Clustering Data

- The **clustering problem**:

  Given a set of objects, find groups of similar objects

- **Cluster:** a collection of data objects
  - Similar to one another within the same cluster
  - Dissimilar to the objects in other clusters

- **What is similar?**

  Define appropriate metrics

- **Applications in**
  - marketing, image processing, biology

2

Clustering Methods

- **K-Means and K-medoids algorithms**
  - PAM, CLARA, CLARANS [Ng and Han, VLDB 1994]
- **Hierarchical algorithms**
  - CURE [Guha et al, SIGMOD 1998]
  - BIRCH [Zhang et al, SIGMOD 1996]
  - CHAMELEON [IEEE Computer, 1999]
- **Density based algorithms**
  - DENCLUE [Hinneburg, Keim, KDD 1998]
  - DBSCAN [Ester et al, KDD 96]
- **Subspace Clustering**
  - CLIQUE [Agrawal et al, SIGMOD 1998]
  - PROCLUS [Agrawal et al, SIGMOD 1999]
  - ORCLUS: [Aggarwal, and Yu, SIGMOD 2000]
  - DOC: [Procopiuc, Jones, Agarwal, and Murali, SIGMOD, 2002]

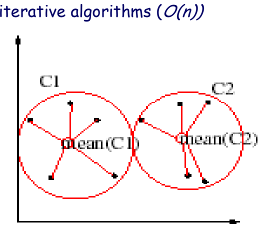
3

K-Means and K-Medoids algorithms

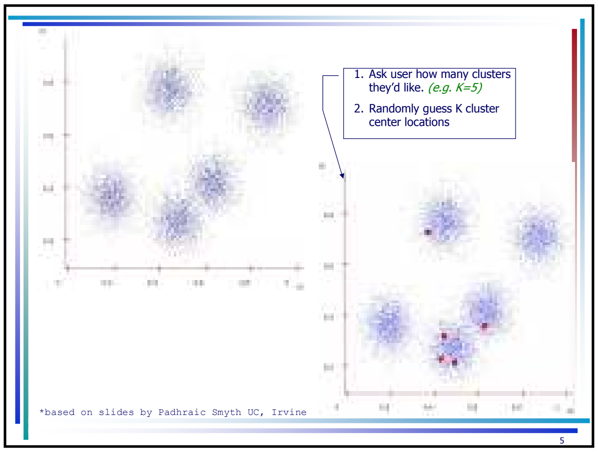
- Minimizes the sum of square distances of points to cluster representative

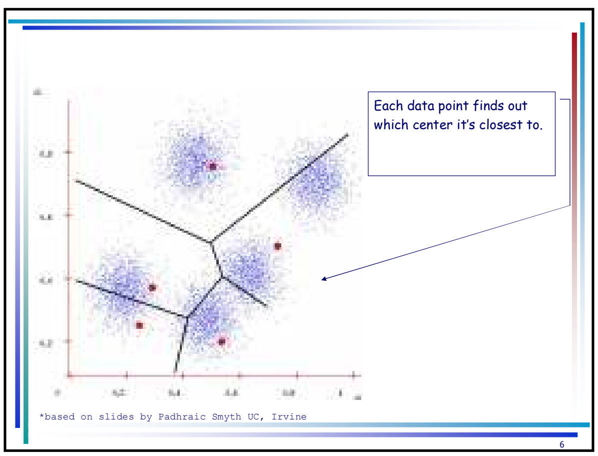
$$E_K = \sum_k \left\| x_k - m_{c(x_k)} \right\|^2$$

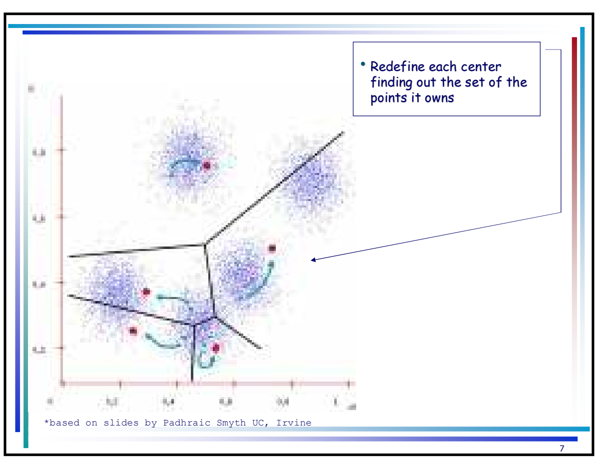
- Efficient iterative algorithms (*O(n)*)



4

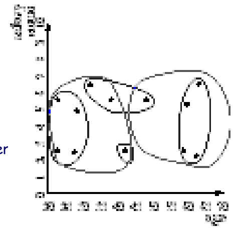

1. Ask user how many clusters they'd like. (e.g. K=5)
2. Randomly guess K cluster center locations
*based on slides by Padhraic Smyth UC, Irvine
5


Each data point finds out which center it's closest to.
*based on slides by Padhraic Smyth UC, Irvine
6


• Redefine each center finding out the set of the points it owns
*based on slides by Padhraic Smyth UC, Irvine
7

Problems with K-Means type algorithms

- <u>Advantages</u>
  - Relatively efficient: ***O(tkn)***,
  - where *n* is the number of objects, ***k*** is the number of clusters, and ***t*** is the number of iterations.
    Normally, ***k***, ***t*** << ***n***.
  - Often terminates at a local optimum.
- <u>Problems</u>
  - Clusters are approximately spherical
  - Unable to handle noisy data and outlier
  - High dimensionality may be a problem
  - The value of *k* is an input parameter



8

Spectral Clustering (I)

- **Algorithms that cluster points using eigenvectors** of matrices derived from the data
- Obtain data representation in the **low-dimensional space** that can be easily clustered
- Variety of methods that use the eigenvectors differently
  [Ng, Jordan, Weiss. NIPS 2001]
  [Belkin, Niyogi, NIPS 2001]
  [Dhillon, KDD 2001]
  [Bach, Jordan NIPS 2003]
  [Kamvar, Klein, Manning. IJCAI 2003]
  [Jin, Ding, Kang, NIPS 2005]

9

Spectral Clustering methods

- **Method #1**
  - Partition using only one eigenvector at a time
  - Use procedure recursively
    - **Example:** Image Segmentation
- **Method #2**
  - Use ***k*** eigenvectors (***k*** chosen by user)
  - Directly compute ***k***-way partitioning
  - Experimentally it has been seen to be "better" ([Ng, Jordan, Weiss. NIPS 2001][Bach, Jordan, NIPS '03]).

10

Kernel-based k-means clustering
(Dhillon et al., 2004)

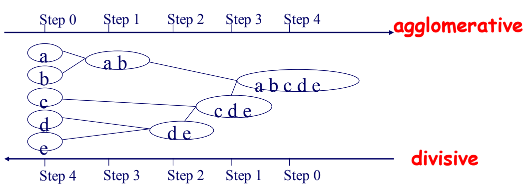
- Data not **linearly separable**
- **Transform data to high-dimensional space** using kernel
  - φ a function that maps X to a high dimensional space
- Use the kernel trick to evaluate the dot products:
  - a kernel function **k (x, y)** computes **φ(x)·φ(y)**
- cluster kernel similarity matrix using **weighted kernel K-Means.**
- The goal is to minimize the following objective function:

$$J\left(\{\pi_c\}_{c=1}^{k}\right)=\sum_{c=1}^{k}\sum_{x_i\in\pi_c}\alpha_i\left\|\varphi(x_i)-m_c\right\|^2$$

$$\text{where}\quad m_c=\frac{\sum_{x_i\in\pi_c}\alpha_i\varphi(x_i)}{\sum_{x_i\in\pi_c}\alpha_i}$$

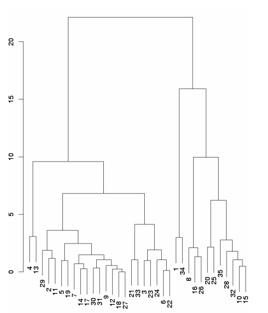
Hierarchical Clustering

- **Two basic approaches:**
  - merging smaller clusters into larger ones **(agglomerative)**,
  - splitting larger clusters **(divisive)**
- visualize both via **"dendograms"**
  - ✓shows nesting structure
  - ✓merges or splits = tree nodes



Hierarchical Clustering: Complexity

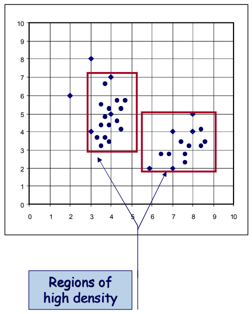
- **Quadratic algorithms**
- **Running time** can be improved using sampling [Guha et al, SIGMOD 1998] or using the triangle inequality (when it holds)



*based on slides by Padhraic Smyth UC, Irvine

Density-based Algorithms

- **Clusters** are **regions of space which have a high density** of points
- Clusters can have **arbitrary shapes**



14

Clustering High Dimensional Data

- Fundamental to all clustering techniques is the choice of distance measure between data points;

$$D(x_i, x_j) = \sum_{k=1}^{q} (x_{ik} - x_{jk})^2$$

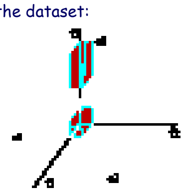
- **Assumption**: All features are **equally important**;
- Such approaches fail in high dimensional spaces
- Feature selection (Dy and Brodley, 2000)

  Dimensionality Reduction

15

Applying Dimensionality Reduction Techniques

Dimensionality reduction techniques (such as **Singular Value Decomposition**) can provide a solution by reducing the dimensionality of the dataset:

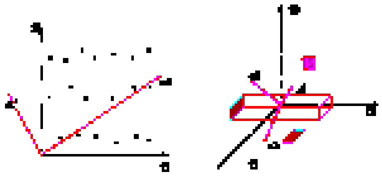


Drawbacks:

- The new dimensions may be difficult to interpret
- They don't improve the clustering in all cases

16

Applying Dimensionality Reduction Techniques



Different dimensions may be relevant to different clusters

In General: Clusters may exist in different subspaces, comprised of different combinations of features

17

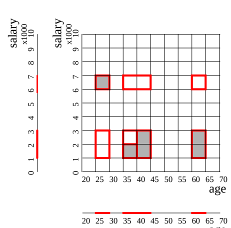
Subspace clustering

- **Subspace clustering** addresses the problems that arise from high dimensionality of data
  - It finds clusters in subspaces: subsets of the attributes
- Density based techniques
  - CLIQUE: Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD'98)
  - DOC: Procopiuc, Jones, Agarwal, and Murali, (SIGMOD, 2002)
- Iterative algorithms
  - PROCLUS: Agrawal, Procopiuc, Wolf, Yu, Park (SIGMOD'99)
  - ORCLUS: Aggarwal, and Yu (SIGMOD 2000).

18

Subspace clustering

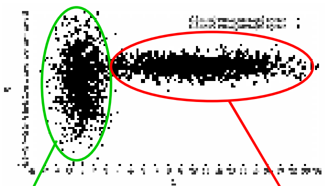
- **Density based clusters:** find dense areas in subspaces
- Identifying the right sets of attributes is hard
- Assuming a global threshold allows bottom-up algorithms
- Constrained monotone search in a lattice space



19

Locally Adaptive Clustering

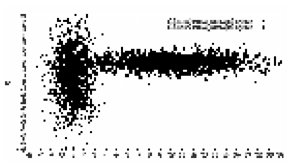
Each cluster is characterized by different attribute weights
(Friedman and Meulman 2002, Domeniconi 2004)



$(w_{1x}, w_{1y}),\ w_{1x} > w_{1y}$ $(w_{2x}, w_{2y}),\ w_{2y} > w_{2x}$

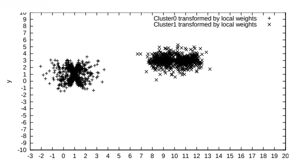
20

Locally Adaptive Clustering : Example



before local
transformations

after local
transformations



21

LAC

[C. Domeniconi et al SDM04]

- **Computing the weights:**

X_{ji} : average squared distance along dimension i of points in S_j from c_j

$$X_{ji} = \frac{1}{|S_j|} \sum_{x \in S_j} (c_{ji} - x_i)^2$$

$$w_{ji} = \frac{e^{-X_{ji}}}{\sum_l e^{-X_{jl}}}$$

Exponential weighting scheme

Result:

$w_1, w_2, \cdots, w_k$

A weight vector for each cluster

22

Convergence of LAC

The ***LAC algorithm*** converges to a local minimum of the error function:

$$E(C,W) = \sum_{j=1}^{k}\sum_{i=1}^{q} w_{ji} e^{-X_{ji}}$$

subject to the constraints $\sum_{i=1}^{q} w_{ji}^2 = 1 \quad \forall j$

$$C = [c_1 \cdots c_k] \qquad W = [w_1 \cdots w_k]$$

EM-like convergence:

Hidden variables: assignments of points to centroids (S_j)

E-step: find the values of S_j given w_{ji}, c_{ji}

M-step: find w_{ji}, c_{ji} that minimize $E(C,W)$ given current estimates S_j.

Semi-Supervised Clustering

- **Clustering** is applicable in many real life scenarios
  - there is typically a large amount of **unlabeled data** available.
- The use of **user input** is critical for
  - the success of the clustering process
  - the evaluation of the clustering accuracy.
- **User input** is given as
  - Labeled data
  - Constraints

Learning approaches that use
labeled data/constraints + ***unlabeled*** data
have recently attracted the interest of researchers

Motivating semi-supervised learning

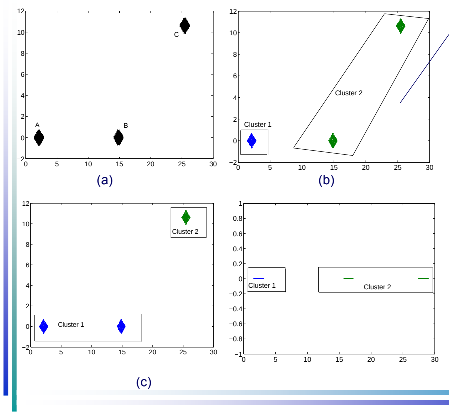
- **Data are correlated.** To recognize clusters, a distance function should reflect such correlations.
- **Different attributes may have different degree of relevance** depending on the application / user requirements

☹ A clustering algorithm does not provide the criterion to be used.

Semi-supervised algorithms: Define clusters taking into account

- ***labeled data or constraints***

if we have "labels" we will convert them to "constraints"



a user may want the points in B and C to belong to the same cluster

→ The right clustering may depend on the user's perspective.

→ Fully automatic techniques are very limited in addressing this problem

Clustering under constraints

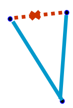
- Use **constraints** to
  - learn a **distance function**
    - Points surrounding a pair of **must-link/cannot-link** points should be close to/far from each other
  - guide the algorithm to a useful solution
    - Two points should be in the same/different clusters

Defining the constraints

- A set of points $X = \{x_1, \ldots, x_n\}$ on which sets of ***must-link(S)*** and ***cannot-link constraints(D)*** have been defined.
- **Must-link constraints**
  - **S:** $\{(x_i, x_j)$ in $X\}$: x_i and x_j **should belong** to the same cluster
- **Cannot-link constraints**
  - **D:** $\{(x_i, x_j)$ in $X\}$: x_i and x_j **cannot belong** to the same cluster
- **Conditional constraints**
  - δ-constraint and ε-constraint

Clustering with constraints: Feasibility issues

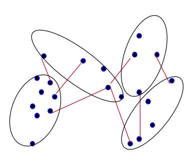
- **Constraints** provide information that should be satisfied.
- Options for **constraint-based clustering**
  - **Satisfy all constraints**
    - **Not always possible:** A with B, B with C, C not with A.



  - **Satisfy as many constraints as possible**

Clustering with constraints: Feasibility issues

- Any combination of constraints involving <u>cannot-link constraints</u> is generally computationally intractable (Davidson & Ravi, ISMB 2000),
  - Reduction to k-colorability problem: Can you cluster (color) the graph with the cannot-link edges using k colors (clusters)?



Feasibility under *ML* and ε

ε-constraint: Any node x should have an ε-neighbor in its cluster (another node y such that **D(x,y)≤ ε**)

$S' = \{x \in S : x$ *does **not** have an* ε *neighbor*$\} = \{s_5, s_6\}$

Each of these should be in their own cluster

x_1 x_2 x_3 x_4 $\boxed{x_5 \quad x_6}$

ML(x_1,x_2), ML(x_3,x_4), ML(x_4,x_5)

Compute the ***Transitive Closure*** on ML={CC_1 … CC_r}

$\boxed{x_1 \quad x_2}$ $\boxed{x_3 \quad x_4 \quad x_5}$ x_6

Infeasible: iff $\exists i, j : x_i \in CC_j, x_i \in S'$

*S. Basu, I. Davidson,turorial ICDM 2005

Clustering based on constraints

- **Algorithm specific approaches**
  - **Incorporate constraints into the clustering algorithm**
    - COP K-Means (Wagstaff et al, 2001)
    - Hierarchical clustering (I. Davidson, S. Ravi, 2005)
  - **Incorporate metric learning into the algorithm**
    - MPCK-Means (Bilenko et al 2004)
    - HMRF K-Means (Basu et al 2004)
- **Learning a distance metric** (Xing et al. '02)
- **Kernel-based constrained clustering** (Kulis et al.'05)

32

COP K-Means (I)

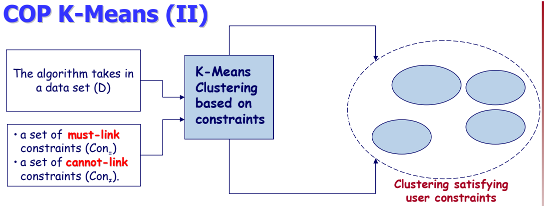
[Wagstaff et al, 2001]

- Semi-supervised variants of K-Means
- **Constraints:** Initial background knowledge
- **Must-link** & **Cannot-link** constraints are used in the clustering process
  - Generate a partition that satisfies all the given constraints

K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl. Constrained k-means clustering with background knowledge. In *ICML*, pages 577-584, 2001.

33

COP K-Means (II)



- **When updating cluster assignments,**
  - we ensure that none of the specified constraints are violated.
- **Assign each point d_i to its closest cluster C_j.** This will succeed unless a constraint would be violated.
  - If there is another point $d_=$ that must be assigned to the same cluster as d, but that is already in some other cluster, or
  - there is another point $d_{\neq}$ that cannot be grouped with *d* but is already in *C*, then *d* cannot be placed in *C*.
- **Constraints are never broken**; if a legal cluster cannot be found for d, the empty partition (f_g) is returned.

34

Hierarchical Clustering based on constraints

[I. Davidson, S. Ravi, 2005]

Instance: A set S of nodes, the (symmetric) distance **d(x,y)≥0** for each pair of nodes x and y and a collection C of constraints

- **Question:** Can we create a dendrogram for S so that all the constraints in C are satisfied?

Davidson I. and Ravi, S. S. "Hierarchical Clustering with Constraints: Theory and Practice", *In PKDD 2005*

Constraints and Irreducible Clusterings

- A **feasible clustering** $C=\{C_1, C_2, \ldots, C_k\}$ of a set S is irreducible if no pair of clusters in C can be merged to obtain a feasible clustering with k-1 clusters.

If mergers are not done correctly, the dendrogram may stop prematurely

- $X=\{x_1, x_2, \ldots, x_k\}$, $Y=\{y_1, y_2, \ldots, y_k\}$, $Z=\{z_1, z_2, \ldots, z_k\}$, $W=\{w_1, w_2, \ldots, w_k\}$
- **CL-constraints**
  - $\forall\{x_i, x_j\}$, $i \neq j$
  - $\forall\{w_i, w_j\}$, $i \neq j$
  - $\forall\{y_i, z_j\}$, $i \leq j$, $j \leq i$

- Feasible clustering with 2k clusters: $\{x_1, y_1\}, \{x_2, y_2\}, \ldots, \{x_k, y_k\}, \{z_1, w_1\}, \{z_2, w_2\}, \ldots, \{z_k, w_k\}$

But then get stuck

- **Alternative is:**
- $\{x_1, w_1, y_1, y_2, \ldots, y_k\}, \{x_2, w_2, z_1, z_2, \ldots, z_k\}, \{x_3, w_3\}, \ldots, \{x_k, w_k\}$

MPCK-Means

[Bilenko et al 2004]

- **Incorporate metric learning directly into the clustering algorithm**
  - Unlabeled data influence the metric learning process
- **Objective function**
  - Sum of total square distances between the points and cluster centroids
  - Cost of violating the pair-wise constraints

M. Bilenko, S. Basu, R. Mooney. "Integrating Constraints and Metric Learning in Semi-supervised clustering. In Proceedings of the 21st ICML Conference, July 2004.

Unifying constraints and Metric learning

Generalized K-means distortion function,
Assumes each cluster is generated by a gaussian with covariance matrix A_{li}^{-1}

$$J_{mpckm} = \boxed{\sum_{x_i \in X} \left\| x_i - \mu_{l_i} \right\|^2_{Al_i} - \log(\det(A_{l_i}))} +$$

$$\boxed{\sum_{(x_i,x_j)\in M} w_{ij} f_M(x_i,x_j) 1[l_i \neq l_j]} + \boxed{\sum_{(x_i,x_j)\in C} \overline{w}_{ij} f_C(x_i,x_j) 1[l_i = l_j]}$$

Violation must-link constraints

Violation cannot-link constraints

Penalty functions

MPCK-Means approach

Initialization:

- Use neighborhoods derived from constraints to initialize clusters

Repeat until convergence (not guaranteed):

1. **E-step:**
   - **Assign** each point *x* to a cluster *to minimize*
     - distance of **x** from the cluster centroid + constraint violations
2. **M-step:**
   - **Estimate** cluster centroids *C* as means of each cluster
   - **Re-estimate** parameters *A (dimension weights)* to minimize constraint violations

Learning a distance metric based on user constraints

- The requirement is :
  - **learn the distance measure** to satisfy user constraints.
- **To simplify the problem consider the weighted Euclidean distance:**
  - different weights are assigned to different dimensions
  - Other formulations that map the points to a new space can be considered, but are significantly more complex to optimize

Distance Learning as Convex Optimization
[Xing et al. '02]

- **Goal:** **Learn a distance metric** between the points in X that satisfies the given constraints
- The problem reduces to the following **optimization problem :**

$$\boxed{\min_A \sum_{(x_i, x_j) \in ML} \|x_i - x_j\|_A^2}$$

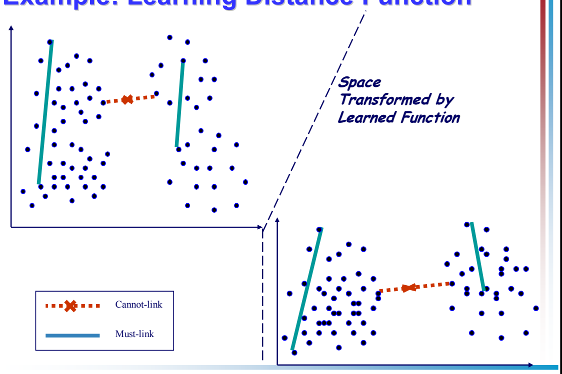
given that

$$\sum_{(x_i, x_j) \in CL} \|x_i - x_j\|_A \geq 1 \quad A \geq 0$$

E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *NIPS*, December 2002.

41

Example: Learning Distance Function



42

Learning Mahalanobis distance

Mahalanobis distance =
Euclidean distance parameterized by matrix *A*

$$|| x - y ||_A^2 = (x - y)^T A(x - y)$$

Typically ***A*** is diagonal

43

The Diagonal *A* Case

- Considering the case of learning ***a diagonal*** A
- we can solve the original ***optimization problem*** using Newton-Raphson to efficiently optimize the following

$$g(A) = \sum_{(x_i,x_j)\in ML} \|x_i - x_j\|_A^2 - \log\left(\sum_{(x_i,x_j)\in CL} \|x_i - x_j\|_A\right)$$

Use **Newton Raphson Technique**:

$x' = x - g(x)/g'(x)$

$A' = A - g(A).J^{-1}(A)$

$A \geq 0$

44

Kernel based Semi-supervised clustering

[Kulis et al.'05]

A non-linear transformation, φ

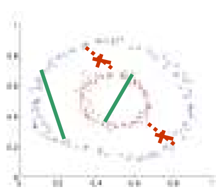
- maps data to a high dimensional space
- the data are expected to be more separable
- a kernel function **k (x, y)** computes **φ(x)·φ(y)**

The user gives constraints

The appropriate kernel is created based on constraints

$$J\left(\{\pi\}_{c=1}^{k}\right) = \sum_{c=1}^{k}\sum_{x_i\in\pi_c}\|\phi(x_i) - m_c\|^2 - \sum_{\substack{x_i,x_j\in ML \\ l_i = l_j}} w_{ij} + \sum_{\substack{x_i,x_j\in CL \\ l_i = l_j}} w_{ij}$$

Reward for constraint satisfaction



45

Semi-Supervised Kernel-KMeans

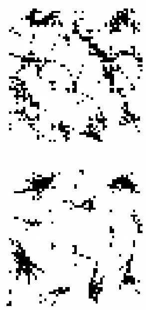
[Kulis et al.'05]

- **Algorithm:**
  - Constructs the appropriate kernel matrix from data and constraints
  - Runs weighted kernel K-Means
- **Input of the algorithm:** Kernel matrix
  - Kernel function on vector data or
  - Graph affinity matrix
- **Benefits:**
  - HMRF-KMeans and Spectral Clustering are special cases
  - Fast algorithm for constrained graph-based clustering
  - Kernels allow constrained clustering with non-linear cluster boundaries

46

Graph-based constrained clustering

- **Constrained graph clustering:**
  - minimize cut in input graph while maximally respecting a given set of constraints



Clustering using constraints and cluster validity criteria

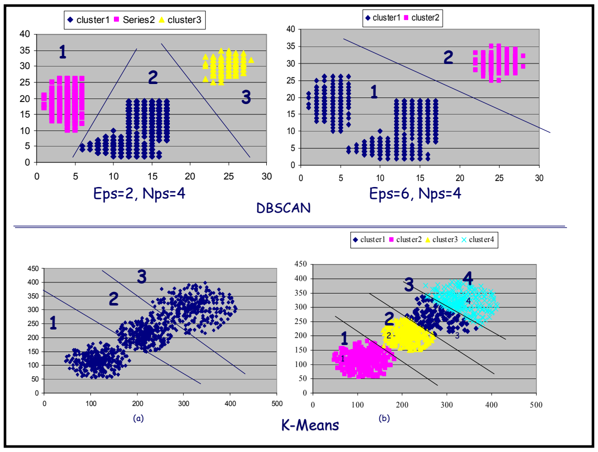
- **Different distance metrics may satisfy the same number of constraints**
- **One solution is to apply a different criterion that evaluates the resulting clustering to choose the right distance metric**
- **A general approach should:**
  - Learn an appropriate distance metric to satisfy the constraints
  - Determine the best clustering w.r.t the defined distance metric.

Cluster Validity

A problem we face in clustering is to

- ✓ define the **"best" partitioning** of a data set, i.e.
  - ✓ number of clusters that fits a data set,
  - ✓ capture the shape of clusters presenting in underlying data set

- The clustering results depend on
  - **the data set** (data distribution)
  - Initial **clustering assumptions**, algorithm **input parameters values**



S_Dbw cluster validity index [Halkidi, Vazirgiannis, ICDM'01]

- **SDbw**: a relative algorithm-independent validity index, based on
  - **S**cattering and **D**ensity **b**et**w**een clusters

Main features of the proposed approach

Validity index **S_Dbw**. Based on the features of the clusters:

- ✓ evaluates the resulting clustering as defined by the algorithm under consideration.
- ✓ selects for each algorithm the optimal set of input parameters with regards to the specific data set.

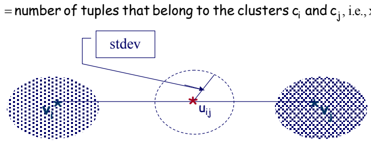
S_Dbw definition: Inter-cluster Density (ID)

Dens_bw: Average density in the area among clusters in relation with the density of the clusters

$$Dens\_bw(c) = \frac{1}{c \cdot (c-1)} \sum_{i=1}^{c} \left(\sum_{\substack{j=1 \\ i \neq j}}^{c} \frac{density(u_{ij})}{\max\{density(v_i), density(v_j)\}} \right),$$

$$\text{density}(u_{ij}) = \sum_{l=1}^{n_{ij}} f(x_l, u_{ij}), \quad f(x,u) = \begin{cases} 0, & \text{if } d(x,u) > \text{stdev} \\ 1, & \text{otherwise} \end{cases}$$

where n_{ij} = number of tuples that belong to the clusters c_i and c_j, i.e., $x_l \in c_i \cup c_j \subseteq S$



S_Dbw definition: Intra-cluster variance

Average scattering of clusters:

$$Scat(c) = \frac{\frac{1}{c}\sum_{i=1}^{c}\left\|\sigma(v_i)\right\|}{\left\|\sigma(X)\right\|}$$

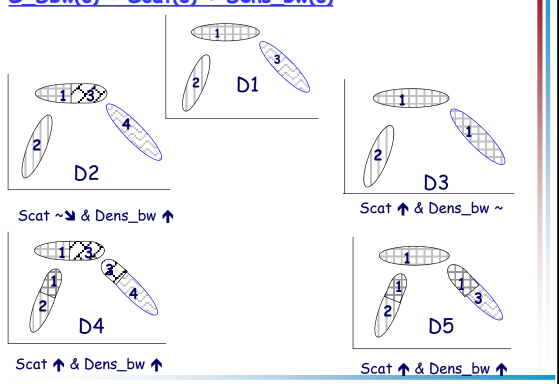
where $\sigma_x^p = \frac{1}{n}\sum_{k=1}^{n}\left(x_k^p - \overline{x}^p\right)^2$

where $\overline{x}^p$ is the p_{th} dimension of $\overline{X} = \frac{1}{n}\sum_{k=1}^{n} x_k, \forall x_k \in X$

$$\sigma_{v_i}^p = \sum_{k=1}^{n_i}\left(x_k^p - v_i^p\right)^2 \Big/ n_i$$

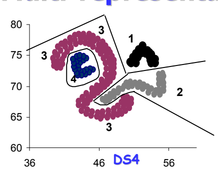
53

S_Dbw(c) = Scat(c) + Dens_bw(c)

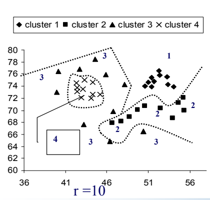


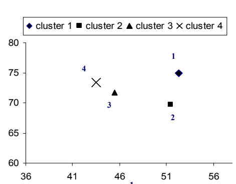
54

Multi-representatives vs. Single



a single representative point cannot efficiently represent the shape of clusters in DS4





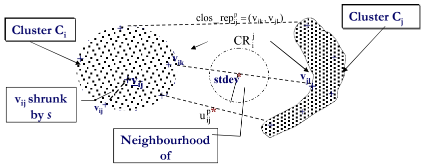
55

Respective Closest Representative points

For each pair of clusters (C_i, C_j) we find the set of closest representatives of C_j with respect to C_i :

for each v_{ik} in C_i = {(v_{ik}, v_{jl})| v_{jx} ∈ C_j and min(dist(v_{ik}, v_{jx}))}

RCR_{ij} = pruning(CR_i^j)



Respective Closest Representative points. The set of respective representative points of the clusters C_i and C_j is defined as the set of mutual closest representatives of the clusters under concern, i.e. RCR_{ij} = {(v_{ik}, v_{jl})| v_{ik} = closest_rep$_i$ (v_{jl}) and v_{jl} = closest_rep$_j$ (v_{ik})} i.e. $RCR_{ij} = CR_j^i \cap CR_j^i$

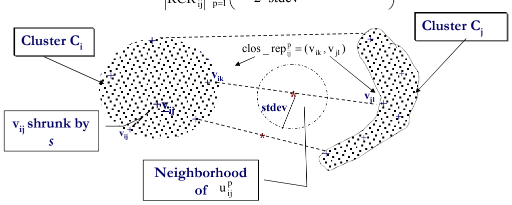
Pruning maintains only the meaningful pairs of representative points

56

Inter cluster density

Clusters' separation implies low density among them

$$\mathrm{Dens}(C_i, C_j) = \frac{1}{|RCR_{ij}|} \sum_{p=1}^{|CR_{ij}|} \left(\frac{d(clos\_rep_{ij}^p)}{2 \cdot stdev} \cdot \mathrm{density}(u_{ij}^p) \right)$$



$$\mathrm{Inter\_dens}(\mathbf{C}) = \frac{1}{c} \cdot \sum_{i=1}^{c} \max_{\substack{j=1,\dots,c \\ j \neq i}} \{\mathrm{Dens}(C_i, C_j)\}$$

57