Clustering
Tips and Tricks

in 45 minutes (maybe more :)

Olfa Nasraoui, University of Louisville

Tutorial for the Data Science for Social Good Fellowship 2015 cohort
@DSSG2015@University of Chicago

https://www.researchgate.net/profile/Olfa_Nasraoui

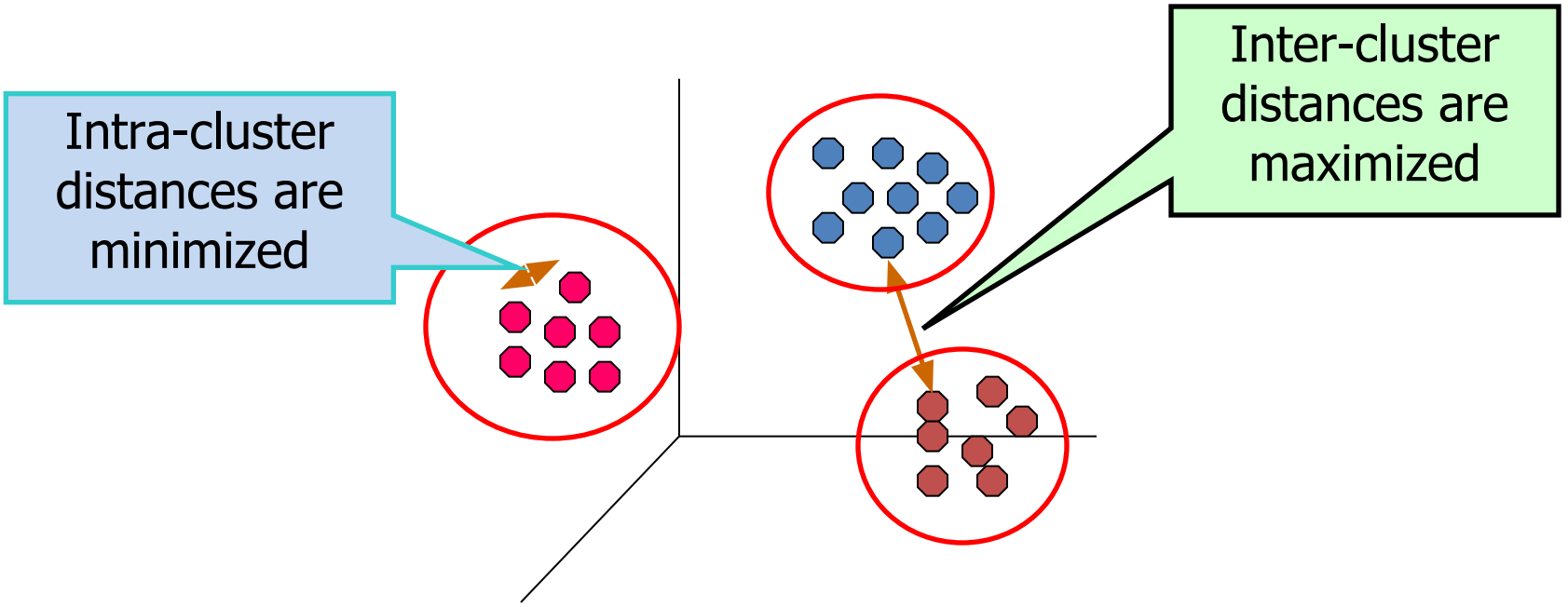
Acknowledgment: many sources collected throughout years of teaching from textbook slides, etc, in particular Tan et al

Roadmap

- Problem Definition and Examples (2 min)
- Major Families of Algorithms (2 min)
- Recipe Book (15 min)
  1. Types of inputs, Nature of data, Desired outputs ➔ Algorithm choice (5 min)
  2. Nature of data ➔ Distance / Similarity measure choice (5 min)
  3. Validation (5 min)
  4. Presentation
  5. Interpretation (5 min)
- Challenges (10 min)
  - Expanding on 1
    - number of clusters, outliers, large data, high dimensional data, sparse data, mixed /heterogeneous data types, nestedness, domain knowledge: a few labels/constraints
- Python Info (10 min)

Definition of Clustering

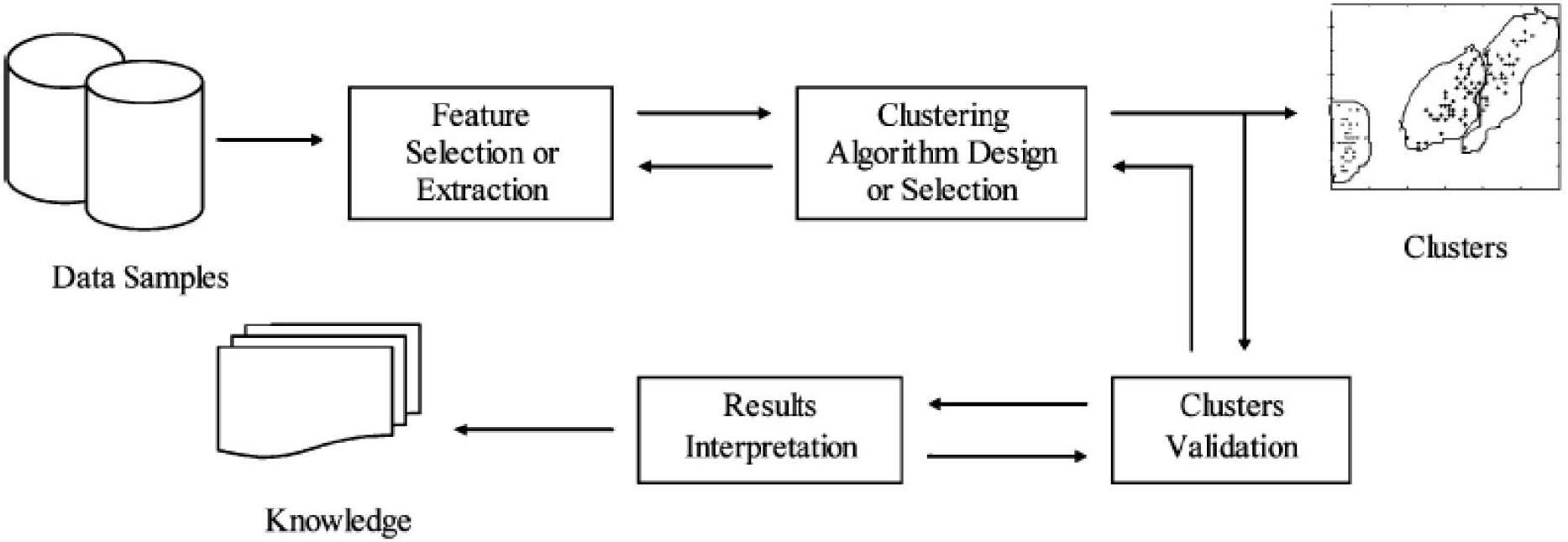
- Finding groups of objects such that:
  - the objects in a group will be similar (or related) to one another and
  - the objects in a group will be different from (or unrelated to) the objects in other groups



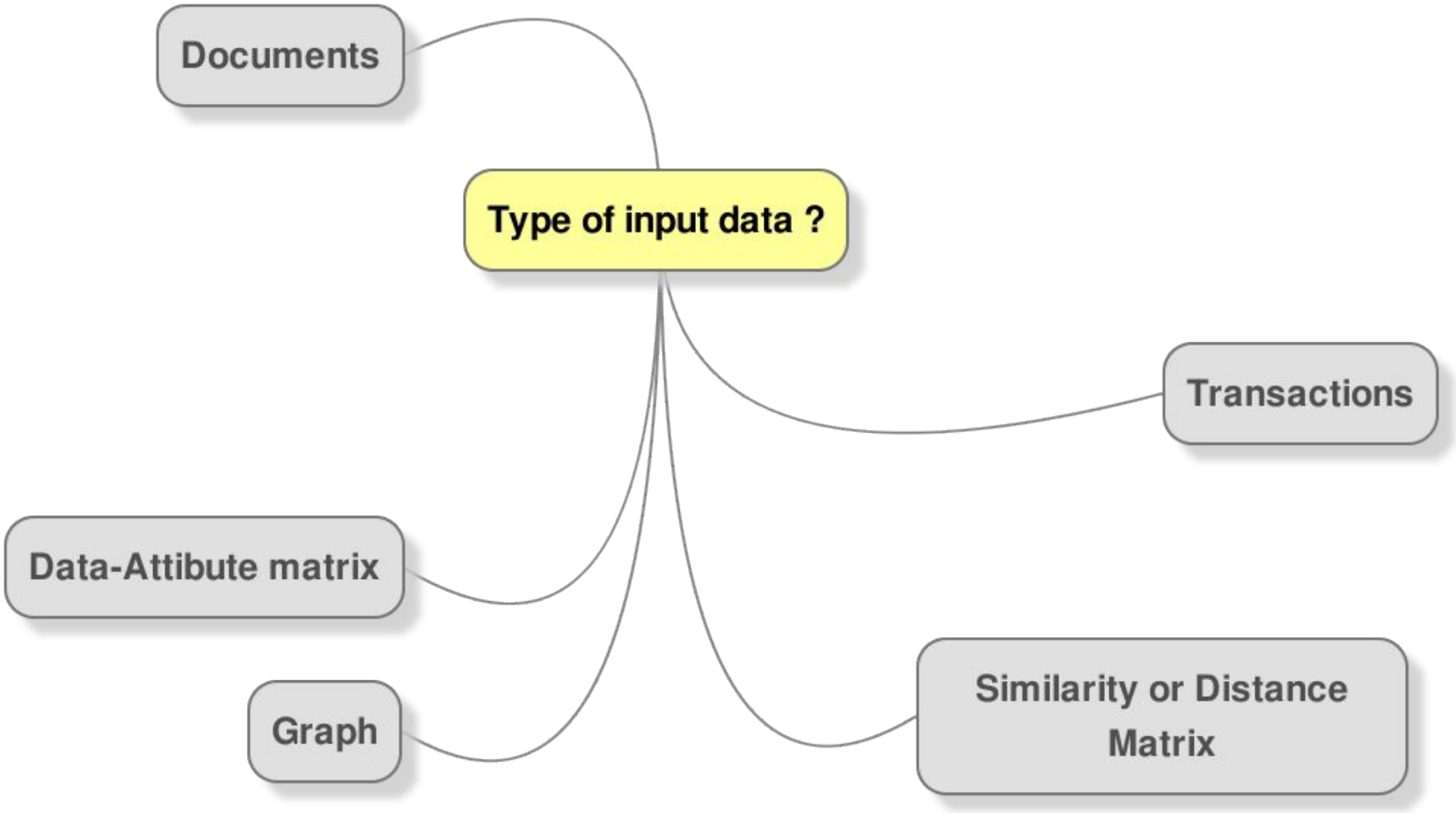
Application Examples

- Marketing: Discover groups of customers with similar purchase patterns or similar demographics
- Land use: Find areas of similar land use in an earth observation database
- City-planning: Identify groups of houses according to their house type, value, and geographical location
- Web Usage Profiling: find groups of users with similar usage or interests on a website
- Document Organization: find groups of documents about similar topics
- **Summarization or Data Reduction:** Reduce size of large data sets
  - **can be better than random sampling**
  - 1) Cluster into large # of clusters
  - 2) then use cluster centroids
- **Discretize Numerical attributes:**
  - 1) Cluster numerical attribute into partitions
  - 2) Then consider each partition (group) ⇔ 1 categorical value
- **Imputate Missing Values :**
  - Cluster attribute values
  - Replace missing value with cluster center

Clustering Steps



What is your input?



Input Data Types for Clustering

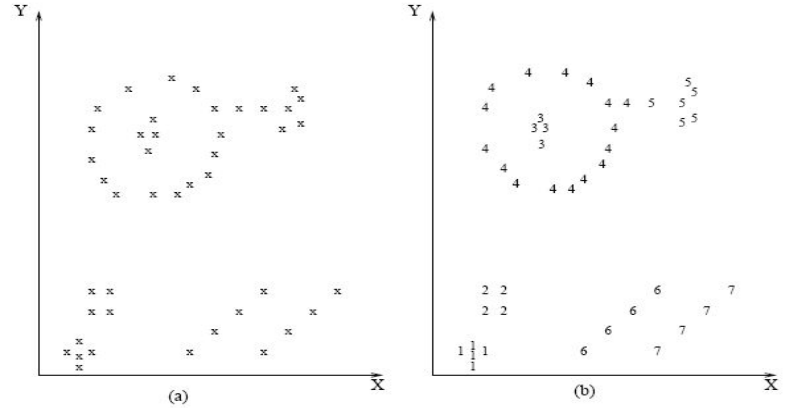
- Data matrix
  - (two modes)

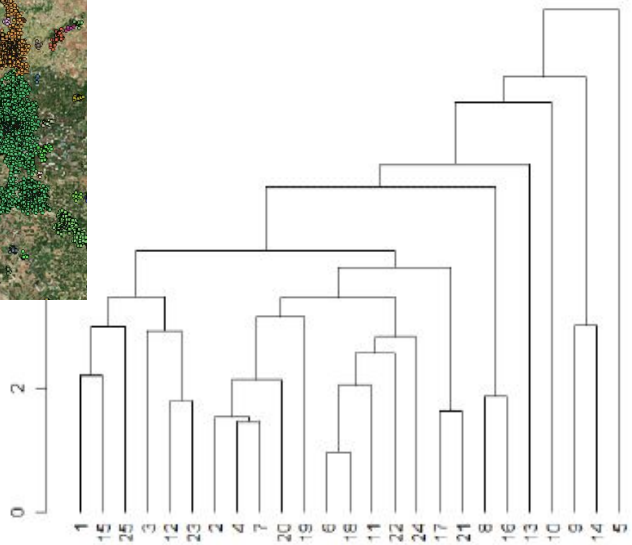
$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

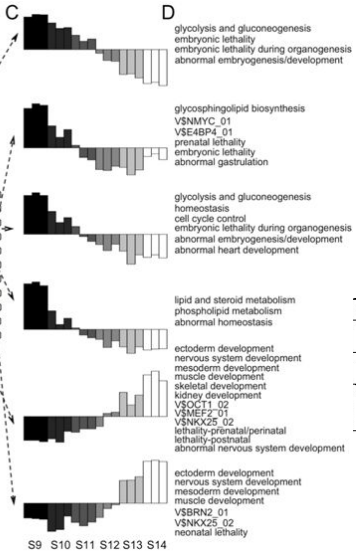
- Dissimilarity matrix
  - (one mode)
  - Example:
    - Kernel matrix (for kernel clustering algorithms)
    - Adjacency or relationship matrix in social graph
  - Or you can compute the distance matrix from every pair of data points

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

What is Your Desired Output?







P(N1)=4/4		P(V|C)
TAILS	ONE	0.50
	TWO	0.50
COLOR	LIGHT	0.50
	DARK	0.50
NUCLEI	ONE	0.25
	TWO	0.50
	THREE	0.25

P(N2)=1/4		P(V|C)
TAILS	ONE	1.0
	TWO	0.0
COLOR	LIGHT	1.0
	DARK	0.0
NUCLEI	ONE	1.0
	TWO	0.0
	THREE	0.0

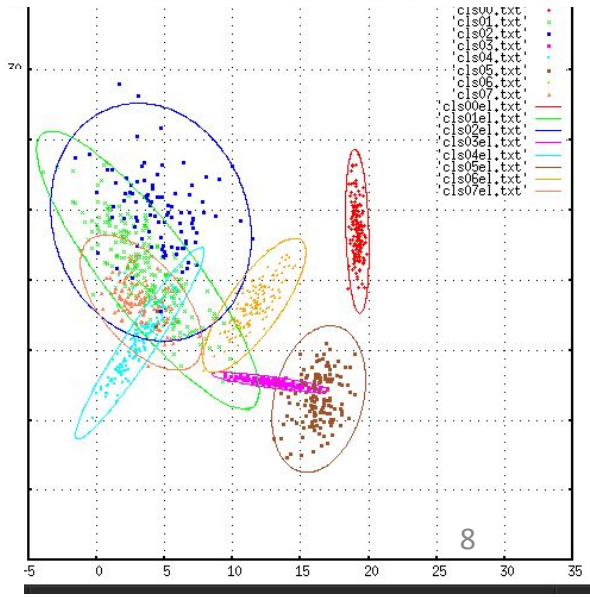
P(N3)=2/4		P(V|C)
TAILS	ONE	0.0
	TWO	1.0
COLOR	LIGHT	0.5
	DARK	0.5
NUCLEI	ONE	0.0
	TWO	1.0
	THREE	0.0

P(N6)=1/4		P(V|C)
TAILS	ONE	1.0
	TWO	0.0
COLOR	LIGHT	0.0
	DARK	1.0
NUCLEI	ONE	0.0
	TWO	0.0
	THREE	1.0

P(N4)=1/2		P(V|C)
TAILS	ONE	0.0
	TWO	1.0
COLOR	LIGHT	1.0
	DARK	0.0
NUCLEI	ONE	0.0
	TWO	1.0
	THREE	0.0

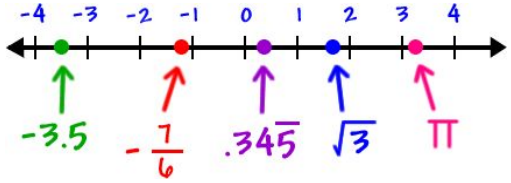
P(N5)=1/2		P(V|C)
TAILS	ONE	0.0
	TWO	1.0
COLOR	LIGHT	0.0
	DARK	1.0
NUCLEI	ONE	0.0
	TWO	0.0
	THREE	1.0

Fig. 3. A sample COBWEB hierarchy with nodes numbered in order of creation.



A Few Specialty Algorithms Depending on the Data

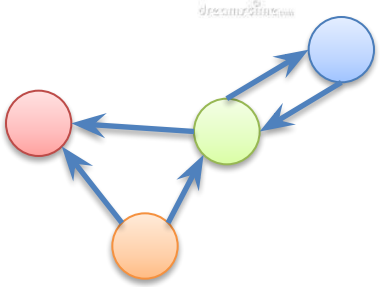
- **Numerical** data
  - k-means, DBSCAN, etc.



- **Categorical** data
  - k-modes, ROCK, CACTUS, etc.

- **Transactional** or **text** data
  - spherical k-means, Bisecting K-means

- **Graph** data
  - KMETIS, spectral clustering, etc.



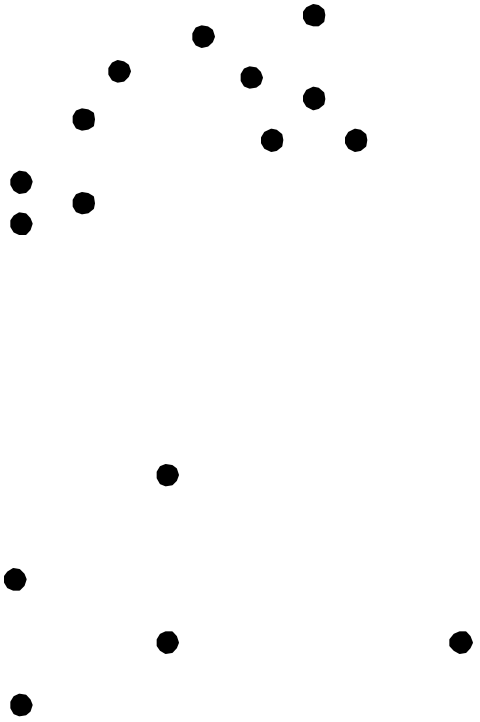
Types of Clustering

- Partitional Clustering
  - A **division** of data objects into non-overlapping subsets (clusters)
    - This division is called a partition
  - Can also be overlapping (soft clusters)
    - Fuzzy clustering
    - Most probabilstic/ model-based clustering

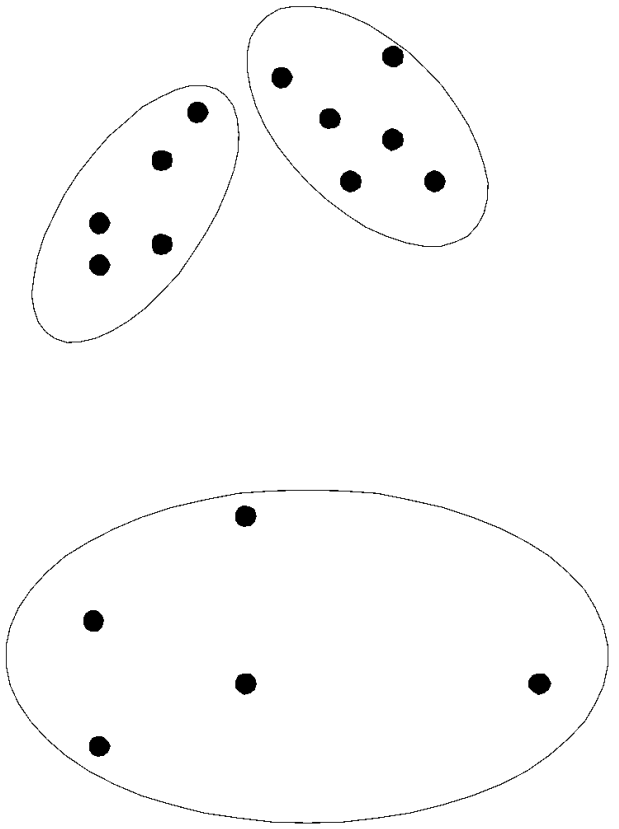
- Hierarchical clustering
  - A set of nested clusters organized as a **hierarchical tree**,
  - Tree is called dendogram

- Density-based clustering
  - 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.

Partitional Clustering

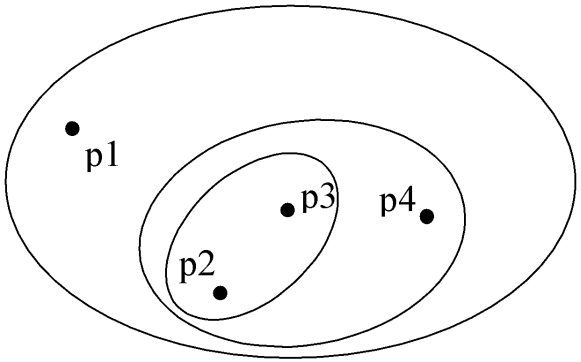


Original Points

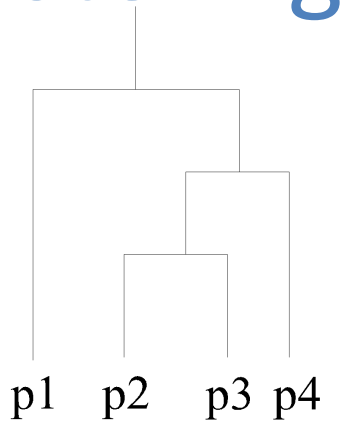


A Partitional Clustering

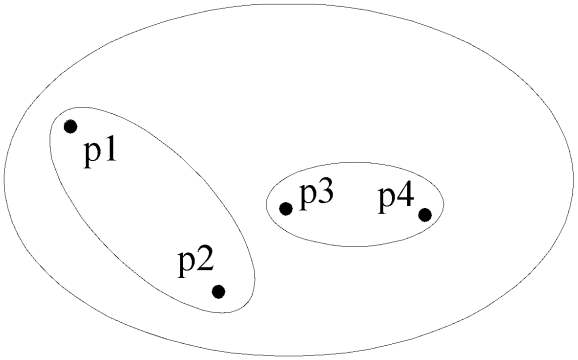
Hierarchical Clustering



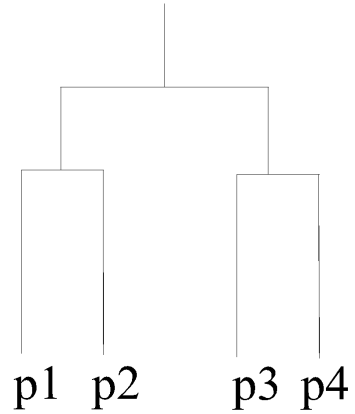
Traditional Hierarchical Clustering



Traditional Dendogram

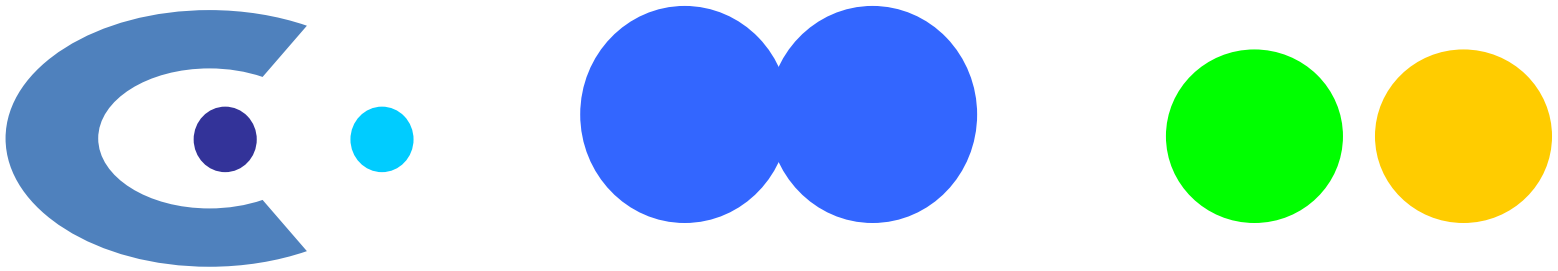


Non-traditional Hierarchical Clustering



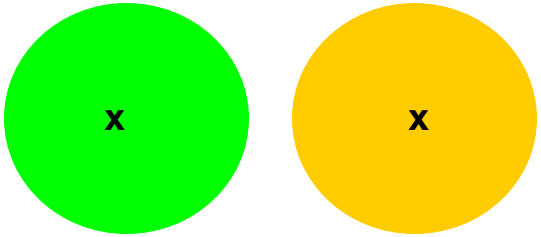
Non-traditional Dendogram

6 Density Based Clusters



K-means Clustering

- Partitional clustering approach
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K, must be specified
- The basic algorithm is very simple



1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

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 // and remove it from list of clusters
4: **for** $i = 1$ to *number_of_iterations* **do** // repeat several trial bisections to find the best
5: Bisect the selected cluster using basic K-means // i.e. split it in 2
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

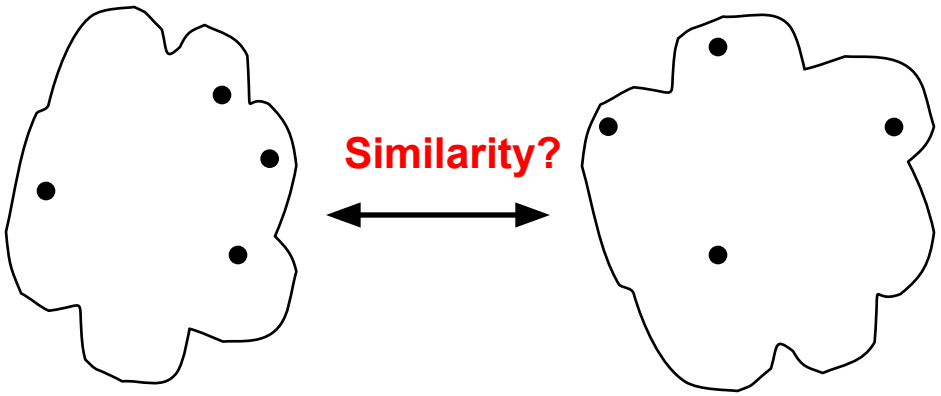
Hierarchical Clustering

- 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

- Most 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

How to Define Inter-Cluster Similarity

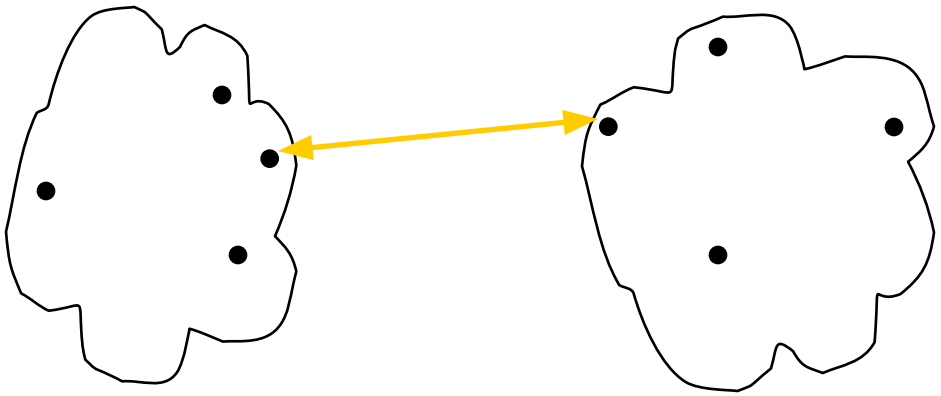


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

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

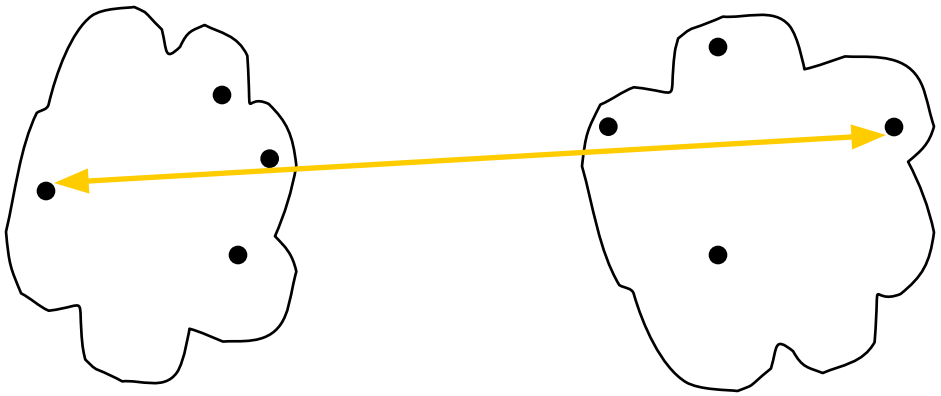


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

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

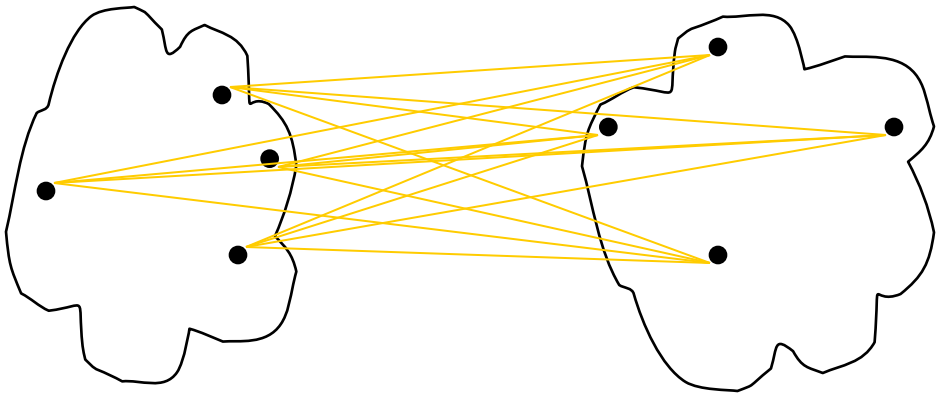


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

How to Define Inter-Cluster Similarity

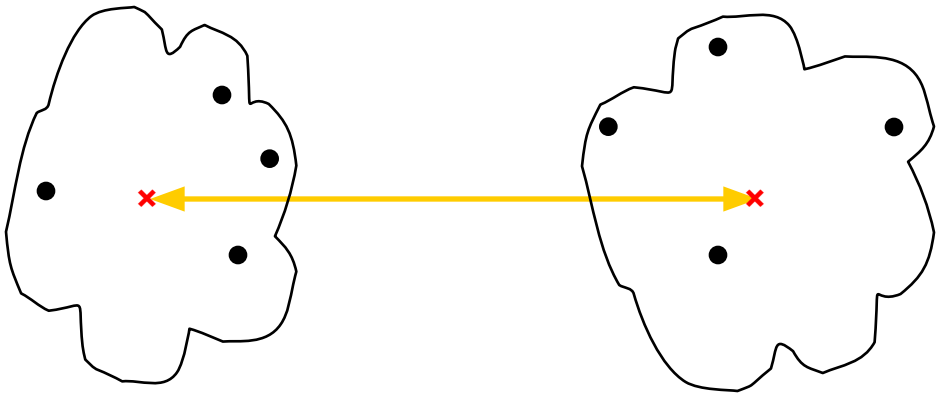


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

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

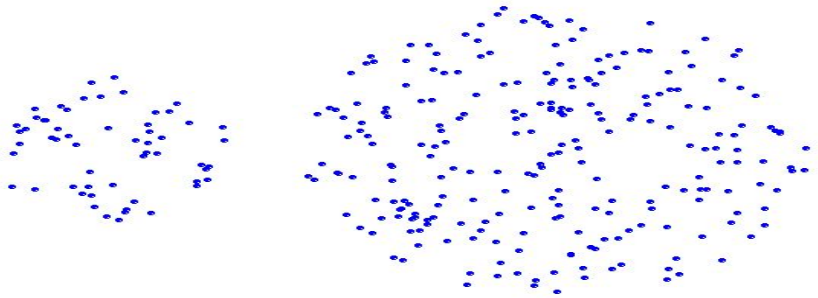


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

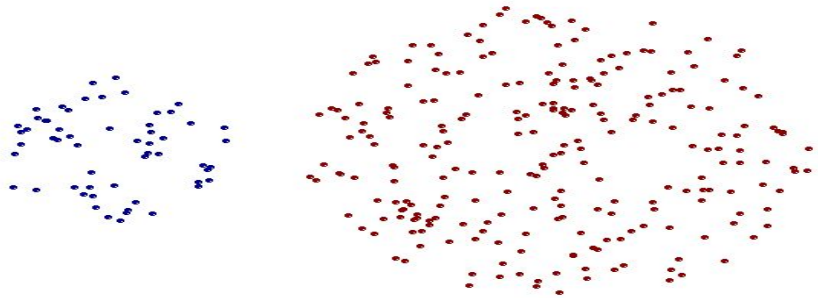
Proximity Matrix

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

Strength of MIN



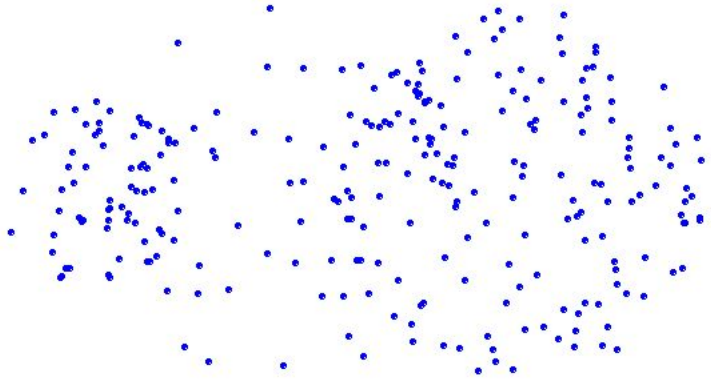
Original Points



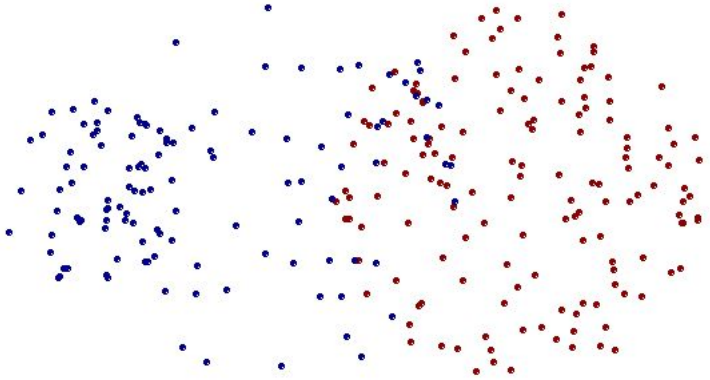
Two Clusters

- **Can handle non-elliptical shapes**

Limitations of MIN



Original Points

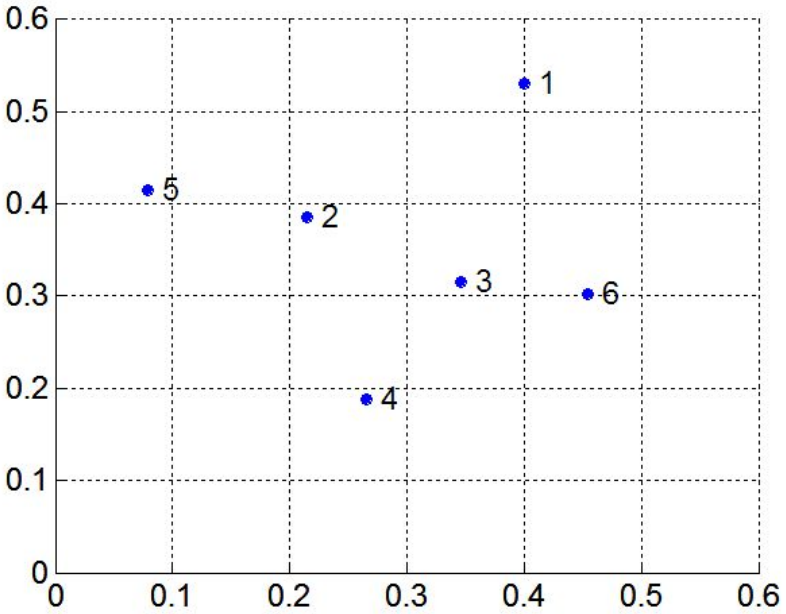


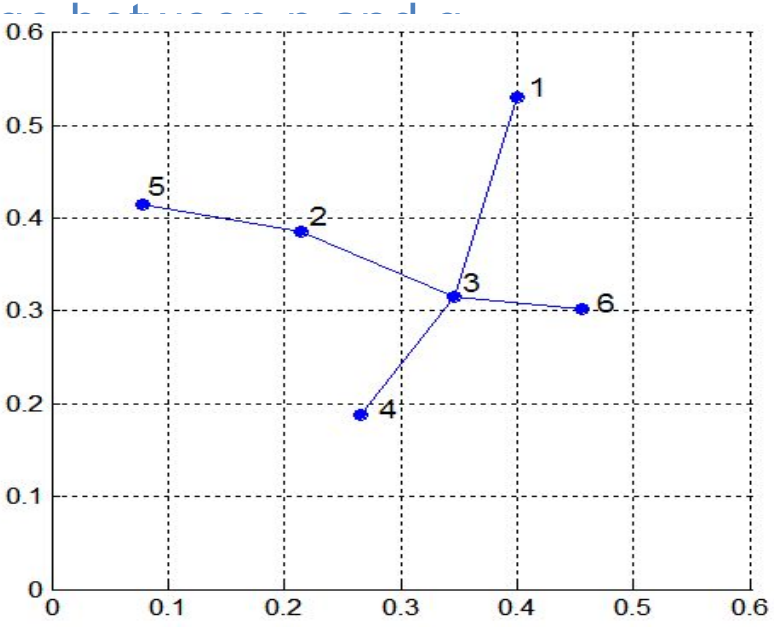
Two Clusters

- **Sensitive to noise and outliers**

MST: Divisive Hierarchical Clustering (2 steps)

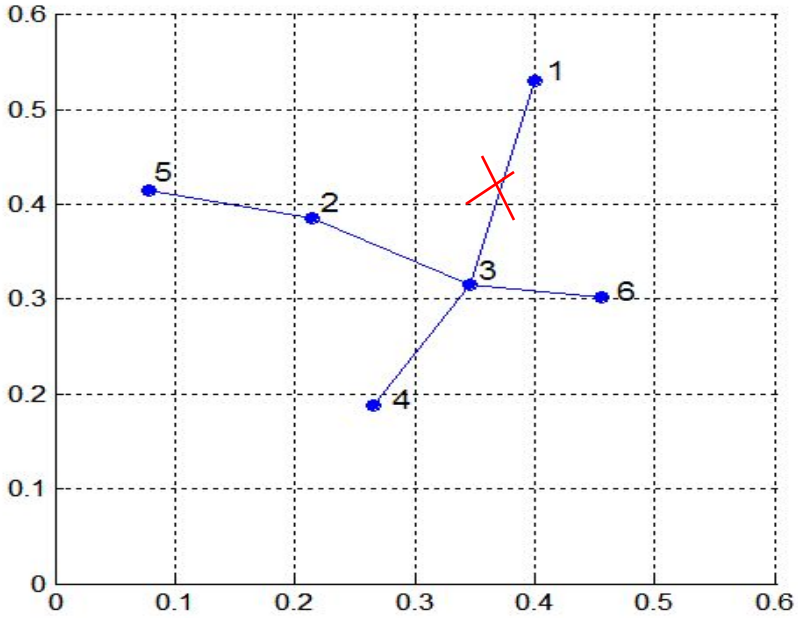
- **STEP 1:** Build **MST** (Minimum Spanning Tree)
  - Start with a tree that consists of any point
  - In successive steps, look for the closest pair of points (p, q) such that one point (p) is in the current tree but the other (q) is not in tree





MST: Divisive Hierarchical Clustering

- **STEP 2: Build Hierarchy**: Use MST for constructing hierarchy of clusters: starts with MST from STEP 1, then gradually removes one edge (the one with highest distance) at a time (thus dividing clusters) to obtain a hierarchy



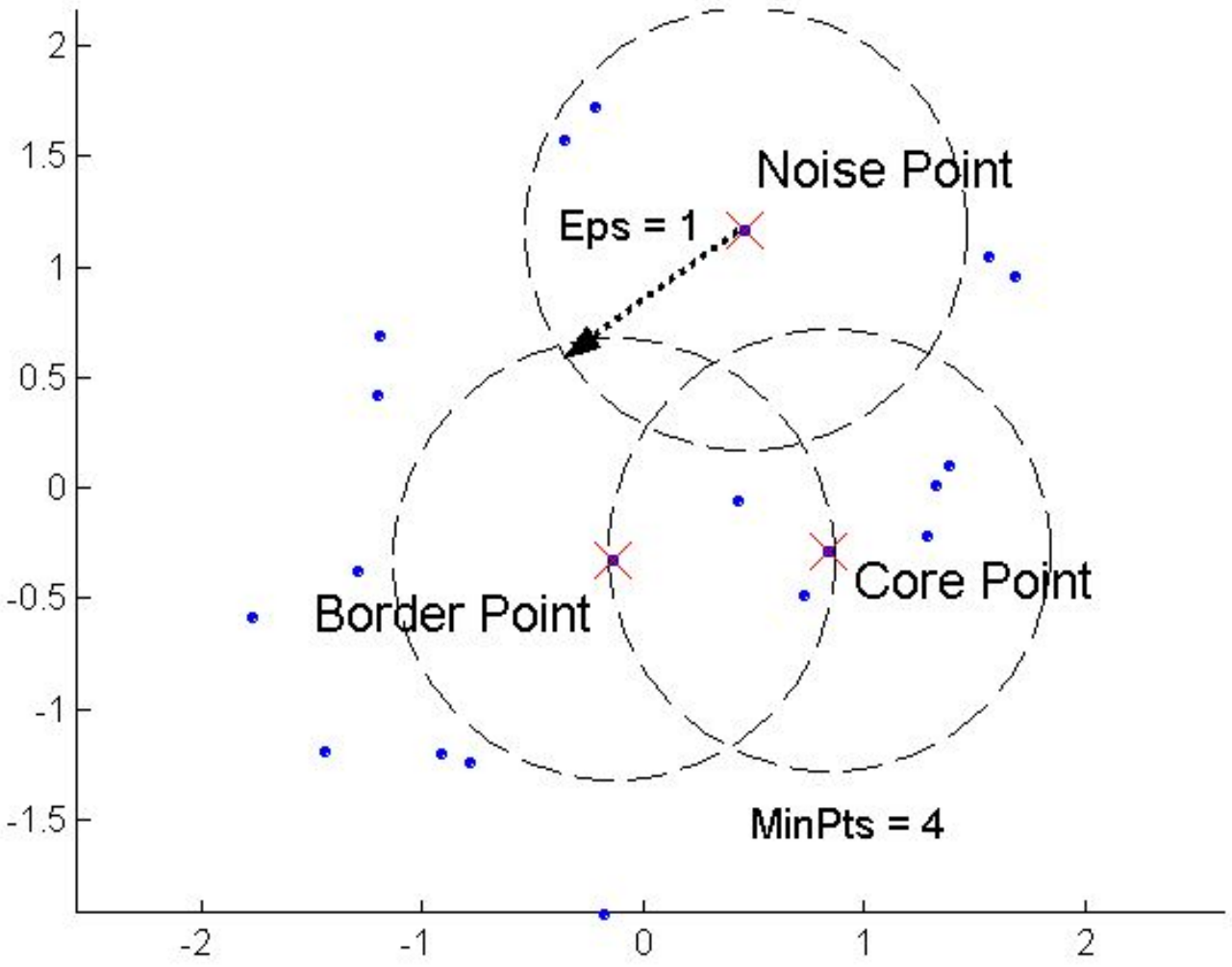
Algorithm 7.5 MST Divisive Hierarchical Clustering Algorithm

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: DBSCAN

- DBSCAN is a density-based algorithm.
  - **Density** = number of points within a specified **radius** (**Eps**)
  - A point is a **core point** if it has more than a specified **Minimum 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
    - i.e.: Not a CORE pt, but close to a CORE pt
  - A **noise point** is any point that is not a core point and not a border point.

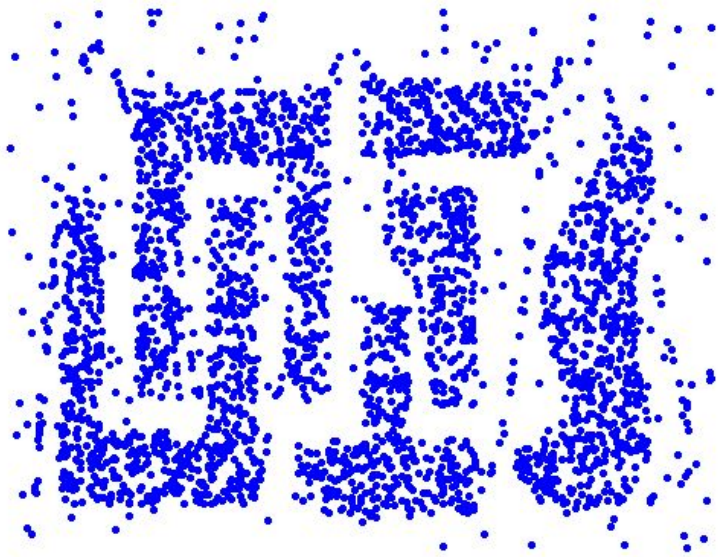
DBSCAN: Core, Border, and Noise Points



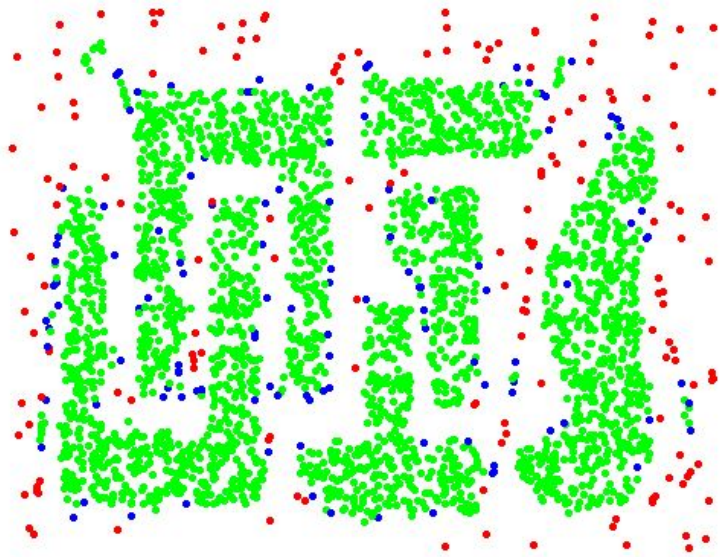
DBSCAN Algorithm

- Label all pts as CORE, BORDER, or NOISE
- Eliminate noise points
- Perform clustering on the remaining points
  - Put an edge between all CORE pts within Eps of each other
  - Make each group of connected CORE pts into a separate cluster (i.e. give them a cluster label)
  - Assign each BORDER pt to one of the clusters of its associated CORE pts

DBSCAN: Core, Border and Noise Points



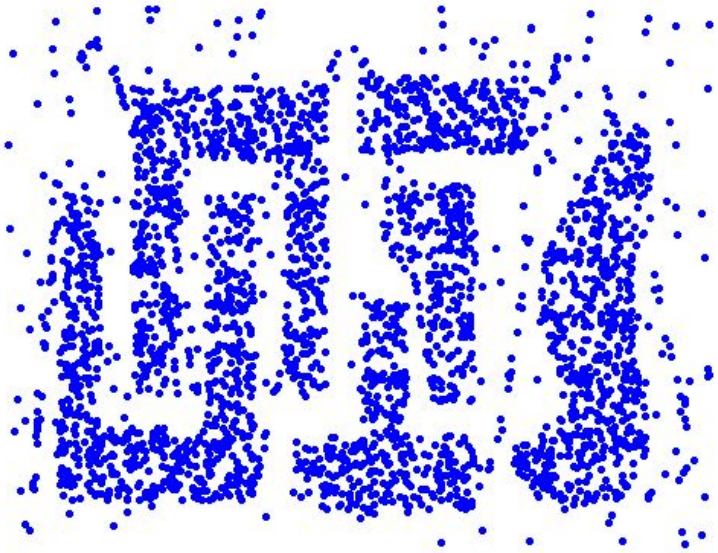
Original Points



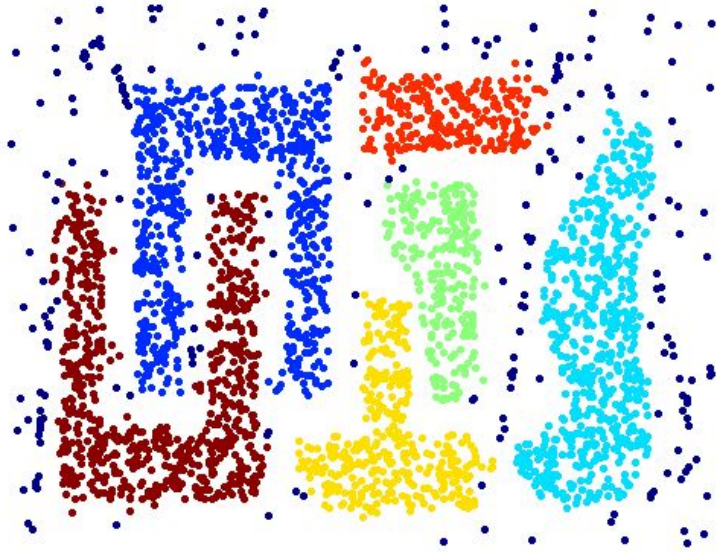
Point types: core, border and noise

Eps = 10, MinPts = 4

When DBSCAN Works Well



Original Points

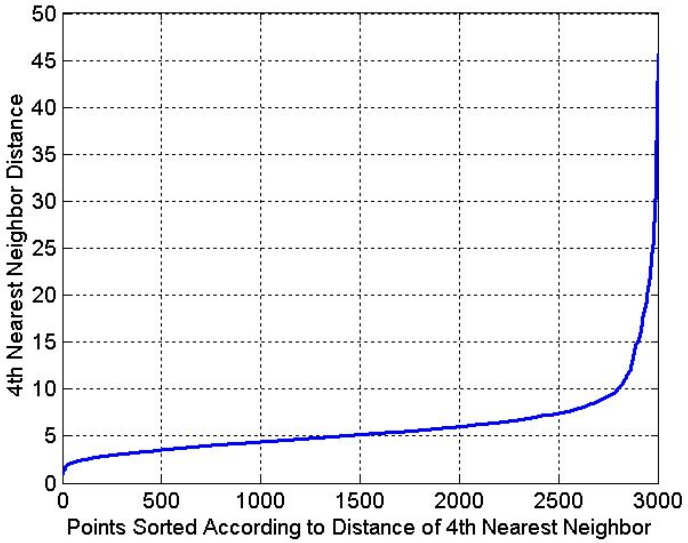


Clusters

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

DBSCAN: Determining EPS and MinPts

- Idea is that for points in a cluster, their k^{th} nearest neighbors are at roughly the same distance
- Noise points have the k^{th} nearest neighbor at farther distance
- So, plot sorted distance of every point to its k^{th} nearest neighbor, choose dist where max jump occurs (the knee)



Category	Hierarchical						
Name	***Type of data***	***Complexity****	***Geometry***	***Outliers***	***Input parameters***	***Results***	***Clustering criterion***
BIRCH	Numerical	O(n)	non-convex shapes	Yes	Radius of clusters, branching factor	CF = (number of points in the cluster *N*, linear sum of the points in the cluster *LS*, the square sum of *N* data points *SS*)	A point is assigned to closest node (cluster) according to a chosen distance metric. Also, the clusters definition is based on the requirement that the number of points in each cluster must satisfy a threshold.
CURE	Numerical	$O(n^2 \log n)$	Arbitrary shapes	Yes	Number of clusters, number of clusters representatives	Assignment of data values to clusters	The clusters with the closest pair of representatives (well scattered points) are merged at each step.
ROCK	Categorical	$O(n^2+nm_m m_a+n^2 \log n)$, $O(n^2, nm_m m_a)$ where m_m is the maximum number of neighbors for a point and m_a is the average number of neighbors for a point	Arbitrary shapes	Yes	Number of clusters	Assignment of data values to clusters	$\max(E_l)$ $E_l = \sum_{i=1}^{k} n_i \sum_{p_q, p_r \in V_i} \frac{link(p_q, p_r)}{n_i^{1+2f(\theta)}}$ - v_i *center of cluster I* - *link* (p_q, p_r) = *the number of common neighbors between* p_l *and* p_r.

Category	Partitional						
Name	***Type of data***	***Complexity****	***Geometry***	***Outliers, noise***	***Input parameters***	***Results***	***Clustering criterion***
K-Mean	Numerical	O(n)	non-convex shapes	No	Number of clusters	Center of clusters	$\min_{v1,v2,\ldots,vk}(E_k)$ $E_k = \sum_{i=1}^{k}\sum_{k=1}^{n} d^2(x_k, v_i)$
K-mode	Categorical	O(n)	non-convex shapes	No	Number of clusters	Modes of clusters	$\min_{Q1,Q2,\ldots,Qk}(E_k)$ $E = \sum_{i=1}^{k}\sum_{l=1}^{n} d(X_l, Q_i)$ $D(X_i,Q_l)$ = distance between categorical objects X_l, and modes Q_i
PAM	Numerical	$O(k(n-k)^2)$	non-convex shapes	No	Number of clusters	Medoids of clusters	min (TC_{ih}) $TC_{ih} = \Sigma_j C_{jih}$
CLARA	Numerical	$O(k(40+k)^2 + k(n-k))$	non-convex shapes	No	Number of clusters	Medoids of clusters	min (TC_{ih}) $TC_{ih} = \Sigma_j C_{jih}$ (C_{jih} = the cost of replacing center i with h as far as Oj is concerned)
CLARANS	Numerical	$O(kn^2)$	non-convex shapes	No	Number of clusters, maximum number of neighbors examined	Medoids of clusters	min (TC_{ih}) $TC_{ih} = \Sigma_j C_{jih}$
FCM Fuzzy C-Means	Numerical	O(n)	Non-convex shapes	No	Number of clusters	Center of cluster, beliefs	$\min_{U,v1,v2,\ldots,vk}(Jm(U,V))$ $J_m(U,V) = \sum_{i=1}^{k}\sum_{j=1}^{n} U_{ik}^m\, d^2(x_j, v_i)$

Category	Density-based						
Name	***Type of data***	***Complexity****	***Geometry***	***Outliers, noise***	***Input parameters***	***Results***	***Clustering criterion***
DBSCAN	Numerical	O(nlogn)	Arbitrary shapes	Yes	Cluster radius, minimum number of objects	Assignment of data values to clusters	Merge points that are density reachable into one cluster.
DENCLUE	Numerical	O(nlogn)	Arbitrary shapes	Yes	Cluster radius σ, Minimum number of objects ξ	Assignment of data values to clusters	$f_{Gauss}^{D}(x^*) = \sum_{x_1 \in near(x^*)} e^{-\frac{d(x^*, x_1)^2}{2\sigma^2}}$ *x^* density attractor for a point x if $F_{Gauss} > \xi$ then x attached to the cluster belonging to x^*.*

Category	Grid-Based						
Name	***Type of data***	***Complexity****	***Geometry***	***Outliers***	***Input parameters***	***Output***	***Clustering criterion***
Wave-Cluster	Spatial data	O(n)	Arbitrary shapes	Yes	Wavelets, the number of grid cells for each dimension, the number of applications of wavelet transform.	Clustered objects	Decompose feature space applying wavelet transformation *Average sub-band* → clusters Detail sub-bands → clusters boundaries
STING	Spatial data	O(K) K is the number of grid cells at the lowest level	Arbitrary shapes	Yes	Number of objects in a cell	Clustered objects	Divide the spatial area into rectangle cells and employ a hierarchical structure. Each cell at a high level is partitioned into a number of smaller cells in the next lower level

Distance and Similarity Measures

Distance functions

- Key to clustering. "similarity" and "dissimilarity" can also commonly used terms.
- There are numerous distance functions for
  - Different types of data
    - Numeric data
    - Nominal data
  - Different specific applications

Distance functions for numeric attributes

- Most commonly used functions are
  - Euclidean distance and
  - Manhattan (city block) distance
- We denote distance with: $dist(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are data points (vectors)
- They are special cases of Minkowski distance. h is positive integer.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = ((x_{i1} - x_{j1})^h + (x_{i2} - x_{j2})^h + ... + (x_{ir} - x_{jr})^h)^{\frac{1}{h}}$$

Euclidean distance and Manhattan distance

- If h = 2, it is the Euclidean distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \ldots + (x_{ir} - x_{jr})^2}$$

- If h = 1, it is the Manhattan distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = | x_{i1} - x_{j1} | + | x_{i2} - x_{j2} | + \ldots + | x_{ir} - x_{jr} |$$

- Weighted Euclidean distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{w_1(x_{i1} - x_{j1})^2 + w_2(x_{i2} - x_{j2})^2 + \ldots + w_r(x_{ir} - x_{jr})^2}$$

Squared distance and Chebychev distance

- Squared Euclidean distance: to place progressively greater weight on data points that are further apart.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = (x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \ldots + (x_{ir} - x_{jr})^2$$

- Chebychev distance: one wants to define two data points as "different" if they are different on any one of the attributes.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \max(|x_{i1} - x_{j1}|, |x_{i2} - x_{j2}|, \ldots, |x_{ir} - x_{jr}|)$$

Distance functions for binary and nominal attributes

- Binary attribute: has two values or states but no ordering relationships, e.g.,
  - Gender: male and female.
- We use a confusion matrix to introduce the distance functions/measures.
- Let the *i*th and *j*th data points be $\mathbf{x}_i$ and $\mathbf{x}_j$ (vectors)

Contingency matrix

Data point i \ Data point j	1	0	
1	a	b	$a+b$
0	c	d	$c+d$
	$a+c$	$b+d$	$a+b+c+d$

(10)

a: the number of attributes with the value of 1 for both data points.

b: the number of attributes for which $x_{if} = 1$ and $x_{jf} = 0$, where x_{if} (x_{jf}) is the value of the fth attribute of the data point $\mathbf{x}_i$ ($\mathbf{x}_j$).

c: the number of attributes for which $x_{if} = 0$ and $x_{jf} = 1$.

d: the number of attributes with the value of 0 for both data points.

Symmetric binary attributes

- A binary attribute is **symmetric** if both of its states (0 and 1) have equal importance, and carry the same weights, e.g., male and female of the attribute Gender
- Distance function: Simple Matching Coefficient, proportion of mismatches of their values

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \frac{b + c}{a + b + c + d}$$

Symmetric binary attributes: example

$\mathbf{x}_1$	1	1	1	0	1	0	0
$\mathbf{x}_2$	0	1	1	0	0	1	0

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \frac{2+1}{2+2+1+2} = \frac{3}{7} = 0.429$$

Asymmetric binary attributes (this is the most important attribute for TEXT, Transactions, Web sessions, etc)

- Asymmetric: if one of the states is more important or more valuable than the other.
  - By convention, state 1 represents the more important state, which is typically the rare or infrequent state.
  - Jaccard coefficient is a popular measure

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \frac{b + c}{a + b + c}$$

  - We can have some variations, adding weights, etc

Nominal attributes

- Nominal attributes: with more than two states or values.
  - the commonly used distance measure is also based on the simple matching method.
  - Given two data points $\mathbf{x}_i$ and $\mathbf{x}_j$, let the number of attributes be r, and the number of values that match in $\mathbf{x}_i$ and $\mathbf{x}_j$ be q.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \frac{r - q}{r}$$

Distance function for text documents

- A text document consists of a sequence of sentences and each sentence consists of a sequence of words.
- To simplify: a document is usually considered a "bag" of words in document clustering.
  - Sequence and position of words are ignored.
- A document is represented with a vector just like a normal data point.
  - TF-IDF (or simple binary), remember to eliminate rare and overly-frequent terms
- It is common to use similarity to compare two documents rather than distance.
  - The most commonly used similarity function is the **cosine similarity**.

Mutual Neighbor Distance

$$MND(\mathbf{x}_i, \mathbf{x}_j) = NN(\mathbf{x}_i, \mathbf{x}_j) + NN(\mathbf{x}_j, \mathbf{x}_i)$$

With *NN(x, y)* the **neighbor number** of *x* with respect to *y*.

MND is not symmetric (it is not a metric)

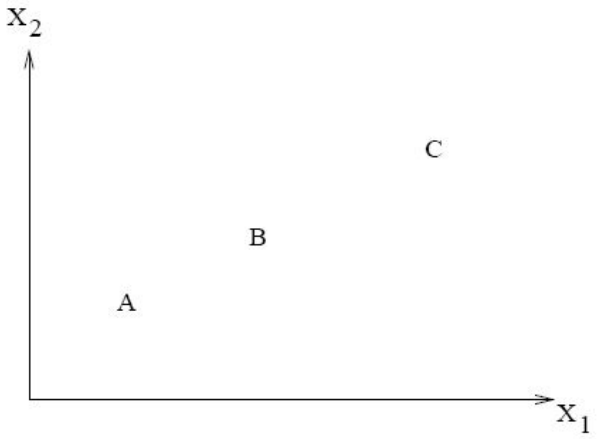


Figure 4. A and B are more similar than A and C.

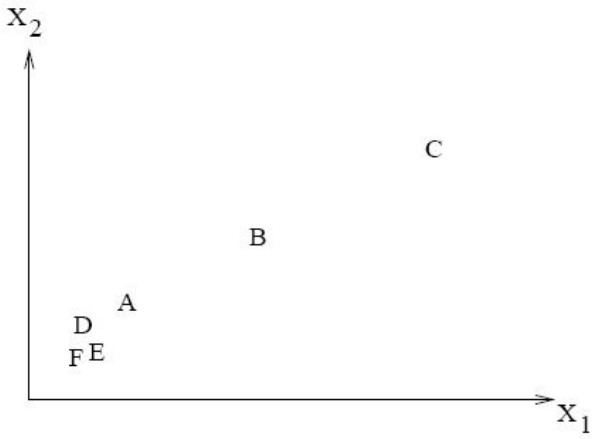


Figure 5. After a change in context, B and C are more similar than B and A.

Cluster Validation

Measures of Cluster Validity

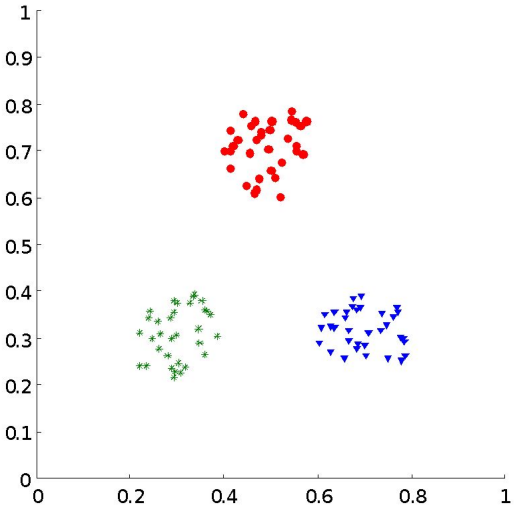
- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.
  - External Index: Used to measure the extent to which cluster labels match externally supplied class labels.
    - Entropy
  - Internal Index: Used to measure the goodness of a clustering structure *without* external information.
    - E.g. 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

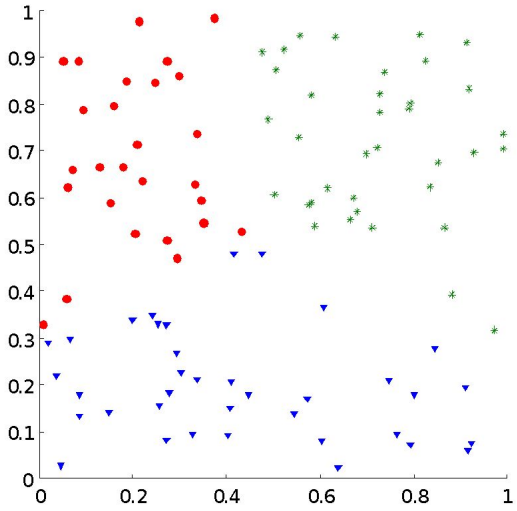
Measuring Cluster Validity Via Correlation

- Two matrices
  - Proximity Matrix (P(i,j) = similarity or distance between pt i and pt j)
  - "Incidence" Matrix (I(i,j) =1 if in same cluster)
    - 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 P and I
  - 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



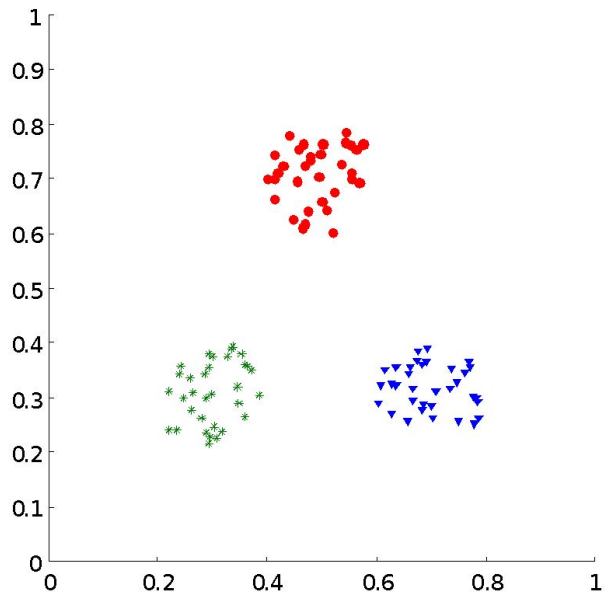


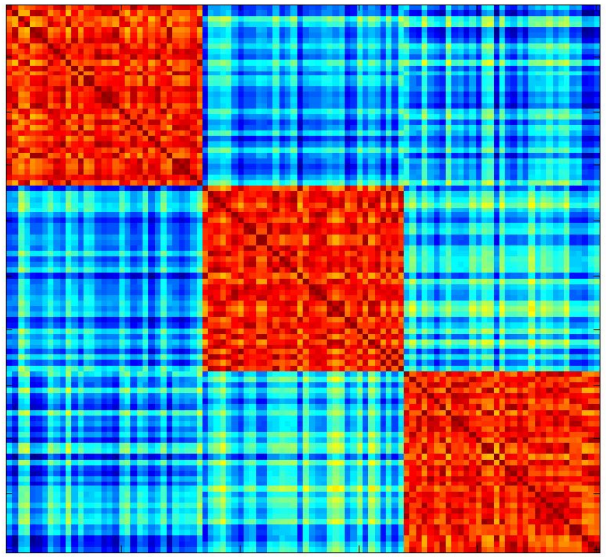
Corr = -0.9235 **(close to -1 because distance instead of similarity was used as proximity)**

Corr = -0.5810 **(closer to 0)**

Using Similarity Matrix for Cluster Validation

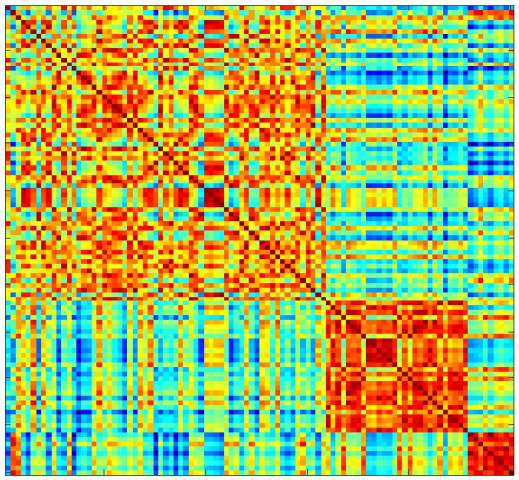
- Order the similarity matrix with respect to cluster labels and inspect visually. → you see clusters/**blocks** in matrix

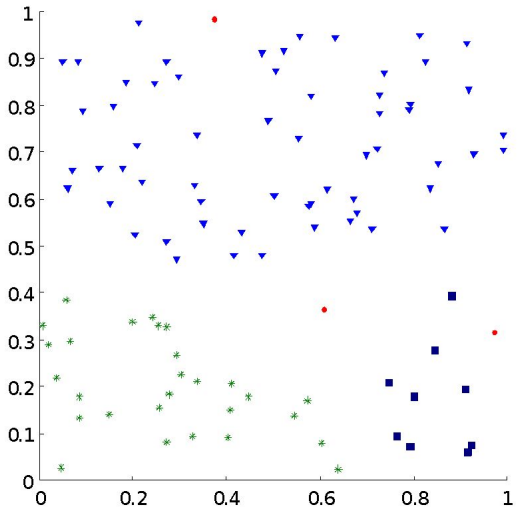




Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp

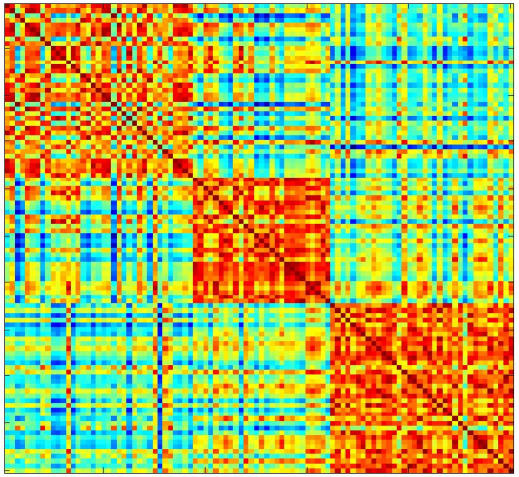


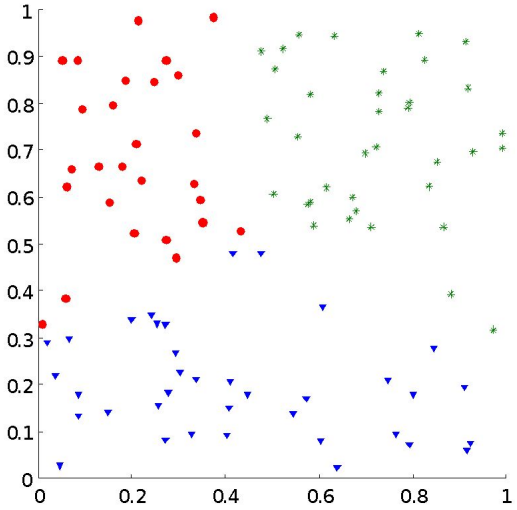


DBSCAN

Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp

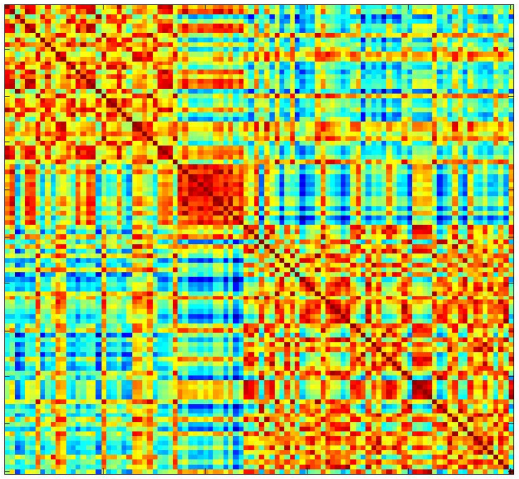


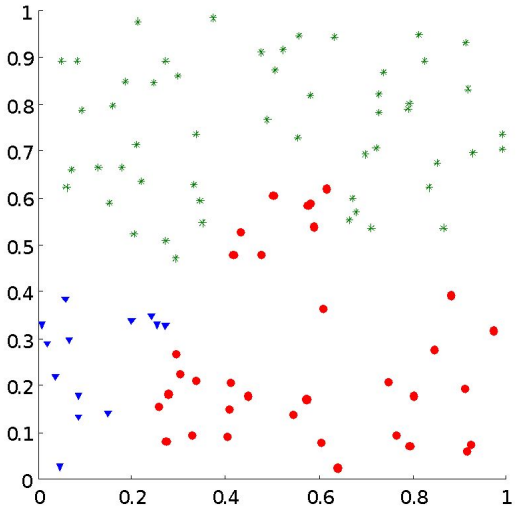


K-means

Using Similarity Matrix for Cluster Validation

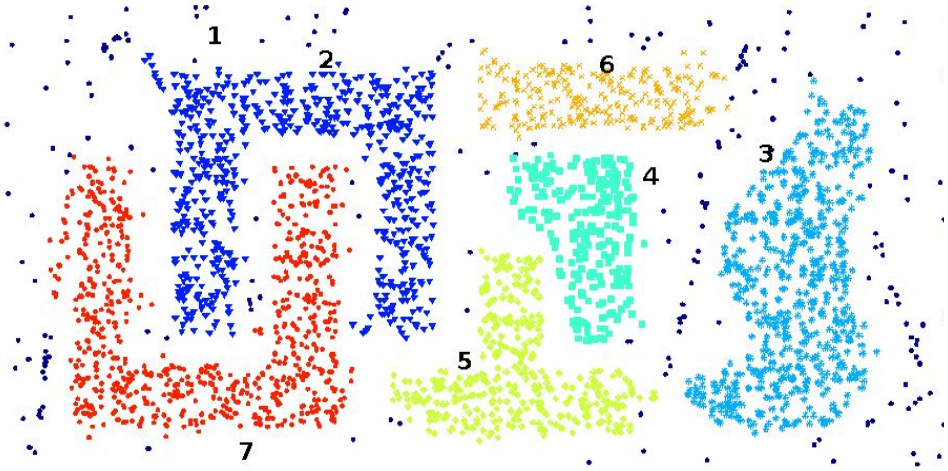
- Clusters in random data are not so crisp

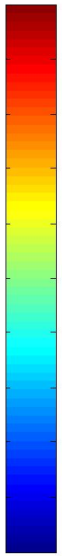




Complete Link

Using Similarity Matrix for Cluster Validation

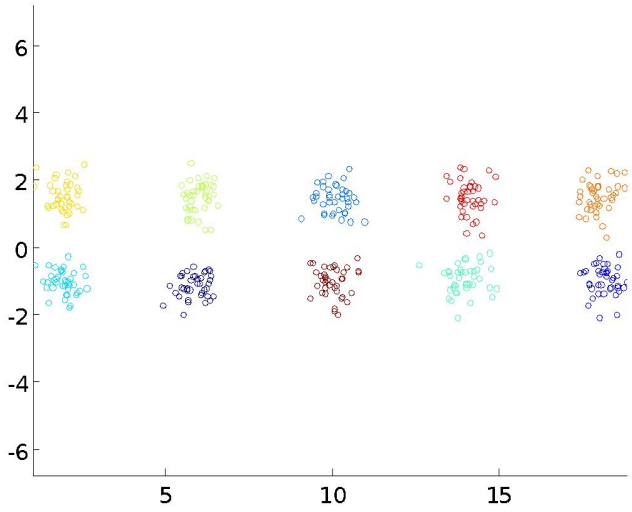


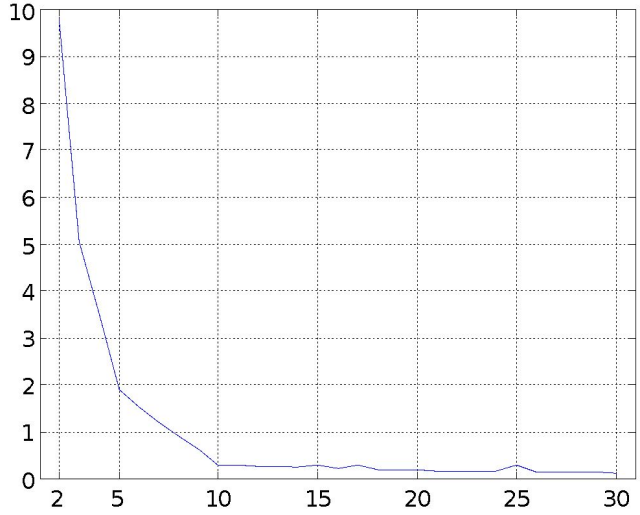


DBSCAN

Internal Measures: SSE

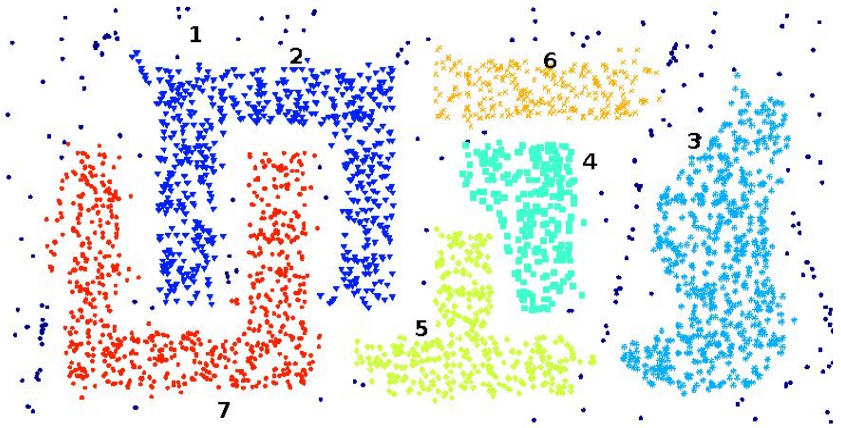
- Clusters in more complicated figures aren't well separated
- Internal Index: Used to measure the goodness of a clustering structure without respect to external information
  - SSE
- SSE is good for comparing two clusterings or two clusters (average SSE).
- Can also be used to estimate the number of clusters (but with reservations since SSE is biased by K!!!)

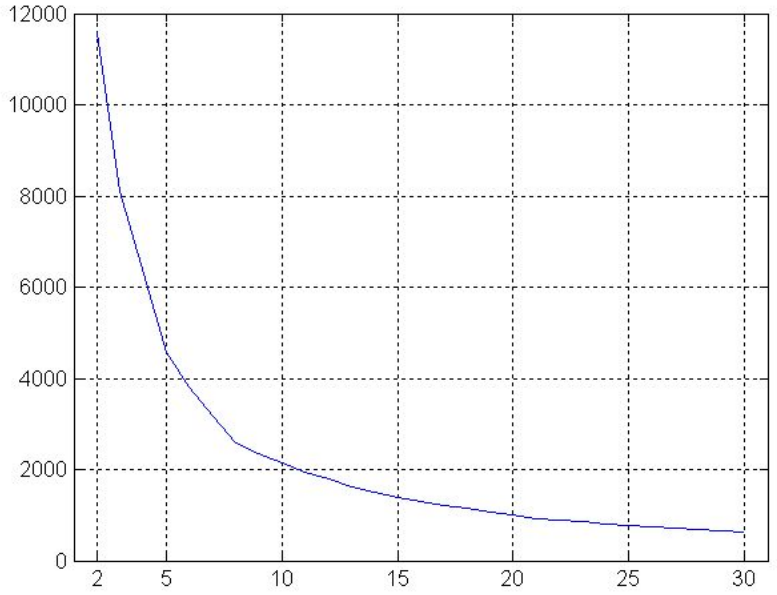




Internal Measures: SSE

- SSE curve for a more complicated data set





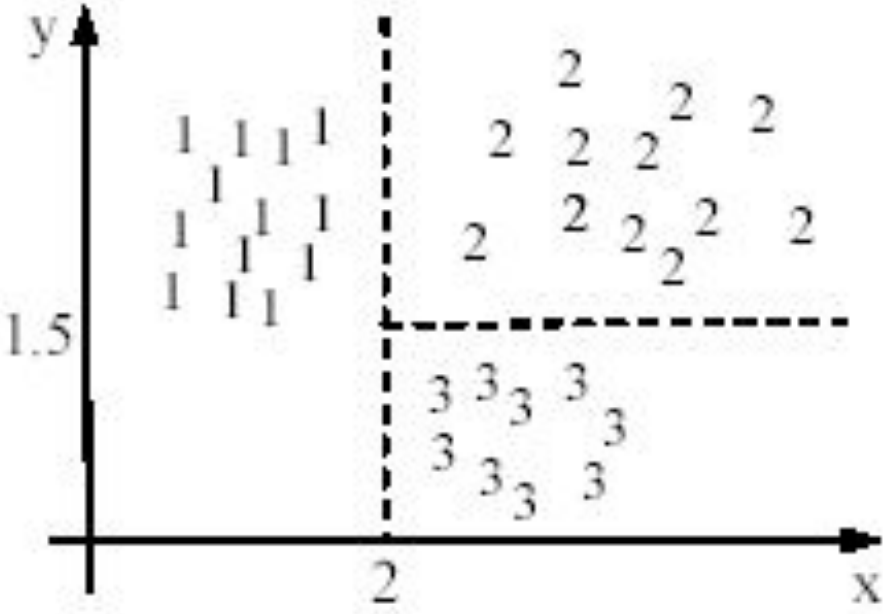
SSE of clusters found using K-means

Cluster Presentation & Interpretation

- Dendogram (hierarchy)
- Cluster Summary
  - Centroid
  - Size of cluster
  - Cluster representative (medoid)
  - Frequent words
  - Frequent categories
  - Pie charts
  - histograms of attributes
- Parallel Coordinates
- Discriminating Rules

Using classification model

- All the data points in a cluster are regarded to have the same class label, e.g., the cluster ID.
  - run a supervised learning algorithm on the data to find a classification model.



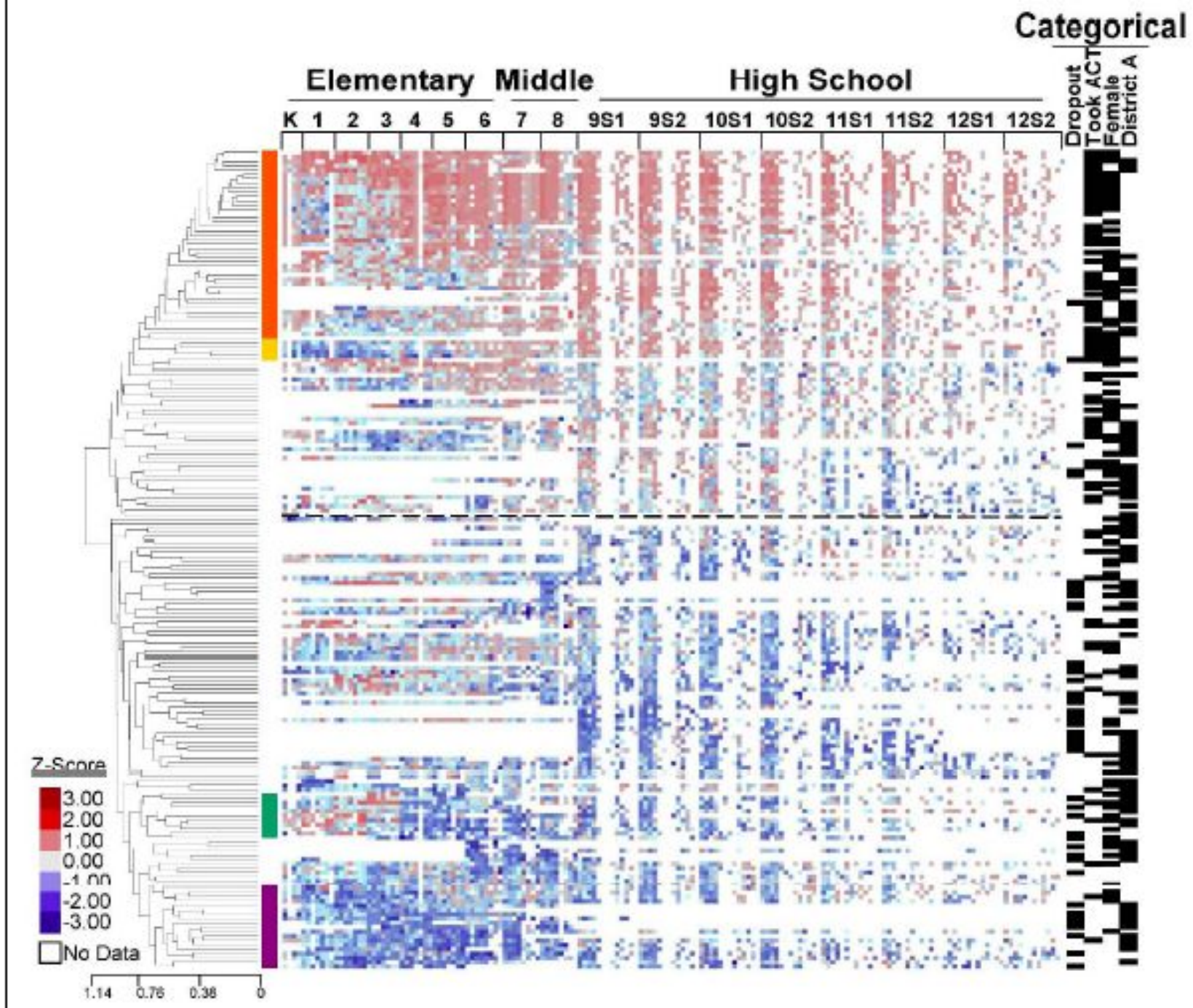
$x \leq 2 \rightarrow$ cluster 1

$x > 2, y > 1.5 \rightarrow$ cluster 2

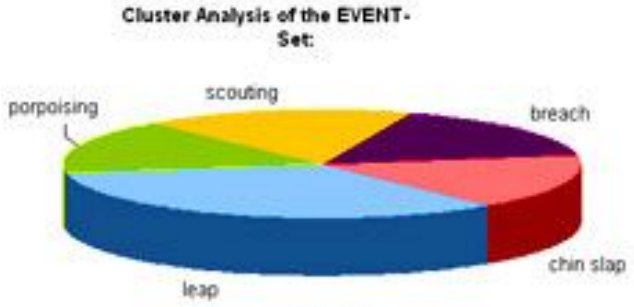
$x > 2, y \leq 1.5 \rightarrow$ cluster 3

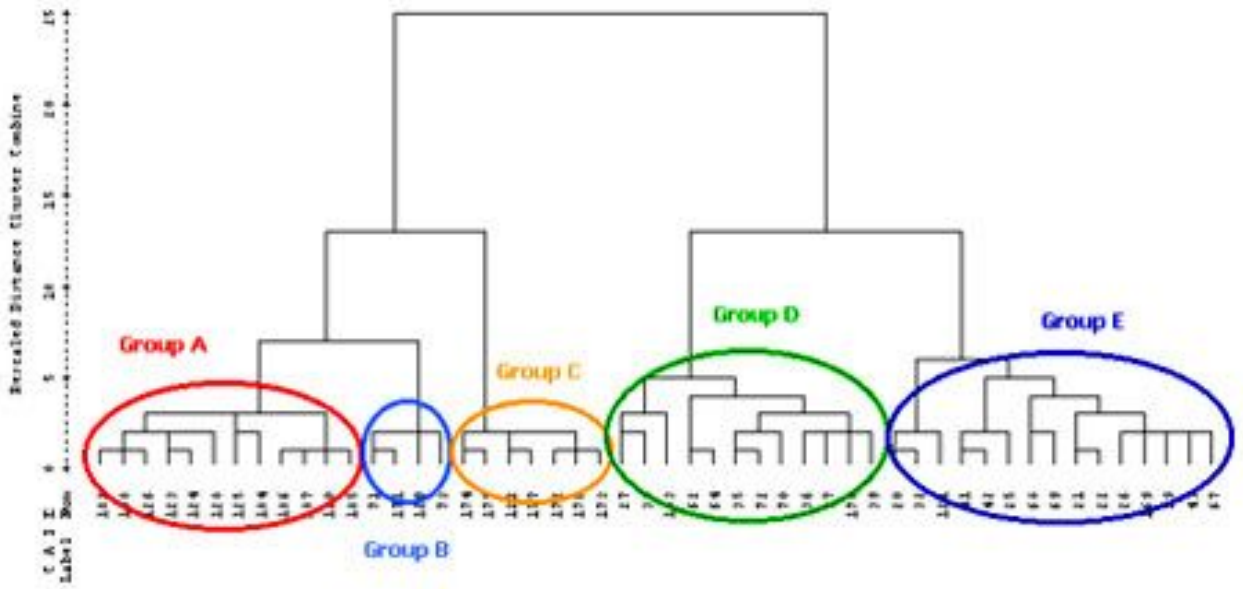
Use frequent values to represent cluster

- This method is mainly for clustering of categorical data (e.g., *k*-modes clustering).
- Main method used in text clustering, where a small set of frequent words in each cluster is selected to represent the cluster.


Categorical
Elementary
Middle
High School
K
1
2
3
4
5
6
7
8
9S1
9S2
10S1
10S2
11S1
11S2
12S1
12S2
Dropout
Took ACT
Female
District A
Z-Score
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
No Data
1.14
0.76
0.38
0

Clustering Dolphins:
Pie charts of the relative frequencies of specific events within the cluster categories and hierarchical Cluster Analysis of the "Behaviours"set





http://www.tethys.org/tethys/tethys-research/cetacean-sanctuary-project/results-slope-area/

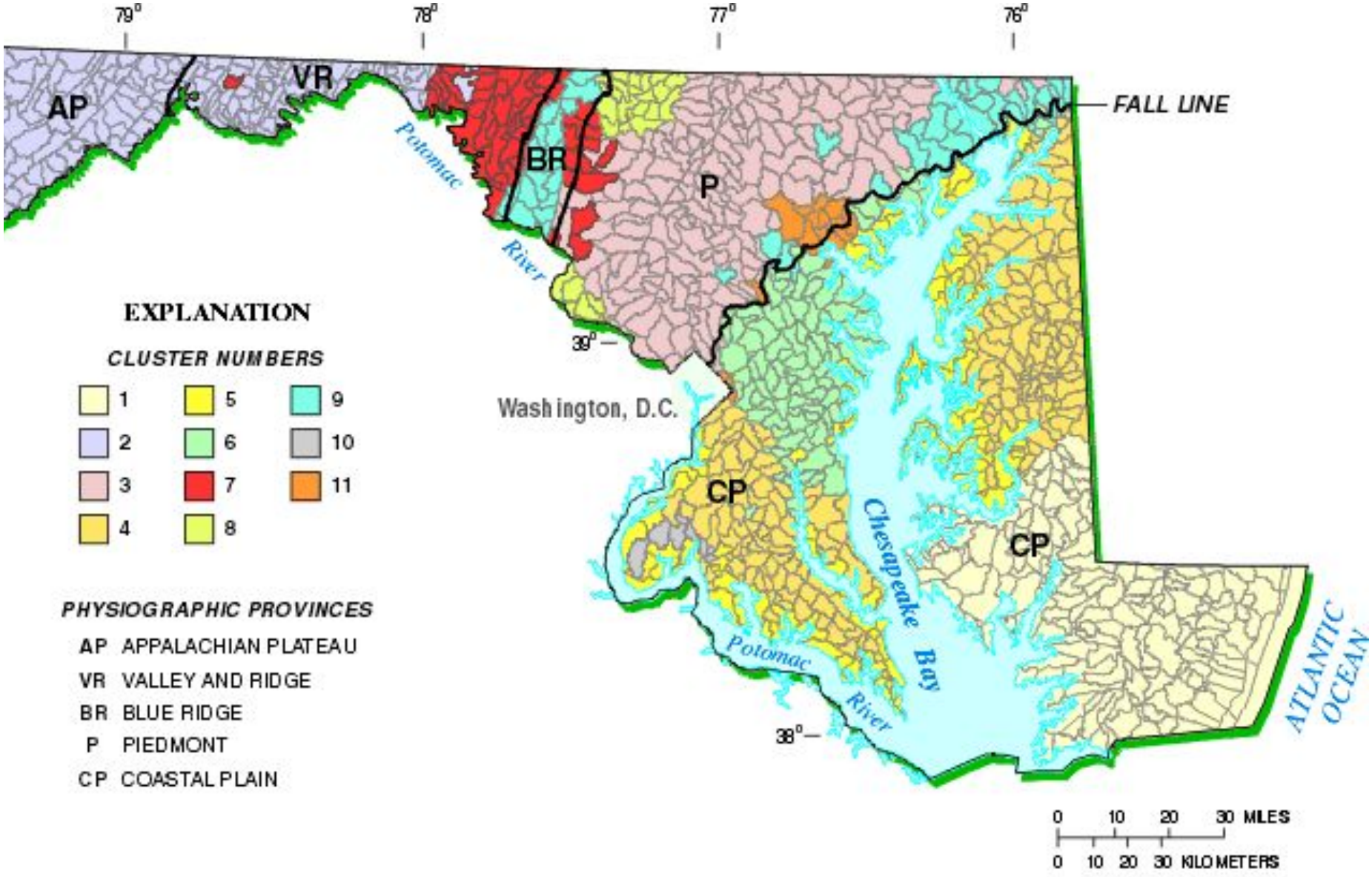


Figure 5. *Results of cluster analysis as applied to geographic data sets for defining hydrochemical response units (HRUs).*

Critical Issues

- Initialization!
- How to determine the number of clusters?
- How to handle outliers?
- How to handle large data sets?
- How to handle high dimensional data?
  - Includes sparse data
- How to handle Mixed /heterogeneous data types?
- Nested
  - Attributes are related in a concept hierarchy
- Domain knowledge:
  - a few labels/constraints (semi-supervised clustering)
  - Cluster must have min size (constrained K-means)
- Need Reliable Clustering?
  - Clustering Ensembles

Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side if too much data + high dimensionality
- Sample data
  - Then use hierarchical clustering to determine initial centroids
- Select more than *K* initial centroids
  - and then select among these initial centroids in the end
  - Select most widely separated
- Ensemble Clustering
  - Produce many different clustering results
  - Combine into consensus
- Bisecting K-means
  - Not as susceptible to initialization issues
- Global Optimization
  - Simulated Annealing;
  - Genetic algorithms

How many clusters?

How many clusters?

6 Clusters

2 Clusters

4 Clusters

Cluster Stability Algorithm: Choose k with most stable cluster results

Algorithm 17.1: Clustering Stability Algorithm for Choosing k

ClusteringStability $(A, t, k^{\max}, \mathbf{D})$:

$n \leftarrow |\mathbf{D}|$

// Generate t samples

for $i = 1, 2, \cdots, t$ **do**

  $\mathbf{D}_i \leftarrow$ sample n points from $\mathbf{D}$ with replacement

// Generate clusterings for different values of k

for $i = 1, 2, \cdots, t$ **do**

  for $k = 2, 3, \cdots, k^{\max}$ **do**

    $\mathcal{C}_k(\mathbf{D}_i) \leftarrow$ cluster $\mathbf{D}_i$ into k clusters using algorithm A

// Compute mean difference between clusterings for each k

foreach *pair* $\mathbf{D}_i, \mathbf{D}_j$ *with* $j > i$ **do**

  $\mathbf{D}_{ij} \leftarrow \mathbf{D}_i \cap \mathbf{D}_j$ // create common dataset using (17.30)

  for $k = 2, 3, \cdots, k^{\max}$ **do**

    $d_{ij}(k) \leftarrow d\big(\mathcal{C}_k(\mathbf{D}_i), \mathcal{C}_k(\mathbf{D}_j), \mathbf{D}_{ij}\big)$ // distance between clusterings

for $k = 2, 3, \cdots, k^{\max}$ **do**

  $\mu_d(k) \leftarrow \frac{2}{t(t-1)} \sum_{i=1}^{t} \sum_{j>i} d_{ij}(k)$ // expected pair-wise distance

// Choose best k

$k^* \leftarrow \arg\min_k \{\mu_d(k)\}$

Gap Statistic

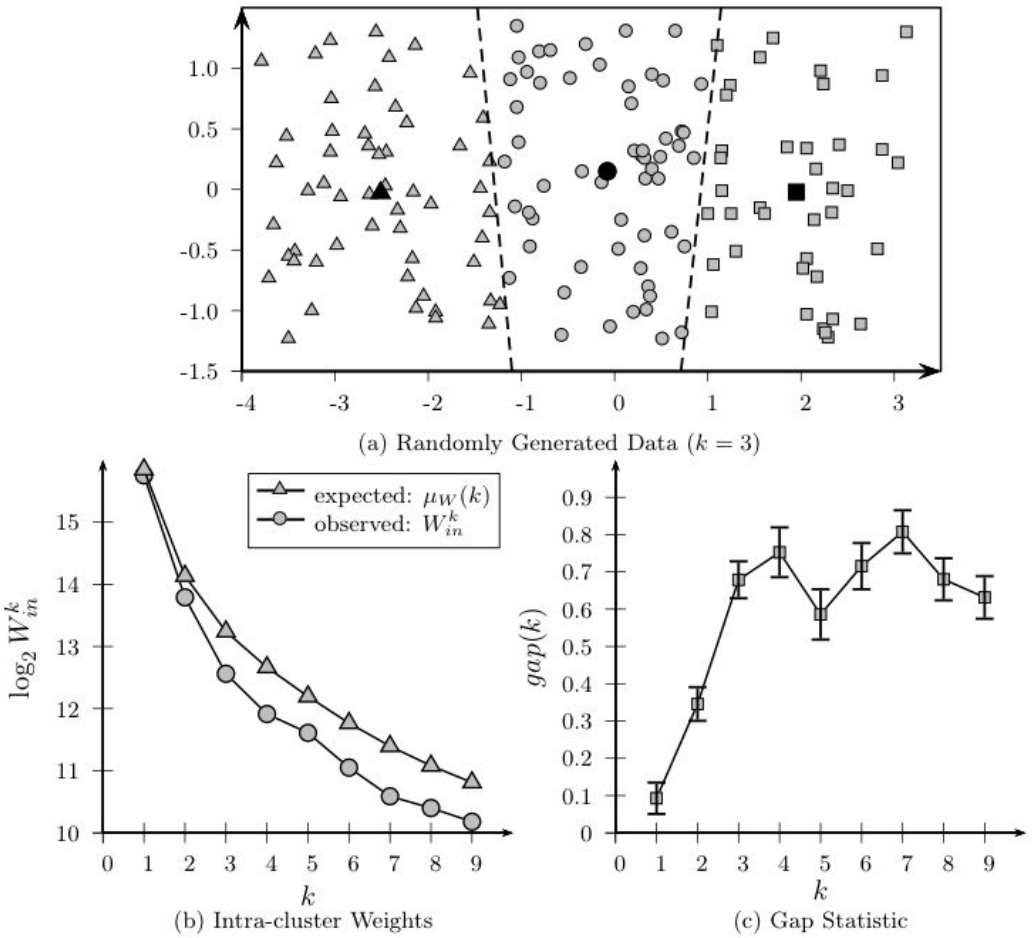
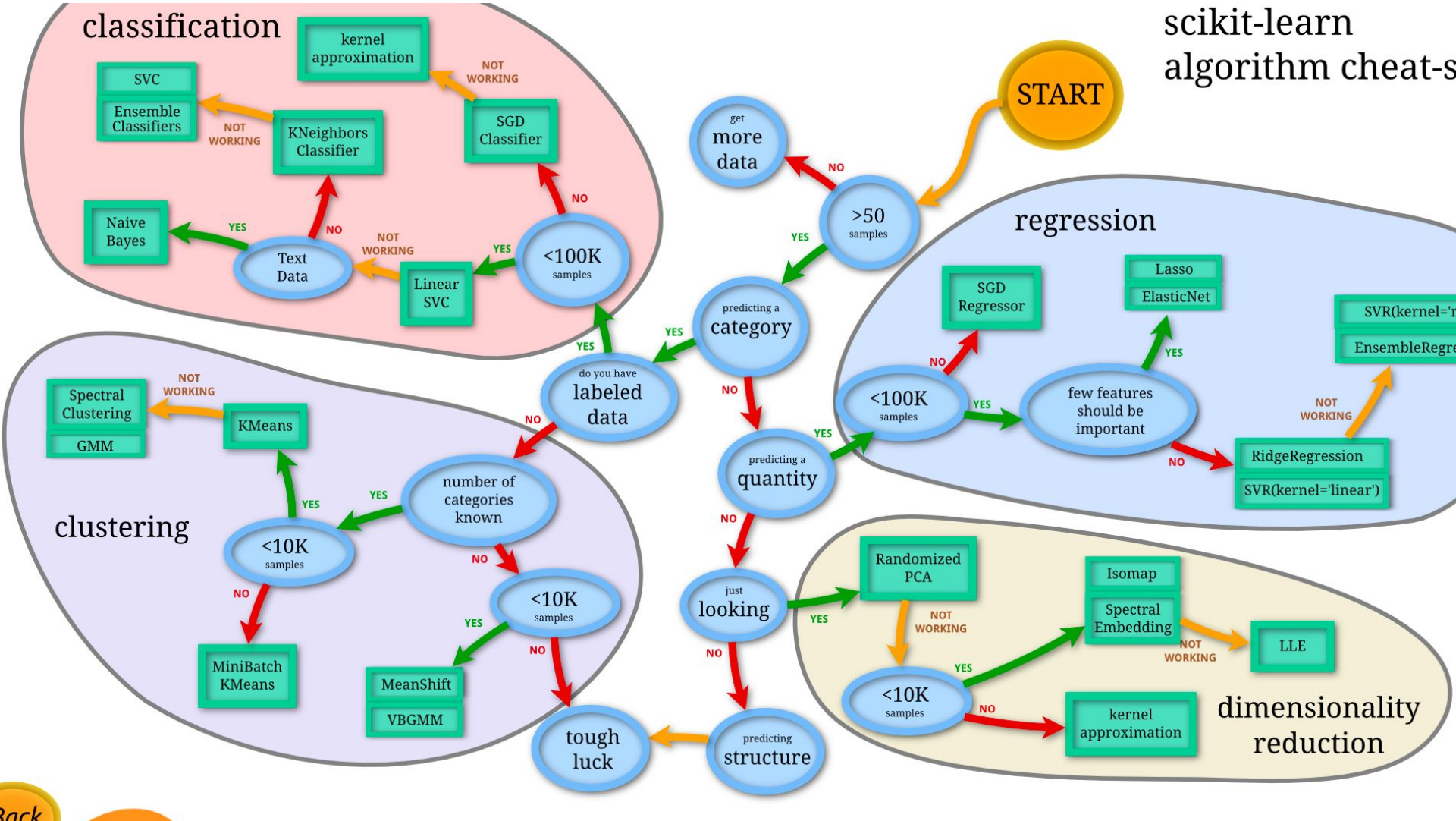


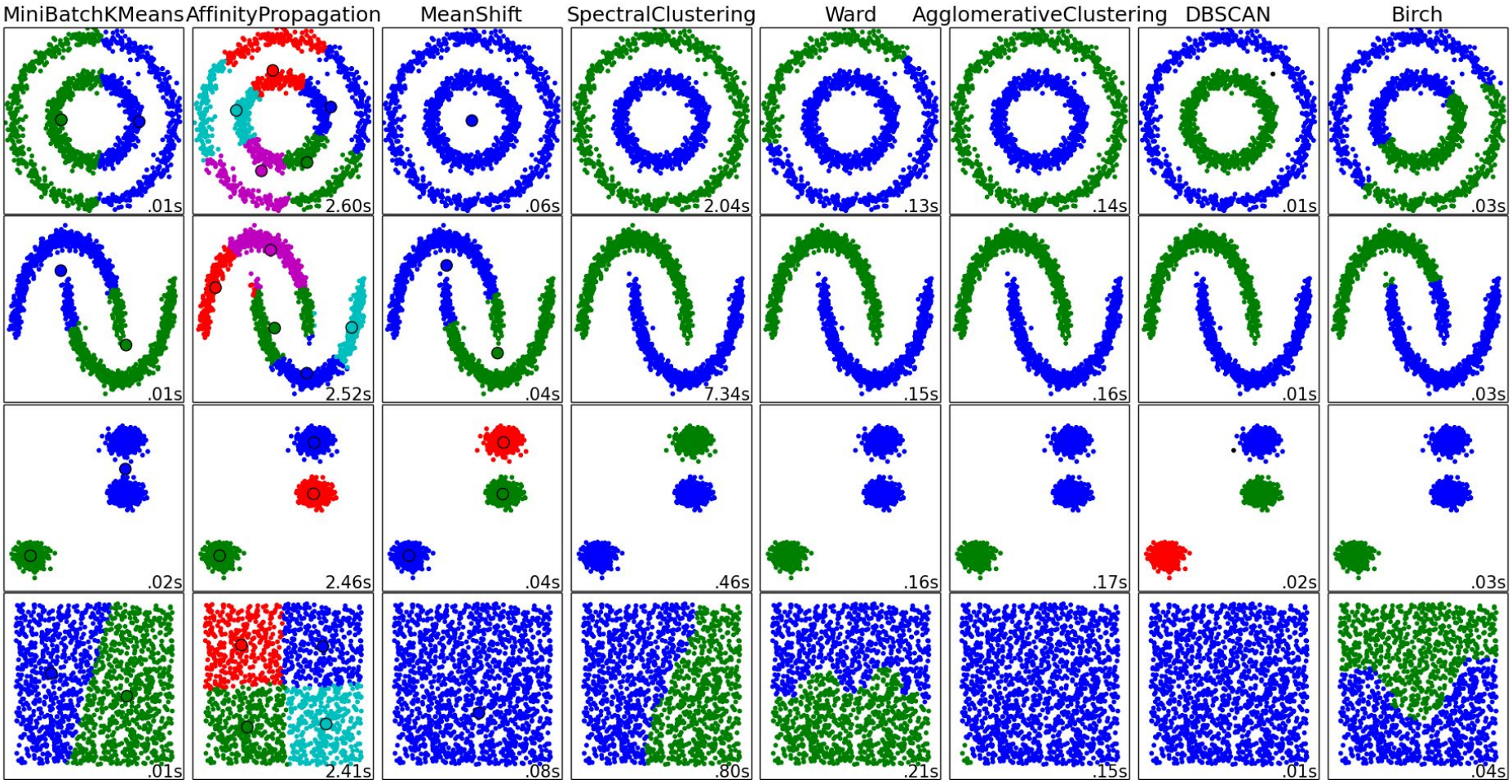
Figure 17.5: Gap Statistic: (a) Randomly generated data, (b) intra-cluster weights for different k, and (c) gap statistic as a function of k.

Python Specific Info

- Each clustering algorithm comes in **two variants**:
  1. **a class**, that implements the fit method to **learn the clusters** on train data, and
  2. **a function**, that, given train data, **returns an array of integer labels** corresponding to the different clusters.
- For the class, the labels over the training data can be found in the labels_ attribute.


scikit-learn
algorithm cheat-s
START
get more data
NO
>50 samples
YES
predicting a category
YES
do you have labeled data
YES
<100K samples
YES
Linear SVC
NOT WORKING
Text Data
NO
Naive Bayes
YES
KNeighbors Classifier
NO
SGD Classifier
NO
kernel approximation
NOT WORKING
SVC
Ensemble Classifiers
NOT WORKING
classification
NO
predicting a quantity
YES
<100K samples
NO
SGD Regressor
YES
few features should be important
YES
Lasso
ElasticNet
NO
RidgeRegression
SVR(kernel='linear')
NOT WORKING
SVR(kernel='r
EnsembleRegre
regression
NO
number of categories known
YES
<10K samples
YES
KMeans
NOT WORKING
Spectral Clustering
GMM
NO
MiniBatch KMeans
NO
<10K samples
YES
MeanShift
VBGMM
NO
clustering
NO
just looking
YES
Randomized PCA
NOT WORKING
<10K samples
YES
Isomap
Spectral Embedding
NOT WORKING
LLE
NO
kernel approximation
dimensionality reduction
NO
predicting structure
tough luck

Scikit-learn



Scikit-learn

Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large `n_samples`, medium `n_clusters` with *MiniBatch code*	General-purpose, even cluster size, flat geometry, not too many clusters	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with `n_samples`	Many clusters, uneven cluster size, non-flat geometry	Distances between points
Spectral clustering	number of clusters	Medium `n_samples`, small `n_clusters`	Few clusters, even cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints	Distances between points
Agglomerative clustering	number of clusters, linkage type, distance	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, non Euclidean distances	Any pairwise distance
DBSCAN	neighborhood size	Very large `n_samples`, medium `n_clusters`	Non-flat geometry, uneven cluster sizes	Distances between nearest points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation	Mahalanobis distances to centers
Birch	branching factor, threshold, optional global clusterer.	Large `n_clusters` and `n_samples`	Large dataset, outlier removal, data reduction.	Euclidean distance between points

Initializing K-means

- K-means can converge to a local minimum, depending on the initialization of the centroids.
- As a result, the computation is often done several times, with different initializations of the centroids.
- k-means++ initialization scheme: method to help address this issue
  - implemented in scikit-learn
  - use the init='kmeans++' parameter.
  - This initializes the centroids to be (generally) distant from each other,
  - leading to provably better results than random initialization, as shown in the reference.

"k-means++: The advantages of careful seeding" Arthur, David, and Sergei Vassilvitskii, Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics (2007)

Mini Batch K-Means

- The MiniBatchKMeans is a variant of the KMeans algorithm which uses mini-batches to reduce the computation time
- while still attempting to optimize the same objective function.
- Mini-batches = subsets of the input data, randomly sampled in each training iteration
- For each sample in the mini-batch, the assigned centroid is updated by taking the streaming average of the sample and all previous samples assigned to that centroid.
- converges faster than KMeans, but the quality of the results is reduced.

Clustering text documents using k-means

- A good example: http://scikit-learn.org/stable/auto_examples/text/document_clustering.html#example-text-document-clustering-py
- 2 feature extraction methods can be used in this example:
  - **TfidfVectorizer**
  - **HashingVectorizer**
- Word count vectors are then normalized to each have L2-norm = 1
  - projected to the Euclidean unit-ball
    => Spherical K-Means
- 2 algorithms are demoed:
  - ordinary **k-means** and
  - more scalable **minibatch k-means**.
- **Several runs** with independent random initialization might be necessary to get a good convergence.
  - **Or use kmeans++**

Clustering text documents using k-means (Feature extraction & transformation details)

- **TfidfVectorizer** uses a in-memory vocabulary (a python dict) to map the most frequent words to features indices
  - hence compute a word occurrence frequency (**sparse**) matrix.
  - **IDF**: The word frequencies are then reweighted using the Inverse Document Frequency (IDF) vector collected feature-wise over the corpus.
    - reduces weight of words that are very common (occur in many documents)
    - They are not helpful to discriminate between clusters or classes
- **HashingVectorizer** hashes word occurrences to a fixed dimensional space, possibly with collisions.
  - HashingVectorizer **does not provide IDF weighting**.
  - **IDF weighting:** If needed, add it by pipelining its output to a **TfidfTransformer** instance.
- **SVD →**
  - reduces dimensionality
  - captures semantics (related words)

Validation

- http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation
  - Several cluster validation metrics including
    - RAND index,
    - Silhouette,
    - Mutual Information Scores,
    - Homogeneity, Compactness, V-Measure
  - Gap statistic also available

Density based (also good for spatial) clustering

- DBSCAN:
- http://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN

Large Data Sets that can't fit in memory: BIRCH

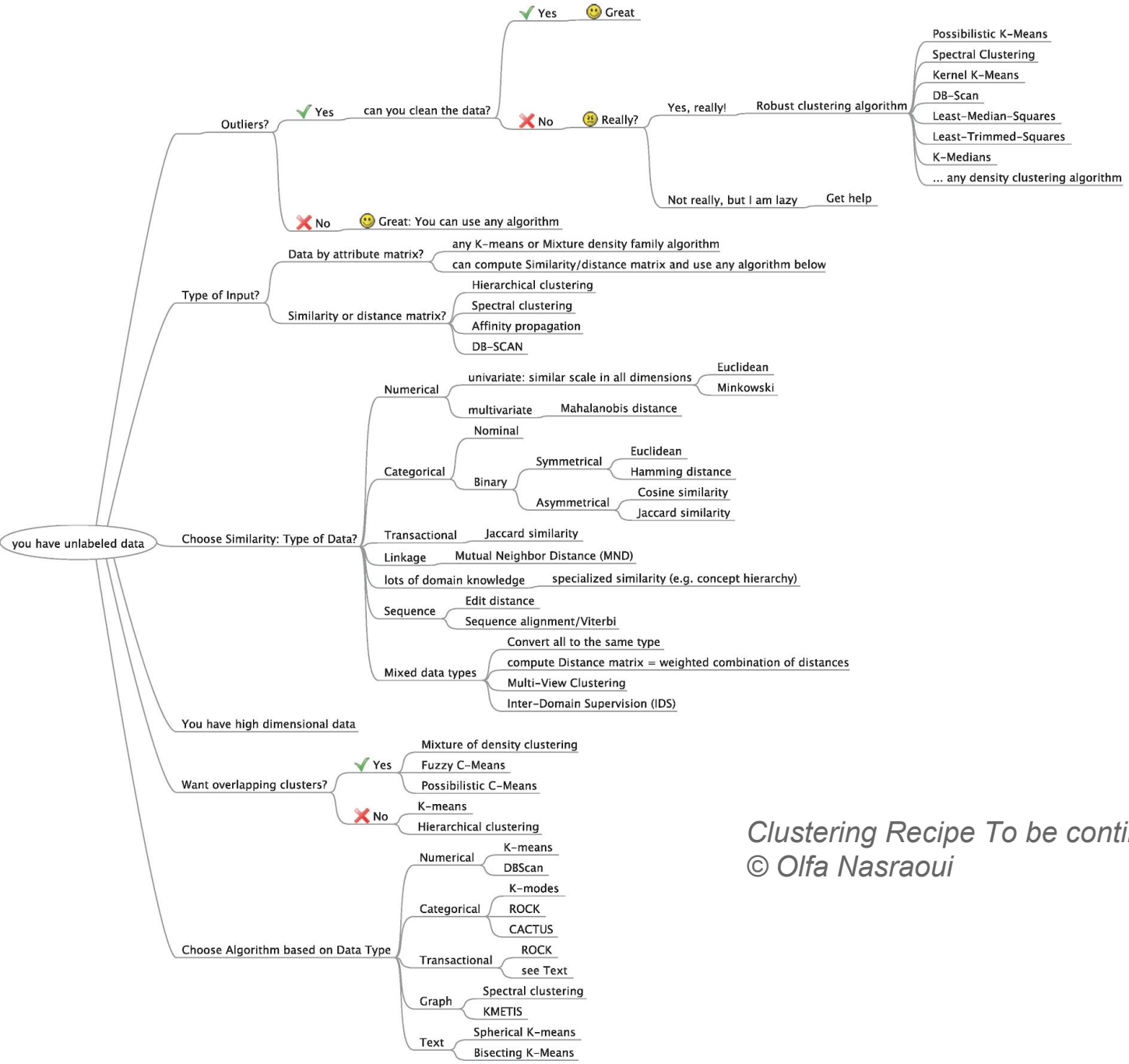
- Birch builds a tree called the Characteristic Feature Tree (CFT) for the given data.
- The data is essentially lossy compressed to a set of Characteristic Feature nodes (CF Nodes).
- The CF Nodes have a number of CF subclusters
- CF Subclusters located in the non-terminal CF Nodes can have CF Nodes as children.
- The CF Subclusters hold the necessary information for clustering which prevents the need to hold the entire input data in memory:
  - Number of samples in a subcluster.
  - Linear Sum - A n-dimensional vector holding the sum of all samples
  - Squared Sum - Sum of the squared L2 norm of all samples.
  - Centroids - To avoid recalculation linear sum / n_samples.
  - Squared norm of the centroids.

Birch or MiniBatchKMeans?

- Birch does not scale very well to high dimensional data.
- As a rule of thumb if n_features > 20
  - → use MiniBatchKMeans.
- Birch is more useful than MiniBatchKMeans:
  - If # data instances needs to be reduced,
  - or need a large number of subclusters either as a preprocessing step or otherwise

Unfinished Recipe for Clustering

Next slide :)



Clustering Recipe To be continued….