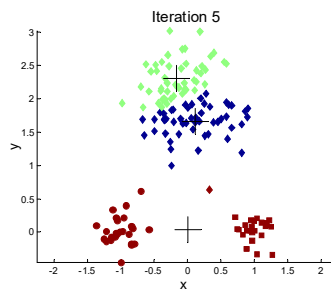
$$SSE = \sum_{i=1}^K \sum_{x \in C_i} dist^2(m_i, x)$$

- $x$  is a data point in cluster  $C_i$  and  $m_i$  is the representative point for cluster  $C_i$ 
  - ♦ can show that  $m_i$  corresponds to the center (mean) of the cluster
- Given two sets of clusters, we can choose the one with the smallest error
- One easy way to reduce SSE is to increase  $K$ , the number of clusters
  - ♦ A good clustering with smaller  $K$  can have a lower SSE than a poor clustering with higher  $K$

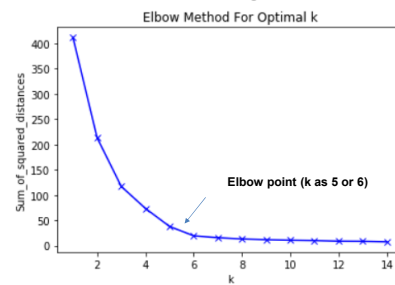
## Importance of Choosing Initial Centroids ...



## Importance of Choosing Initial Centroids ...



## Choosing K



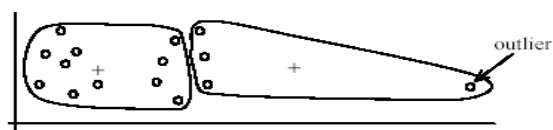
## Why use K-means?

- Strengths:
  - Simple: easy to understand and to implement
  - Efficient: Time complexity:  $O(tkn)$ , where  $n$  is the number of data points,  $k$  is the number of clusters, and,  $t$  is the number of iterations.
  - Since both  $k$  and  $t$  are small,  $k$ -means is considered a linear algorithm.
- K-means is the **most popular** clustering algorithm.
- Note: it terminates at a **local optimum** if SSE is used. The **global optimum** is hard to find due to complexity.

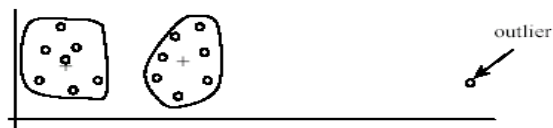
## Weaknesses of K-means

- The algorithm is only applicable if the **mean** is Defined (**why?**)
  - For categorical data,  $k$ -mode - the centroid is represented by most frequent values.
- The user needs to specify  $k$ .
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values.

## Outliers



(A): Undesirable clusters

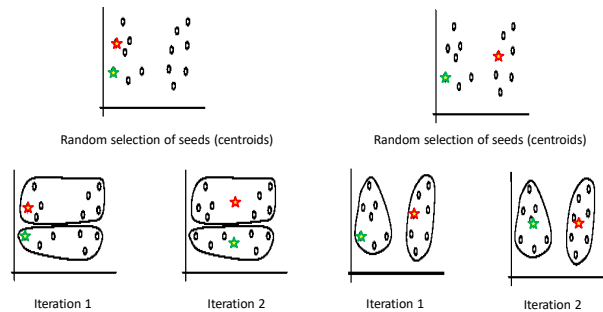


(B): Ideal clusters

## Dealing with outliers

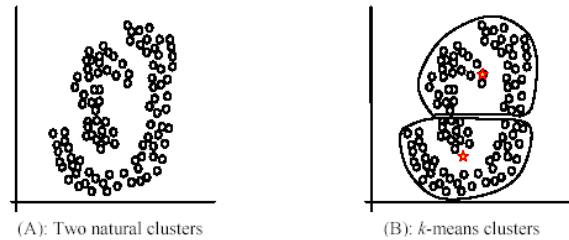
- Remove some data points that are much further away from the centroids than other data points
  - To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.
- Perform random sampling: by choosing a small subset of the data points, the chance of selecting an outlier is much smaller
  - Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

## Sensitivity to initial seeds



## Special data structures

- The  $k$ -means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).



## K-means summary

- Despite weaknesses,  $k$ -means is still the **most popular algorithm** due to its simplicity and efficiency
- No clear evidence that any other clustering algorithm performs better in general
- Comparing different clustering algorithms is a difficult task. **No one knows the correct clusters!**

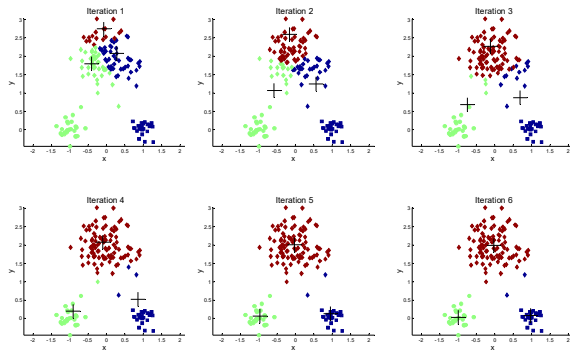
## Problems with Selecting Initial Points

- 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

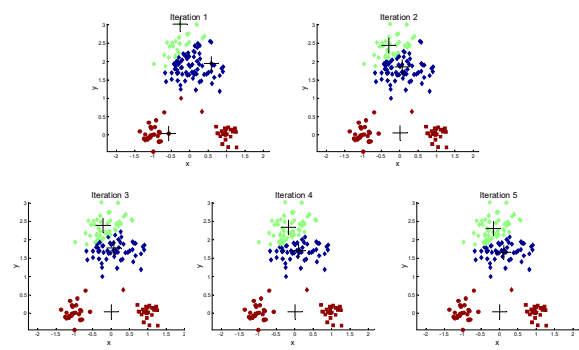
$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

- For example, if  $K = 10$ , then probability =  $10!/10^{10} = 0.00036$
- Sometimes the initial centroids will readjust themselves in 'right' way, and sometimes they don't
- Consider an example of five pairs of clusters

## Importance of Choosing Initial Centroids

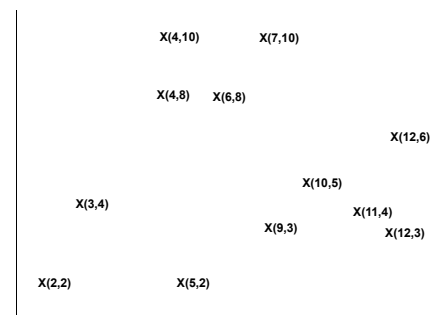


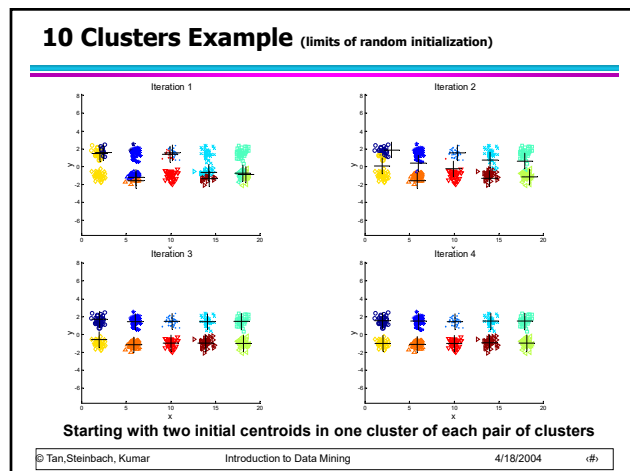
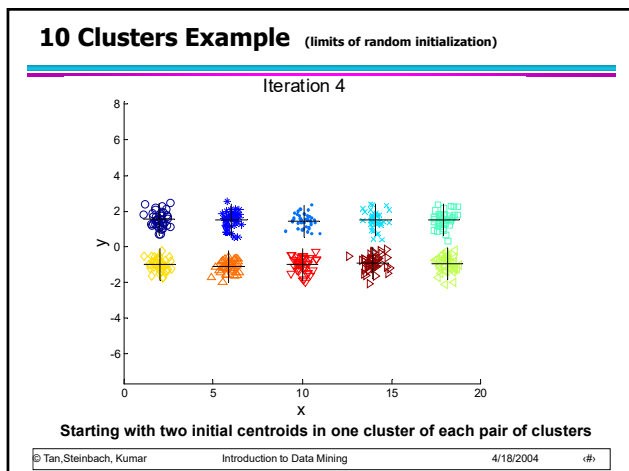
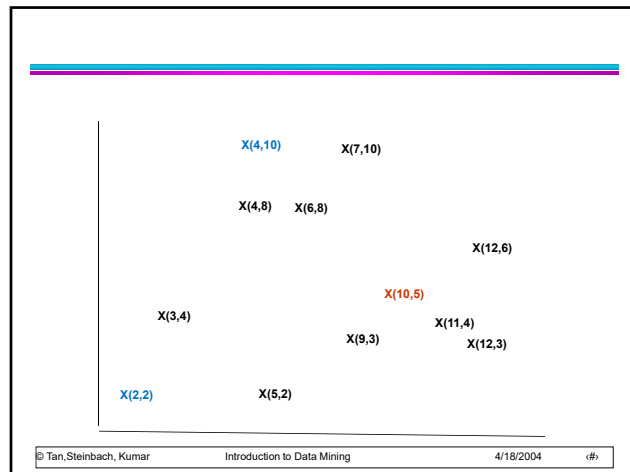
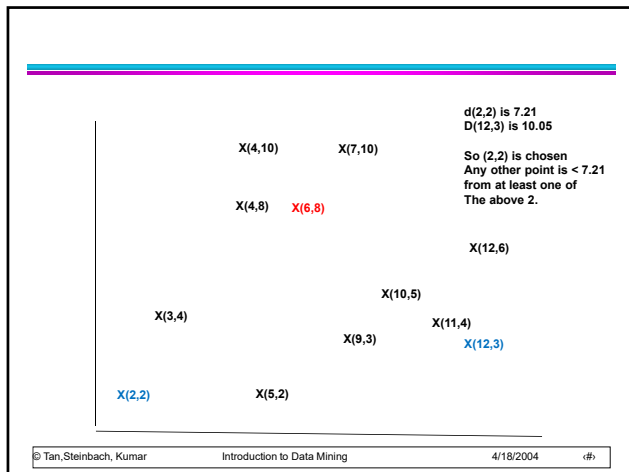
## Importance of Choosing Initial Centroids ...



## Selecting Initial Points

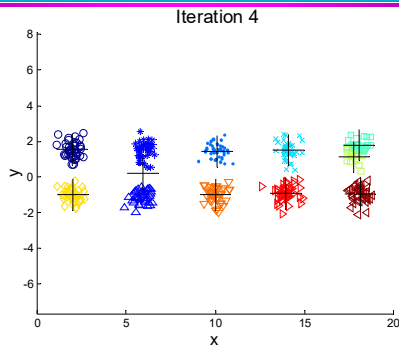
- Two approaches
  - Pick points that are as far away from one another as possible
  - Cluster a sample of the data, perhaps hierarchically, so there are  $k$  clusters. Pick a point from each cluster, perhaps the point closest to the centroid of the cluster (we will see this later)
- First approach:
  - Pick the first point at random;
  - While there are fewer than  $k$  points do
    - Add the point whose minimum distance from the selected points is as large as possible
  - end



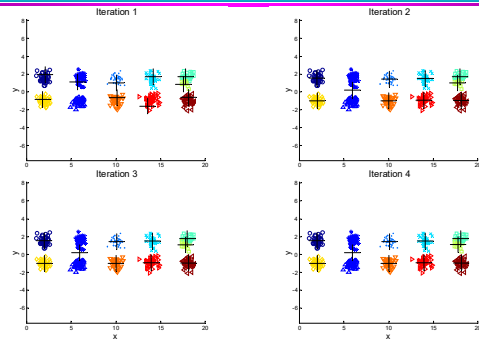




## 10 Clusters Example (limits of random initialization)



## 10 Clusters Example



## Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side
- Sample and use hierarchical clustering to determine initial centroids
  - Works under limited cases (small sample size, small k)
- Select more than k initial centroids and then select among these initial centroids
  - Select most widely separated
- Select a centroid for all; then other centroids farther from that! Can select outliers!
- Post processing – “fix up” the set of clusters produced!
- Bisecting K-means
  - Not as susceptible to initialization issues

## Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side
- Sample and use hierarchical clustering to determine initial centroids
  - Works under limited cases (small sample size, small k)
- Select more than k initial centroids and then select among these initial centroids
  - Select most widely separated
- Select a centroid for all; then other centroids farther from that! Can select outliers!
- Post processing – “fix up” the set of clusters produced!
- Bisecting K-means
  - Not as susceptible to initialization issues

## K-means++ algorithm

- K-means++ algorithm is guaranteed to find a k-means clustering and is shown to be optimal within a factor of  $O(\log(k))$
- Pick a centroid randomly
- Use distance square as probabilities for each point with respect to its closest centroid
- Pick the next centroid based on weighted probabilities
- If there are outliers, this is not a good approach
  - Remove outliers and use this

## K-means++ algorithm

- k-means++ initialization algorithm
- *Pick the 1<sup>st</sup> centroid randomly*
- *For  $i = 1$  to number of trials do*
  - *Compute the distance  $d(x)$ , of each point to its closest centroid*
  - *Assign each point a probability proportional to each point's  $d(x)^2$*
  - *Pick new centroid from the remaining points using the weighted probabilities*
- *End for*

## Handling Empty Clusters

- Basic K-means algorithm can yield empty clusters
  - If no points are allocated to a cluster during the initial step!
  - To get k clusters, a replacement strategy is needed
- Several strategies
  - Choose the point that contributes most to SSE
  - Choose a point from the cluster with the highest SSE
    - ♦ May split the cluster
  - If there are several empty clusters, the above can be repeated several times.

## Handling outliers

- SSE can be unduly influenced by the outliers
  - With outliers, typical centroids are not representative
- Approaches
  - Discover outliers and eliminate before hand
  - Also keeping in mind that in some domains outliers should not be eliminated!
    - ♦ High net worth or profitable individuals in financial domain
- Identification of outliers
  - Will be discussed in module 4
  - Outliers can be identified during post processing rather than eliminating before clustering

## 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
    - ◆ The point moves to a new cluster (2 updates) or stays in the same cluster (zero updates)
  - More expensive
  - Introduces an order dependency
  - Never get an empty cluster
  - Can use “weights” to change the impact

## Reducing SSE with post processing

- Increasing k is likely to reduce SSE
- Can SSE be improved without increasing k?
- “fixing up” resulting clusters
- Focus on individual clusters as SSE is a sum (total SSE and cluster SSE)
- Splitting and mergers of clusters and alternating between them
  - Split: cluster with largest SSE (or largest std for an attribute)
  - Introduce new cluster centroid. Typically, a point farthest from any cluster center is chosen by keeping track of SSE for each point
    - ◆ Randomly choose from all points in the largest SSE

## Reducing the number of clusters

- Decreasing the number of clusters while minimizing SSE
  - Disperse a cluster
    - ◆ Removing the centroid of a cluster and re-assigning all points
    - ◆ This should increase the total sse the least!
  - Merge two clusters
    - ◆ Clusters with the closest centroids are typically chosen
    - ◆ That result in the smallest increase in total SSE.
    - ◆ This strategy is used in hierarchical clustering as well.

## 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
    - ◆ ISODATA

## Bisecting K-means Approach

- A combination of K-means and hierarchical clustering
- Instead of partitioning data into k clusters in each iteration, bisecting k-means splits one cluster into two sub clusters at each bisecting step (using the original k-means) until k clusters are obtained!
- Note that running **Bisecting K-Means** with the same data does not always generate the same result because **Bisecting K-Means** initializes clusters randomly.
- The ITER specifies how many times the algorithm should repeat a split to keep the best split. If it is set to a high value it should provide better results but it would be more slow. Splits are evaluated using the Squared Sum of Errors (SSE).

77

Copyright ©2007-2017 The University of Texas at Arlington. All Rights Reserved.

## 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 consisting of all points
  2. **Repeat**
  3.   remove a cluster from the list of clusters
  4.   {perform several "trial" bisections of the chosen cluster}
  5.   **for**  $l = 1$  to **number of trials** **do**
  6.     Bisect the selected cluster using basic K-means
  7.   **end for**
  8.   Select the two clusters from the bisection with the **lowest total SSE**
  9.   add the two clusters to the list of clusters
  10. **until** the list of clusters contains K clusters

© Tan, Steinbach, Kumar

Introduction to Data Mining

4/18/2004

4/18

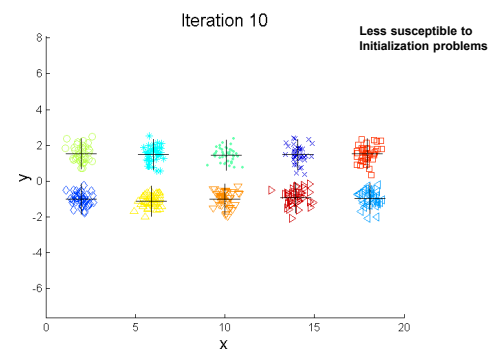
## Bisecting K-means Approach

- There are a number of ways to choose which cluster to split.
  - ❖ choose the largest cluster at each step
  - ❖ Choose the one with the largest SSE
  - ❖ Use a criterion based on both size and SSE
- Different choices result in different clusters
- Because we are using the K-means algorithm "locally" to bisect individual clusters, the final set does not represent a local minimum with respect to total SSE
  - ❖ This is partially true for each bisect but not overall!
- The clusters can be improved by using the cluster centroids as initial centroids for the standard K-means algorithm

79

Copyright ©2007-2017 The University of Texas at Arlington. All Rights Reserved.

## Bisecting K-means Example



© Tan, Steinbach, Kumar

Introduction to Data Mining

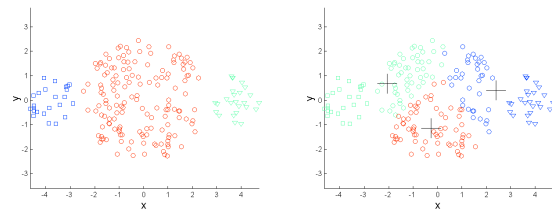
4/18/2004

4/18

## Limitations of K-means

- K-means is better at detecting “natural” clusters
  - Globular clusters (equal size and density)
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes
- K-means has problems when the data contains outliers.

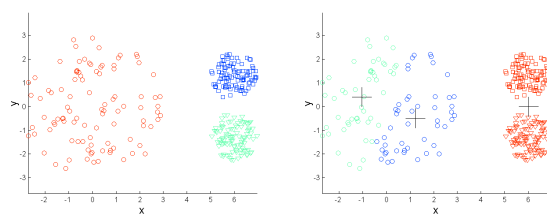
## Limitations of K-means: Differing Sizes



Original Points

K-means (3 Clusters)

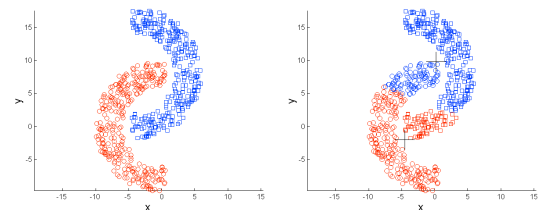
## Limitations of K-means: Differing Density



Original Points

K-means (3 Clusters)

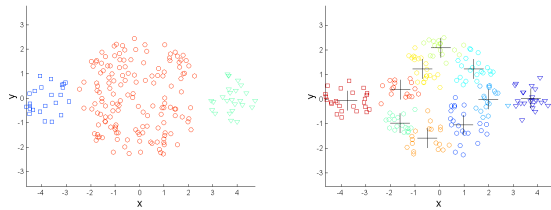
## Limitations of K-means: Non-globular Shapes



Original Points

K-means (2 Clusters)

## Overcoming K-means Limitations

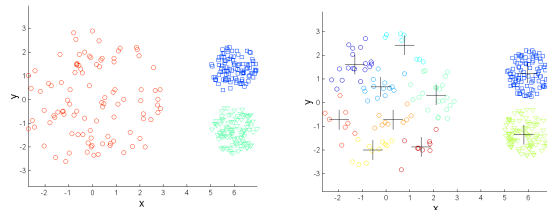


Original Points

K-means Clusters

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

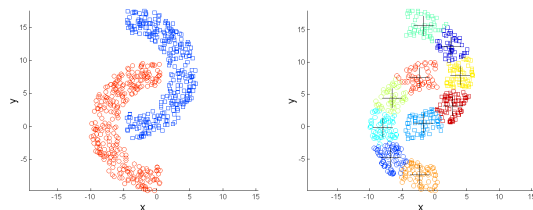
## Overcoming K-means Limitations



Original Points

K-means Clusters

## Overcoming K-means Limitations



Original Points

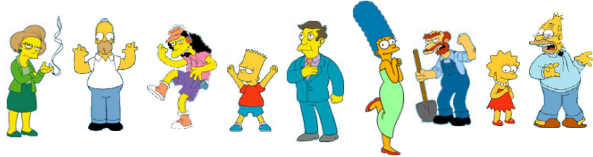
K-means Clusters

## K-means Summary

- K-means is better at detecting “natural” clusters
  - Globular clusters (equal size and density)
- K-means is efficient
- K-means has problems when the data contains outliers.
- K-means is NOT suitable for all types of data
  - Cannot handle non-globular clusters
  - Cannot handle clusters of different sizes
  - Cannot handle Irregular shapes

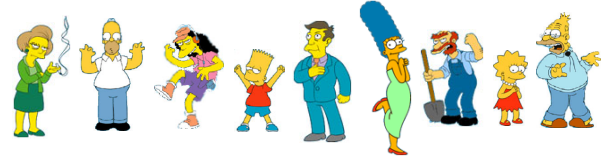
## Other ways of clustering

Slide from Eamonn Keogh



## What is a natural grouping of these objects?

Slide from Eamonn Keogh



Clustering is **subjective**



Simpson's Family



School Employees



Females



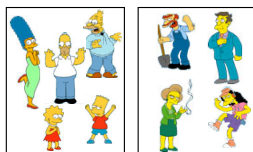
Males

## Two Types of Clustering

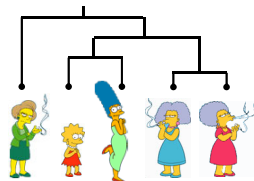
Slide based on one by Eamonn Keogh

- **Partitional algorithms:** Construct various partitions and then evaluate them by some criterion
- **Hierarchical algorithms:** Create a hierarchical decomposition of the set of objects using some criterion

### Partitional

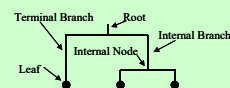


### Hierarchical

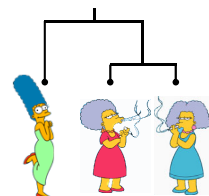
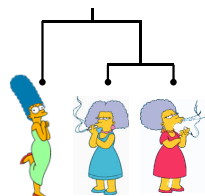


## Dendrogram: A Useful Tool for Summarizing Similarity Measurements

Slide based on one by Eamonn Keogh

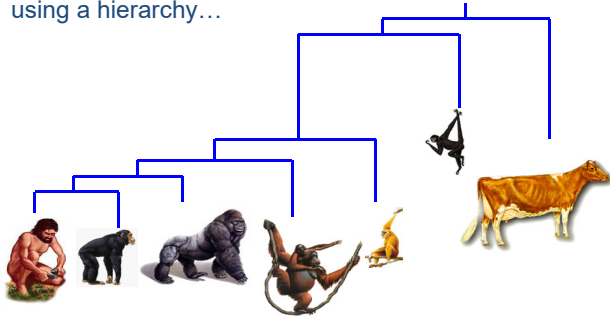


The similarity between two objects in a dendrogram is represented as the height of the lowest internal node they share.



Slide based on one by Eamonn Keogh

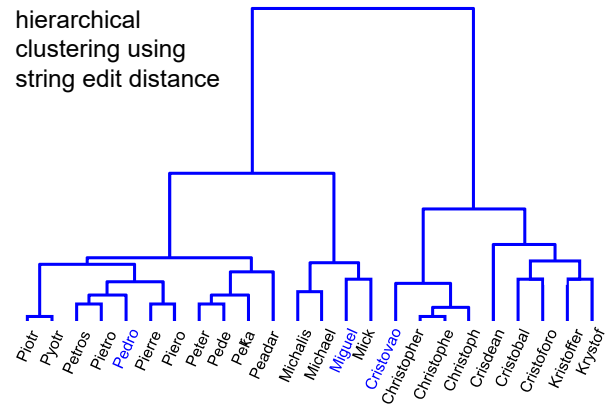
There is only one dataset that  
can be perfectly clustered  
using a hierarchy...



(Bovine:0.69395, (Spider Monkey 0.390, (Gibbon:0.36079,(Orang:0.33636,(Gorilla:0.17147,(Chimp:0.19268, Human:0.11927):0.08386):0.06124):0.15057):0.54939);

Slide based on one by Eamonn Keogh

A demonstration of  
hierarchical  
clustering using  
string edit distance

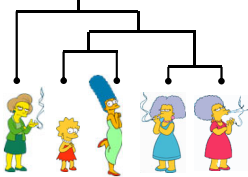


## Hierarchical Clustering

Slide based on one by Eamonn Keogh

The number of dendrograms with  $n$   
leaves =  $(2n-3)! / [(2^{n-2}) (n-2)!]$

Number of Leaves	Number of Possible Dendrograms
2	1
3	3
4	15
5	105
...	...
10	34,459,425



Since we cannot test all possible trees we will have to do a heuristic search of all possible trees. We could do this..

### Bottom-Up (agglomerative):

Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

**Top-Down (divisive):** Starting with all the data in a single cluster, consider every possible way to divide the cluster into two. Choose the best division and recursively operate on both sides.

Slide based on one by Eamonn Keogh

We begin with a distance matrix which contains the distances between every pair of objects in our database.

$$D(\text{Piotr}, \text{Pyotr}) = 8$$

$$D(\text{Piotr}, \text{Pyotr}) = 1$$

0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0



**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

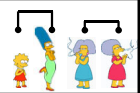
This slide and next 4 based on slides by Eamonn Keogh

Consider all possible merges... Choose the best



**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Consider all possible merges... Choose the best

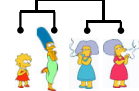


Consider all possible merges... Choose the best

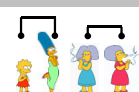


**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Consider all possible merges... Choose the best



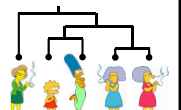
Consider all possible merges... Choose the best



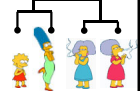
Consider all possible merges... Choose the best



**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



Consider all possible merges... Choose the best



Consider all possible merges... Choose the best

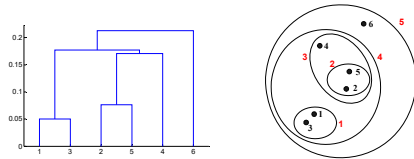


Consider all possible merges... Choose the best



## Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree like diagram that records the sequences of merges or splits



## 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 dendrogram at the proper level
- They may correspond to meaningful **taxonomies**
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, ...)

## Hierarchical Clustering

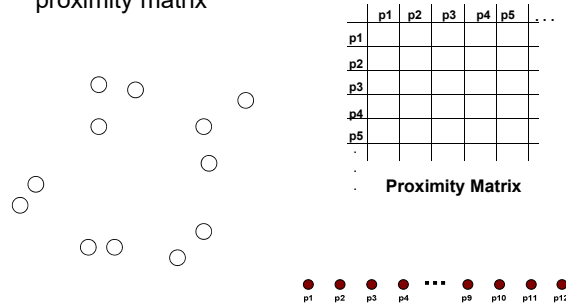
- Two main types of hierarchical clustering
  - Agglomerative:
    - ◆ Start with the points as individual clusters
    - ◆ At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - Divisive:
    - ◆ Start with one, all-inclusive cluster
    - ◆ At each step, split a cluster until each cluster contains a point (or there are k clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

## Agglomerative Clustering 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

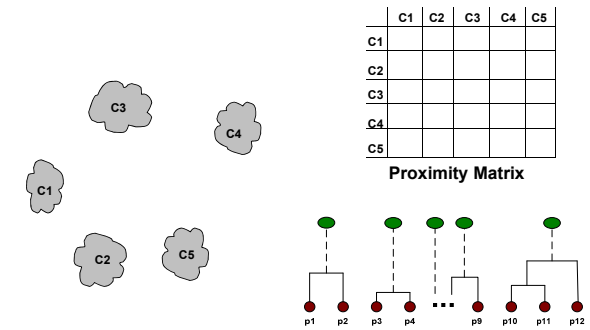
## Starting Situation

- Start with clusters of individual points and a proximity matrix



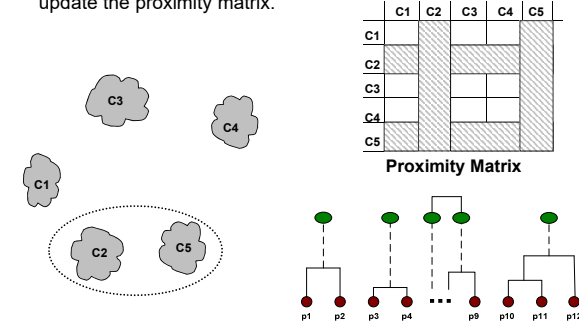
## Intermediate Situation

- After some merging steps, we have some clusters



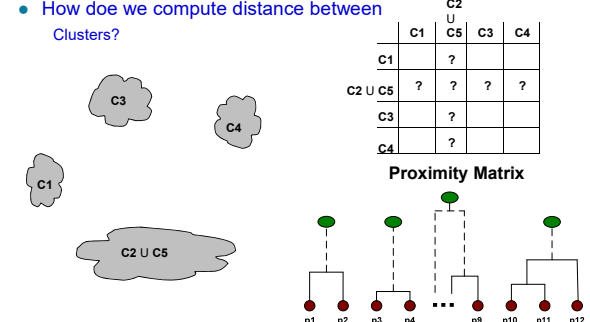
## Intermediate Situation

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

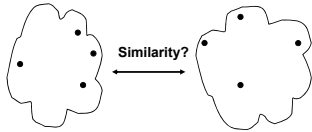


## After Merging

- The question is "How do we update the proximity matrix?"
- How do we compute distance between clusters?



## How to Define Inter-Cluster Similarity



- MIN (single linkage)
- MAX (complete linkage)
- Group Average linkage
- 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						
.						

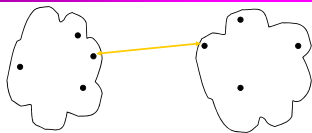
Proximity Matrix

Slide based on one by Eamonn Keogh

We know how to measure the distance between two objects, but **defining the distance between an object and a cluster, or defining the distance between two clusters is non obvious.**

- **MIN or Single linkage (nearest neighbor):** In this method the distance between two clusters is determined by the distance of the **two closest objects** (nearest neighbors) in the different clusters.
- **MAX or Complete linkage (furthest neighbor):** In this method, the distances between clusters are determined by the **greatest distance between any two objects in the different clusters** (i.e., by the "furthest neighbors").
- **Group average linkage:** In this method, the distance between two clusters is calculated as the **average distance between all pairs of objects** in the two different clusters.

## How to Define Inter-Cluster Similarity

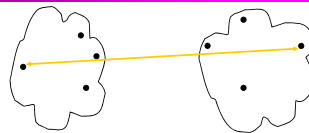


- **MIN**
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

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

Proximity Matrix

## How to Define Inter-Cluster Similarity

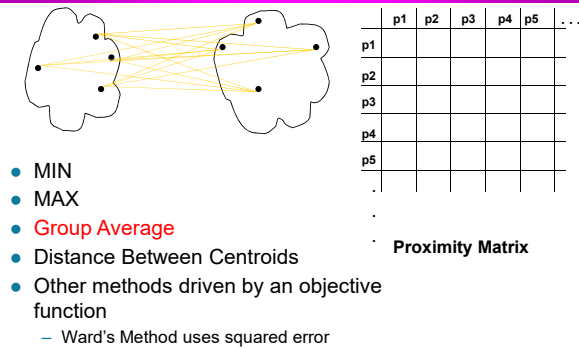


- MIN
- **MAX**
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

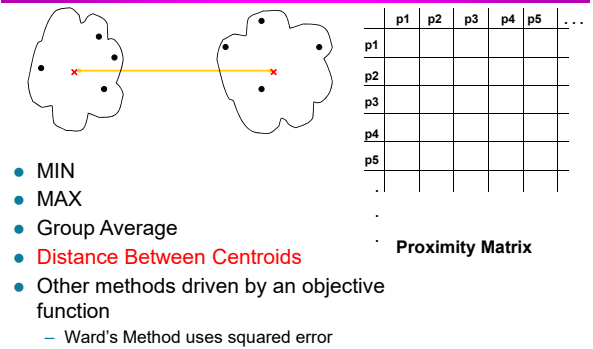
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						

Proximity Matrix

## How to Define Inter-Cluster Similarity



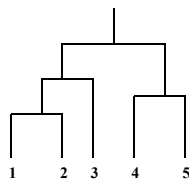
## How to Define Inter-Cluster Similarity



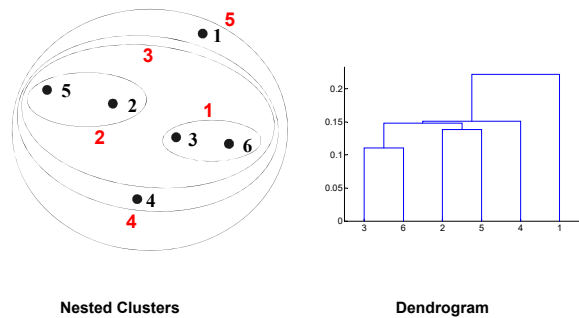
## Cluster Similarity: MIN or Single Link

- 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.

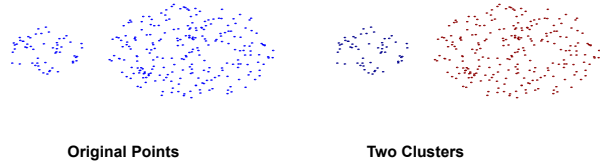
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



## Hierarchical Clustering: MIN

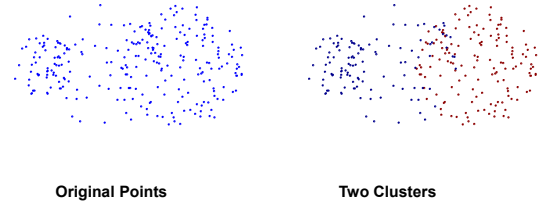


## Strength of MIN



- Can handle non-elliptical shapes

## Limitations of MIN

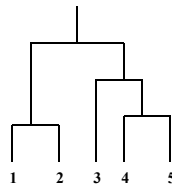


- Sensitive to noise and outliers

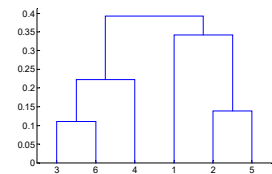
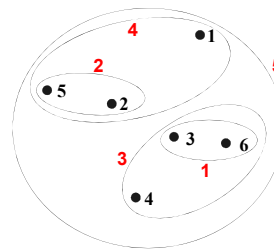
## Cluster Similarity: MAX or Complete Linkage

- 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

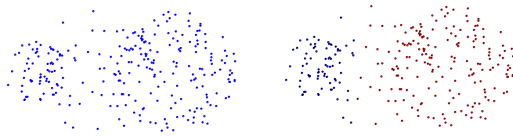
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



## Hierarchical Clustering: MAX



## Strength of MAX

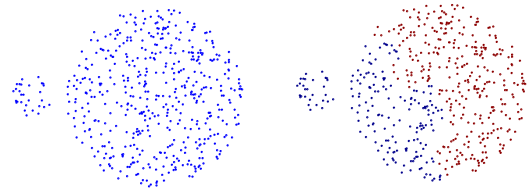


Original Points

Two Clusters

- Less susceptible to noise and outliers

## Limitations of MAX



Original Points

Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

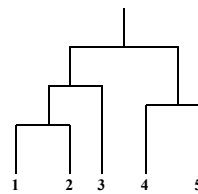
## Cluster Similarity: Group Average

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

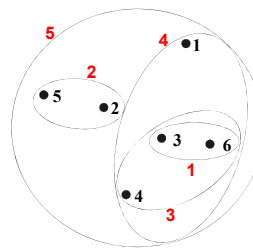
$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{p_i \in \text{Cluster}_i, p_j \in \text{Cluster}_j} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

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

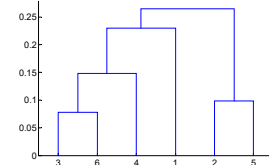
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



## Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram

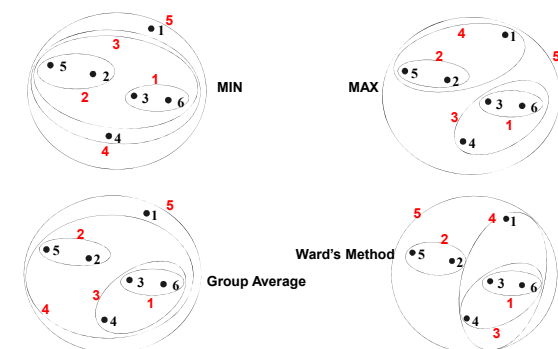
## Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise and outliers
- Limitations
  - Biased towards globular 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



## Hierarchical Clustering: Time and Space requirements

- $O(N^2)$  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 and 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

## Practical Issues of Classification

- Underfitting and Overfitting
- Missing Values
- Costs of Classification

## Overfitting



- Overfitting is a modeling error which occurs when a function is too closely fit to a limited set of data points.
- Intuitively, generalization or extrapolations that is NOT borne out by sample data!
  - ❖ For instance, a common problem is using computer [algorithms](#) to search extensive databases of historical market data in order to find patterns. Given enough study, it is often possible to develop elaborate theorems which appear to predict things such as returns in the [stock market](#) with close accuracy.
  - ❖ However, when applied to data outside of the sample, such theorems may likely prove to be merely the overfitting of a model to what were in reality just chance occurrences. In all cases, it is [important to test a model against data which is outside of the sample used to develop it](#).
- A [statistical fit](#) refers to how well you approximate the target function!

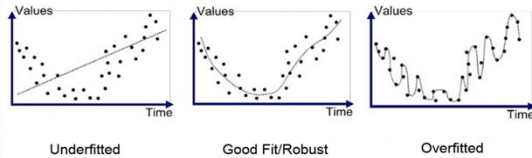
## Underfitting



- Underfitting refers to a model that can neither model the training data nor generalize to new data
- An underfit machine learning model is not a suitable model and will be obvious as it will have [poor performance on the training data](#).
- Underfitting is often not discussed as it is easy to detect given a good performance metric. The remedy is to move on and try alternate machine learning algorithms. Nevertheless, it does provide a good contrast to the problem of overfitting.
- Ideally, you want to select a model at the [sweet spot](#) between underfitting and overfitting.

## Summary

- **Overfitting:** Good performance on the training data, poor generalization to other data.
- **Underfitting:** Poor performance on the training data and poor generalization to other data

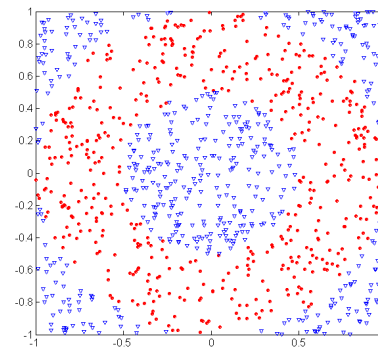


- Straightline is NOT representative of data; poor predictive capabilities
- Left one **intercepts** every data point! This model fits the data perfectly. Unless future data points follow the past perfectly, this model will have a poor predictive value!

133

Copyright ©2007-2017 The University of Texas at Arlington. All Rights Reserved.

## Underfitting and Overfitting (Example)



500 circular and 500 triangular data points.

Circular points:  
 $0.5 \leq \sqrt{x_1^2 + x_2^2} \leq 1$

Triangular points:  
 $\sqrt{x_1^2 + x_2^2} > 0.5$  or  
 $\sqrt{x_1^2 + x_2^2} < 1$

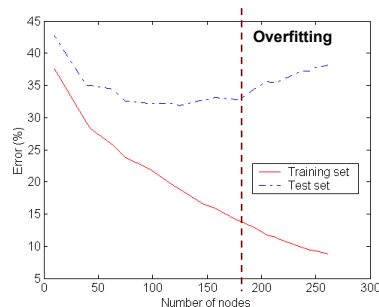
© Tan,Steinbach, Kumar

Introduction to Data Mining

4/18/2004

©#

## Underfitting and Overfitting



**Underfitting:** when model is too simple, both training and test errors are large

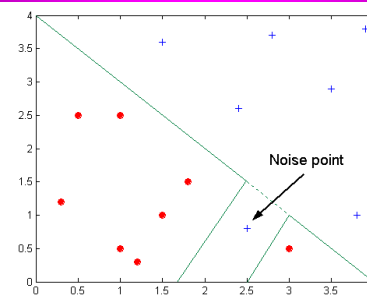
© Tan,Steinbach, Kumar

Introduction to Data Mining

4/18/2004

©#

## Overfitting due to Noise



**Decision boundary is distorted by noise point**

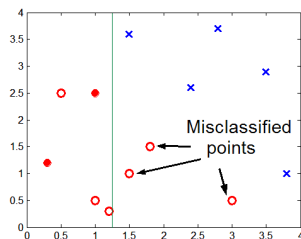
© Tan,Steinbach, Kumar

Introduction to Data Mining

4/18/2004

©#

## Overfitting due to Insufficient Examples



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

## Overfitting and underfitting

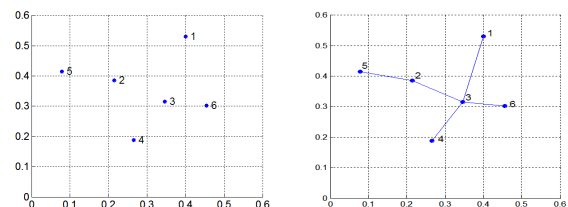
- What is overfitting in decision trees?
  - Trees are **more complex** than necessary
  - More than necessary breadth and depth!
- What is underfitting in decision trees?
  - Trees are less complex than necessary
  - Trees shallow and fewer splits!
- What is overfitting in k-means?
  - Corresponds to choosing larger k than necessary!
- What is underfitting in K-means?
  - Corresponds to a smaller k than necessary!

## Occam's Razor

- Given two models of **similar generalization errors**, one should prefer the simpler model over the more complex model
- For complex models, there is a greater chance that it was fitted accidentally by errors in data
- Therefore, one should include model complexity when evaluating a model

## 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



## MST: Divisive Hierarchical Clustering

- Use MST for constructing hierarchy of clusters

---

### Algorithm 7.5 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
- 

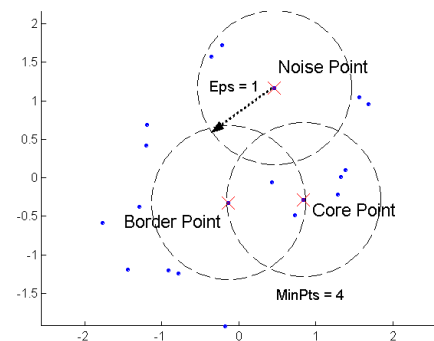
## Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as densely-connected points
- Major features:
  - Discover clusters of **arbitrary shape**
  - Handles noise
  - **One scan** (why is this important?)
  - 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) (more grid-based)

## DBSCAN -- a density-based algorithm

- Two parameters:
  - **Eps**: Specifies radius of the neighborhood
  - **MinPts**: Minimum number of points in an Eps-neighborhood of that point
- Density = number of points within Eps radius
- 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 neither a core point nor a border point.

## DBSCAN: Core, Border, and Noise Points



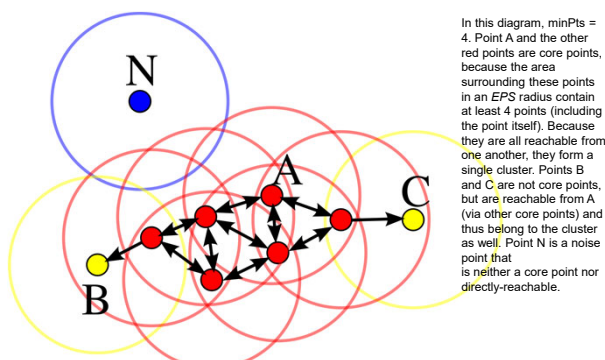


- In k-means clustering, each cluster is represented by a **centroid**, and points are assigned to whichever centroid they are closest to. In DBSCAN, there are **no centroids**, and clusters are formed by **linking nearby points** to one another.
- k-means requires specifying the number of clusters, 'k'. DBSCAN does not, but does require specifying two parameters which influence the decision of whether two nearby points should be linked into the same cluster. These two parameters are a distance threshold, **Eps (epsilon)**, and "**MinPts**" (minimum number of points).
- k-means **runs over many iterations** to converge on a good set of clusters, and cluster assignments can change on each iteration. DBSCAN makes **only a single pass** through the data, and once a point has been assigned to a particular cluster, it never changes.

## DBSCAN Algorithm (Center-based)

- Label all points as core 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**
- Understand this algorithm clearly!

## Example

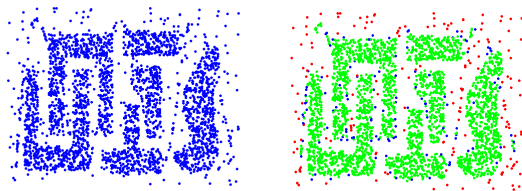


## DBSCAN algorithm (tree-based view)



1. Start with an **arbitrary seed point** which has at least  $MinPts$  in  $Eps$ 
    - a. Do a breadth-first search along **each of these nearby points**
    - b. If it has fewer than  $MinPts$  neighbors, this becomes a leaf and we do not grow this further
    - c. Add all points that have  $MinPts$  to a FIFO queue (directly-reachable from a core point)
  2. Continue this until the queue is empty.
  3. All points used in this BFS become a cluster (including the leaf)
  4. Continue this process with a **new seed point** not part of another cluster
  5. Until all points are assigned
- If a point has fewer than  $MinPts$  and it is not a leaf node, then it is labeled as noise!

## DBSCAN: Core, Border and Noise Points

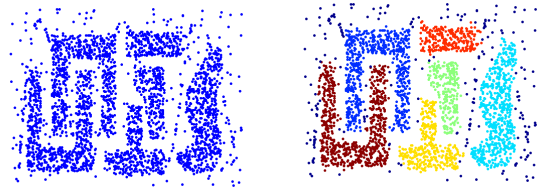


Original Points

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

**Eps = 10, MinPts = 4 (how are these determined?)**

## When DBSCAN Works Well

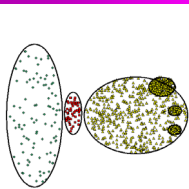


Original Points

Clusters

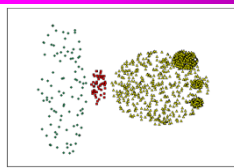
- Resistant to Noise
- Can handle clusters of different shapes and sizes

## When DBSCAN Does NOT Work Well

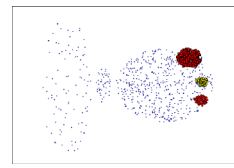


Original Points

- Varying densities
- High-dimensional data



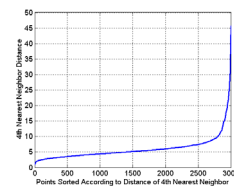
(MinPts=4, Eps=9.75).



(MinPts=4, Eps=9.92)

## DBSCAN: Determining EPS and MinPts

- Idea is that for points in a cluster, their  $k^{\text{th}}$  nearest neighbors are at roughly the same distance
- Noise points have the  $k^{\text{th}}$  nearest neighbor at farther distance
- So, plot sorted distance of every point to its  $k^{\text{th}}$  nearest neighbor

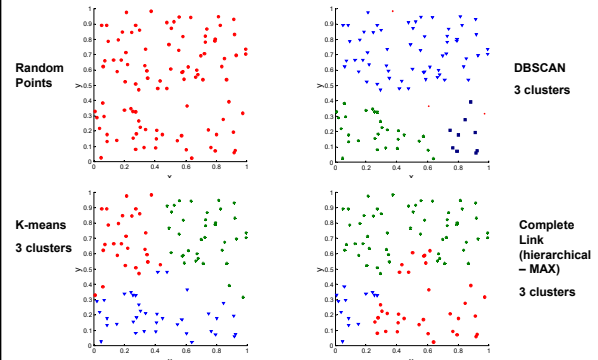


Using this MinPts as 4  
And Eps as 10 is chosen!

## Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall
- For cluster analysis, the analogous question is how to evaluate the “goodness” of the resulting clusters?
- But “clusters are in the eye of the beholder”!
- Then 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

## Clusters found in Random Data



## Different Aspects of Cluster Validation

1. Determining the **clustering tendency** of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.
3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
  - Use only the data
4. Comparing the results of two different sets of cluster analyses to determine which is better.
5. Determining the 'correct' number of clusters.

For 2, 3, and 4, we can further distinguish whether we want to evaluate the entire clustering or just individual clusters.

## Measures of Cluster Validity

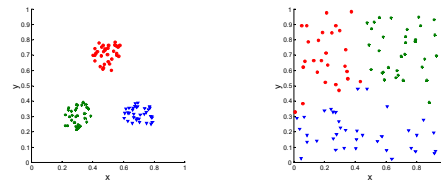
- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.
  - **External Index:** Used to measure the extent to which cluster labels match externally supplied class labels.
    - ♦ Entropy
  - **Internal Index:** Used to measure the goodness of a clustering structure *without* respect to external information.
    - ♦ Sum of Squared Error (SSE)
  - **Relative Index:** Used 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.

## Measuring Cluster Validity Via Correlation

- Two matrices
  - Proximity Matrix (or similarity matrix)
  - "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
  - Since the matrices are symmetric, only the correlation between  $n(n-1)/2$  entries needs to be calculated.
- 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.

## Measuring Cluster Validity Via Correlation

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

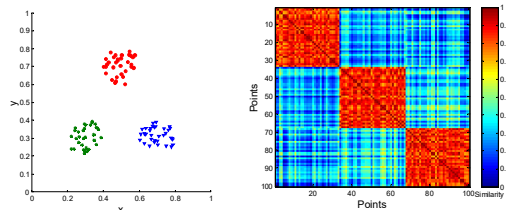


Corr = -0.9235

Corr = -0.5810

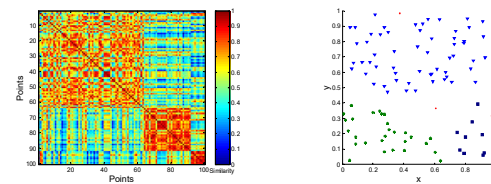
## Using Similarity Matrix for Cluster Validation

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



## Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp

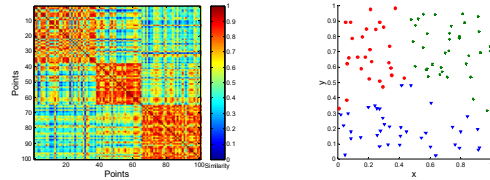


DBSCAN



## Using Similarity Matrix for Cluster Validation

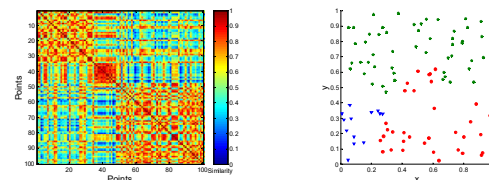
- Clusters in random data are not so crisp



K-means

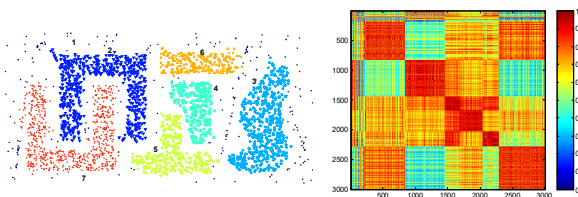
## Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp



Complete Link

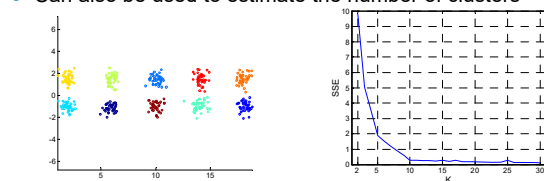
## Using Similarity Matrix for Cluster Validation



DBSCAN

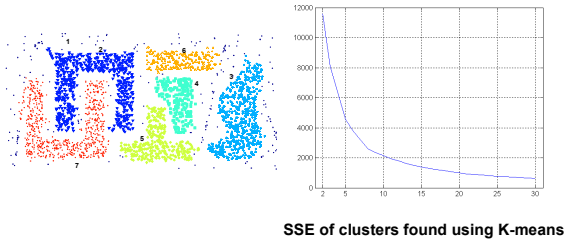
## Internal Measures: SSE

- 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 is good for comparing two clusterings or two clusters (average SSE).
- Can also be used to estimate the number of clusters



## Internal Measures: SSE

- SSE curve for a more complicated data set

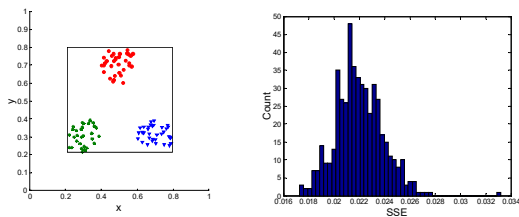


## Framework for Cluster Validity

- 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.
    - If the value of the index is unlikely, then the cluster results are valid
  - 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

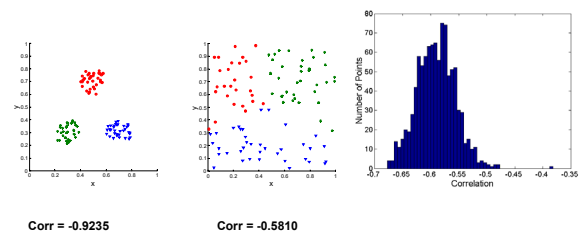
## Statistical Framework for SSE

- 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



## Statistical Framework for Correlation

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



### Internal Measures: Cohesion and Separation

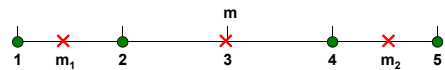
- **Cluster Cohesion:** Measures how closely related are objects in a cluster
  - Example: SSE
- **Cluster Separation:** Measure how distinct or well-separated a cluster is from other clusters
- Example: Squared Error
  - Cohesion is measured by the within cluster sum of squares (SSE)
 
$$WSS = \sum_i \sum_{x \in C_i} (x - m_i)^2$$
  - Separation is measured by the between cluster sum of squares
 
$$BSS = \sum_i |C_i| (m - m_i)^2$$

$m$  is the centroid for the whole cluster

– Where  $|C_i|$  is the size of cluster  $i$

### Internal Measures: Cohesion and Separation

- Example: SSE
  - $BSS + WSS = \text{constant}$



**K=1 cluster:**

$$WSS = (1-3)^2 + (2-3)^2 + (4-3)^2 + (5-3)^2 = 10$$

$$BSS = 4 \times (3-3)^2 = 0$$

$$Total = 10 + 0 = 10$$

**K=2 clusters:**

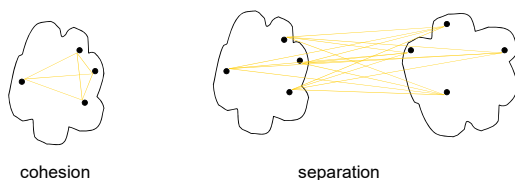
$$WSS = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$

$$BSS = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$

$$Total = 1 + 9 = 10$$

### Internal Measures: Cohesion and Separation

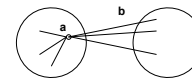
- A proximity graph based approach can also be used for cohesion and separation.
  - Cluster cohesion is the sum of the weight of all links within a cluster.
  - Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.



### Internal Measures: Silhouette Coefficient

- Silhouette Coefficient combine ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- 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 is then given by
 
$$s = 1 - a/b \quad \text{if } a < b, \quad (\text{or } s = b/a - 1 \quad \text{if } a \geq b, \text{ not the usual case})$$

- Typically between 0 and 1.
- The closer to 1 the better.



- Can calculate the Average Silhouette width for a cluster or a clustering

## External Measures of Cluster Validity: Entropy and Purity

Table 5.9. K-means Clustering Results for LA Document Data Set

Cluster	Entertainment	Financial	Foreign	Metro	National	Sports	Entropy	Purity
1	3	5	40	506	96	27	1.2270	0.7474
2	4	7	280	29	39	2	1.1472	0.7756
3	1	1	1	7	4	671	0.1813	0.9796
4	10	162	3	119	73	2	1.7487	0.4390
5	331	22	5	70	13	23	1.3976	0.7134
6	5	358	12	212	48	13	1.5523	0.5525
Total	354	555	341	943	273	738	1.1450	0.7203

**entropy** For each cluster, the class distribution of the data is calculated first, i.e., for cluster  $j$  we compute  $p_{ij}$ , the 'probability' that a member of cluster  $j$  belongs to class  $i$  as follows:  $p_{ij} = m_{ij}/m_j$ , where  $m_j$  is the number of values in cluster  $j$  and  $m_{ij}$  is the number of values of class  $i$  in cluster  $j$ . Then using this class distribution, the entropy of each cluster  $j$  is calculated using the standard formula  $e_j = -\sum_{i=1}^L p_{ij} \log_2 p_{ij}$ , where the  $L$  is the number of classes. The total entropy for a set of clusters is calculated as the sum of the entropies of each cluster weighted by the size of each cluster, i.e.,  $e = \sum_{j=1}^K \frac{m_j}{m} e_j$ , where  $m_j$  is the size of cluster  $j$ ,  $K$  is the number of clusters, and  $m$  is the total number of data points.

**purity** Using the terminology derived for entropy, the purity of cluster  $j$ , is given by  $\text{purity}_j = \max_i p_{ij}$  and the overall purity of a clustering is  $\text{purity} = \sum_{j=1}^K \frac{m_j}{m} \text{purity}_j$ .

## Final Comment on Cluster Validity

"The validation of clustering structures is the most difficult and frustrating part of cluster analysis.

Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage."

*Algorithms for Clustering Data, Jain and Dubes*

## K-Medoids Method

- 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
- Handling categorical data: *k-modes*
  - 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* method

## Centroid, Radius and Diameter of a Cluster (for numerical data sets)

- Centroid: the "middle" of a cluster 
$$C_m = \frac{\sum_{i=1}^N (t_{ip})}{N}$$
- Radius: square root of average distance from any point of the cluster to its centroid 
$$R_m = \sqrt{\frac{\sum_{i=1}^N (t_{ip} - c_m)^2}{N}}$$
- Diameter: square root of average mean squared distance between all pairs of points in the cluster

$$D_m = \sqrt{\frac{\sum_{i=1}^N \sum_{j=1}^N (t_{ip} - t_{jq})^2}{N(N-1)}}$$

## Clustering Summary



### ➤ Partitioning approach:

- ❖ Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
- ❖ Typical methods: k-means, k-medoids, CLARANS

### ➤ Hierarchical approach:

- ❖ Create a hierarchical decomposition of the set of data (or objects) using some criterion
- ❖ Typical methods: Diana, Agnes, BIRCH, CAMELEON

### ➤ Density-based approach:

- ❖ Based on connectivity and density functions
- ❖ Typical methods: DBSACN, OPTICS, DenClue

### ➤ Grid-based approach:

- ❖ based on a multiple-level granularity structure
- ❖ Typical methods: STING, WaveCluster, CLIQUE

177

## Clustering Summary



### ➤ Model-based:

- A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
- Typical methods: EM, SOM, COBWEB

### ➤ Frequent pattern-based:

- Based on the analysis of frequent patterns
- Typical methods: p-Cluster

### ➤ User-guided or constraint-based:

- Clustering by considering user-specified or application-specific constraints
- Typical methods: COD (obstacles), constrained clustering

### ➤ Link-based clustering:

- Objects are often linked together in various ways
- Massive links can be used to cluster objects: SimRank, LinkClus

178

Thank You !!!



For more information visit:

<http://itlab.uta.edu>



BDA 2018 (Warangal)

13 December 2018



179

