CS6510
Applied Machine Learning

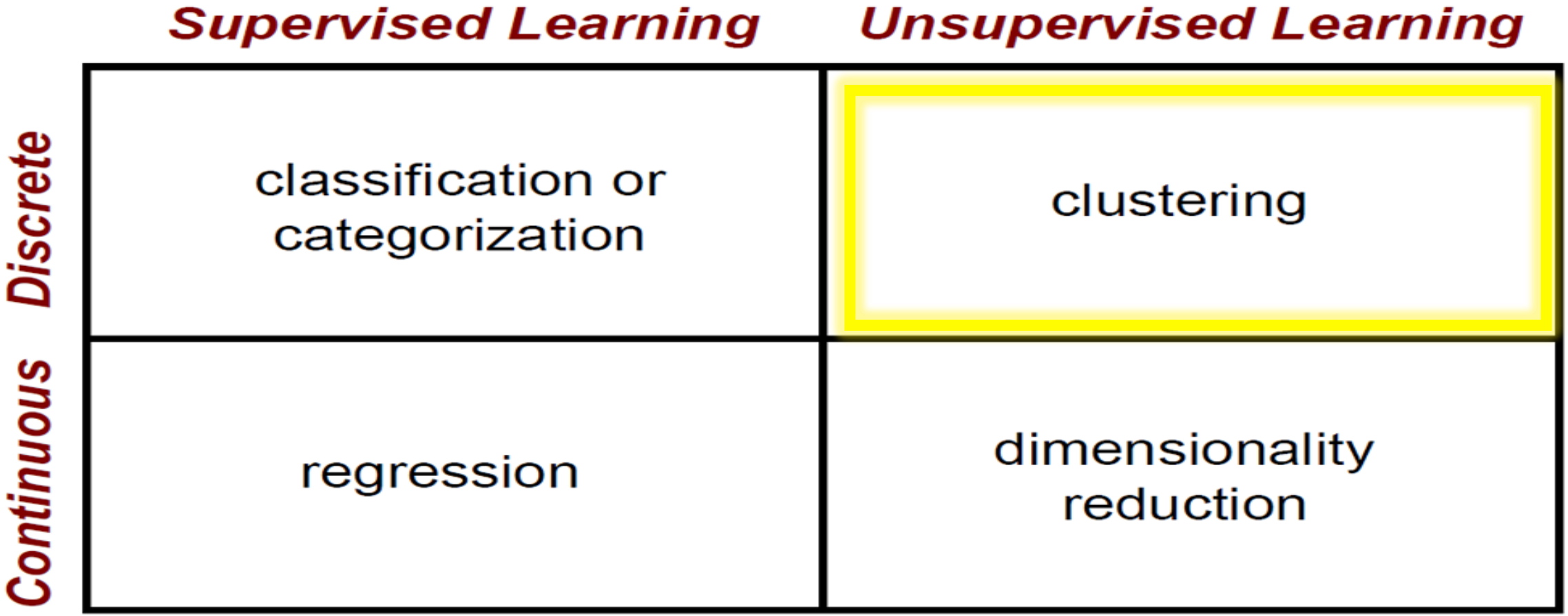
Clustering

14 Oct 2017

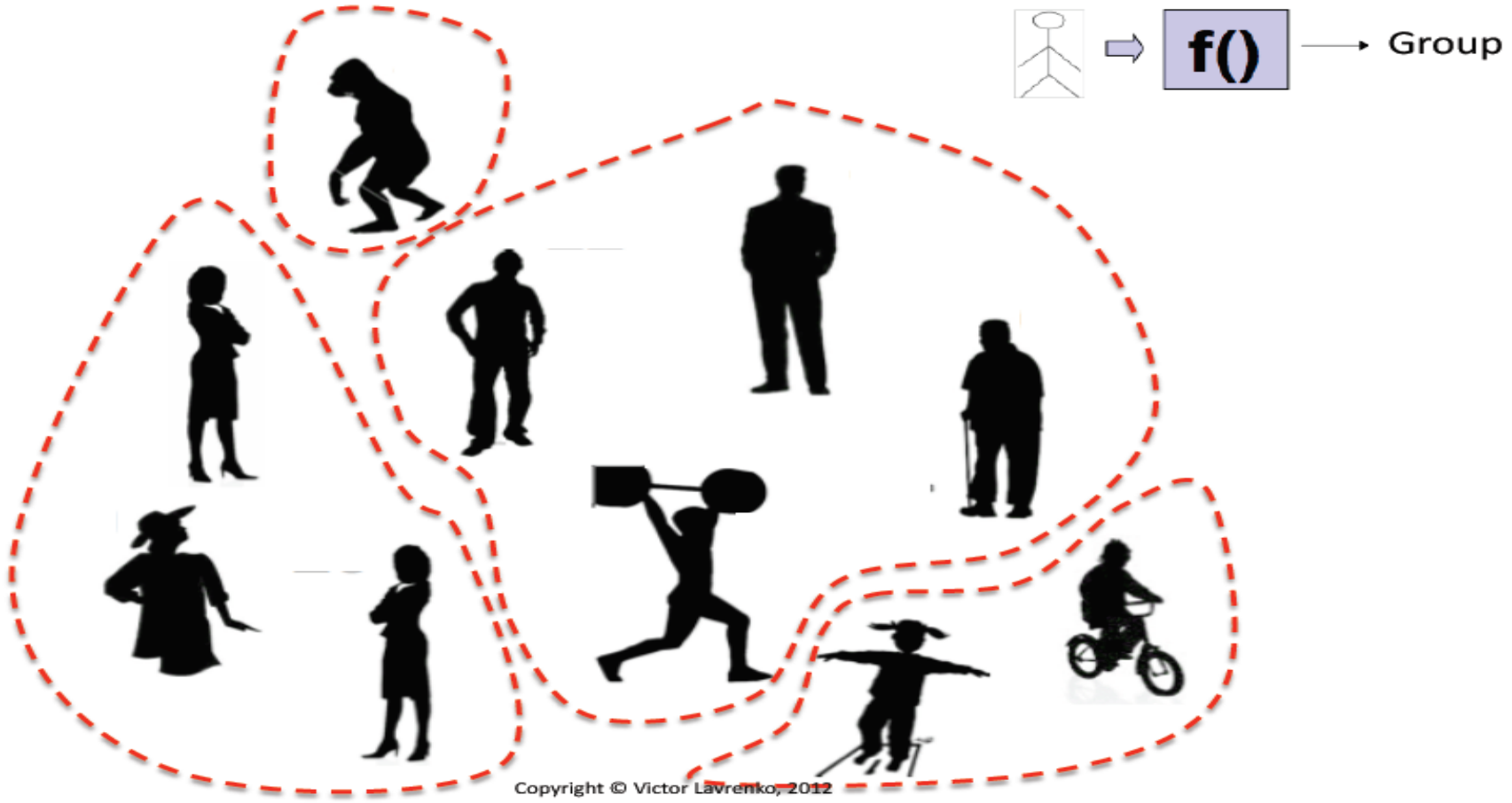
Vineeth N Balasubramanian

आई आई टी हैदराबाद
IIT Hyderabad

ML Problems


Supervised Learning
Unsupervised Learning
Discrete
Continuous
classification or categorization
clustering
regression
dimensionality reduction

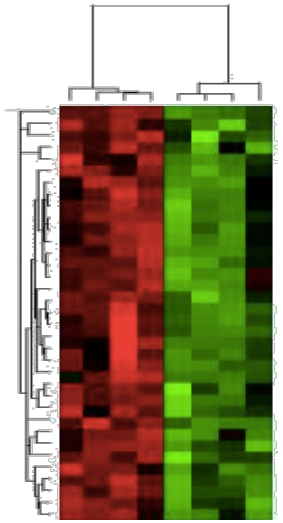
Clustering (Unsupervised Learning)

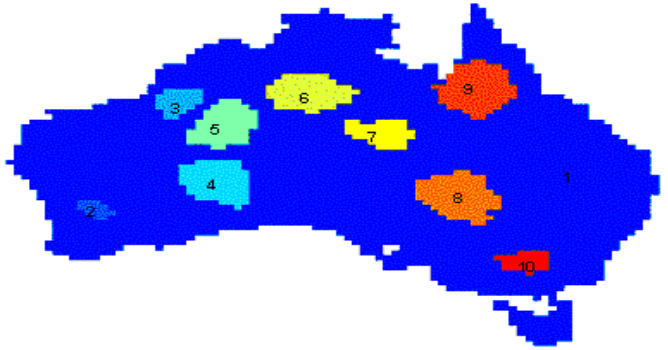


Where is Clustering used?

- **Understanding**
  - Group genes and proteins that have similar functionality, or group stocks with similar price fluctuations
- **Summarization**
  - Reduce the size of large data sets

More real-world applications?





Clustering precipitation in Australia

Where is Clustering Used?

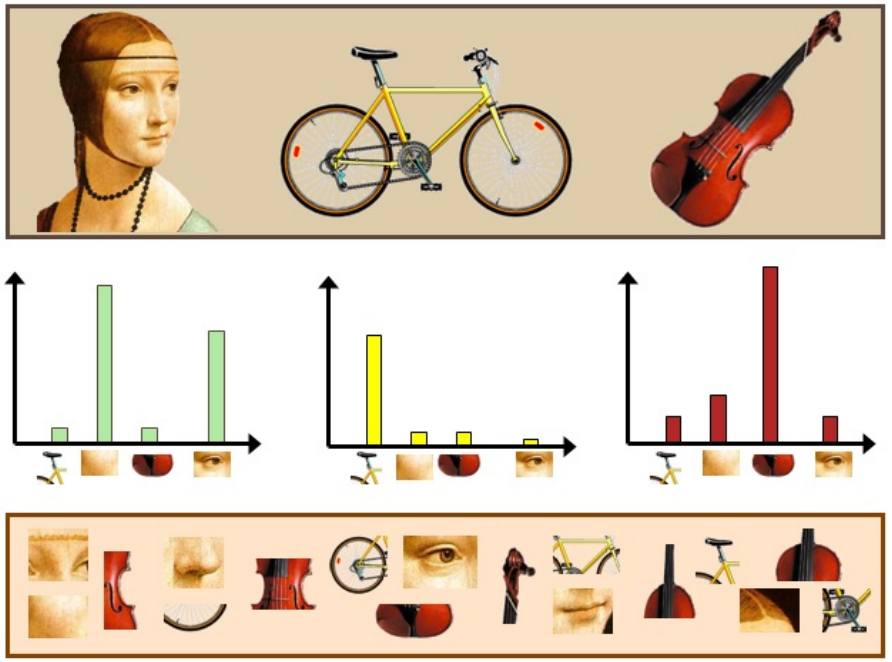
- Bank/Internet Security: fraud/spam pattern discovery
- Biology: taxonomy of living things such as kingdom, phylum, class, order, family, genus and species
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Climate change: understanding earth climate, find patterns of atmospheric and ocean
- Finance: stock clustering analysis to uncover correlation among underlying shares
- Image Compression/segmentation: coherent pixels grouped
- Information retrieval/organization: Google search, topic-based news
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- Social network mining: special interest group automatic discovery

Clustering: Objectives

- Discover underlying structure of data
- What sub-populations exist in the data?
  - How many are there?
  - What are their sizes?
  - Do elements in a sub-population have any common properties?
  - Are sub-populations cohesive? Can they be further split?
  - Are there outliers?

Clustering as Preprocessing

- Popular application of clustering
- Estimated group labels h_j (soft) or b_j (hard) may be seen as the dimensions of a new k dimensional space, where we can then learn our discriminant or regressor
- E.g. Bag-of-words representation in images

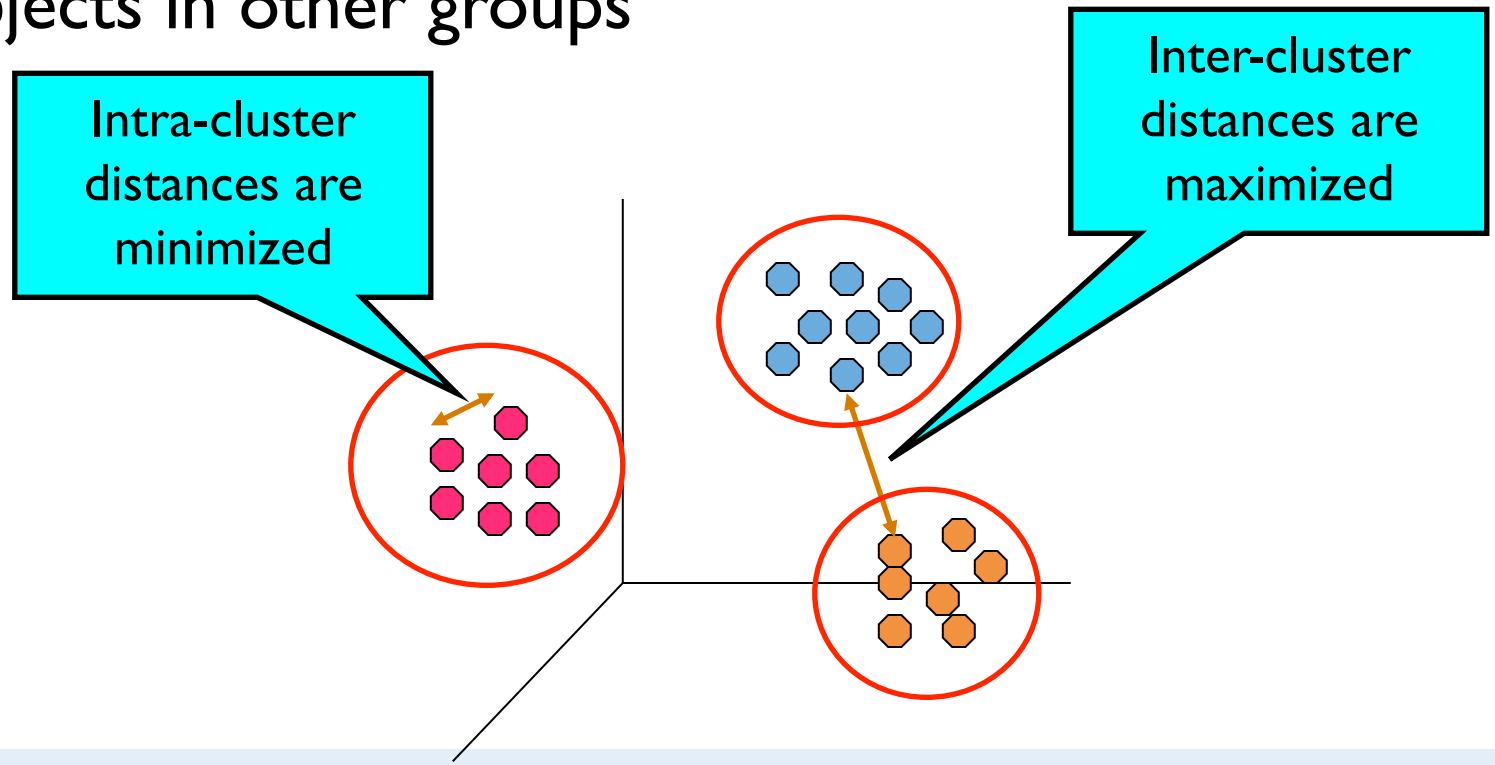


Types of Clustering Methods

- **In terms of objective:**
  - Monothetic: cluster members have some common property
    - E.g. All are males aged 20-35, or all have X% response to test B
  - Polythetic: cluster members are similar to each other
    - Distance between elements defines membership
- **In terms of overlap of clusters**
  - Hard clustering: clusters do not overlap
  - Soft clustering: clusters may overlap
    - "Strength of association" between element and cluster
- **In terms of methodology**
  - Flat/partitioning (vs) hierarchical: Set of groups (vs) taxonomy
  - Density-based (vs) Model/Distribution-based: DBSCAN vs GMMs
  - Connectionist (vs) Centroid-based: k-means vs Hierarchical clustering

Clustering Methods

- 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
- How?



Outline

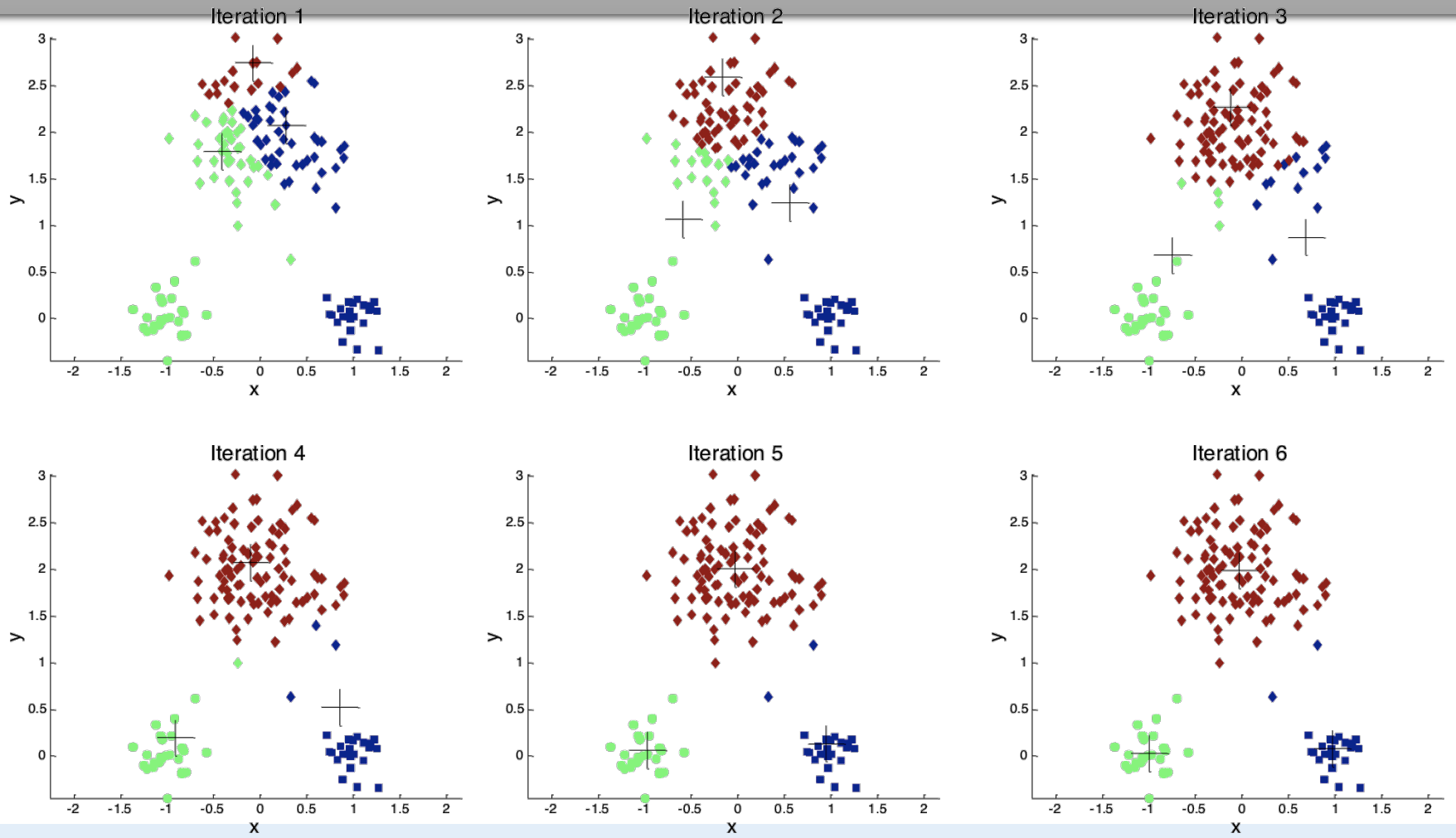
- K-Means
- Hierarchical Clustering
- Graph-based/Spectral Clustering
- DBSCAN
- Model-based Clustering (GMM and Expectation Maximization)
- Evaluation of Clustering Algorithms

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

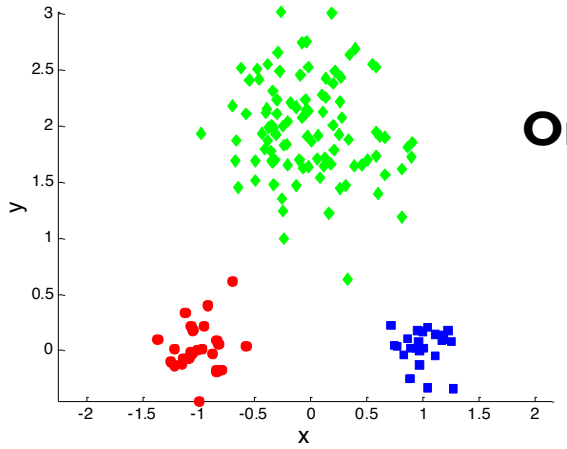
k-Means: Illustration



k-Means Clustering

- Initial centroids are often chosen randomly.
  - Clusters produced can 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 (**local minimum** though)
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to 'Until relatively few points change clusters'
- Nearby points may not end up in the same cluster! Example?

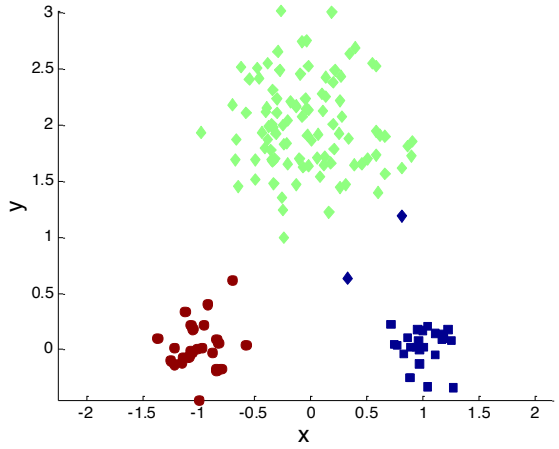
Two different k-Means clusterings

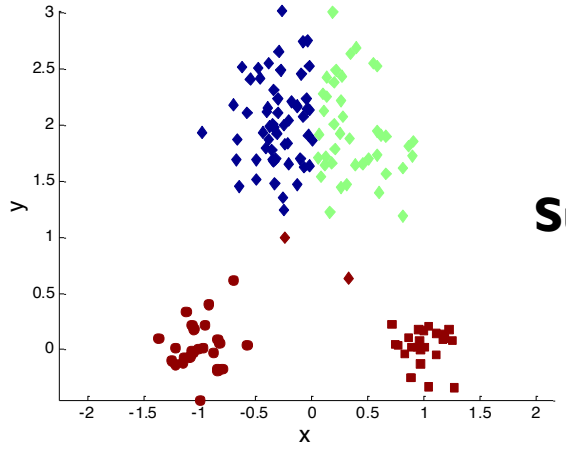


Original Points

What's the problem?

Optimal Clustering





Sub-optimal Clustering

Selecting Initial Centroids

- 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

Possible Solutions

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

Evaluating k-Means Clusters

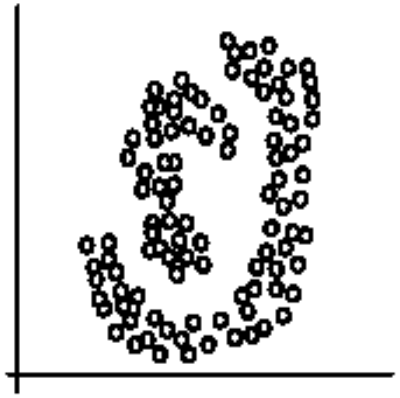
- Most common measure is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, we square these errors and sum them.

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

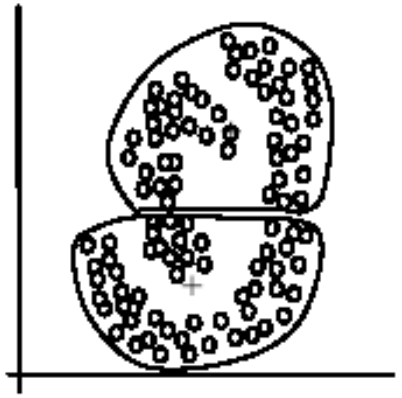
  - x is a data point in cluster Ci and m_i is the representative point for cluster Ci
  - Can show that m_i corresponds to the center (mean) of the cluster
- Given two clusterings, we can choose the one with the smaller error
- One easy way to reduce SSE is to increase K, the number of clusters
- A good clustering with smaller K can have a lower SSE than a poor clustering with higher K
- Relatively faster than other clustering methods: O(# iterations * # clusters * # instances * # dimensions)

Limitations

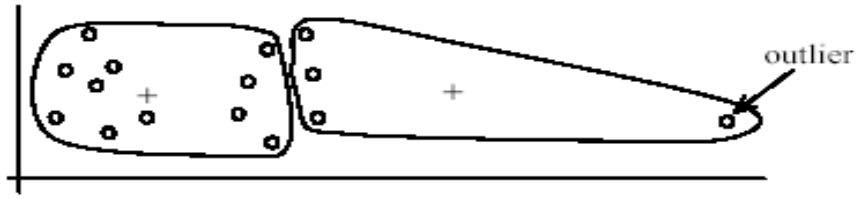
- k-Means has problems when clusters are of differing
  - Sizes, Densities, Non-globular shapes
- Sensitive to outliers
- The number of clusters (K) is difficult to determine



(A): Two natural clusters



(B): *k*-means clusters



(A): Undesirable clusters

(B): Ideal clusters

Extensions

- Use of various distance metrics
  - Euclidean distance

$$d(\mathbf{x},\mathbf{y}) = \sqrt{|x_1 - y_1|^2 + |x_2 - y_2|^2 \cdots + |x_n - y_n|^2}$$

  - Manhattan (city-block) distance

$$d(\mathbf{x},\mathbf{y}) = |x_1 - y_1| + |x_2 - y_2| \cdots + |x_n - y_n|$$

  - Cosine distance

$$\cos(\mathbf{x},\mathbf{y}) = \frac{x_1 y_1 + \cdots + x_n y_n}{\sqrt{x_1^2 + \cdots + x_n^2}\sqrt{y_1^2 + \cdots + y_n^2}}$$

$$d(\mathbf{x},\mathbf{y}) = 1 - \cos(\mathbf{x},\mathbf{y})$$

  - Chebyshev distance

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

Extensions

- k-Medioids
- Bisecting k-Means
- k-Means++

Outline

- K-Means
- Hierarchical Clustering
- Graph-based/Spectral Clustering
- DBSCAN
- Model-based Clustering (GMM and Expectation Maximization)
- Evaluation of Clustering Algorithms

Challenge

How many clusters?

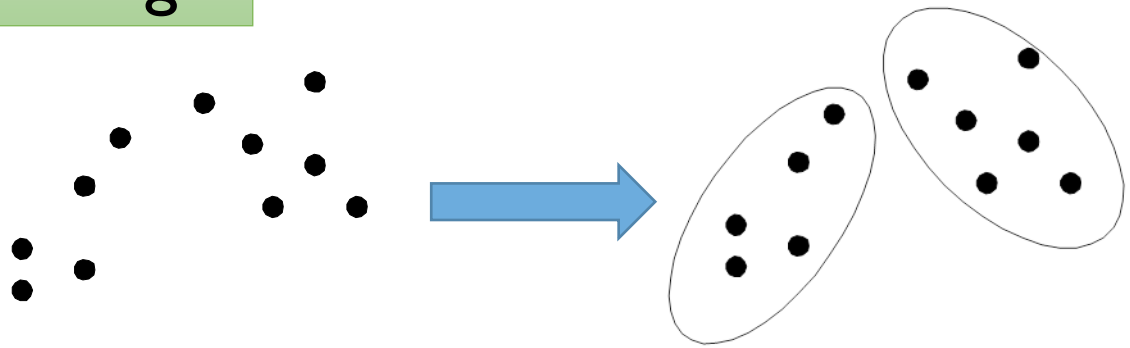
Six Clusters

Two Clusters

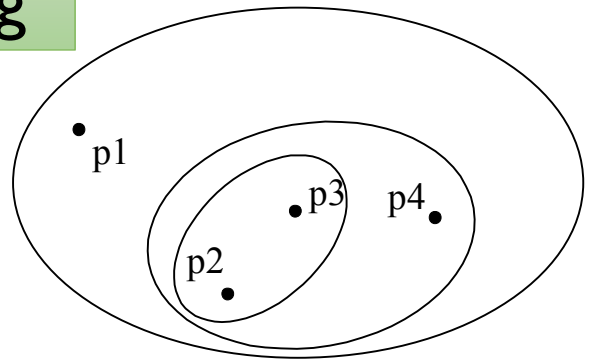
Four Clusters

Types of Clustering Methods

Partitional Clustering

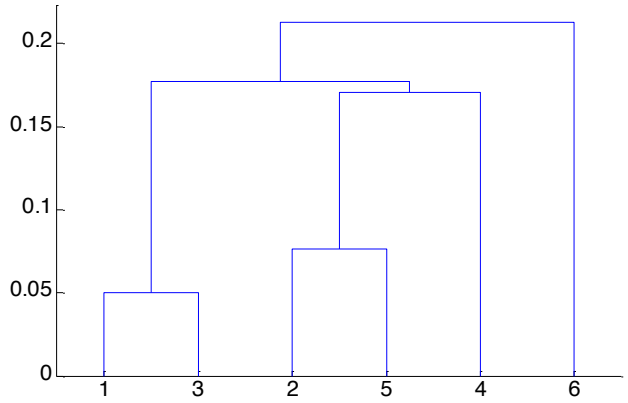


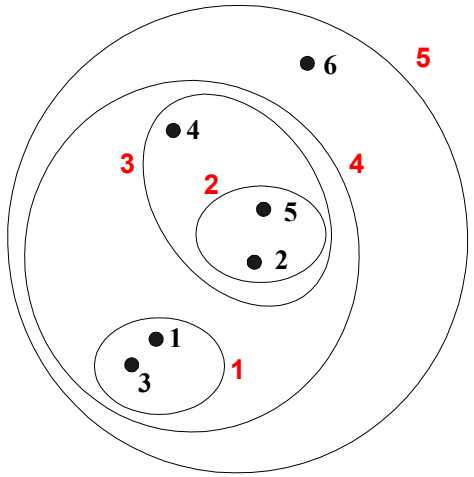
Hierarchical Clustering



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

- 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

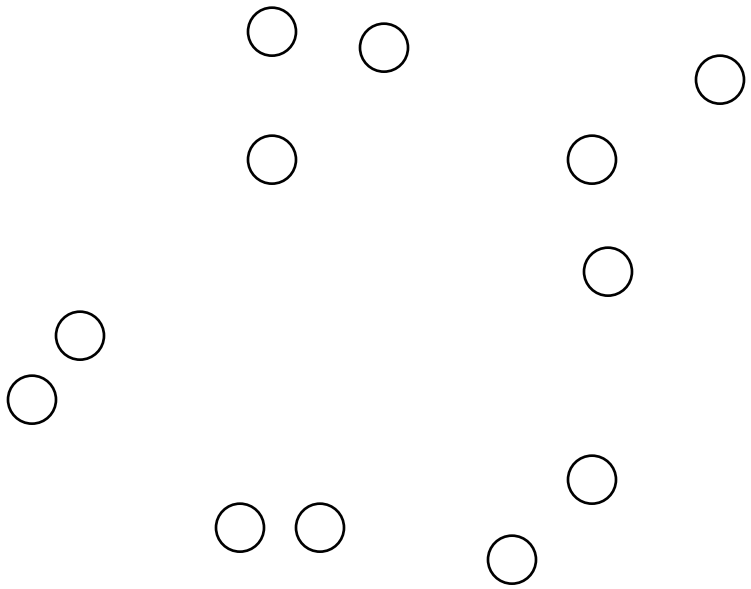
- Two main types of hierarchical clustering
  - **Agglomerative:**
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - **Divisive:**
    - Start with one, all-inclusive cluster
    - At each step, split 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
     1. Merge the two closest clusters
     2. Update the proximity matrix
  4. 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

Methodology

- Start with clusters of individual points and a proximity matrix

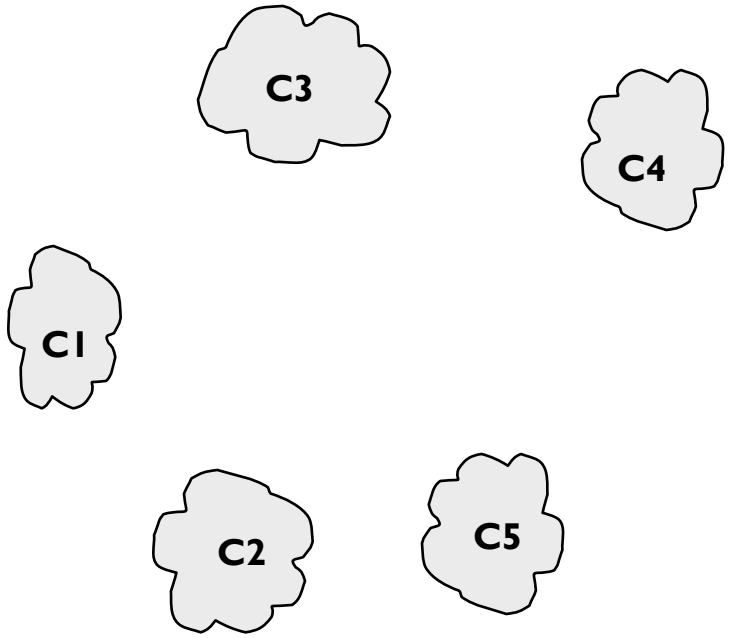


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

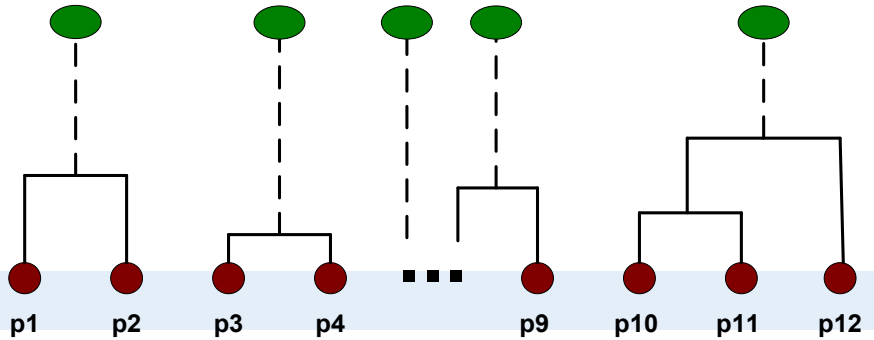
Methodology

- After some merging steps, we have some clusters



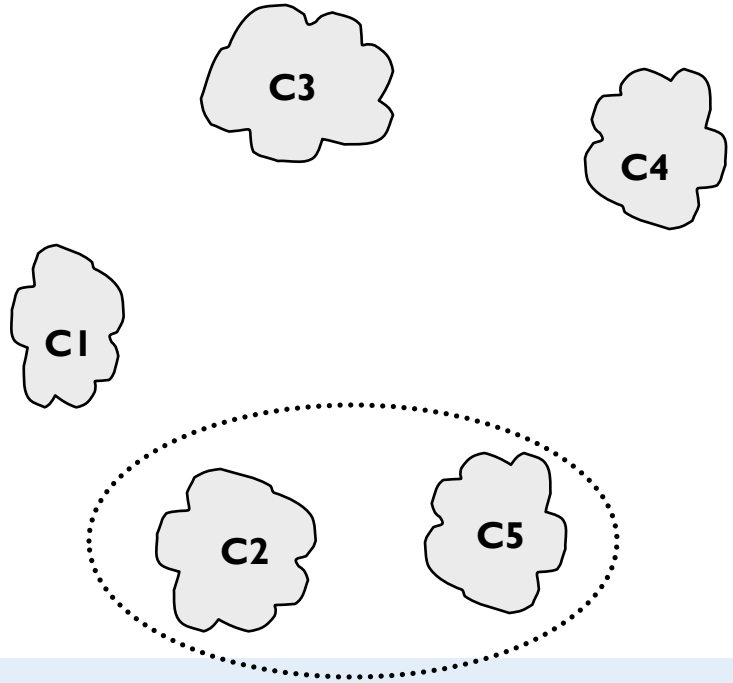
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



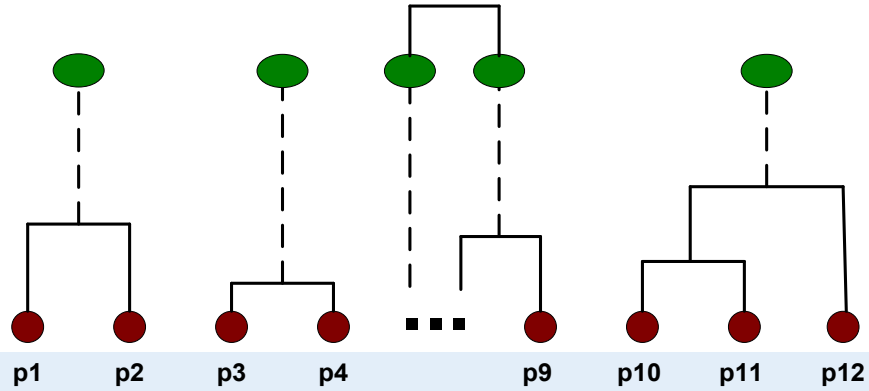
Methodology

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



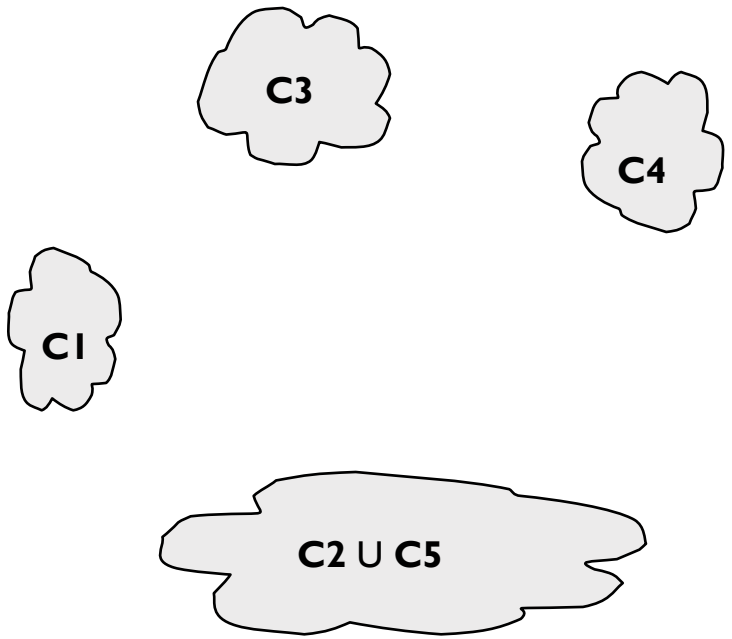
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



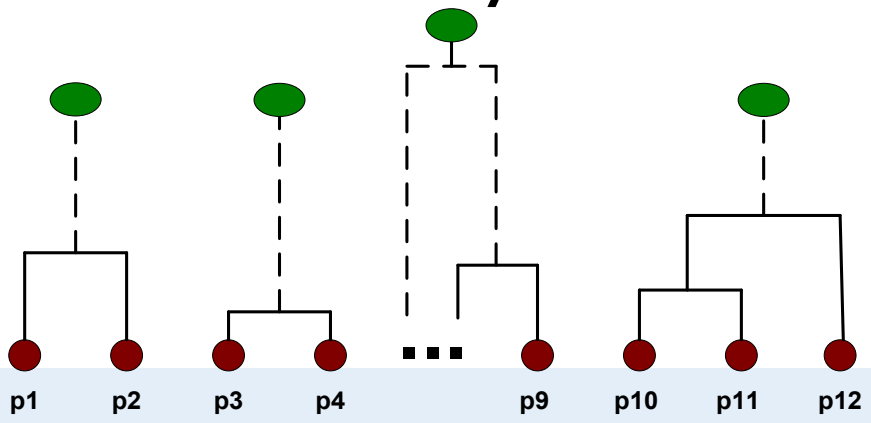
Methodology

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

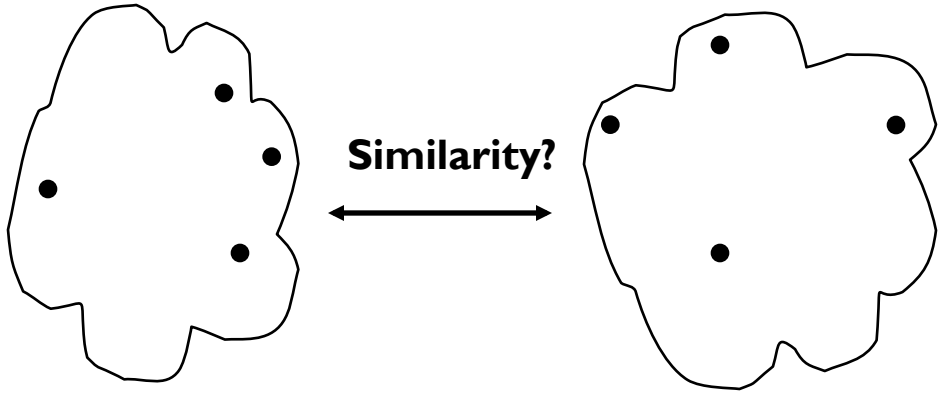


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



Defining Inter-cluster Similarity

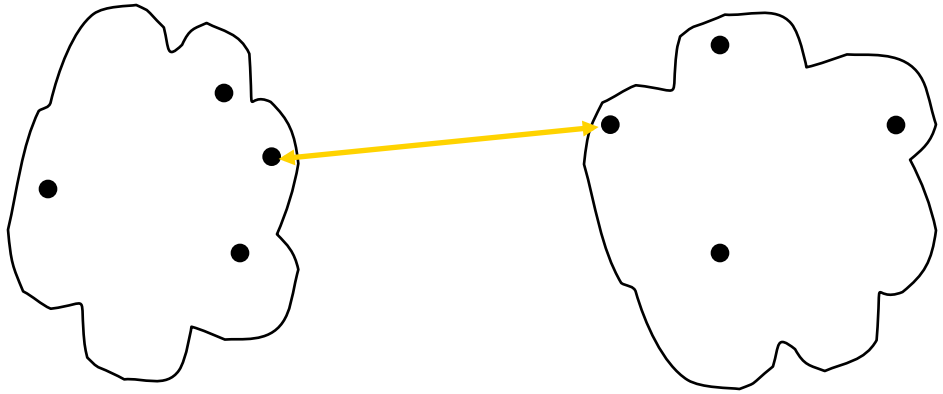


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						

Proximity Matrix

- MIN (Single-link)
- MAX (Complete-link)
- Group Average (Average-link)
- Distance Between Centroids

Defining Inter-cluster Similarity

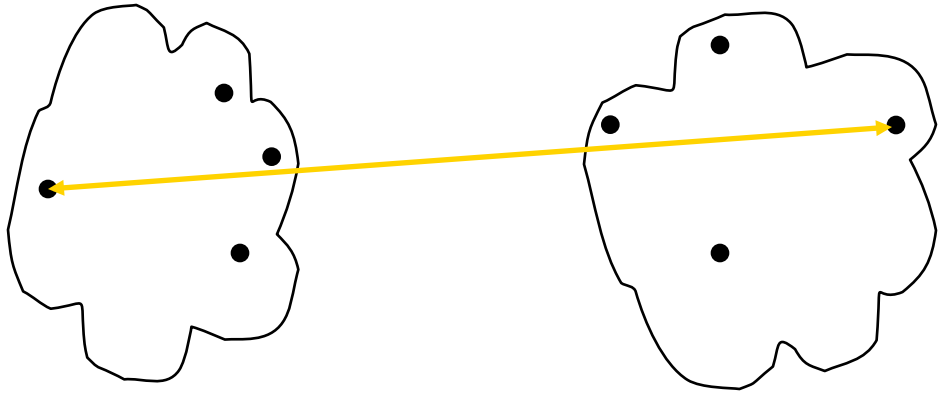


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- **MIN (Single-link)**
- MAX (Complete-link)
- Group Average (Average-link)
- Distance Between Centroids

Defining Inter-cluster Similarity

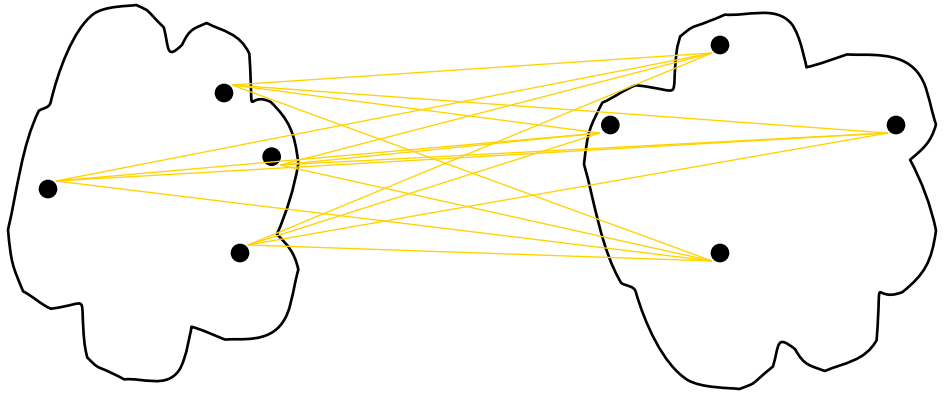


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN (Single-link)
- **MAX (Complete-link)**
- Group Average (Average-link)
- Distance Between Centroids

Defining Inter-cluster Similarity

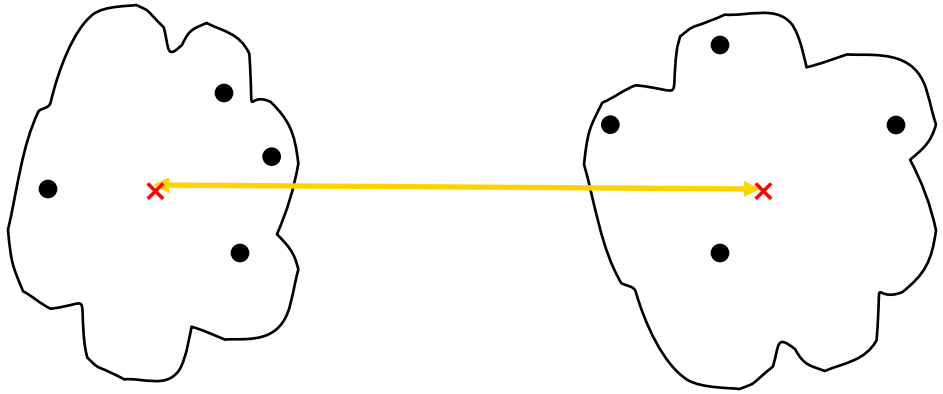


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN (Single-link)
- MAX (Complete-link)
- **Group Average (Average-link)**
- Distance Between Centroids

Defining Inter-cluster Similarity



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

Proximity Matrix

- MIN (Single-link)
- MAX (Complete-link)
- Group Average (Average-link)
- **Distance Between Centroids**

Hierarchical Clustering: Limitations

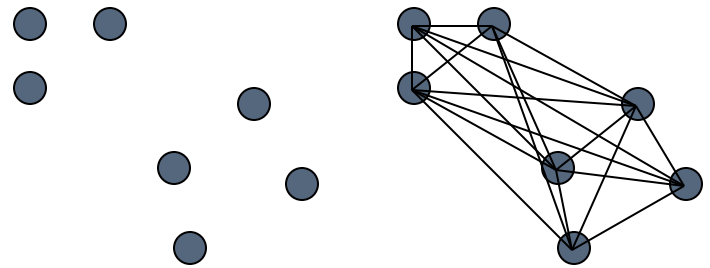
- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers (MIN)
  - Difficulty handling different sized clusters and non-convex shapes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

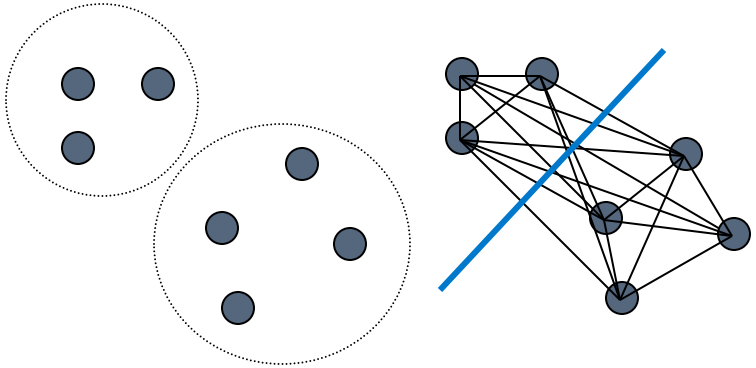
Outline

- K-Means
- Hierarchical Clustering
- Graph-based/Spectral Clustering
- DBSCAN
- Model-based Clustering (GMM and Expectation Maximization)
- Evaluation of Clustering Algorithms

Graph-Based/Spectral Clustering

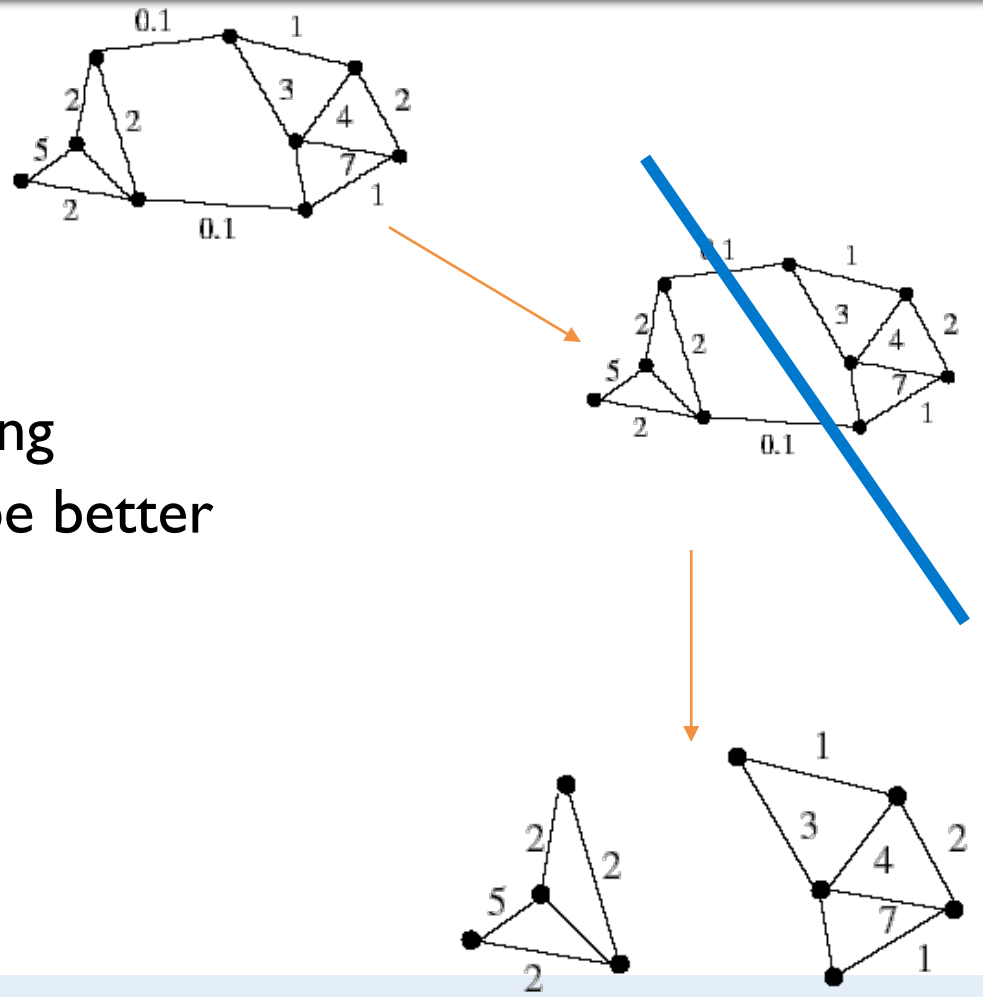
- Associate each data item with a vertex in a weighted graph
  - weights on the edges between elements are large if the elements are similar and small if they are not.
- Cut the graph into connected components with relatively large interior weights by cutting edges with relatively low weights.
- Clustering becomes a graph cut problem.





Graph-based/Spectral Clustering

- Method #1
  - Partition into two clusters
  - Use procedure recursively
- Method #2
  - Directly compute k-way partitioning
  - Experimentally has been seen to be better
- Examples
  - Minimum Cut
  - Normalized Cut

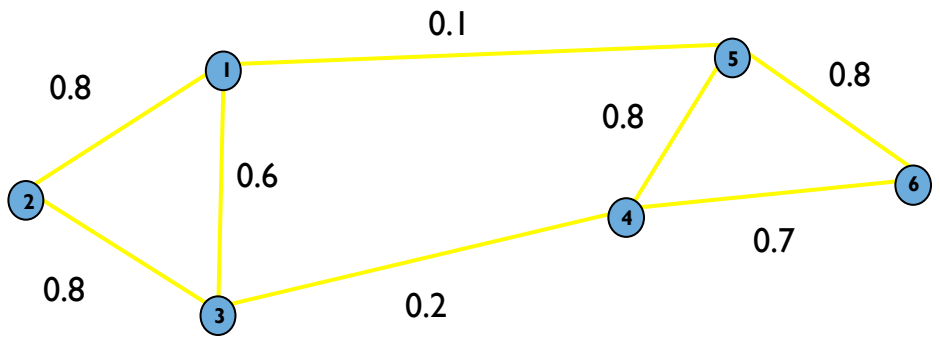


Graph-Based/Spectral Clustering

- Represent dataset as a weighted graph *G(V,E)*

$V=\{x_i\}$ Set of *n* vertices representing data points

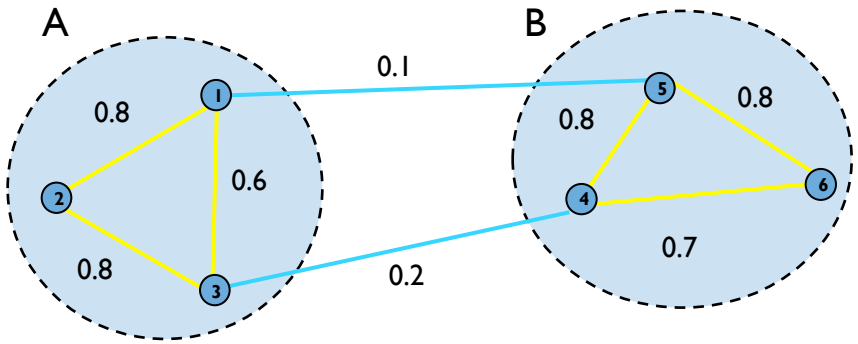
$E=\{W_{ij}\}$ Set of weighted edges indicating pair-wise similarity between points (Wii=0)



Note that there are many ways of defining similarity here again

Graph Cuts

- Express partitioning objectives as a function of the "edge cut" of the partition.
- *Cut:* Set of edges with only one vertex in a group. The groups that provide the minimal cut would be the partition.

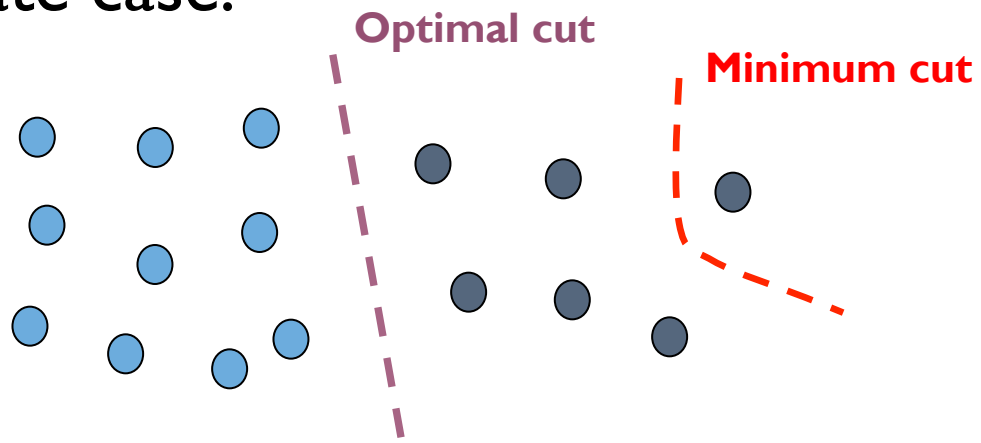


$$cut(A,B) = \sum_{i \in A, j \in B} w_{ij}$$

$$cut(A,B) = 0.3$$

Graph Cuts

- **Criterion: Minimum-cut**
  - Minimise weight of connections between groups: min *cut(A,B)*
- Degenerate case:



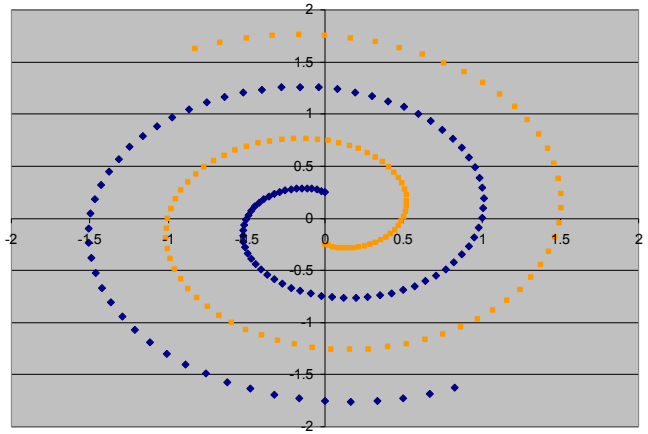
- Problem:
  - Only considers external cluster connections
  - Does not consider internal cluster density

Graph Cuts

- **Criterion: Normalised-cut** (Shi & Malik,'97)
  - Consider the connectivity between groups relative to the density of each group: $$\min Ncut(A,B) = \frac{cut(A,B)}{vol(A)} + \frac{cut(A,B)}{vol(B)}$$
  - Normalise the association between groups by *volume*.
    - *Vol(A)*: The total weight of the edges originating from group *A*.
  - Why use this criterion?
    - Minimising the normalised cut is equivalent to maximising normalised association.
    - Produces more balanced partitions.

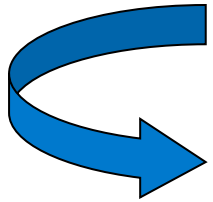
These are NP-hard! Spectral clustering provides a relaxation to these

Graph-based/Spectral Clustering

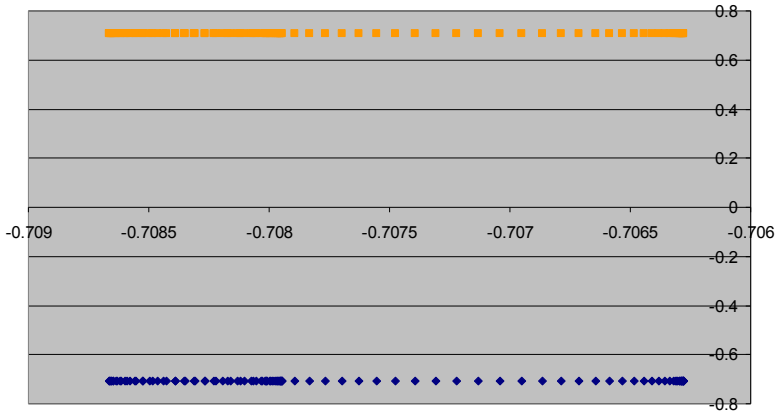


Dataset exhibits complex cluster shapes

⇒ K-means (and similar methods) perform very poorly in this space due bias toward dense spherical clusters.



In the embedded space given by two leading eigenvectors, clusters are trivial to separate.



Spectral Graph Theory

- Possible approach
  - Represent a similarity graph as a matrix
  - Apply knowledge from Linear Algebra…

- The *eigenvalues* and *eigenvectors* of a matrix provide global information about its structure.

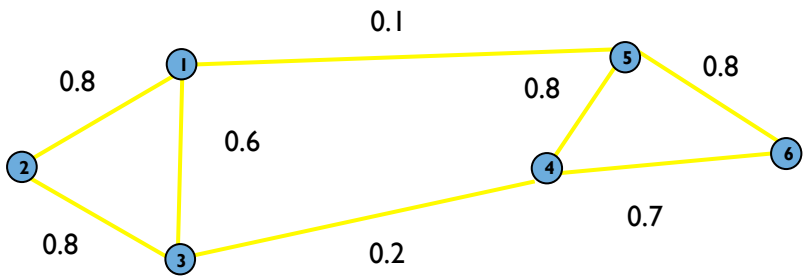
$$\begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & & \vdots \\ w_{n1} & \cdots & w_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

- *Spectral Graph Theory*
  - Analyse the "spectrum" of matrix representing a graph.
  - *Spectrum*: The eigenvectors of a graph, ordered by the magnitude(strength) of their corresponding eigenvalues.

$$\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$$

Spectral Graph Theory

- **Adjacency matrix (W)**
  - *n x n* matrix
  - $W = [w_{ij}]$: edge weight between vertex x_i and x_j



	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	0.8	0.6	0	0.1	0
x_2	0.8	0	0.8	0	0	0
x_3	0.6	0.8	0	0.2	0	0
x_4	0	0	0.2	0	0.8	0.7
x_5	0.1	0	0	0.8	0	0.8
x_6	0	0	0	0.7	0.8	0

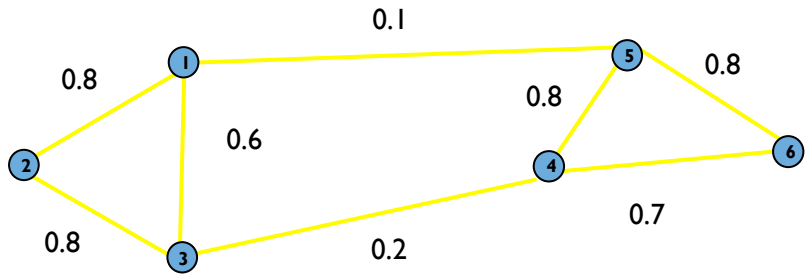
- Important properties:
  - Symmetric matrix

  ⇒ Eigenvalues are <u>real</u>

  ⇒ Eigenvectors form <u>orthogonal basis</u>

Spectral Graph Theory

- **Degree matrix (D)**
  - *n x n* diagonal matrix
  - $D(i,i) = \sum_j w_{ij}$: total weight of edges incident to vertex x_i

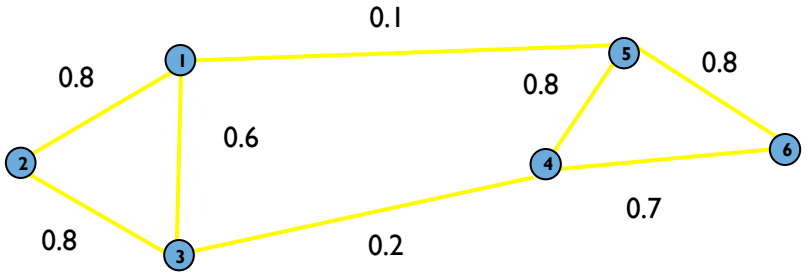


	x_1	x_2	x_3	x_4	x_5	x_6
x_1	1.5	0	0	0	0	0
x_2	0	1.6	0	0	0	0
x_3	0	0	1.6	0	0	0
x_4	0	0	0	1.7	0	0
x_5	0	0	0	0	1.7	0
x_6	0	0	0	0	0	1.5

- Important application:
  - Normalise adjacency matrix

Spectral Graph Theory

- **Laplacian matrix (L):** $L = D - W$
  - *n x n* symmetric matrix

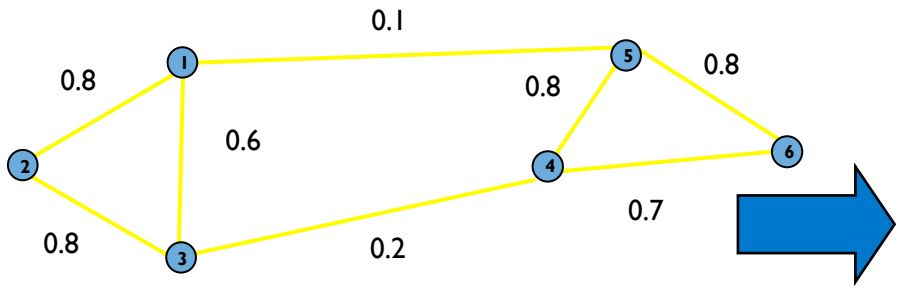


	x_1	x_2	x_3	x_4	x_5	x_6
x_1	1.5	-0.8	-0.6	0	-0.1	0
x_2	-0.8	1.6	-0.8	0	0	0
x_3	-0.6	-0.8	1.6	-0.2	0	0
x_4	0	0	-0.2	1.7	-0.8	-0.7
x_5	-0.1	0	0	0.8-	1.7	-0.8
x_6	0	0	0	-0.7	-0.8	1.5

- Important properties:
  - Eigenvalues are non-negative real numbers
  - Smallest eigenvalue is zero; corresponding eigenvector is $\mathbf{1}^T$ ($L\mathbf{1} = D\mathbf{1} - W\mathbf{1} = 0$)

Spectral Graph Theory

- Normalized Laplacian matrix (L): $D^{-0.5}.(D-W).D^{-0.5}$
  - *n x n* symmetric matrix

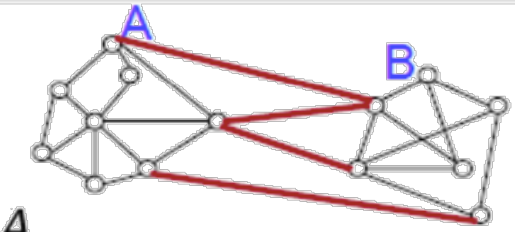


1.00	-0.52	-0.39	0.00	-0.06	0.00
-0.52	1.00	-0.50	0.00	0.00	0.00
-0.39	-0.50	1.00	-0.12	0.00	0.00
0.00	0.00	-0.12	1.00	-0.47	-0.44
-0.06	0.00	0.00	0.47-	1.00	-0.50
0.00	0.00	0.00	-0.44	-0.50	1.00

- Important properties:
  - Eigenvectors are real and normalizes
  - Each W_{ij} (which i,j is not equal) = W_{ij}/D_{ii}

Spectral Clustering

$$\text{cut}(A, B) := \sum_{i \in A, j \in B} w_{ij}$$



$$\text{Choose } f = (f_1, \ldots, f_n)' \text{ with } f_i = \begin{cases} 1 & \text{if } X_i \in A \\ -1 & \text{if } X_i \in B \end{cases}$$

$$\text{cut}(A, B) = \sum_{i \in A, j \in B} w_{ij} = \frac{1}{4} \sum_{i,j} w_{ij}(f_i - f_j)^2 = \frac{1}{2} f^T(D\text{-}W) f$$

$$\text{2RHS} = f^T(D\text{-}W) f = f^T D f - f^T W f = \sum_i d_i f_i^2 - \sum_{i,j} f_i f_j w_{ij}$$

$$= \frac{1}{2}\left(\sum_i (\sum_j w_{ij}) f_i^2 - 2\sum_{ij} f_i f_j w_{ij} + \sum_j (\sum_i w_{ij}) f_j^2\right)$$

$$= \frac{1}{2}\sum_{ij} w_{ij}(f_i - f_j)^2 \quad = \text{2LHS}$$

Slide Courtesy: Aarti Singh, CMU

Balanced Min-cut

$$\min_{A,B} \text{cut}(A, B) \text{ s.t. } |A| = |B|$$

$$\min_{f \in \{-1,1\}^n} f^T L f \quad \text{s.t.} \quad f^T 1 = 0$$

(since $\sum f_i = \sum 1_{i \in A} - 1_{i \in B} = 0$)

Above formulation is still NP-Hard, so we relax f not to be binary:

$$\min_{f \in R^n} f^T L f \quad \text{s.t.} \quad f^T 1 = 0, \quad f^T f = n$$

$$\min_{f \in R^n} \frac{f^T L f}{f^T f} \quad \text{s.t.} \quad f^T 1 = 0$$

Slide Courtesy: Aarti Singh, CMU

Relaxation of Balanced Min-cut

$$\min_{f \in R^n} \frac{f^T L f}{f^T f} \quad \text{s.t.} \quad f^T 1 = 0$$

$$\Downarrow$$

$\lambda_{min}(L)$ - smallest eigenvalue of L (Rayleigh-Ritz theorem)

If f is eigenvector of L, then

$$\frac{f^T L f}{f^T f} = \frac{f^T \lambda f}{f^T f} = \lambda$$

- Recall that smallest eigenvalue of L is 0 with corresponding eigenvector 1. But f can't be 1 according to constraint $f^T 1 = 0$
- Therefore, solution f is the eigenvector of L corresponding to second smallest eigenvalue, a.k.a second eigenvector.

Slide Courtesy: Aarti Singh, CMU

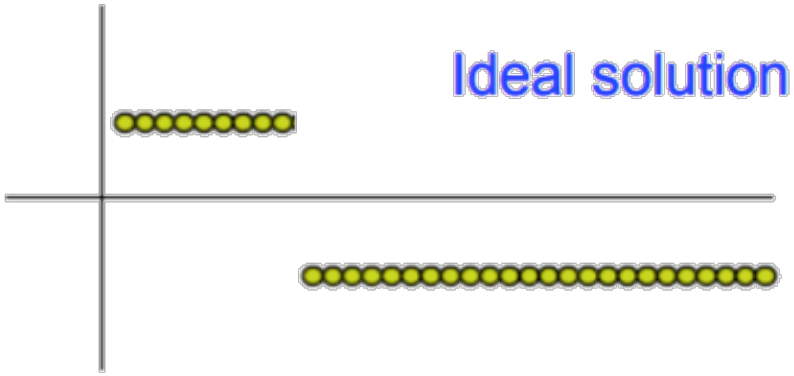
Relaxation of Balanced Min-cut

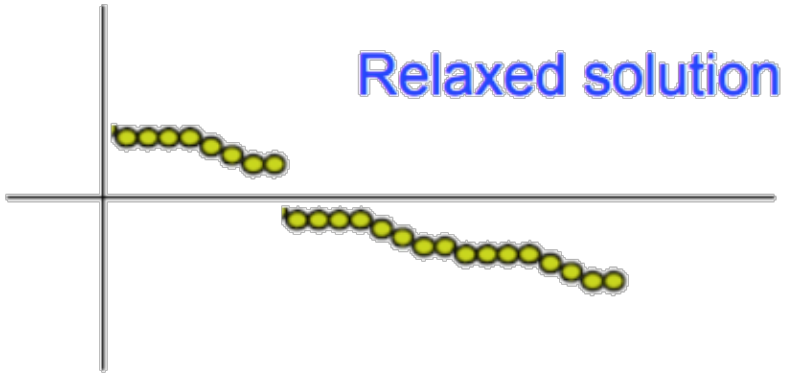
$$\min_{A,B} \text{cut}(A,B) \text{ s.t. } |A| = |B|$$

Let f be the second eigenvector of the unnormalized graph Laplacian L.

Recover binary partition as follows:

$$i \in A \quad if \quad f_i \geq 0$$
$$i \in B \quad if \quad f_i < 0$$



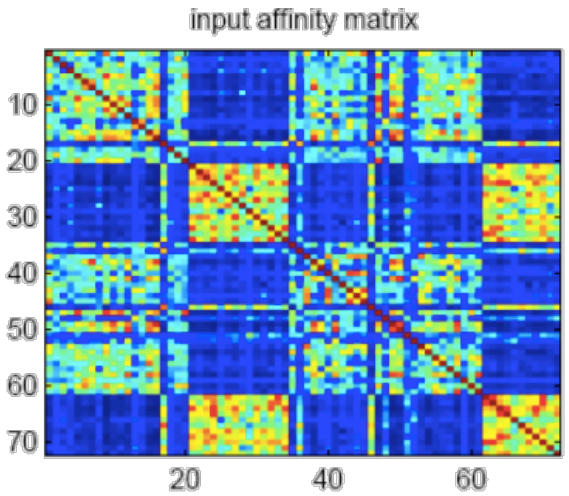


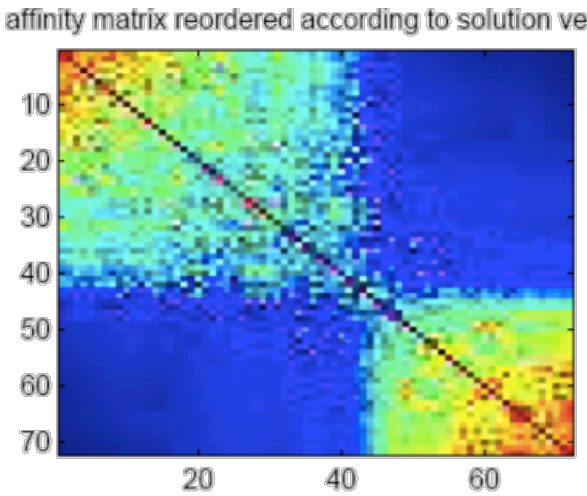
Similar relaxations work for other cut problems:

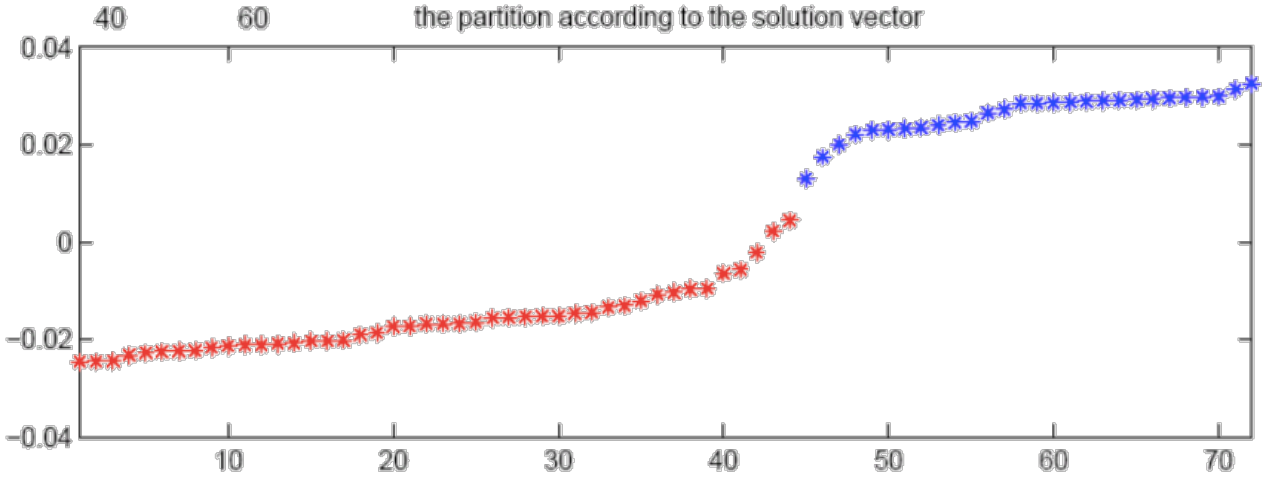
- E.g. Normalized cut – second eigenvector of normalized Laplacian
  $\mathbf{L' = I - D^{-1}W}$

Slide Courtesy: Aarti Singh, CMU

Example







Slide Courtesy: Aarti Singh, CMU

How to partition into k clusters?

Input: Similarity matrix W, number k of clusters to construct

- Build similarity graph
- Compute the first k eigenvectors $v_1, \ldots, v_k$ of the matrix

$$\begin{cases} L & \text{for unnormalized spectral clustering} \\ L' & \text{for normalized spectral clustering} \end{cases}$$

- Build the matrix $V \in \mathbb{R}^{n \times k}$ with the eigenvectors as columns
- Interpret the rows of V as new data points $Z_i \in \mathbb{R}^k$

	v_1	v_2	v_3
Z_1	v_{11}	v_{12}	v_{13}
$\vdots$	$\vdots$	$\vdots$	$\vdots$
Z_n	v_{n1}	v_{n2}	v_{n3}

Dimensionality Reduction
n x n → n x k

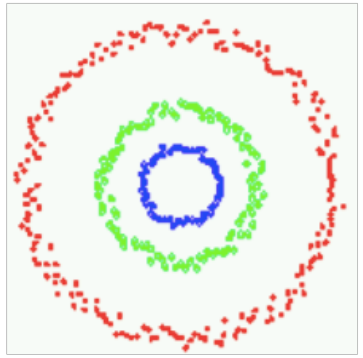
- Cluster the points Z_i with the k-means algorithm in $\mathbb{R}^k$.

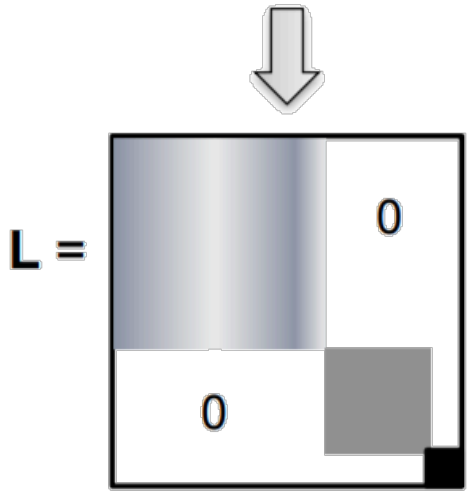
Slide Courtesy: Aarti Singh, CMU

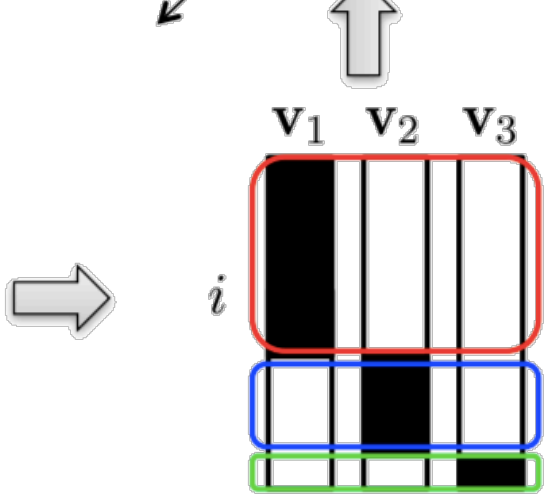
Understanding Spectral Clustering

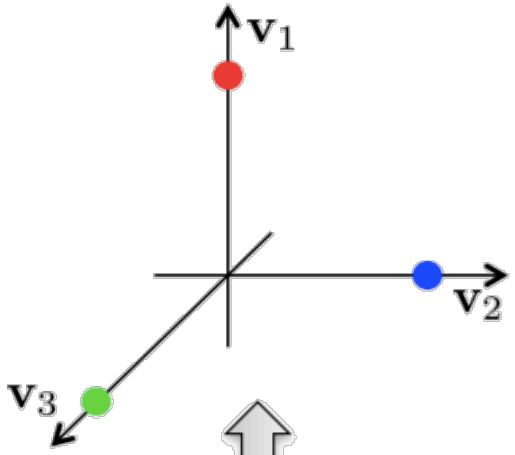
Eigenvectors of the Laplacian matrix provide an embedding of the data based on similarity.

Disconnected subgraphs









Points are easy to cluster in embedded space e.g. using k-means

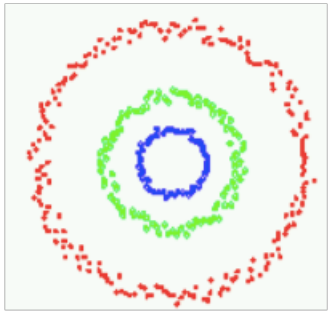
Embedding of point i

$[\mathbf{v}_1(i), \mathbf{v}_2(i), \mathbf{v}_3(i)]$

Slide Courtesy: Aarti Singh, CMU

Understanding Spectral Clustering

- If graph is connected, first Laplacian evec is constant (all 1s)
- If graph is disconnected (k connected components), Laplacian is block diagonal and first k Laplacian evecs are:



OR

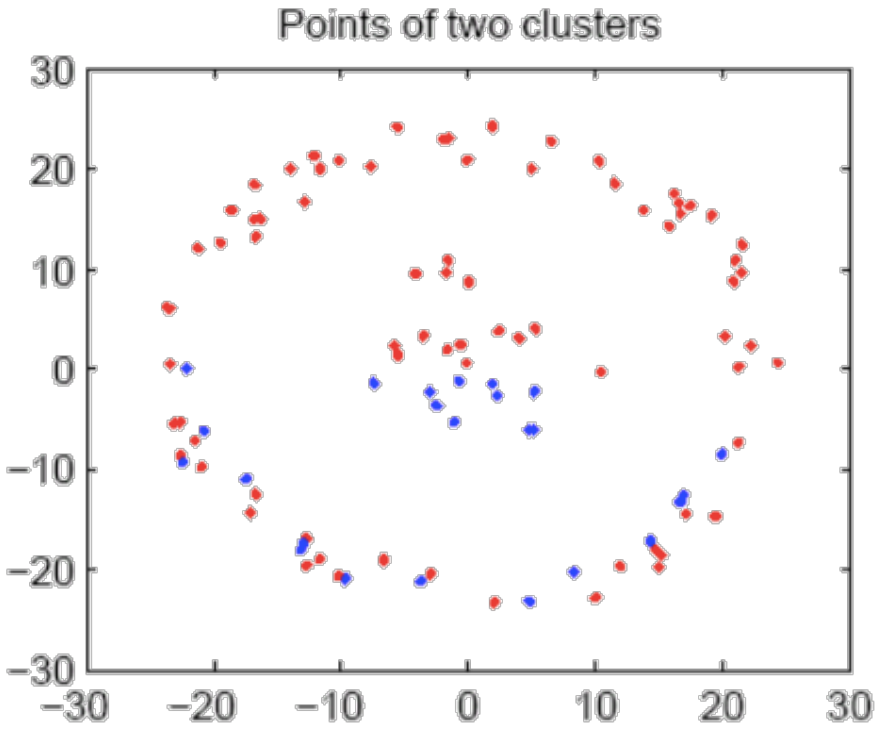
$$L = \begin{bmatrix} L_1 & & 0 \\ & L_2 & \\ 0 & & L_3 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

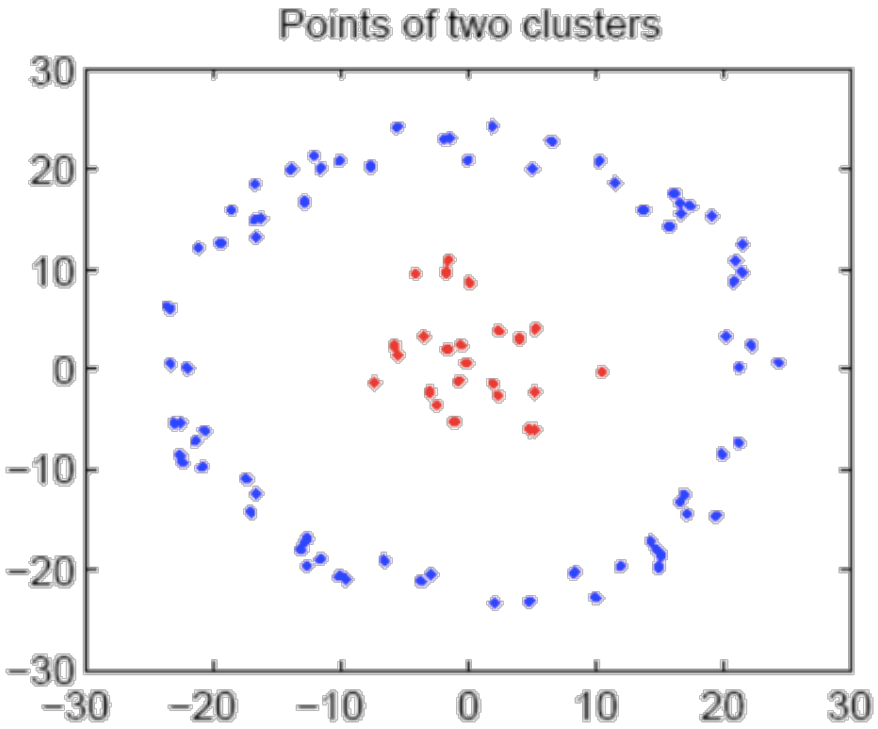
First three eigenvectors

Slide Courtesy: Aarti Singh, CMU

K-means vs Spectral Clustering



k-means output



Spectral clustering output

Slide Courtesy: Aarti Singh, CMU

Outline

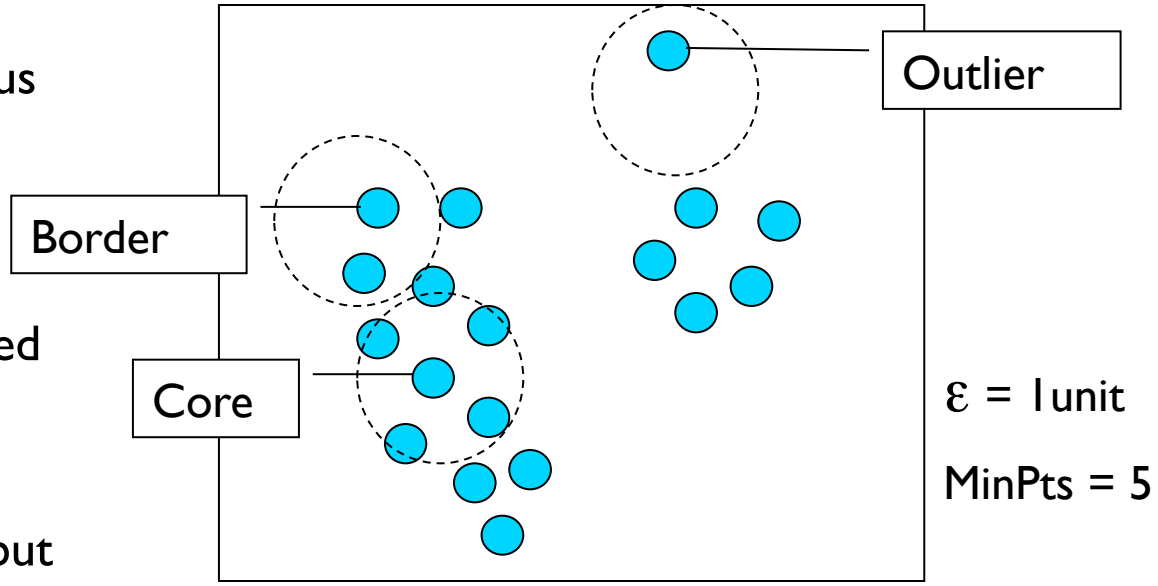
- K-Means
- Hierarchical Clustering
- Graph-based/Spectral Clustering
- DBSCAN
- Model-based Clustering (GMM and Expectation Maximization)
- Evaluation of Clustering Algorithms

DBSCAN

Density-based Clustering: locates regions of high density that are separated from one another by regions of low density.

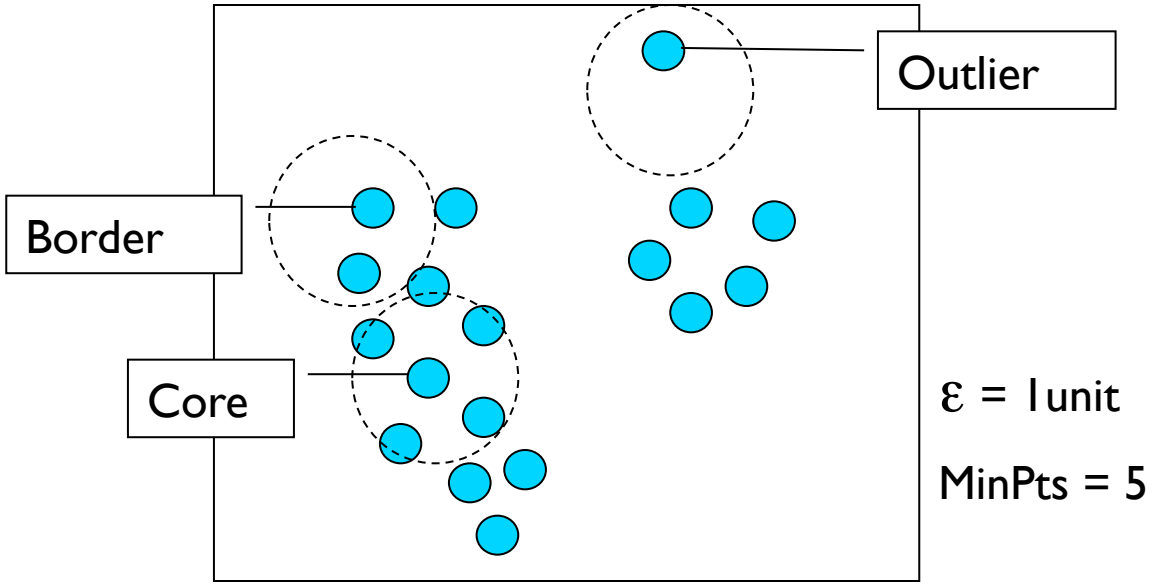
- Density = number of points within a specified radius (Eps)

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



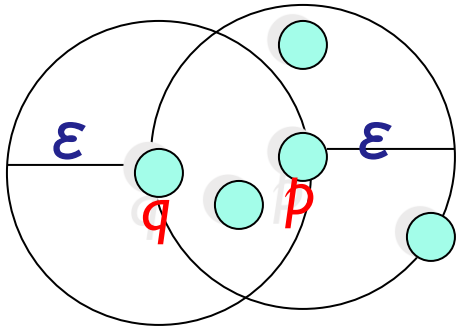
DBSCAN

- Key idea
  - Any two core points are close enough– within a distance *Eps* of one another – are put in the same cluster
  - Any border point that is close enough to a core point is put in the same cluster as the core point
  - Noise points are discarded



DBSCAN

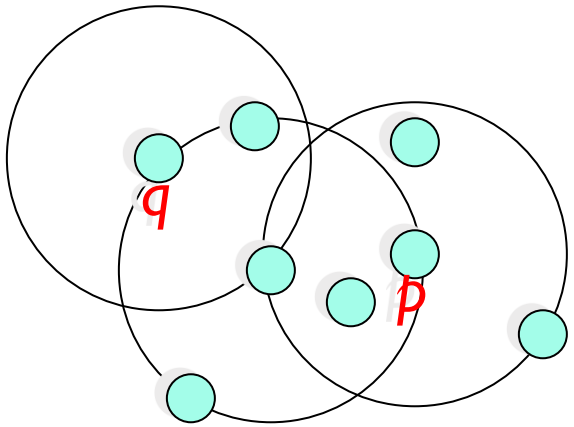
- **Directly density-reachable**
  - An object q is directly density-reachable from object p if q is within the ε - Neighborhood of p and p is a core object.



- q is directly density-reachable from p
- p is not directly density- reachable from q

DBSCAN

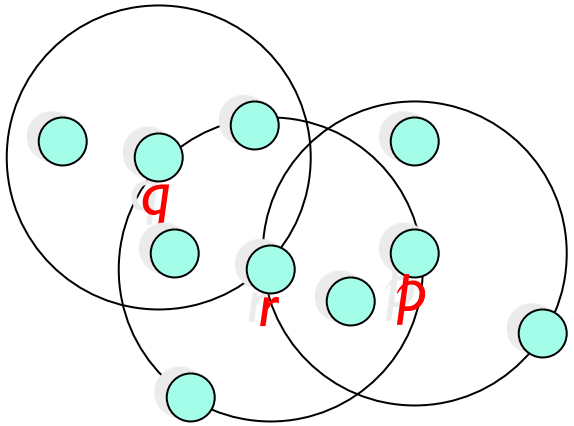
- **Density-reachable:**
  - An object p is density-reachable from q w.r.t ε and *MinPts* if there is a chain of objects $p_1,\ldots,p_n$, with $p_1=q, p_n=p$ such that p_{i+1} is directly density-reachable from p_i w.r.t ε and *MinPts* for all 1 <= i <= n



- q is density-reachable from p
- p is not density- reachable from q
- Transitive closure of direct density-Reachability, asymmetric

DBSCAN

- **Density-connectivity**
  - Object *p* is density-connected to object *q* w.r.t ε and *MinPts* if there is an object *o* such that both *p* and *q* are density-reachable from *o* w.r.t ε and *MinPts*

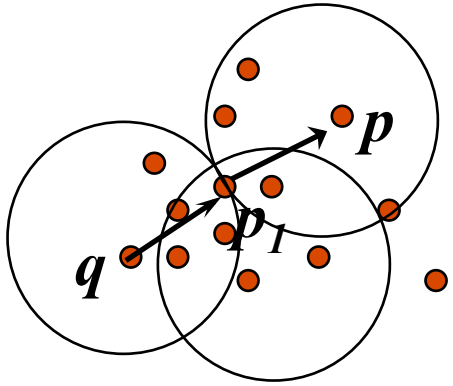


- *P* and *q* are density-connected to each other by *r*
- Density-connectivity is symmetric

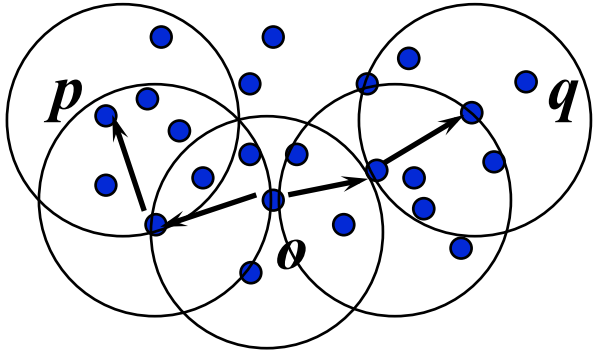
DBSCAN

- **Density-connectivity (vs) Density-reachability**

Density-reachable



Density-connected



DBSCAN

- **Cluster**: a cluster ***C*** in a set of objects ***D*** w.r.t ε and *MinPts* is a non empty subset of ***D*** satisfying
  - **Maximality:** For all p, q if $p \in \boldsymbol{C}$ and if q is density-reachable from p w.r.t ε and *MinPts*, then also $q \in \boldsymbol{C}$.
  - **Connectivity:** for all $p, q \in \boldsymbol{C}$, p is density-connected to q w.r.t ε and *MinPts* in ***D***.
  - ***Note:*** cluster contains *core objects* as well as *border objects*

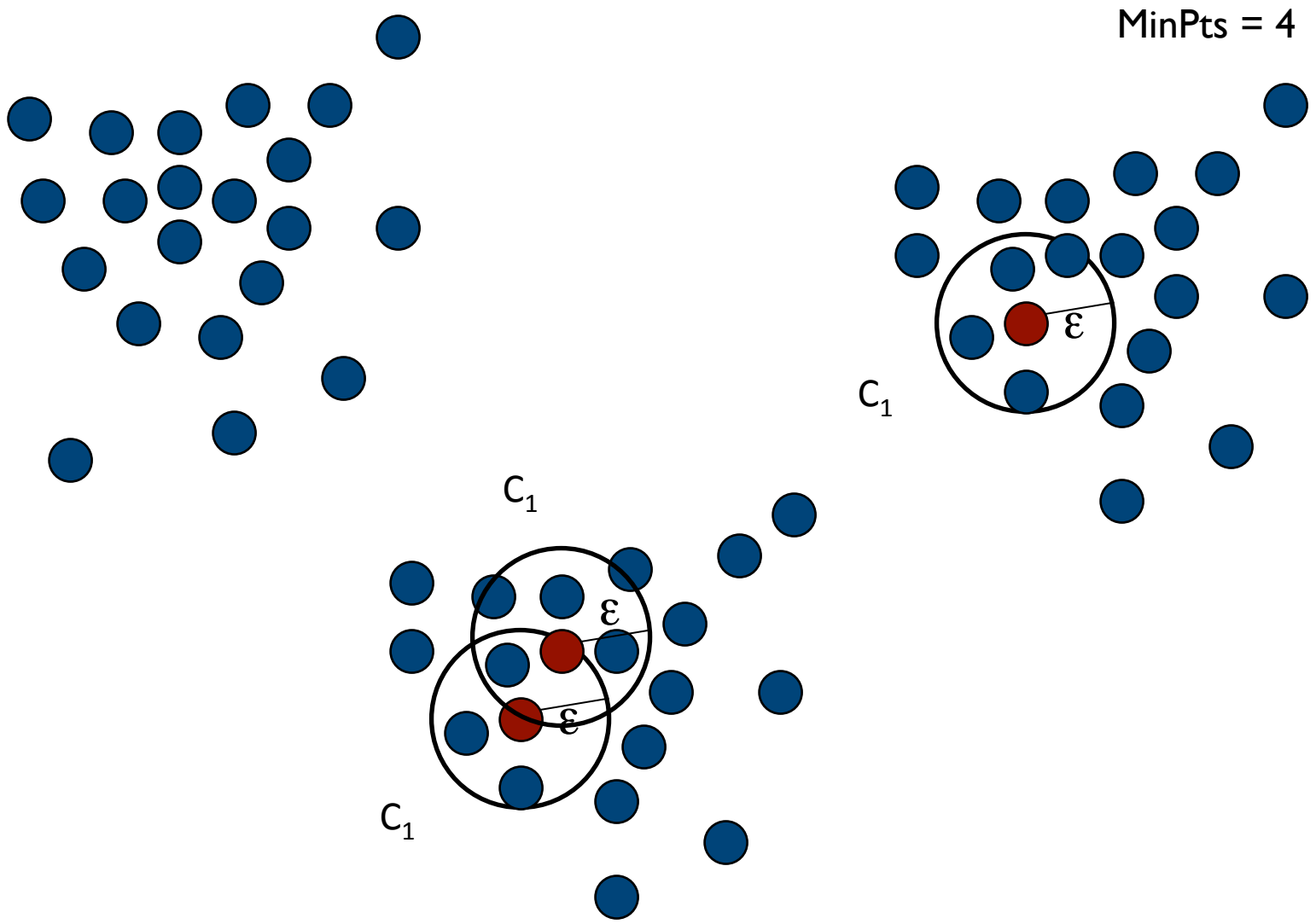
- **Noise:** objects which are not directly density-reachable from at least one core object.

DBSCAN Algorithm

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

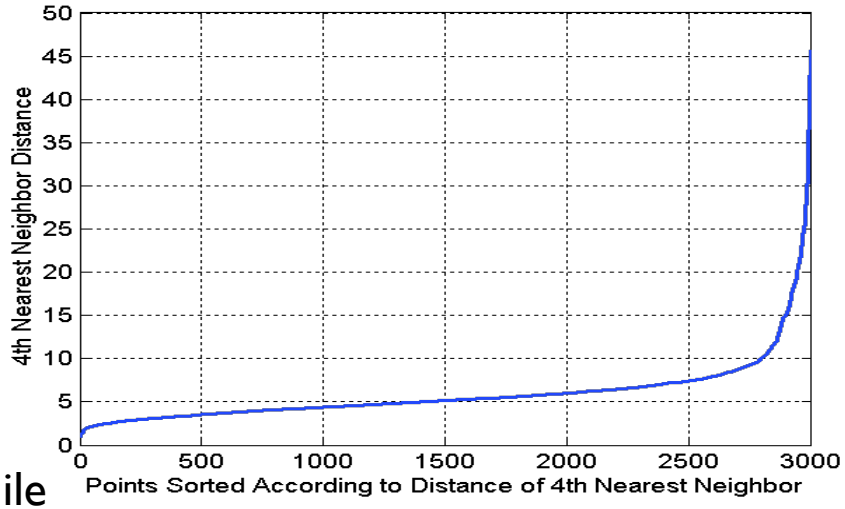
Result is independent of the order of processing the points

DBSCAN: Example



DBSCAN: Determining Eps and Minpts

- Distance from a point to its k^{th} nearest neighbor=>k-dist
- Compute k-dist for all points for some k
- Sort them in increasing order and plot sorted values
- A sharp change at the value of k-dist that corresponds to suitable value of eps and the value of k as MinPts
  - Points for which k-dist is less than eps will be labeled as core points while other points will be labeled as noise or border points



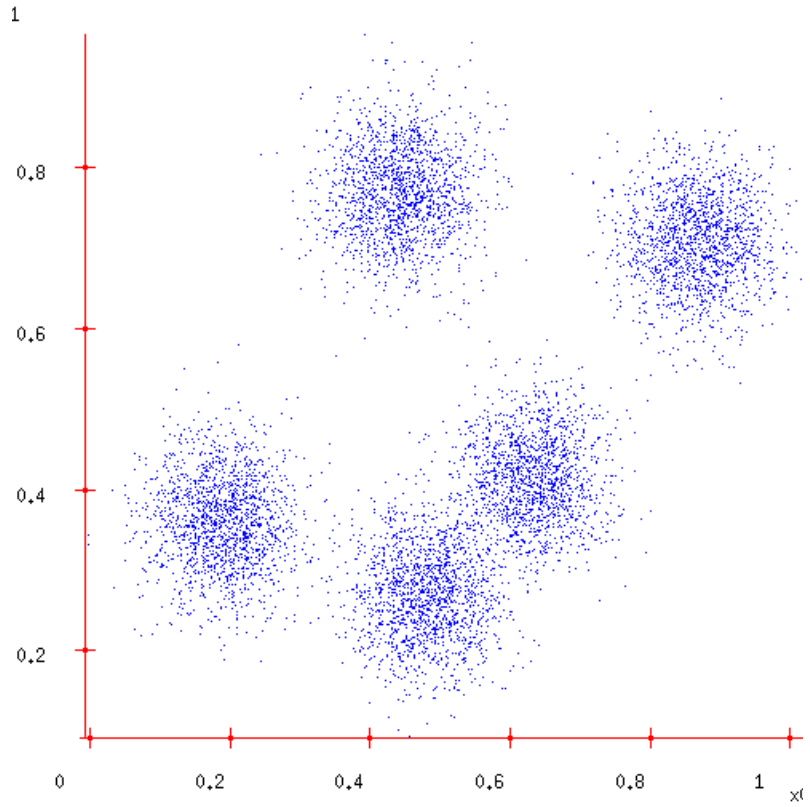
If k is too large=> small clusters (of size less than k) are likely to be labeled as noise
If k is too small=> Even a small number of closely spaced that are noise or outliers will be incorrectly labeled as clusters

Outline

- K-Means
- Hierarchical Clustering
- Graph-based/Spectral Clustering
- DBSCAN
- Model-based Clustering (GMM and Expectation Maximization)
- Evaluation of Clustering Algorithms

Model-based Clustering: Gaussian Mixture Model

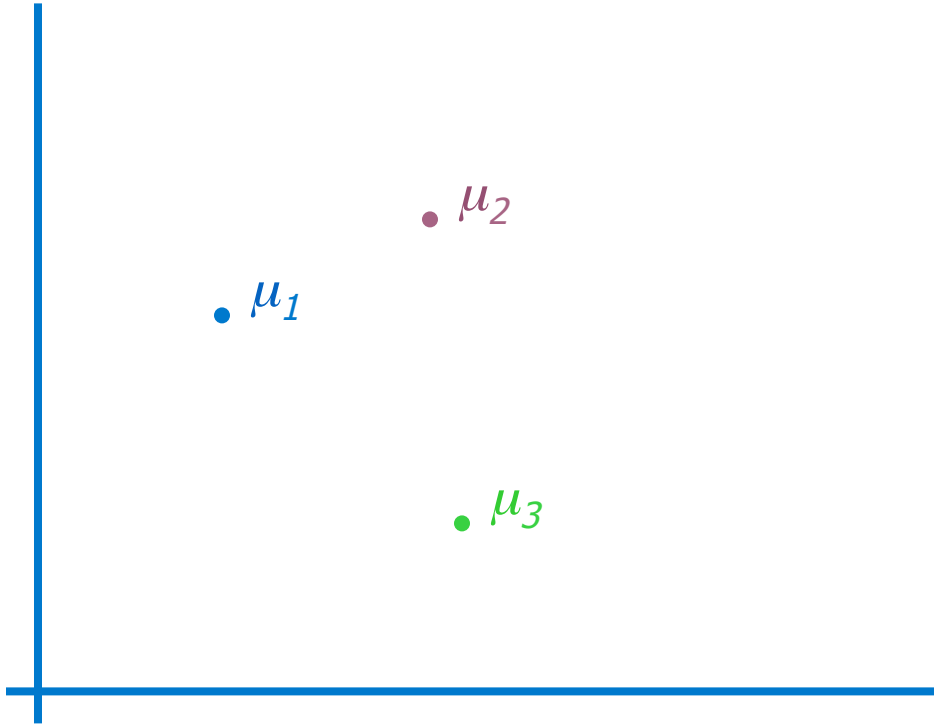
- Density estimation with multimodal/clumpy data



Slide Courtesy: Andrew Moore, CMU

Gaussian Mixture Model (GMM)

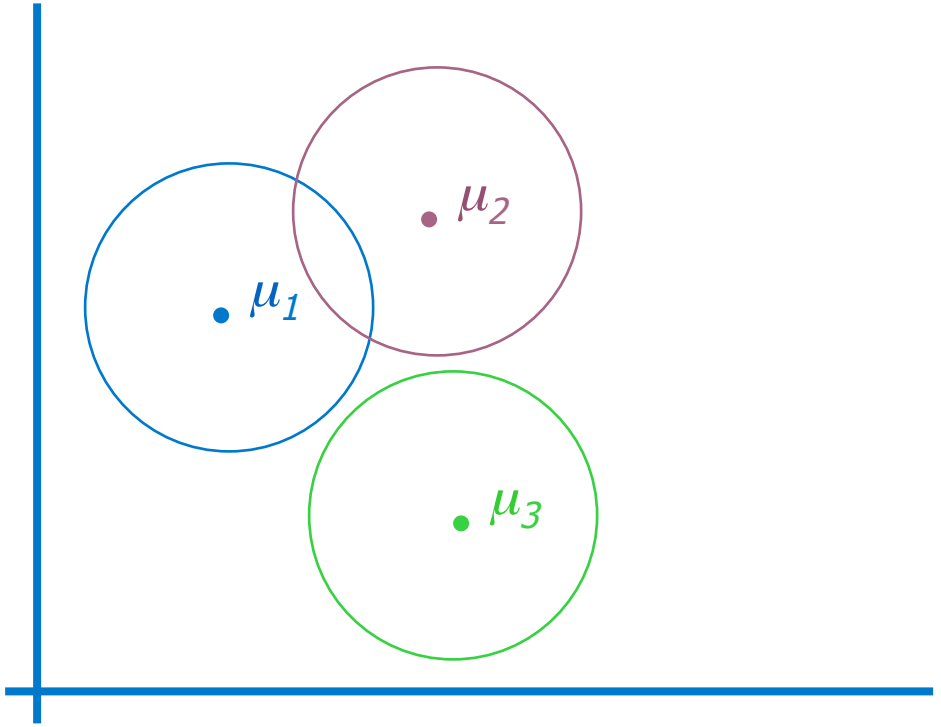
- The GMM assumption
- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i



Slide Courtesy: Andrew Moore, CMU

Gaussian Mixture Model (GMM)

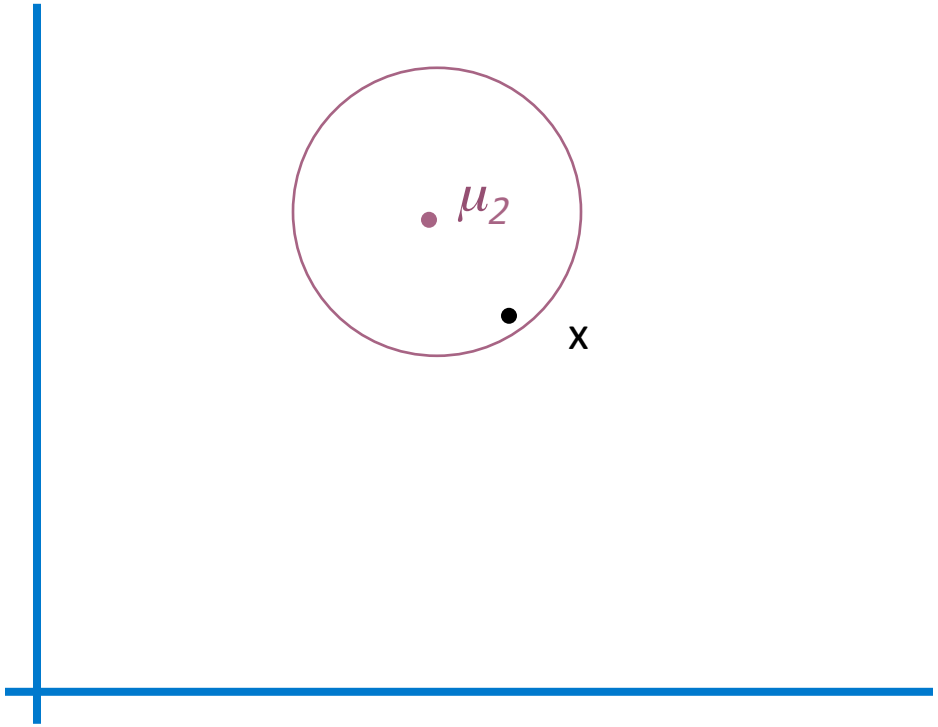
- The GMM assumption
- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 I$



Slide Courtesy: Andrew Moore, CMU

Gaussian Mixture Model (GMM)

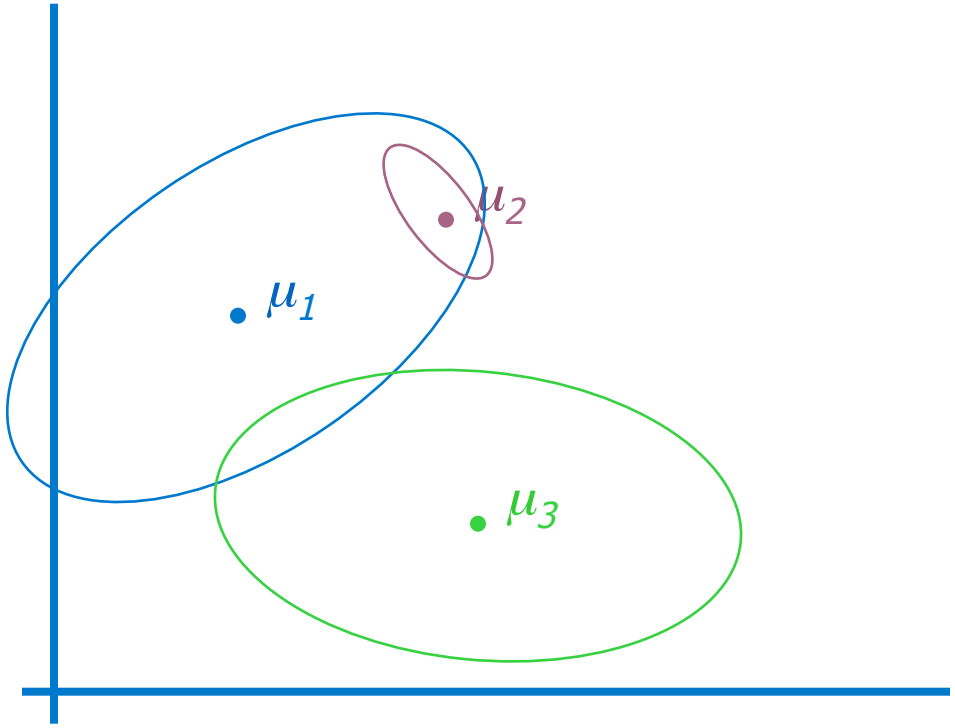
- The GMM assumption
- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 I$
- Assume that each datapoint is generated according to the following recipe:
  - Pick a component at random. Choose component i with probability $P(\omega_i)$.
  - Datapoint ~ $N(\mu_i, \Sigma_i)$



Slide Courtesy: Andrew Moore, CMU

Gaussian Mixture Model (GMM)

- The GMM assumption
- There are k components. The i'th component is called ω_i
- Component ω_i has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean μ_i and covariance matrix $\sigma^2 I$
- Assume that each datapoint is generated according to the following recipe:
  - Pick a component at random. Choose component i with probability $P(\omega_i)$.
  - Datapoint ~ $N(\mu_i, \Sigma_i)$



Slide Courtesy: Andrew Moore, CMU

Gaussian Mixture Model (GMM)

- Given the means, we can compute P(data | $\mu_1, \mu_2 .. \mu_k$). How do we find the μ_i s which give max.likelihood?
- The normal max likelihood trick:

  Set $\frac{d}{d\mu_i}$ log Prob (….) = 0

  and solve for μ_i's.
- Use gradient descent
  - Slow but doable
- Use a much faster and popular method: EM

Slide Courtesy: Andrew Moore, CMU

Expectation Maximization (EM)

- We'll get back to unsupervised learning/clustering/GMM soon.
- The EM algorithm was explained and given its name in a classic 1977 paper by Arthur Dempster, Nan Laird, and Donald Rubin.
- They pointed out that the method had been "proposed many times in special circumstances" by earlier authors.
- EM is typically used to compute maximum likelihood estimates given incomplete samples.
  - An excellent way of doing our unsupervised learning problem, as we'll see
  - Many, many other uses, including inference of Hidden Markov Models (future lecture)
- The EM algorithm estimates the parameters of a model iteratively. Starting from some initial guess, each iteration consists of
  - an E step (Expectation step)
  - an M step (Maximization step)

Slide Courtesy: Andrew Moore, CMU

EM: Trivial Example

Let events be "grades in a class"

w_1 = Gets an A P(A) = ½

w_2 = Gets a B P(B) = μ

w_3 = Gets a C P(C) = 2μ

w_4 = Gets a D P(D) = ½-3μ

(Note 0 ≤ μ ≤1/6)

Assume we want to estimate μ from data. In a given class, there were

a A's
b B's
c C's
d D's

What's the maximum likelihood estimate of μ given a,b,c,d ?

Slide Courtesy: Andrew Moore, CMU

EM: Trivial Example

P(A) = ½ P(B) = μ P(C) = 2μ P(D) = ½-3μ

P(*a,b,c,d* | μ) = $(½)^a(μ)^b(2μ)^c(½-3μ)^d$

log P(*a,b,c,d* | μ) = *a*log ½ + *b*log μ + *c*log 2μ + *d*log (½-3μ)

$$\text{FOR MAX LIKE } \mu, \text{SET } \frac{\partial \text{LogP}}{\partial \mu} = 0$$

$$\frac{\partial \text{LogP}}{\partial \mu} = \frac{b}{\mu} + \frac{2c}{2\mu} - \frac{3d}{1/2 - 3\mu} = 0$$

$$\text{Gives max like } \mu = \frac{b + c}{6(b + c + d)}$$

So if class got

A	B	C	D
14	6	9	10

$$\text{Max likelihood estimate} : \mu = \frac{1}{10}$$

Slide Courtesy: Andrew Moore, CMU

EM: Same Example with Hidden Info

Someone tells us that

Number of High grades (A's + B's) = h

Number of C's = c

Number of D's = d

What is the max likelihood estimate of μ now?

REMEMBER

P(A) = ½

P(B) = μ

P(C) = 2μ

P(D) = ½-3μ

Slide Courtesy: Andrew Moore, CMU

EM: Same Example with Hidden Info

Someone tells us that

Number of High grades (A's + B's) = h

Number of C's = c

Number of D's = d

What is the max likelihood estimate of μ now?

We can answer this circularly as below

REMEMBER

P(A) = ½

P(B) = μ

P(C) = 2μ

P(D) = ½-3μ

EXPECTATION

If we know the value of μ we could compute the expected value of a and b

Since the ratio a:b should be the same as the ratio ½ : m

$$a = \frac{1/2}{1/2 + \mu} h \qquad b = \frac{\mu}{1/2 + \mu} h$$

MAXIMIZATION

If we know the expected values of a and b we could compute the maximum likelihood value of μ

$$\mu = \frac{b + c}{6(b + c + d)}$$

Slide Courtesy: Andrew Moore, CMU

EM: Solution for Trivial Example

We begin with a guess for μ

We iterate between EXPECTATION and MAXIMALIZATION to improve our estimates of μ and *a* and *b*.

Define μ(t) the estimate of μ on the t^{th} iteration

b(t) the estimate of *b* on t^{th} iteration

E-step

$$\mu(0) = \text{initial guess}$$

$$b(t) = \frac{\mu(t)h}{1/2 + \mu(t)} = \mathrm{E}[b \mid \mu(t)]$$

M-step

$$\mu(t+1) = \frac{b(t) + c}{6(b(t) + c + d)}$$

$$= \max \text{ like est of } \mu \text{ given } b(t)$$

Continue iterating until converged.

Good news: Converging to local optimum is assured.

Bad news: "local" optimum.

Slide Courtesy: Andrew Moore, CMU

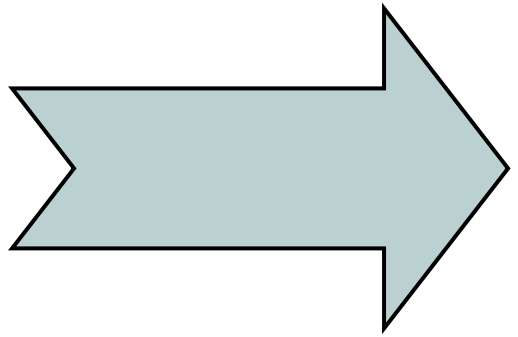
EM: Convergence

- Convergence proof based on fact that Prob(data | μ) must increase or remain same between each iteration [NOT OBVIOUS]
- But it can never exceed 1 [OBVIOUS]

So it must therefore converge [OBVIOUS]

In our example, suppose we had

h = 20
c = 10
d = 10
μ(0) = 0



t	μ(t)	b(t)
0	0	0
1	0.0833	2.857
2	0.0937	3.158
3	0.0947	3.185
4	0.0948	3.187
5	0.0948	3.187
6	0.0948	3.187

Convergence is generally linear: error decreases by a constant factor each time step.

Slide Courtesy: Andrew Moore, CMU

Back to GMM

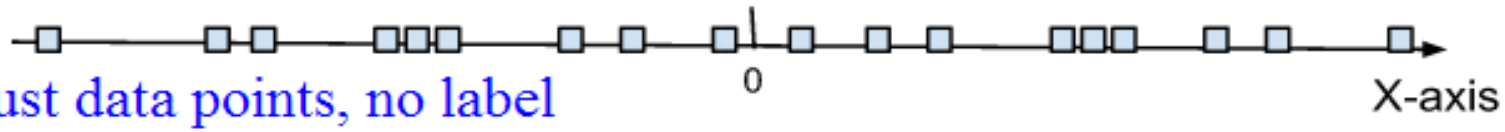
Given a training data set: X={x(1),x(2),...,x(n)}
Z={z(1),z(2),...,z(n)}
z(i) is the calss/group label of sample x(i).
As we are in Clustering setting,
X is Given and Z is unknown

Now, we model the data by specifying a joint distribution p(x(i), z(i))=p(x(i)|z(i))p(z(i))

$$z^{(i)} \sim \text{Multinomial}(\phi)$$
$$\phi_j \geq 0, \sum_{j=1}^{k} \phi_j = 1$$
$$k = \#\text{ of } z^{(i)}\text{'s values}$$
$$\phi_j = p(z^{(i)} = j)$$
$$x^{(i)} | z^{(i)} = j \sim \mathcal{N}(\mu_j, \Sigma_j)$$

⇨ each $x^{(i)}$ was generated by randomly choosing $z^{(i)}$ from $\{1, \ldots, k\}$, and then $x^{(i)}$ was drawn from one of k Gaussians.

The parameters of our model are thus ϕ, μ and Σ.



Just data points, no label

Slide Courtesy: Andrew Moore, CMU

EM for GMM

Incomplete Data

X={x(1),x(2),...,x(n)} Given
Z={z(1),z(2),...,z(n)} unknown

unknown

The parameters of our model ϕ, μ, Σ

What is the value of z(i)?

We can answer this question circularly:

EXPECTATION

If we know the expected values of Z we could compute the maximum likelihood value of ϕ, μ, Σ

MAXIMIZATION

If we know the values of ϕ, μ, Σ we could compute the expected values of Z

We begin with a guess for ϕ, μ, Σ, and then iterate between EXPECTATION and MAXIMALIZATION to improve our estimates of ϕ, μ, Σ and Z

Continue iterating until converged.

Slide Courtesy: Andrew Moore, CMU

EM for GMM

$$\ell(\phi, \mu, \Sigma) = \sum_{i=1}^{m} \log p(x^{(i)}|z^{(i)}; \mu, \Sigma) + \log p(z^{(i)}; \phi)$$

Maximizing this with respect to ϕ, μ and Σ gives the parameters:

$$\begin{aligned}
\phi_j &= \frac{1}{m}\sum_{i=1}^{m} 1\{z^{(i)} = j\}, \\
\mu_j &= \frac{\sum_{i=1}^{m} 1\{z^{(i)} = j\}x^{(i)}}{\sum_{i=1}^{m} 1\{z^{(i)} = j\}}, \\
\Sigma_j &= \frac{\sum_{i=1}^{m} 1\{z^{(i)} = j\}(x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^{m} 1\{z^{(i)} = j\}}.
\end{aligned}$$

Slide Courtesy: Andrew Moore, CMU

EM for GMM

Repeat until convergence: {

(E-step) For each i, j, set

$$w_j^{(i)} := p(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma)$$

(M-step) Update the parameters:

$$\phi_j := \frac{1}{m} \sum_{i=1}^{m} w_j^{(i)},$$

$$\mu_j := \frac{\sum_{i=1}^{m} w_j^{(i)} x^{(i)}}{\sum_{i=1}^{m} w_j^{(i)}},$$

$$\Sigma_j := \frac{\sum_{i=1}^{m} w_j^{(i)} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^{m} w_j^{(i)}}$$

}

Slide Courtesy: Andrew Moore, CMU

GMM vs k-Means

Given a training data set: X={x(1),x(2),...,x(n)}
Z={z(1),z(2),...,z(n)}
z(i) is the calss/group label of sample x(i).
As we are in Clustering setting,
X is Given and Z is unknown

Model of EM

EM model the data by specifying a joint distribution p(x(i), z(i))=p(x(i)|z(i))p(z(i))

$$z^{(i)} \sim \text{Multinomial}(\phi)$$
$$\phi_j \geq 0, \sum_{i=1}^{k} \phi_j = 1$$
$$k = \#\ \text{of}\ z^{(i)}\text{'s values}$$
$$\phi_j = p(z^{(i)} = j)$$
$$x^{(i)} | z^{(i)} = j \sim \mathcal{N}(\mu_j, \Sigma_j)$$

⇨ each $x^{(i)}$ was generated by randomly choosing $z^{(i)}$ from $\{1, \ldots, k\}$, and then $x^{(i)}$ was drawn from one of k Gaussians.

K-mans is a simplified EM, it assumes that

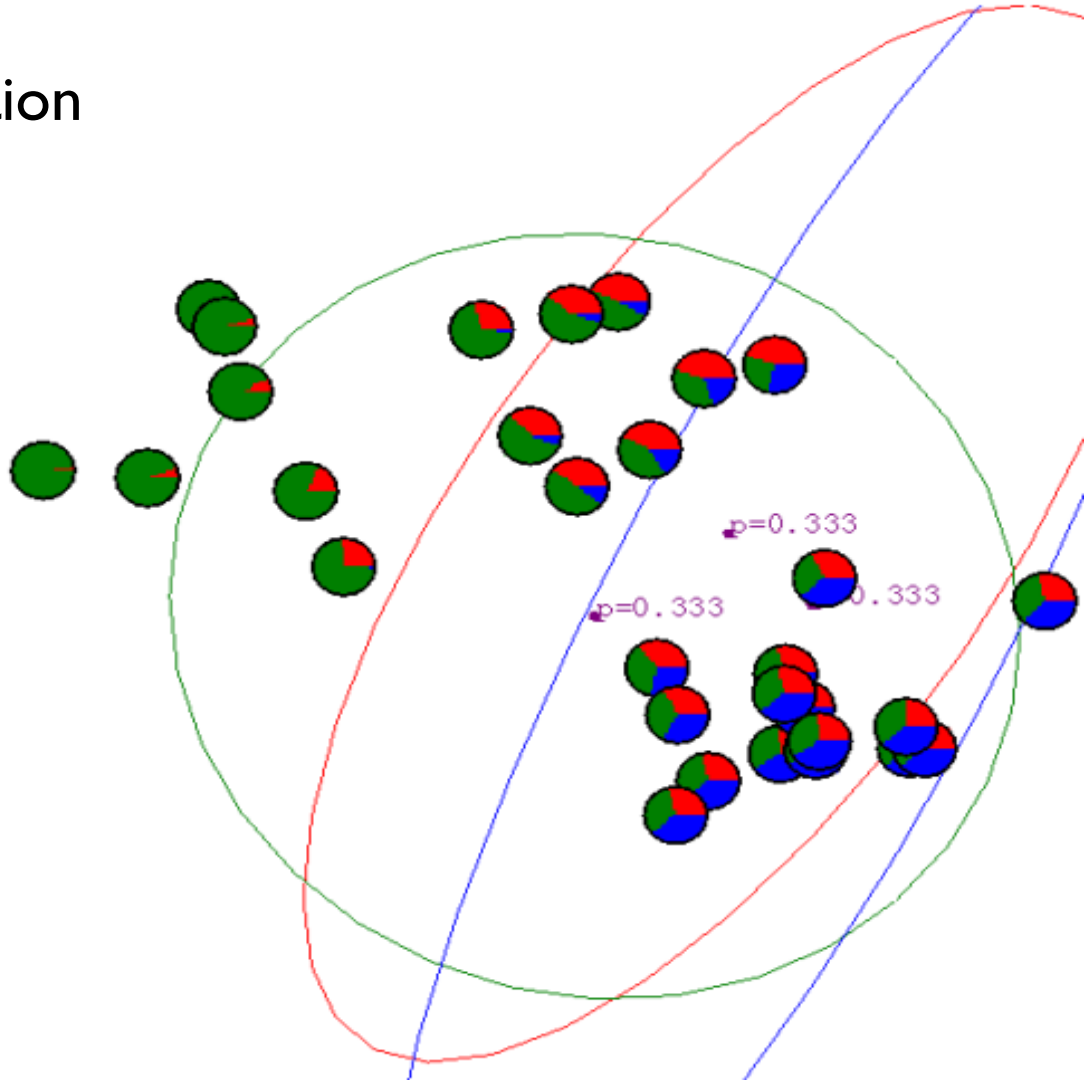
$\phi_j = \phi_i = 1/k$, and $\Sigma_j = \Sigma_i$ for i, j=1,2,...k
k is given by user

⇨ μ_1, μ_2 ..... μ_k are the only unknown parameters of the model (the means of clusters)

Slide Courtesy: Andrew Moore, CMU

GMM: Example

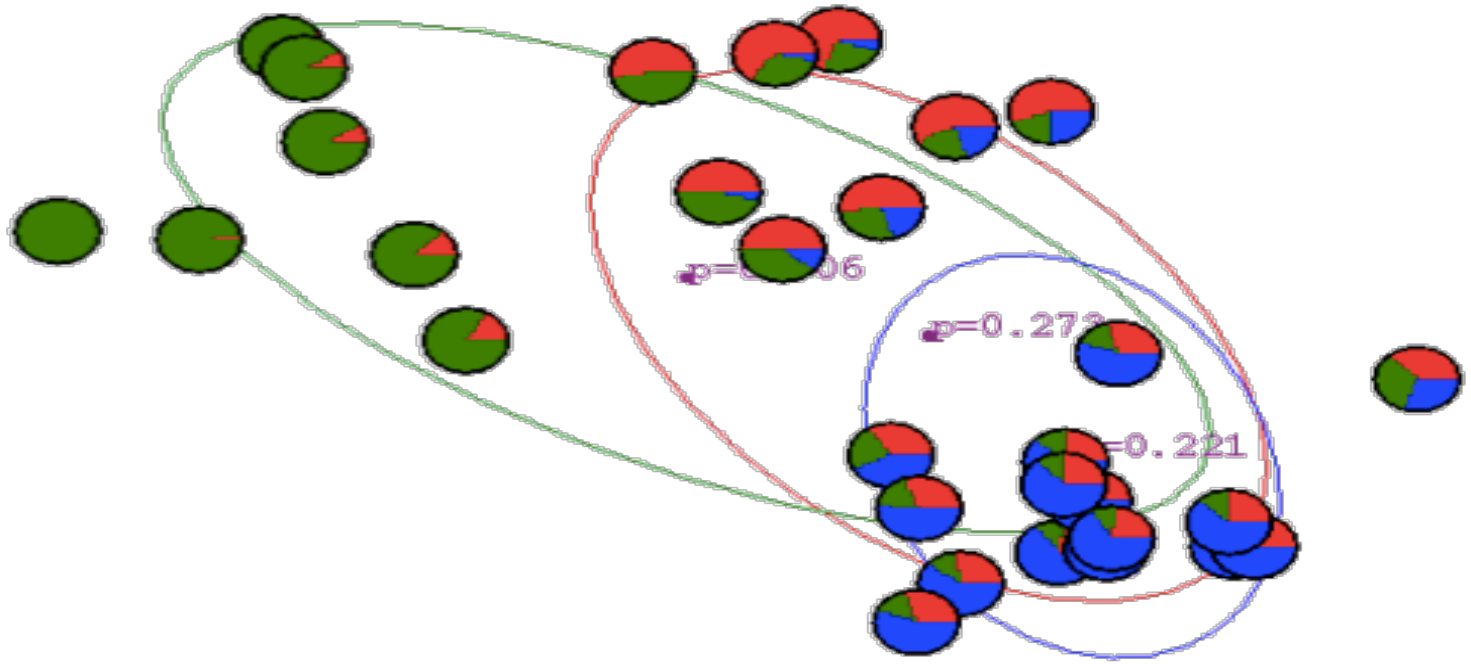
Start: 0^{th} iteration



Slide Courtesy: Andrew Moore, CMU

GMM: Example

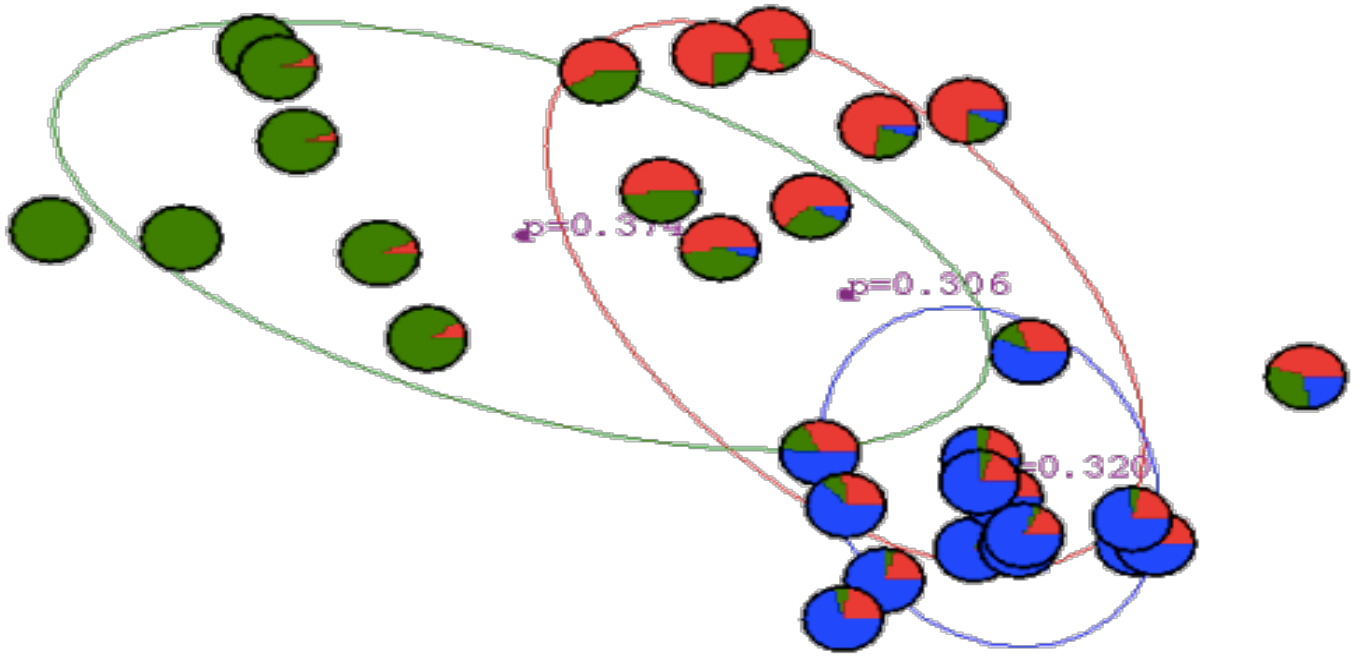
After 1st iteration



Slide Courtesy: Andrew Moore, CMU

GMM: Example

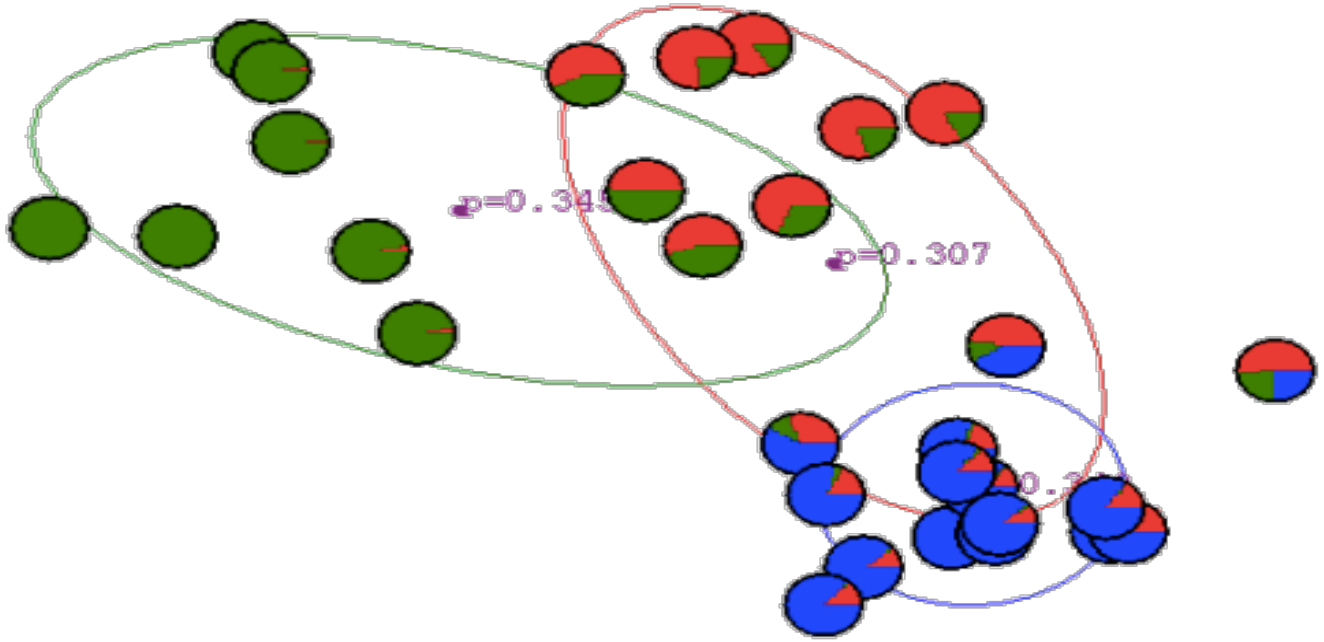
After 2nd iteration



Slide Courtesy: Andrew Moore, CMU

GMM: Example

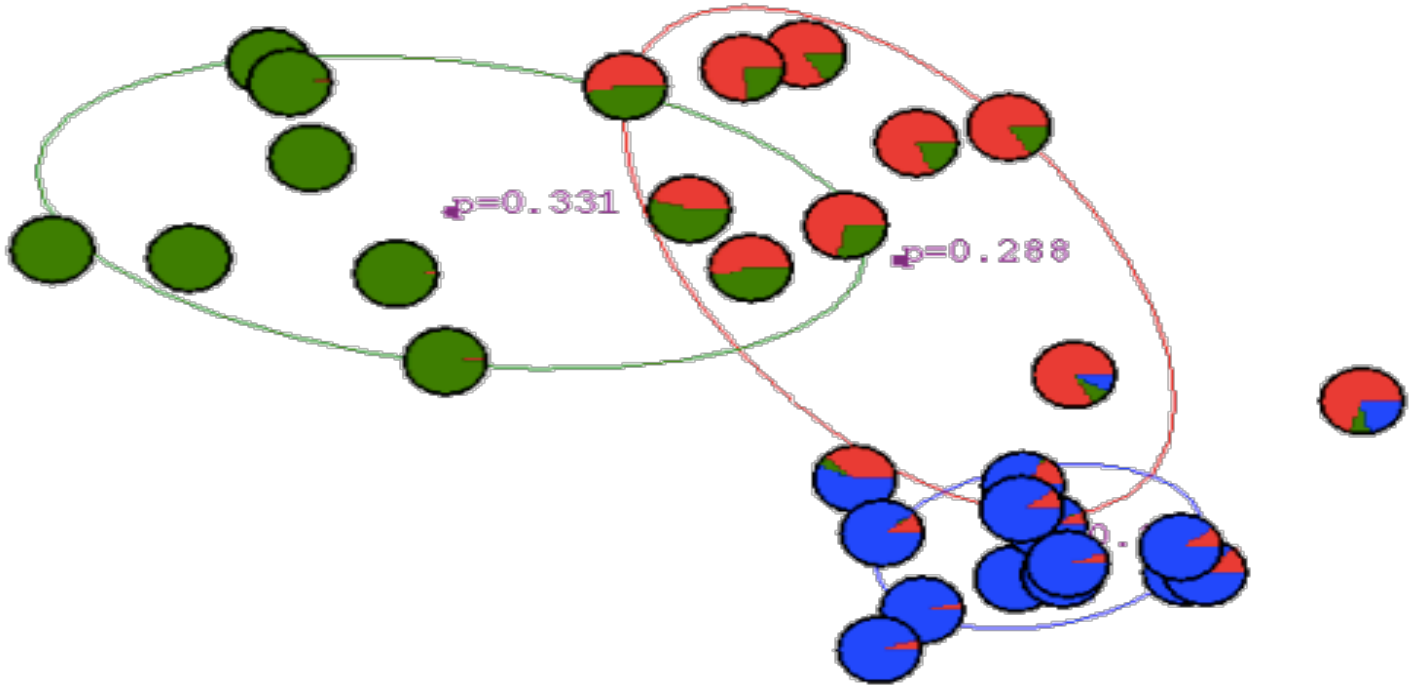
After 3rd iteration



Slide Courtesy: Andrew Moore, CMU

GMM: Example

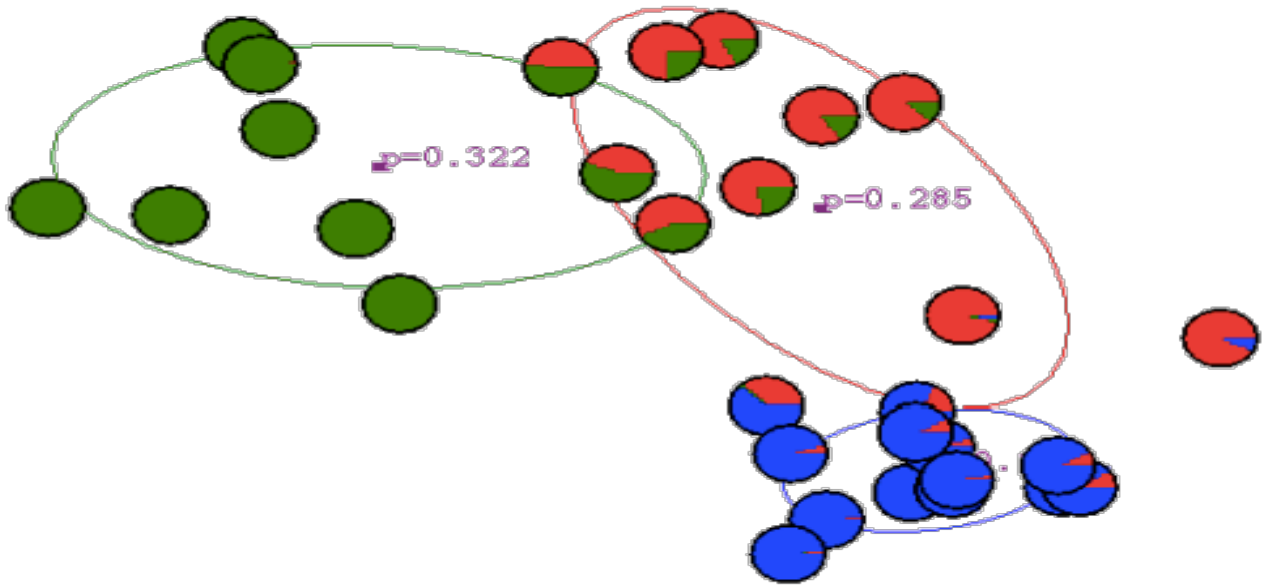
After 4th iteration



Slide Courtesy: Andrew Moore, CMU

GMM: Example

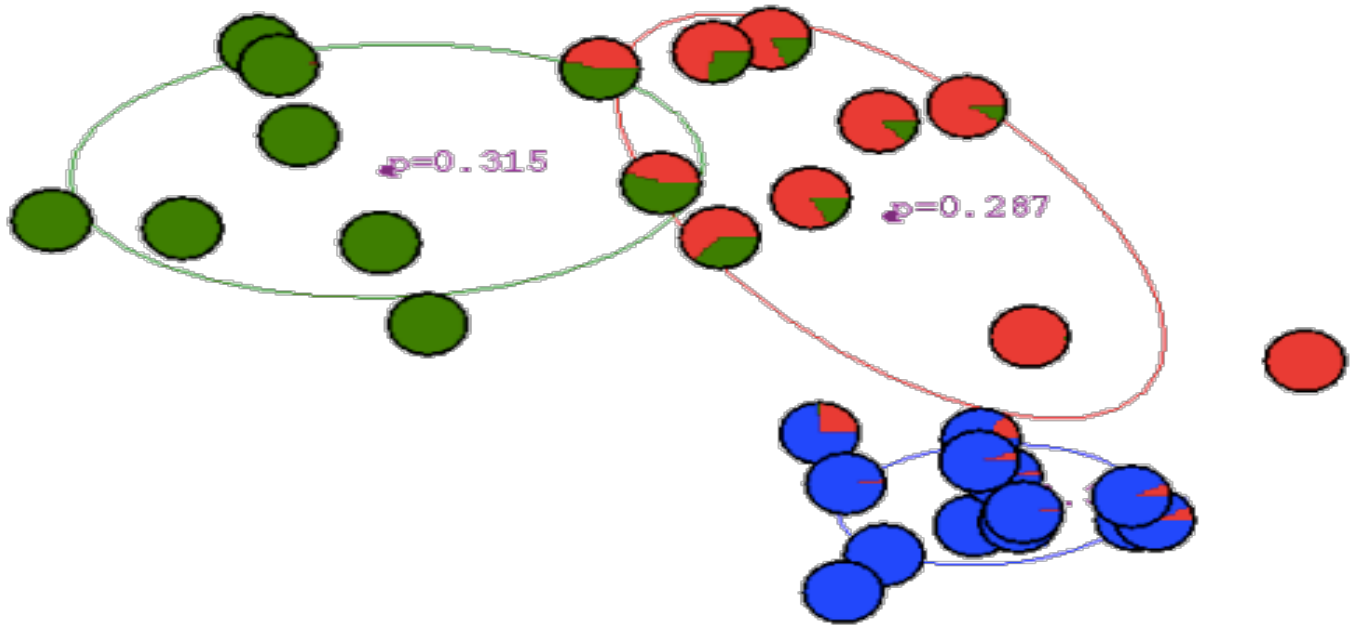
After 5th iteration



Slide Courtesy: Andrew Moore, CMU

GMM: Example

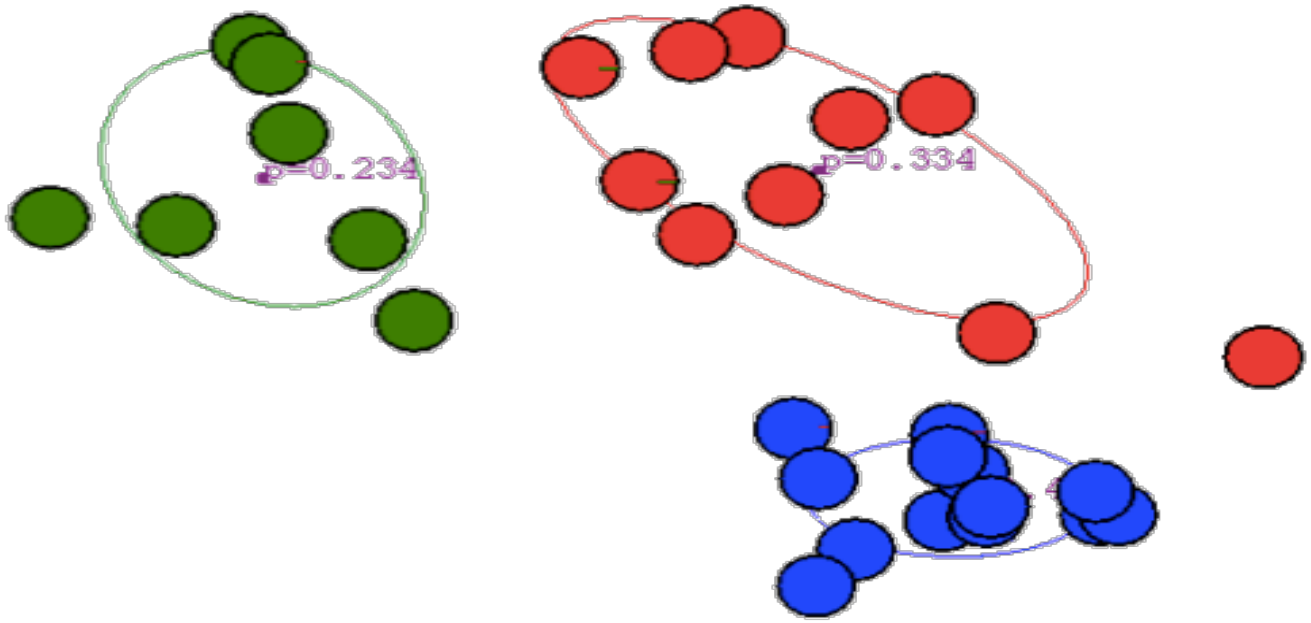
After 6th iteration



Slide Courtesy: Andrew Moore, CMU

GMM: Example

After 20th iteration



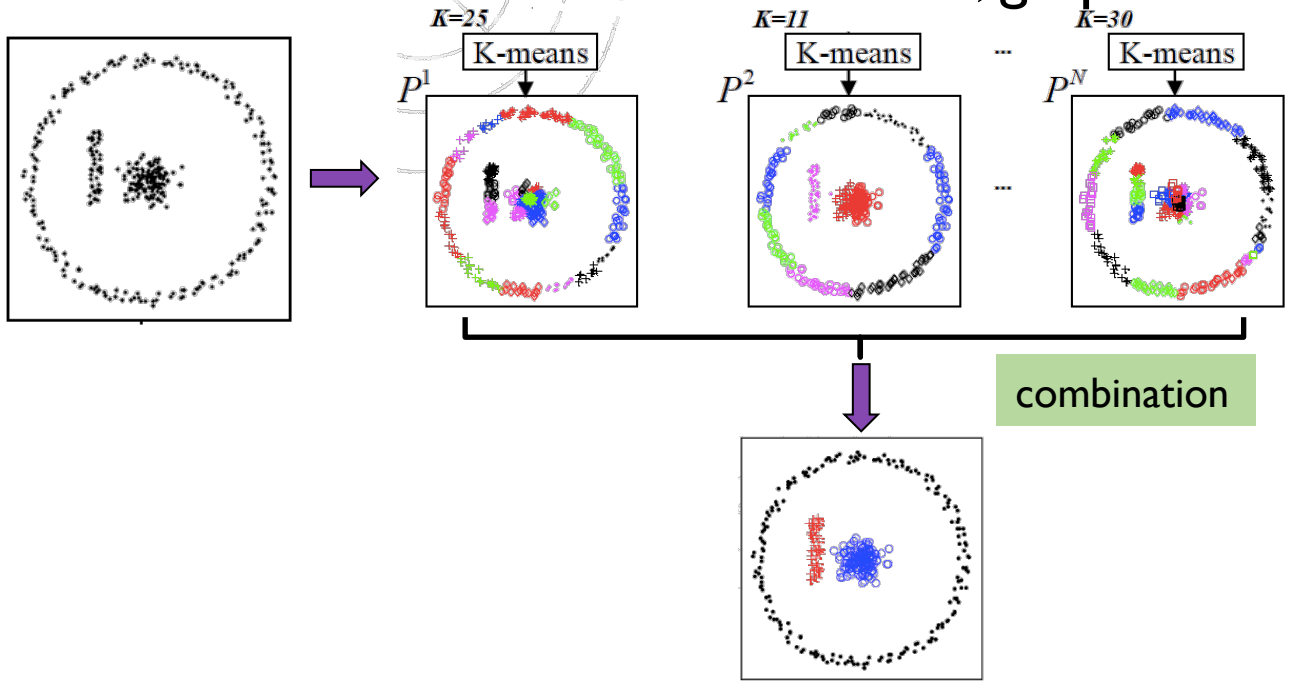
Slide Courtesy: Andrew Moore, CMU

More on EM Algorithm

- What are the EM algorithm initialization methods?
  - Random guess.
  - Any general classifier that builds a parameterized probability distribution model (i.e. naive Bayes).
  - Initialized by k-means. After a few iterations of k-means, using the parameters to initialize EM
- What are the main advantages of parametric methods?
  - You can easily change the model to adapt to different distribution of data sets.
  - Knowledge representation is very compact. Once the model is selected, the model is represented by a specific number of parameters.
  - The number of parameters does not increase with the increasing of training data .

Clustering Ensembles

- Clustering ensemble approach
  - Combine multiple clustering results (different partitions)
  - Typical methods: Evidence-accumulation based, graph-based ……

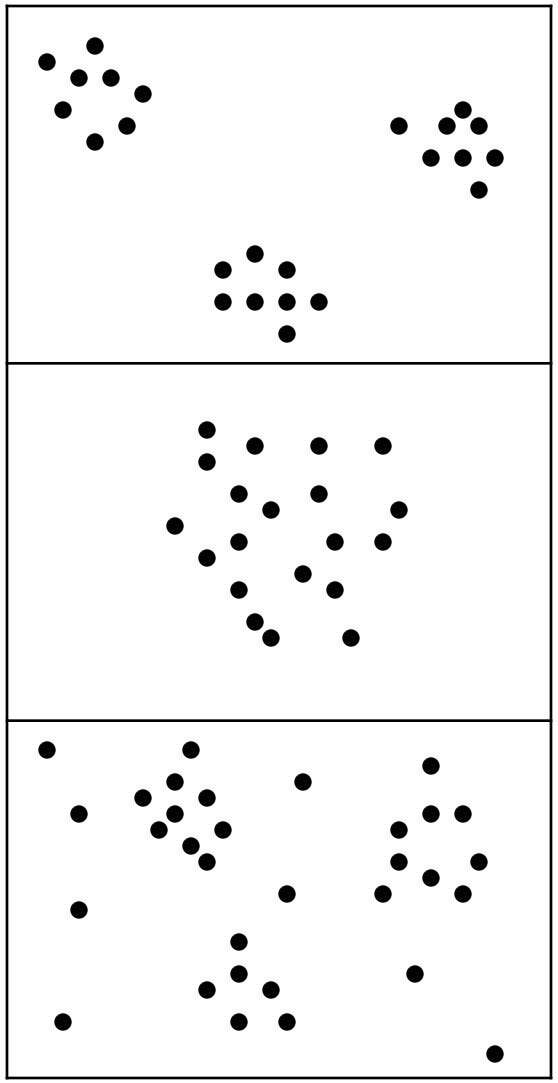


Clustering Methods: Summary

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of pdf
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

As we go down this chart, the clustering strategies have more tendency to transitively group points even if they are not nearby in feature space

Clustering Methods: Summary



Sometimes easy

Sometimes impossible

and sometimes in between

Outline

- K-Means
- Hierarchical Clustering
- Graph-based/Spectral Clustering
- DBSCAN
- Model-based Clustering (GMM and Expectation Maximization)
- Evaluation of Clustering Algorithms

Cluster Validity

- 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

Internal Measures

- 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)

$$WSS = \sum_{i} \sum_{x \in C_i} (x - m_i)^2$$

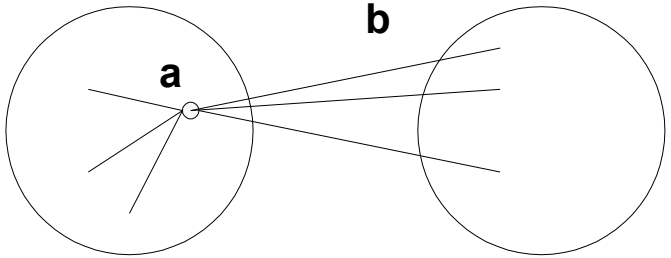
  - Separation is measured by the between cluster sum of squares

$$BSS = \sum_{i} |C_i| (m - m_i)^2$$

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

Internal Measures: Silhouette Coefficient

- Combines ideas of both cohesion and separation, but for individual points as well as clusters
- 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
    $s = 1 - a/b$ if $a < b$, (or $s = b/a - 1$ if $a \geq b$, not the usual case)
  - Typically between 0 and 1.
  - The closer to 1 the better.



External Measures

Table 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., $e = \sum_{i=1}^{K} \frac{m_i}{m} e_j$, 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 $purity_j = \max p_{ij}$ and the overall purity of a clustering by $purity = \sum_{i=1}^{K} \frac{m_i}{m} purity_j$.

Open Issues with Clustering Methods

- Finding the number of "natural" clusters with arbitrary shapes
- Dealing with mixed types of features
- Handling massive amount of data – Big Data
- Coping with data of high dimensionality
- Performance evaluation (especially when no ground-truth available)
- "Holes" in the dataset – how to find?
  - E.g. in a disease database, we may find that:
    - certain symptoms and/or test values do not occur together, or
    - when a certain medicine is used, some test values never go beyond certain ranges
  - Discovery of such information can be important
    - Could mean the discovery of a cure to a disease or some biological laws

Readings

- "Introduction to Machine Learning" by Ethem Alpaydin, Chapter 7