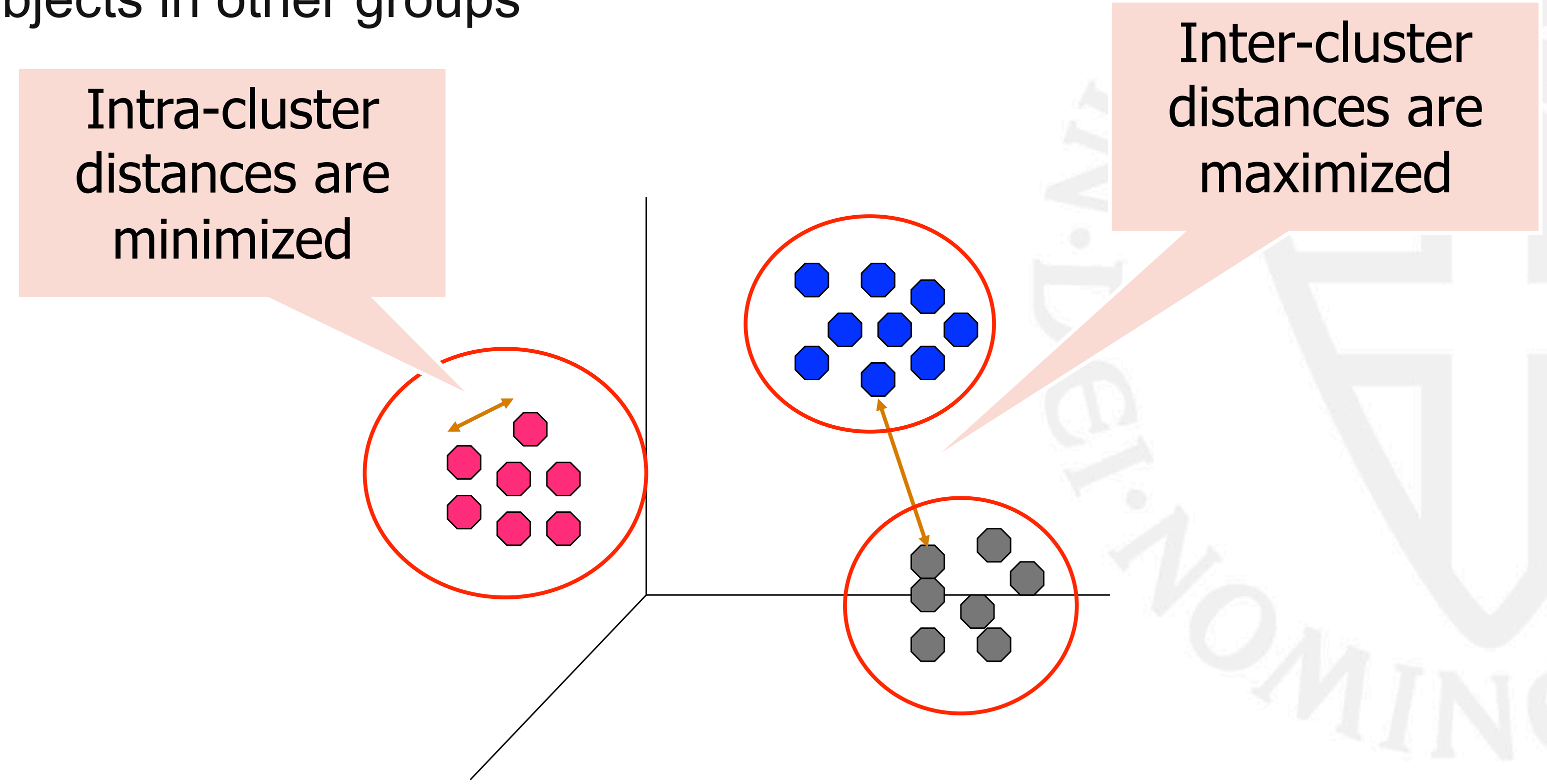
Data Mining: Clustering

Tom Claassen

What is Cluster Analysis?

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


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

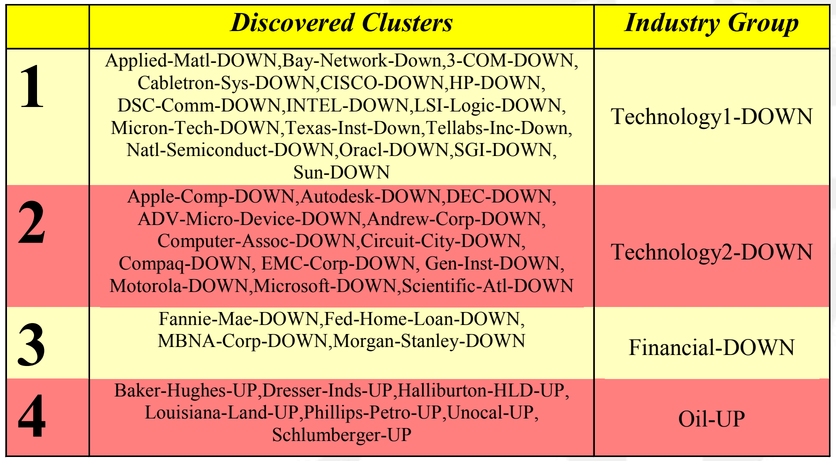
Applications of Cluster Analysis

Understanding

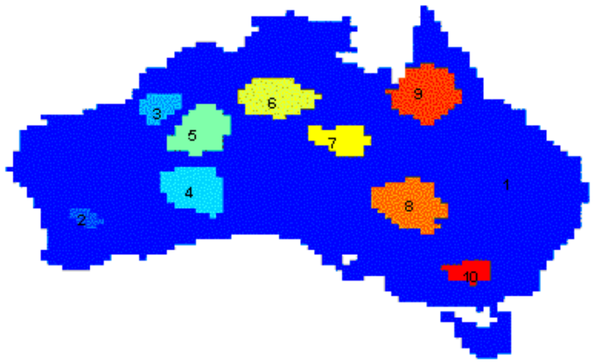
- Group related documents for browsing, group genes and proteins that have similar functionality, or group stocks with similar price fluctuations

Summarization

- Reduce the size of large data sets



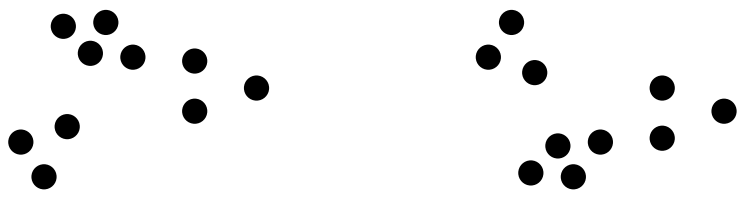
	Discovered Clusters	***Industry Group***
1	Applied-Matl-DOWN,Bay-Network-Down,3-COM-DOWN, Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN, Micron-Tech-DOWN,Texas-Inst-Down,Tellabs-Inc-Down, Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN	Technology1-DOWN
2	Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN, ADV-Micro-Device-DOWN,Andrew-Corp-DOWN, Computer-Assoc-DOWN,Circuit-City-DOWN, Compaq-DOWN, EMC-Corp-DOWN, Gen-Inst-DOWN, Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN	Technology2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP	Oil-UP

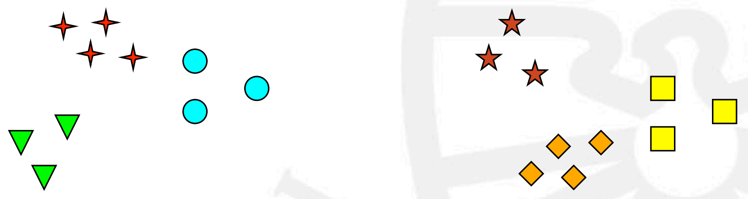


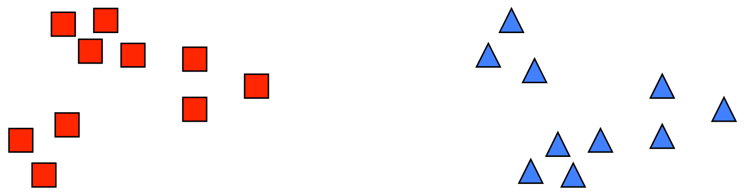
What is not Cluster Analysis?

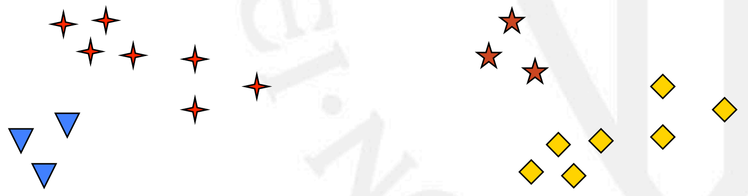
- Supervised classification
  - Have class label information
- Simple segmentation
  - Dividing students into different registration groups alphabetically, by last name
- Results of a query
  - Groupings are a result of an external specification
- Graph partitioning
  - Some mutual relevance and synergy, but areas are not identical

Notion of a Cluster can be Ambiguous


How many clusters?


Six Clusters

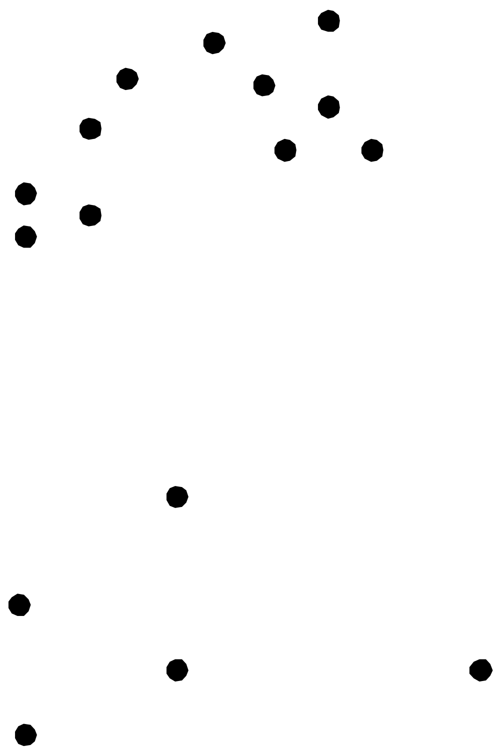

Two Clusters

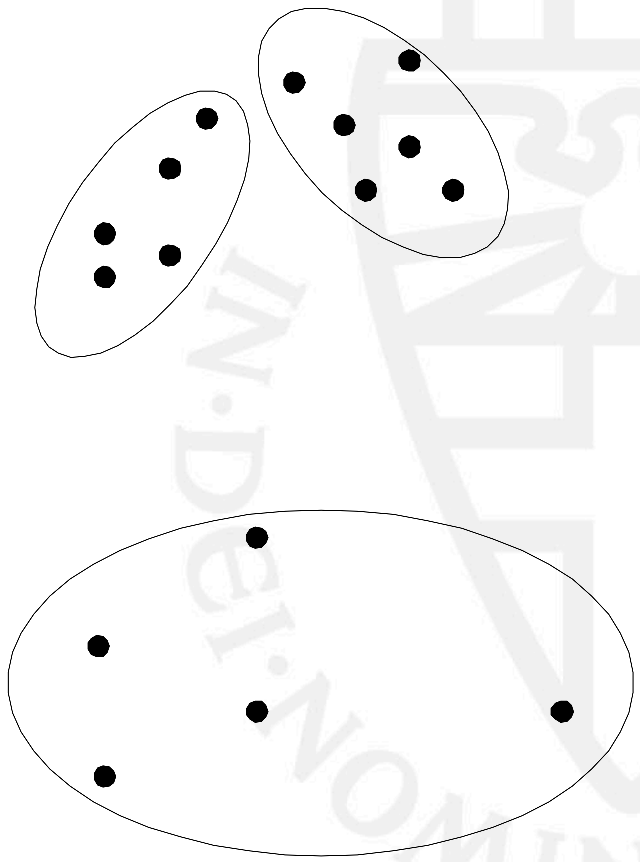

Four Clusters

Types of Clusterings

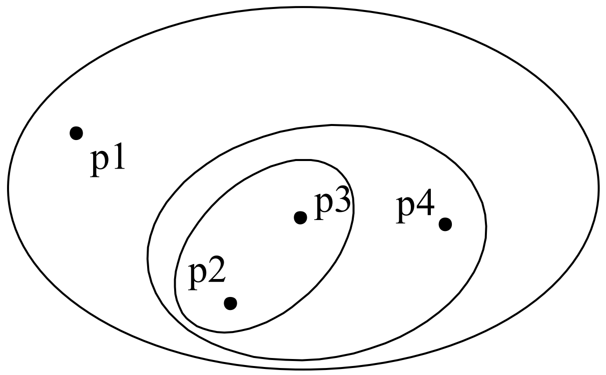
- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- Partitional Clustering
  - A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- Hierarchical clustering
  - A set of nested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

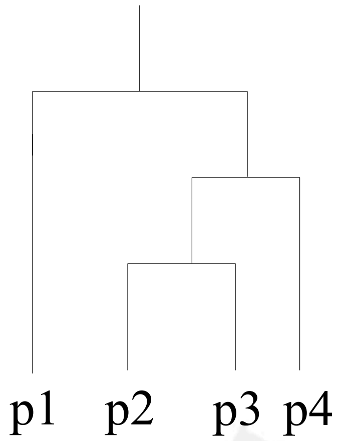
Partitional Clustering

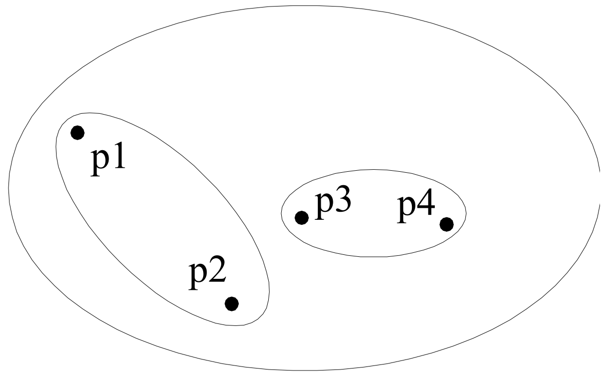

Original Points


Partitional Clustering

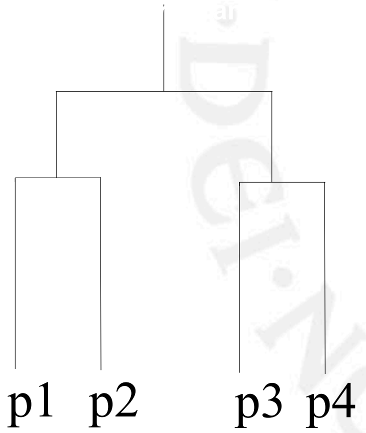
Hierarchical Clustering







Hierarchical Clustering



Dendrogram

Other Distinctions Between Sets of Clusters

- Exclusive versus non-exclusive
  - In non-exclusive clusterings, points may belong to multiple clusters.
  - Can represent multiple classes or ‘border’ points

- Fuzzy versus non-fuzzy
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
  - Weights must sum to 1
  - Probabilistic clustering has similar characteristics

- Partial versus complete
  - In some cases, we only want to cluster some of the data

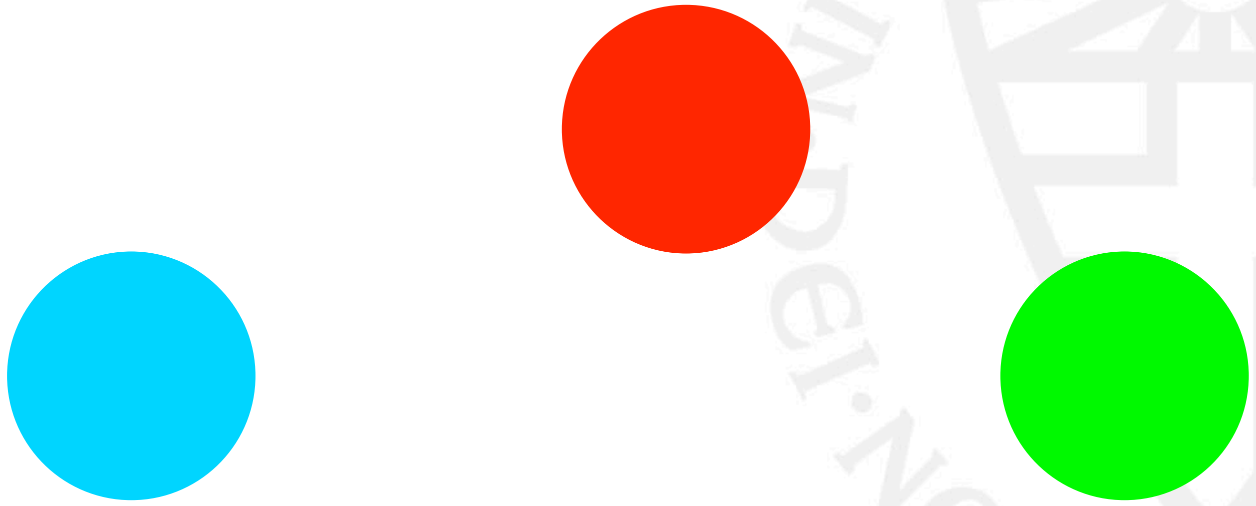
- Heterogeneous versus homogeneous
  - Cluster of widely different sizes, shapes, and densities

Types of Clusters

- Well-separated clusters
- Center-based clusters
- Contiguous clusters
- Density-based clusters
- Property or Conceptual
- Based upon an objective function

Types of Clusters: Well-Separated

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

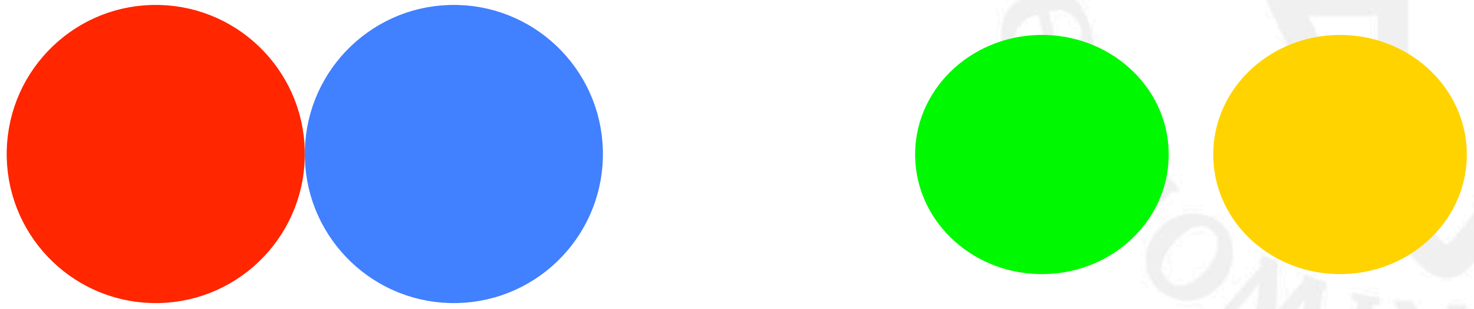


3 well-separated clusters

Types of Clusters: Center-Based

- A center-based cluster is a set of objects such that an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster

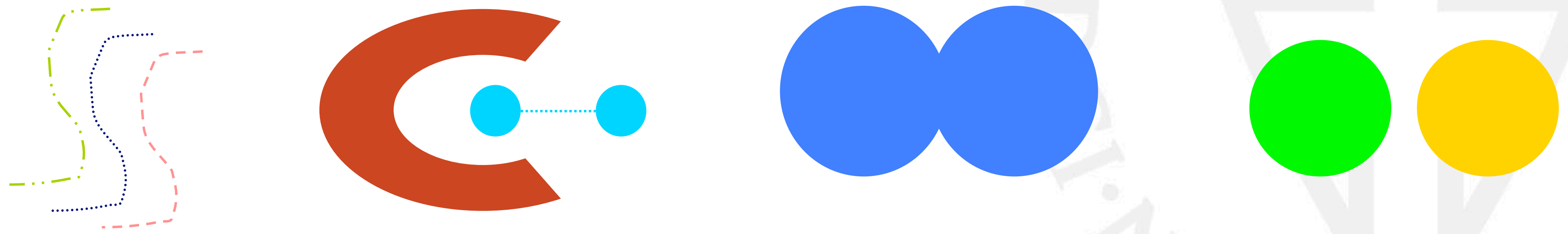
- The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most "representative" point of a cluster



4 center-based clusters

Types of Clusters: Contiguity-Based

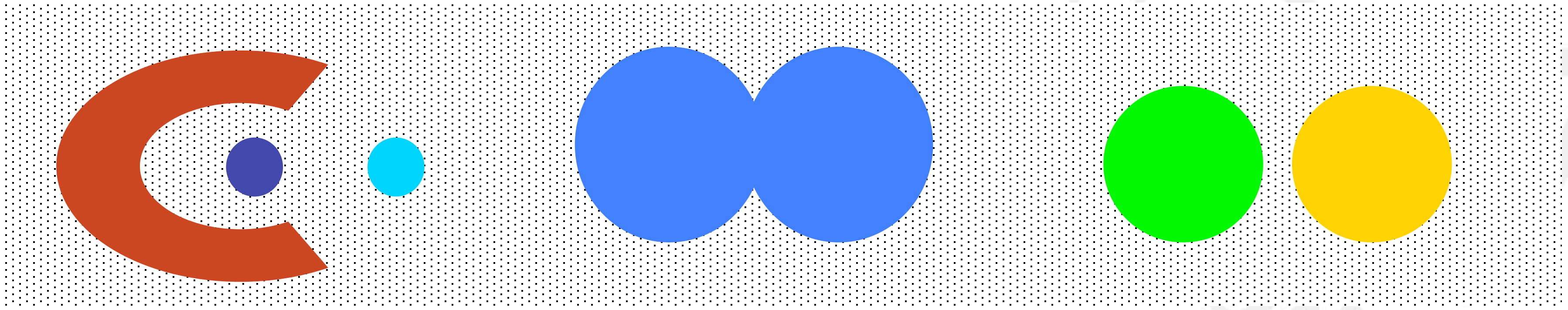
A contiguous (nearest-neighbor, transitive) cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster



8 contiguous clusters

Types of Clusters: Density-Based

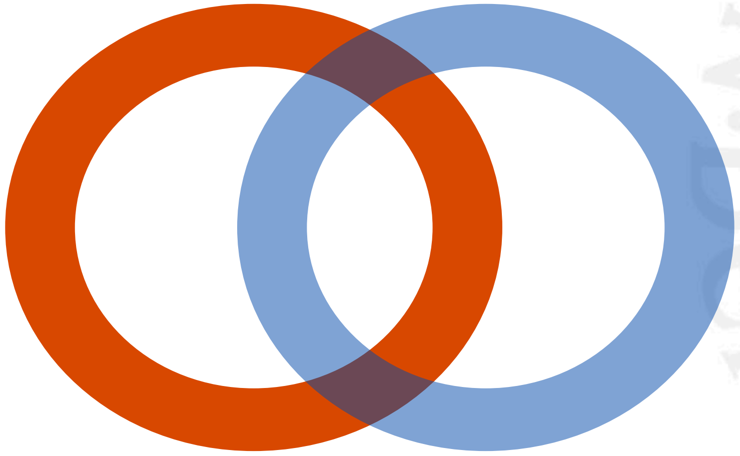
- A density-based cluster is a dense region of points, which is separated by low-density regions, from other regions of high density
- Used when the clusters are irregular or intertwined, and when noise and outliers are present



6 density-based clusters

Types of Clusters: Conceptual Clusters

A conceptual (shared-property) cluster contains points that jointly share some common property or represent a particular concept



2 overlapping circles

Types of Clusters: Based upon an Objective Function (1)

- Many objective functions that define the “goodness” of a clustering
- Specific algorithms for optimizing such an objective functions
- Finding the optimal solution often requires enumerating all possible ways of dividing the points into clusters (NP Hard)
- Can have global or local objectives
  - Hierarchical clustering algorithms typically have local objectives
  - Partitional algorithms typically have global objectives

Types of Clusters: Based upon an Objective Function (2)

- A variation of the global objective function approach is to fit the data to a parameterized statistical model
  - Parameters for the model are determined from the data.
  - Mixture models assume that the data is a ‘mixture' of a number of statistical distributions.

- Alternative approach is to map the clustering problem to a different domain and solve a related problem in that domain
  - E.g., proximity matrix defines a weighted graph, where the nodes are the points being clustered, and the weighted edges represent the proximities between points
  - Clustering is equivalent to breaking the graph into connected components, one for each cluster: minimum spanning tree algorithm

Characteristics of the Input Data Are Important

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

Clustering Algorithms

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

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 Clustering – Details

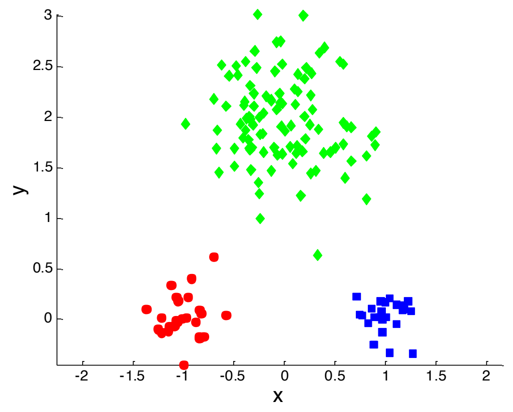
- Initial centroids are often chosen randomly
  - Clusters produced vary from one run to another
- The centroid is (typically) the mean of the points in the cluster
- ‘Closeness’ is measured by Euclidean distance, cosine similarity, correlation, etc.
- MATLAB:

```
% X: data matrix, K: number of clusters

[idx,C] = kmeans(X,K)
```

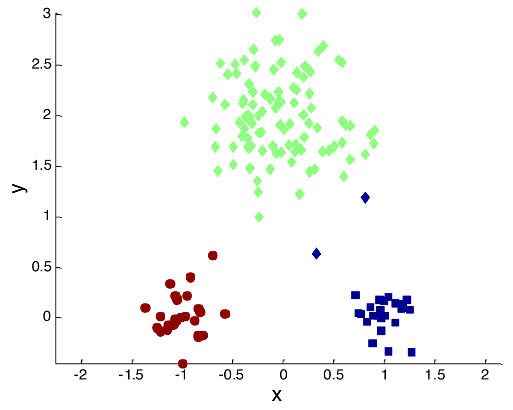
K-means Clustering – Convergence and Complexity

- K-means will converge for common similarity measures
- Most of the convergence happens in the first few iterations
- Often the stopping condition is changed to ‘Until relatively few points change clusters’
- Complexity is O(n K I d)
  - n = number of points, K = number of clusters, I = number of iterations, d = number of attributes

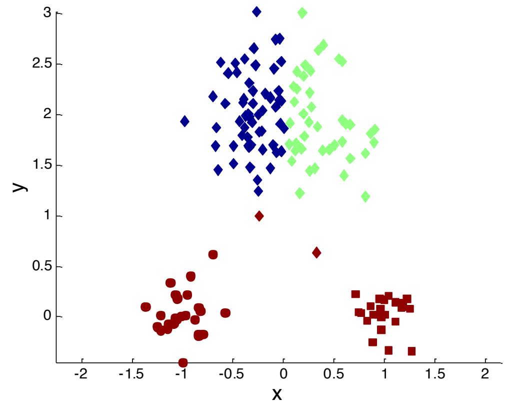
Two different K-means Clusterings



Original Points

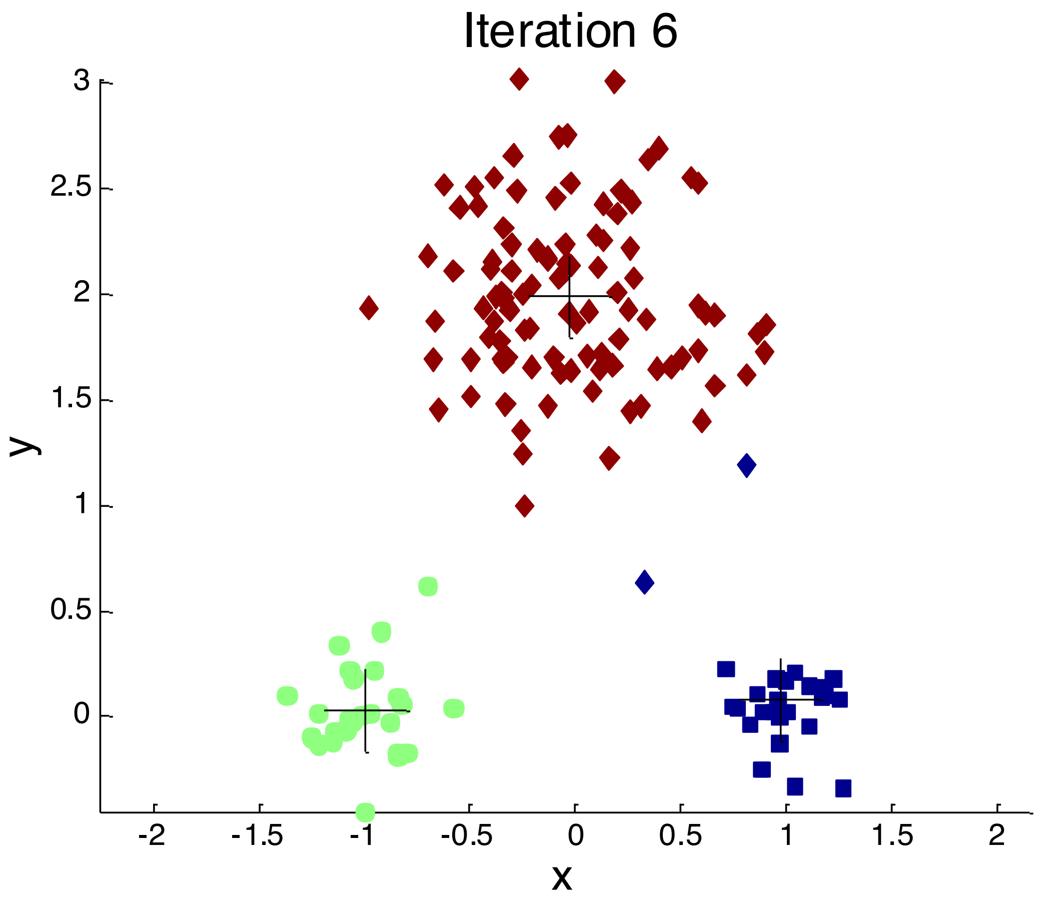


Optimal Clustering

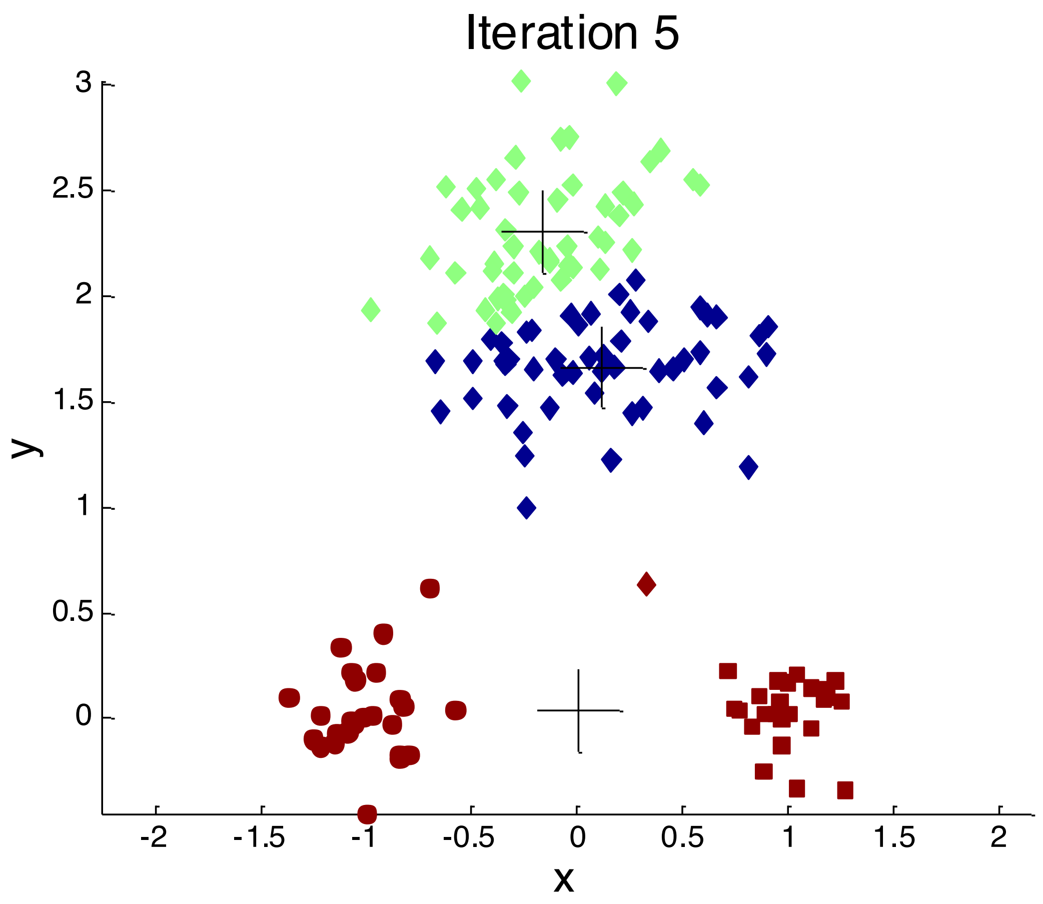


Sub-optimal Clustering

Importance of Choosing Initial Centroids (1)



Importance of Choosing Initial Centroids (2)



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:

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} |\mathbf{m}_i - \mathbf{x}|^2$$

  - $\mathbf{x}$ is a data point in cluster C_i and $\mathbf{m}_i$ is the representative point for cluster C_i

- Given two clusterings, the one with the smallest error is “better”

- One easy way to reduce SSE is to increase *K*, the number of clusters, so it typically only makes sense to compare *SSE*’s for clusterings with the same *K*

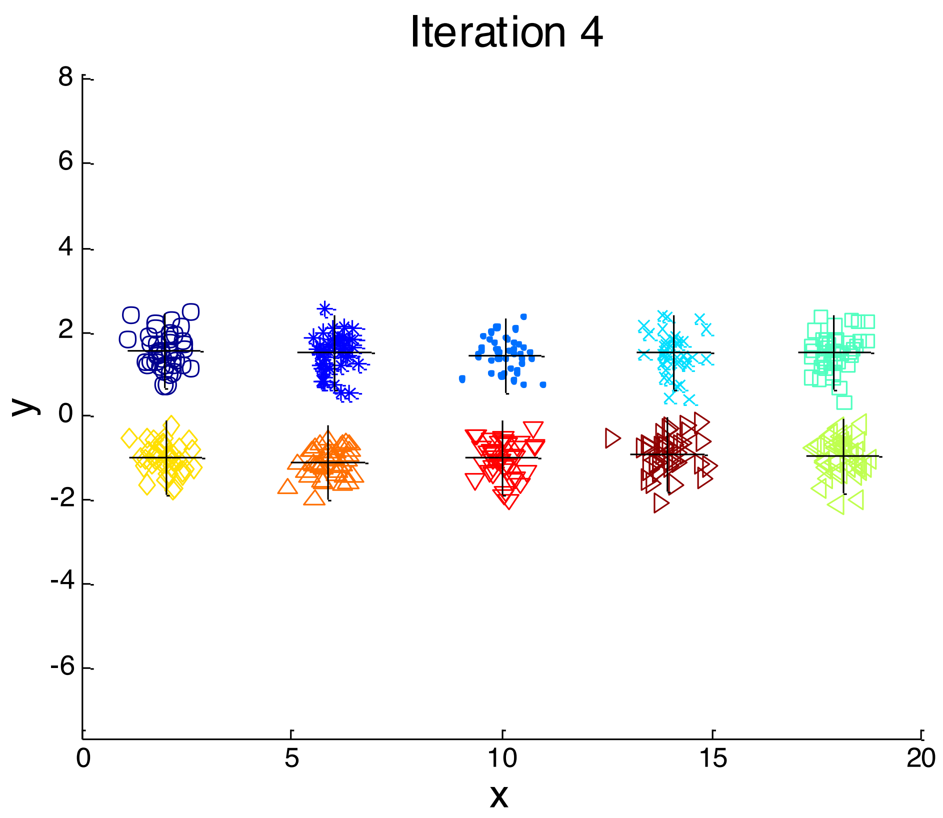
Problems with Selecting Initial Points

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

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

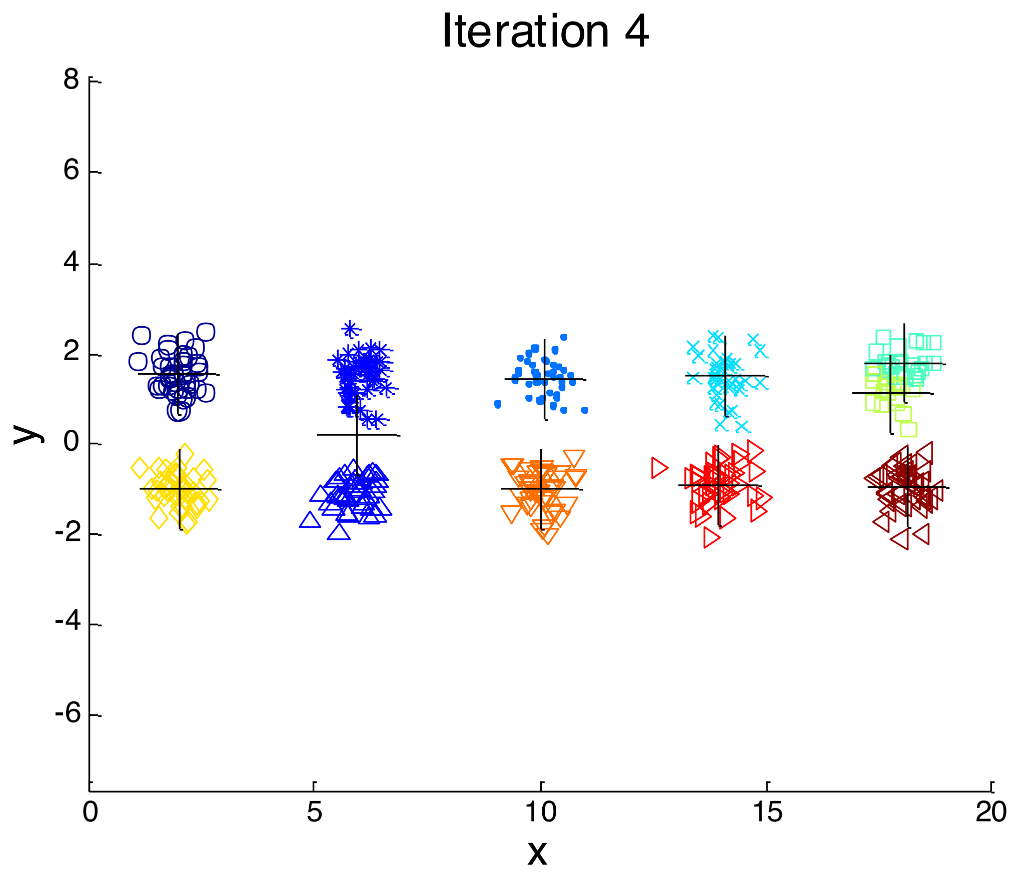
  - For example, if *K* = 10, then probability = 10!/1010 = 0.00036
- Sometimes the initial centroids will readjust themselves in ‘right’ way, and sometimes they don’t...

10 Clusters Example (1)



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

10 Clusters Example (2)



Starting with some pairs of clusters having three initial centroids, others only one

Solutions to Initial Centroids Problem

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

Handling Empty Clusters

- Basic K-means algorithm can yield empty clusters
- Idea: set the cluster center of an empty cluster equal to one of the data points and continue updating
- Several strategies
  - Choose the data point that contributes most to SSE
  - Choose a data point from the cluster with the highest SSE
  - If there are several empty clusters, the above can be repeated several times

Updating Centers Incrementally

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

Pre-processing and Post-processing

- Pre-processing
  - Normalize the data
  - Eliminate outliers

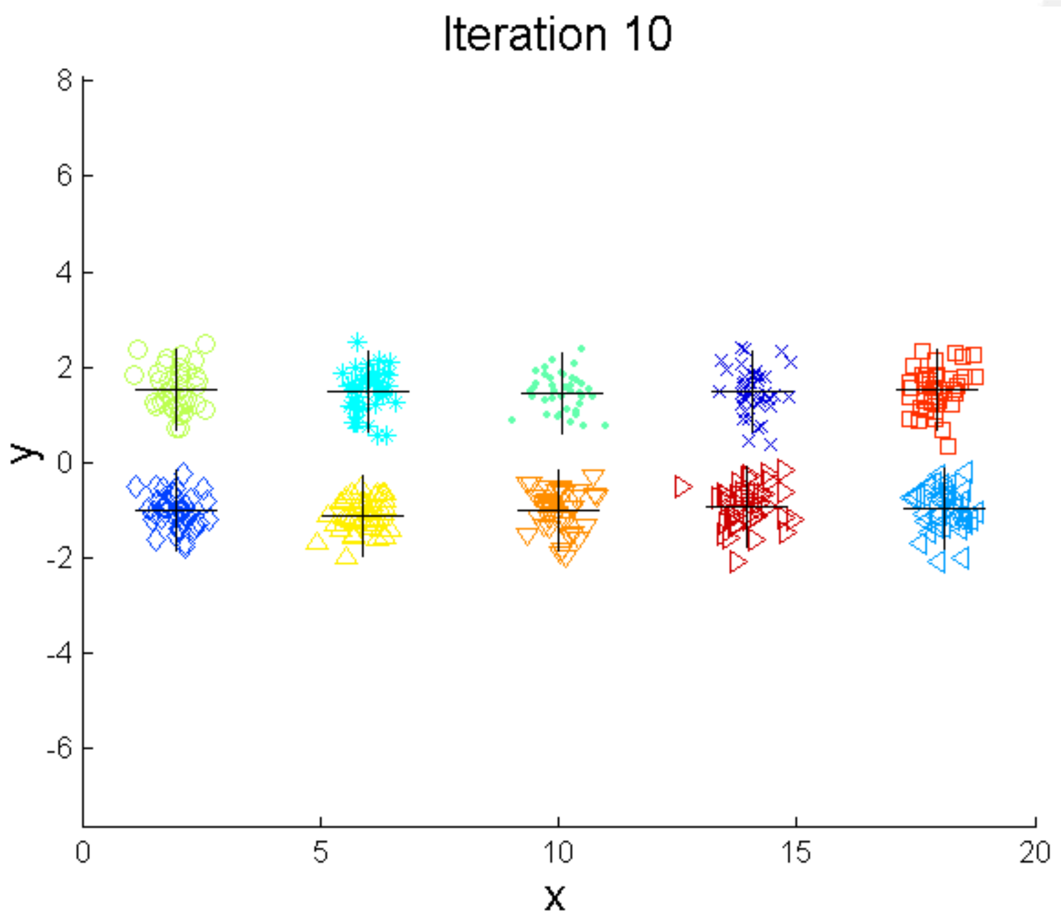
- Post-processing
  - Eliminate small clusters that may represent outliers
  - Split ‘loose’ clusters, i.e., clusters with relatively high SSE
  - Merge clusters that are ‘close’ and that have relatively low SSE
  - Can use these steps during the clustering process (done in an algorithm called ISODATA)

Bisecting K-means

Variant of K-means that can produce a partitional or a hierarchical clustering

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

Bisecting K-means Example

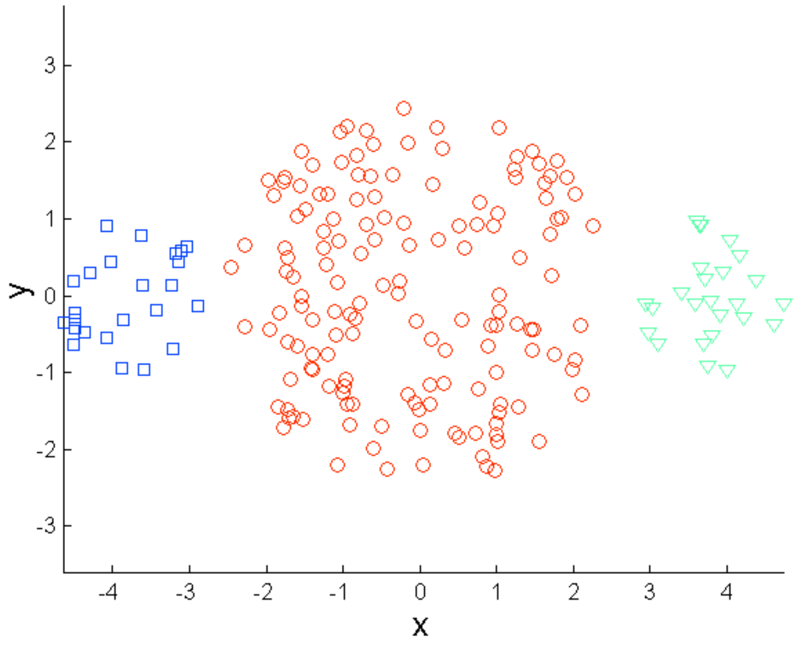


Limitations of K-means

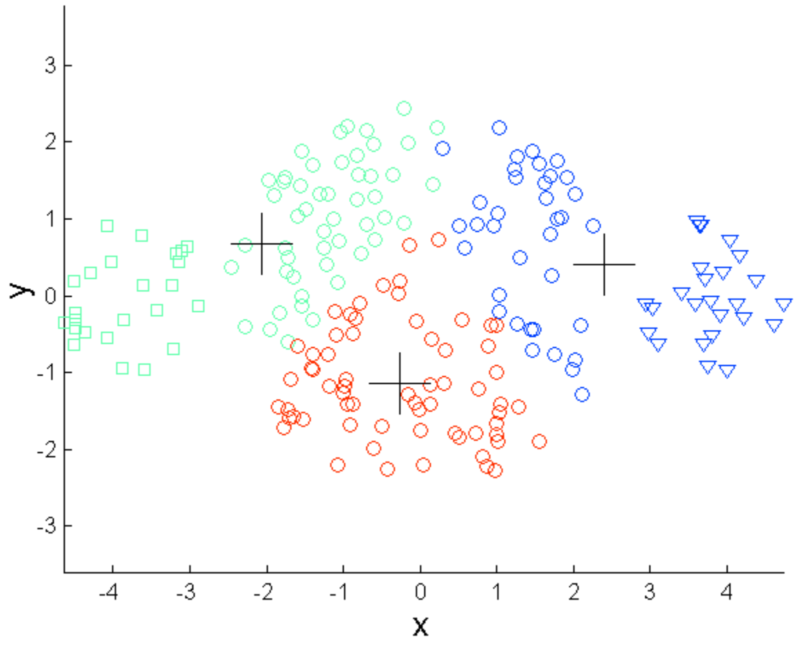
- K-means has problems when clusters are of differing
  - sizes
  - densities
  - non-globular shapes

- K-means has problems when the data contains outliers

Limitations of K-means: Differing Sizes

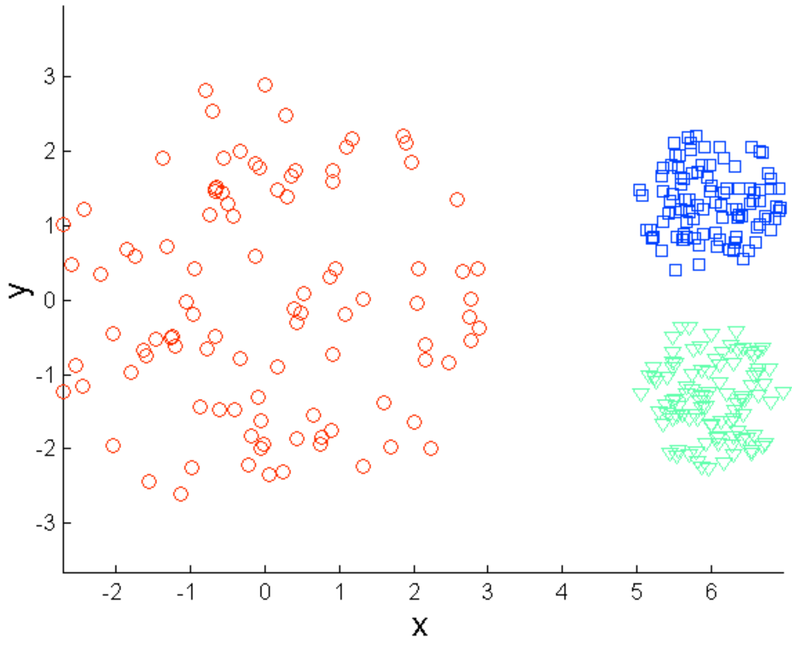


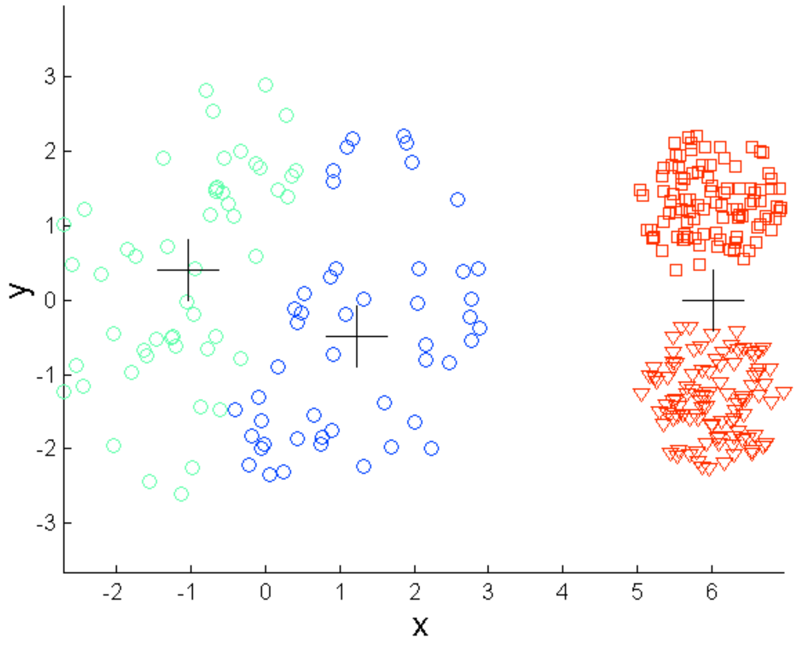
Original Points



K-means (3 Clusters)

Limitations of K-means: Differing Density

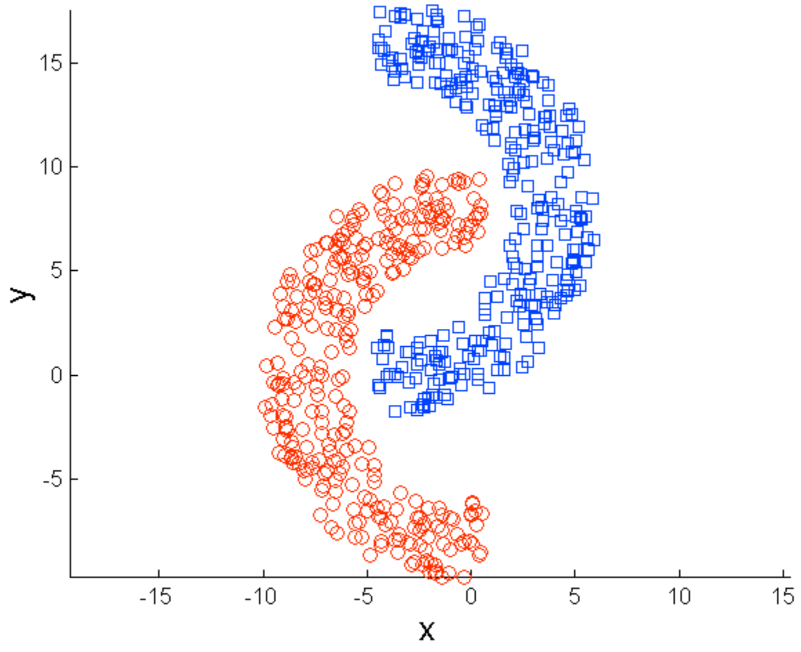




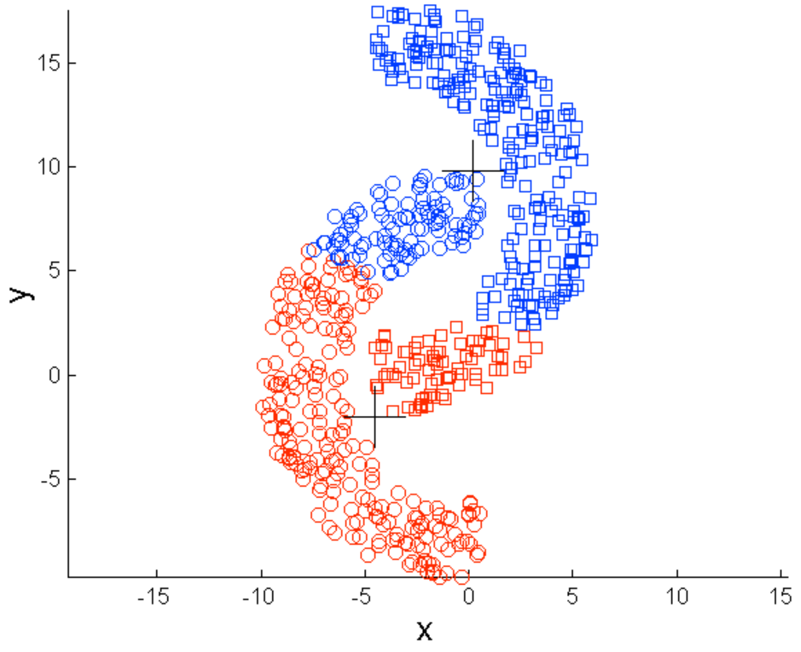
Original Points

K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes



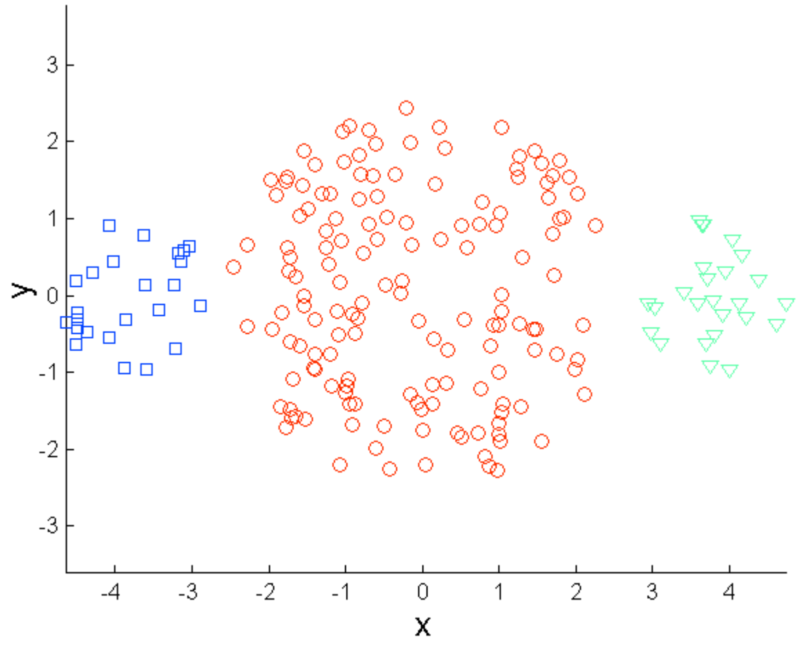
Original Points



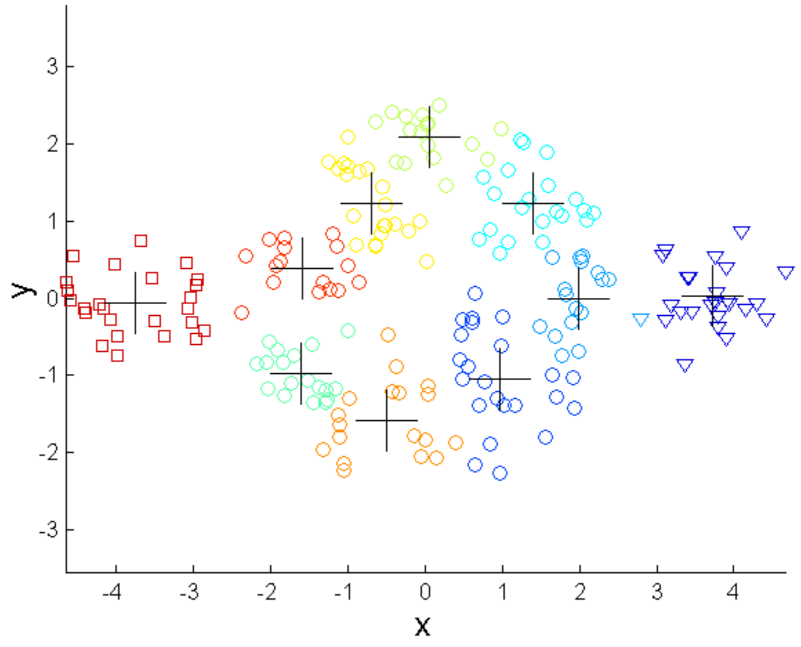
K-means (2 Clusters)

Overcoming K-means Limitations (1)

One solution is to use many clusters: find parts of clusters, but then you need to put these together

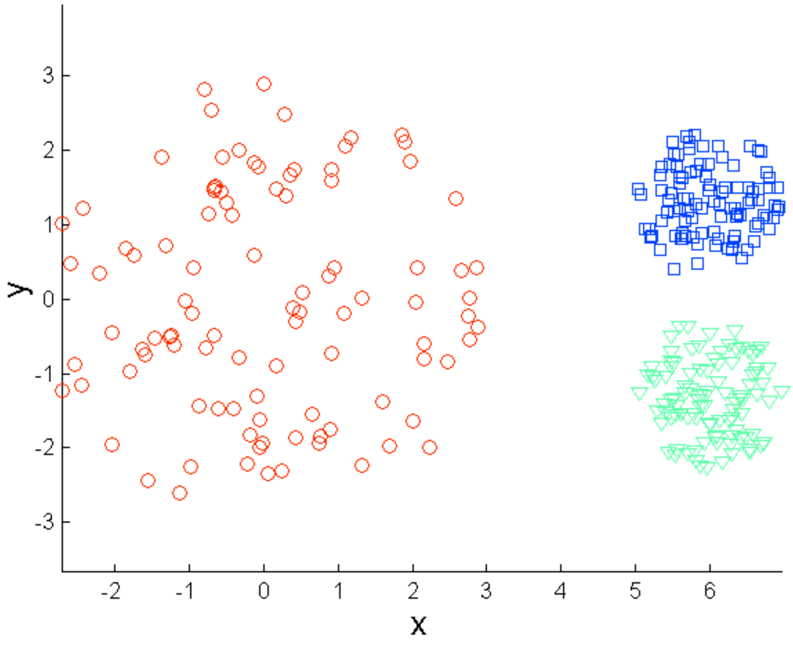


Original Points

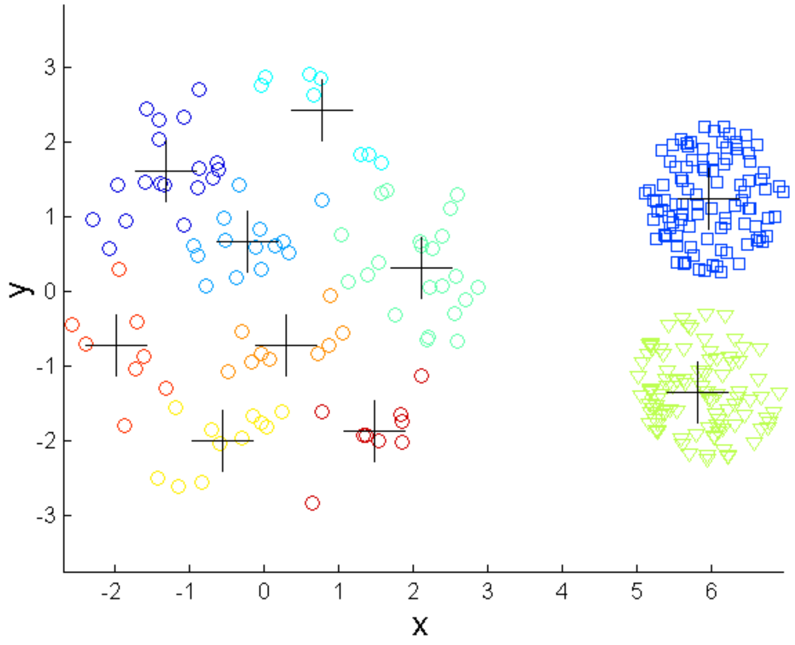


K-means Clusters

Overcoming K-means Limitations

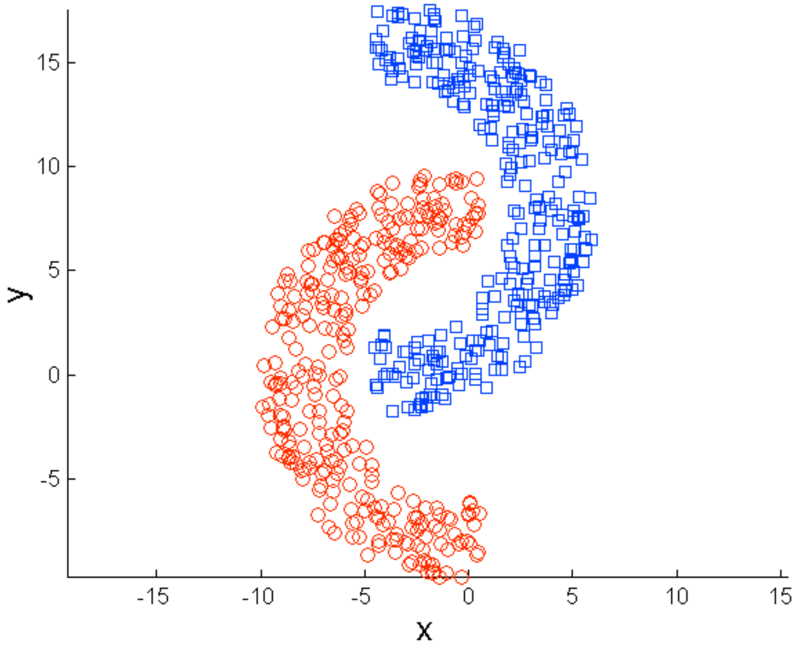


Original Points

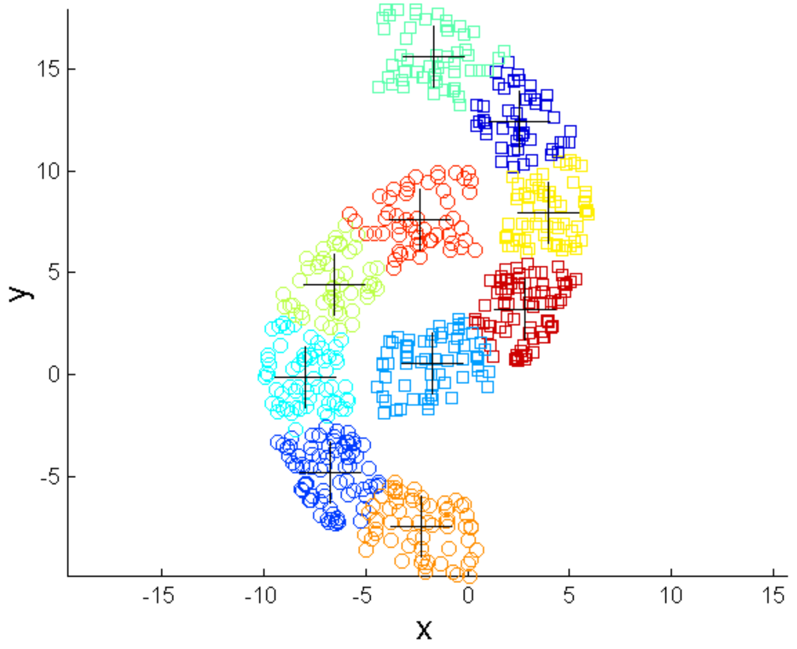


K-means Clusters

Overcoming K-means Limitations



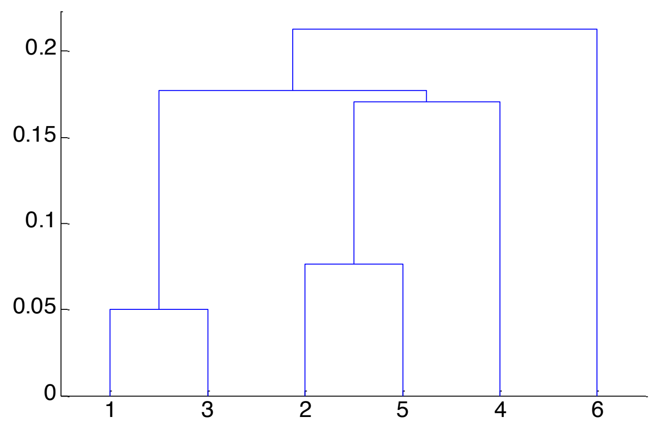
Original Points

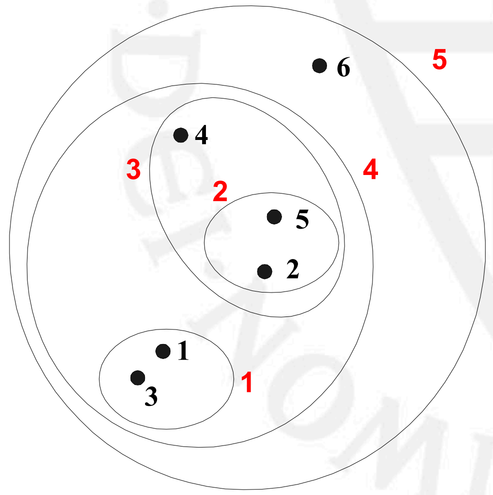


K-means Clusters

Hierarchical Clustering

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





Strengths of Hierarchical Clustering

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

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

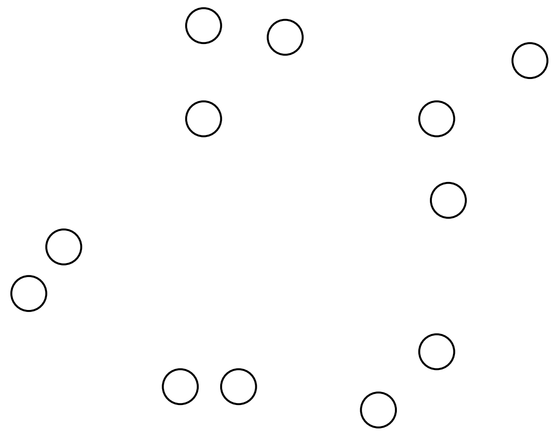
- Basic algorithm is straightforward:

```
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
```

- Different approaches to defining the distance between two clusters in step 5 distinguish the different algorithms

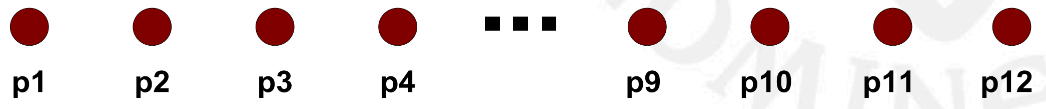
Starting Situation

- Start with clusters of individual points and a proximity matrix



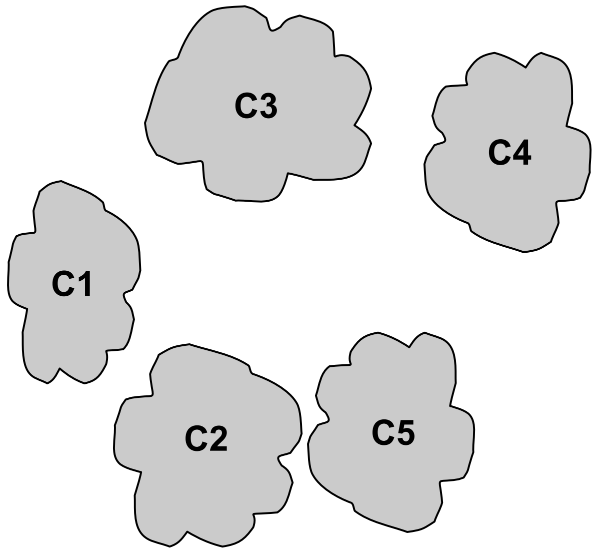
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
...						

Proximity Matrix



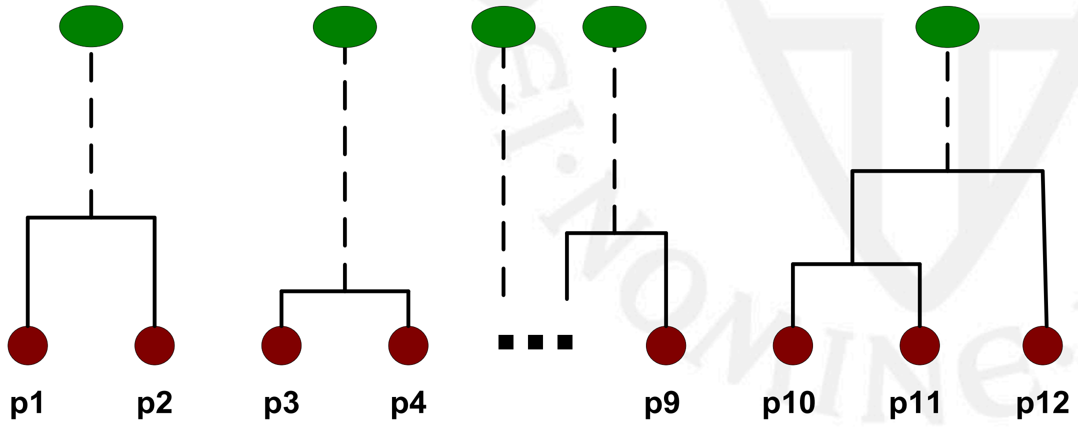
Intermediate Situation (1)

- After some merging steps we have some clusters



	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix

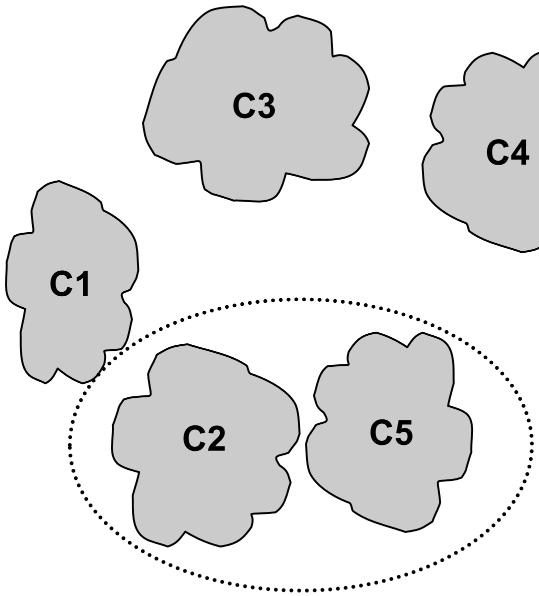


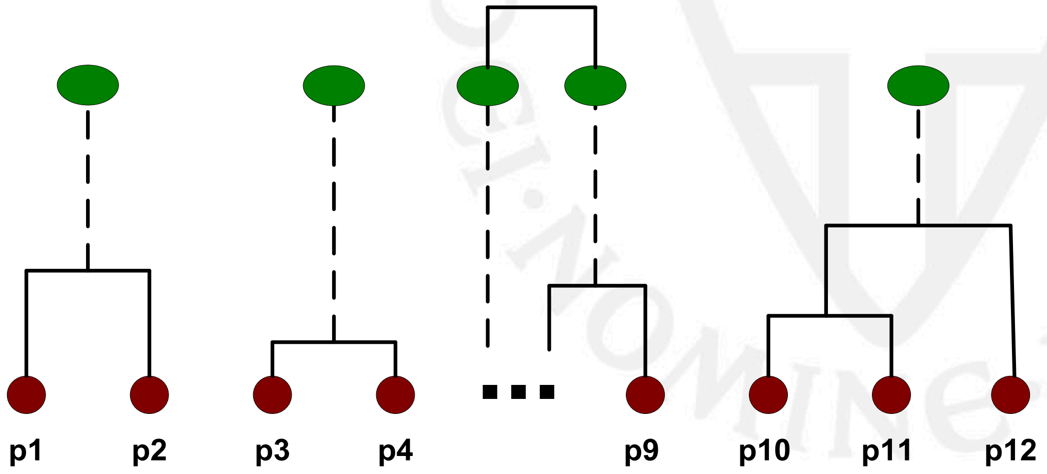
Intermediate Situation (2)

- 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



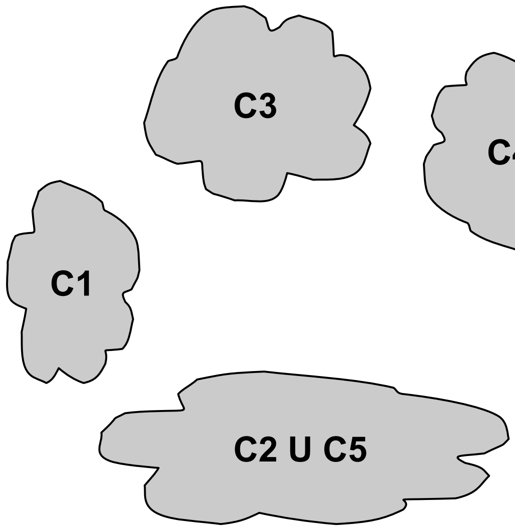


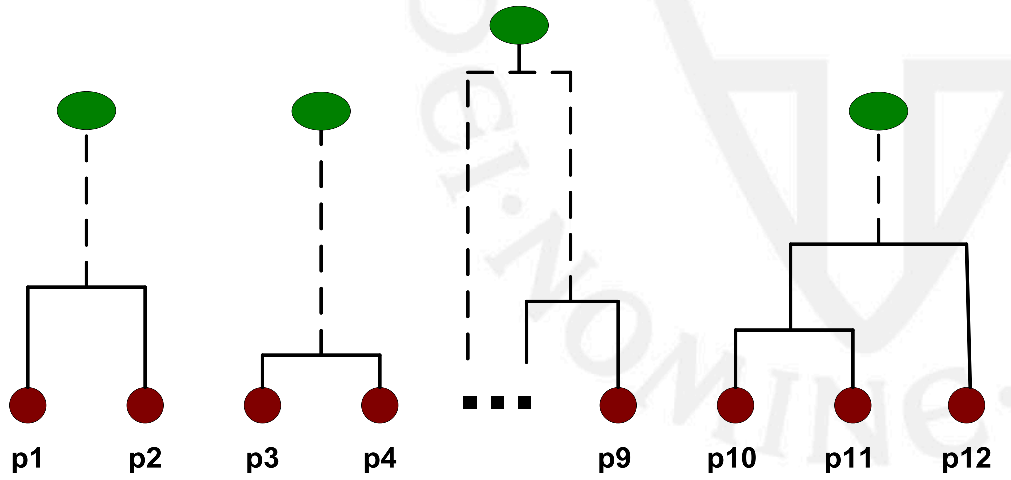
After merging

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

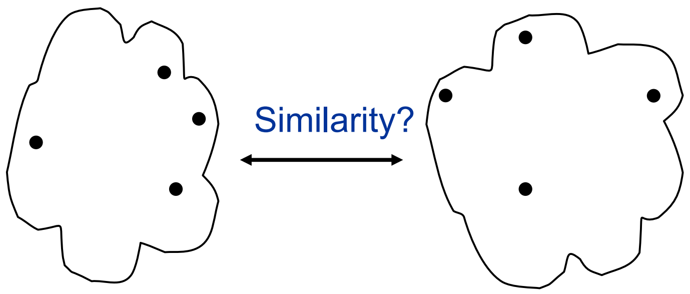
	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?		?	?
C3		?		
C4		?		

Proximity Matrix





How to define cluster similarity

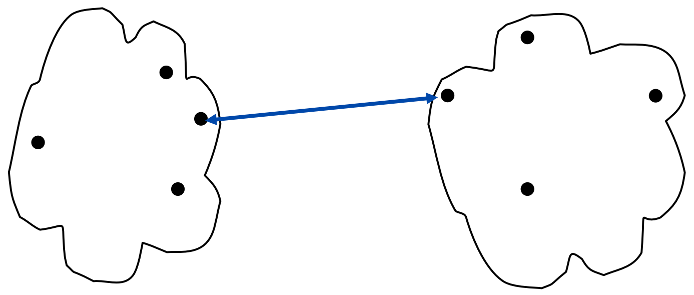


	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
...						

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 cluster similarity

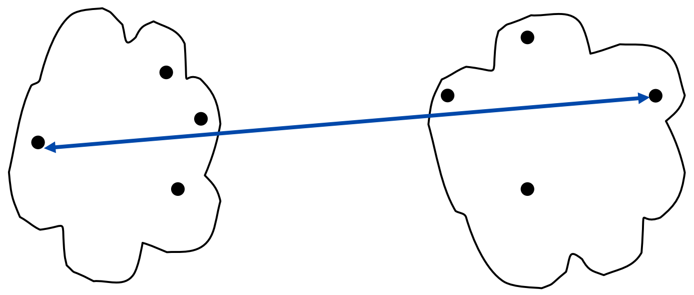


	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
...						

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 cluster similarity

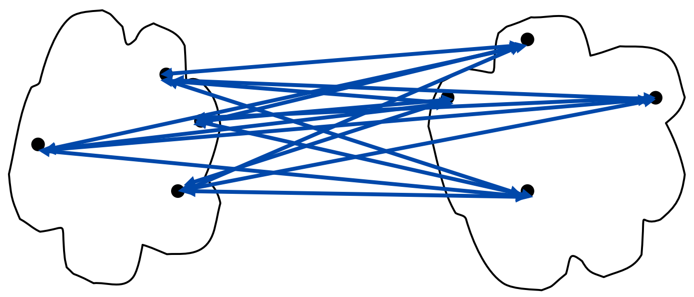


	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
...						

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 cluster similarity

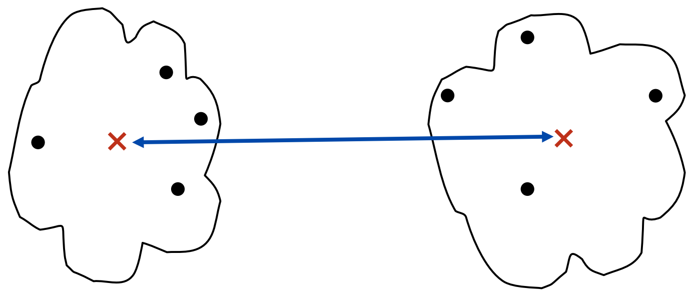


	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
...						

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 cluster similarity



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

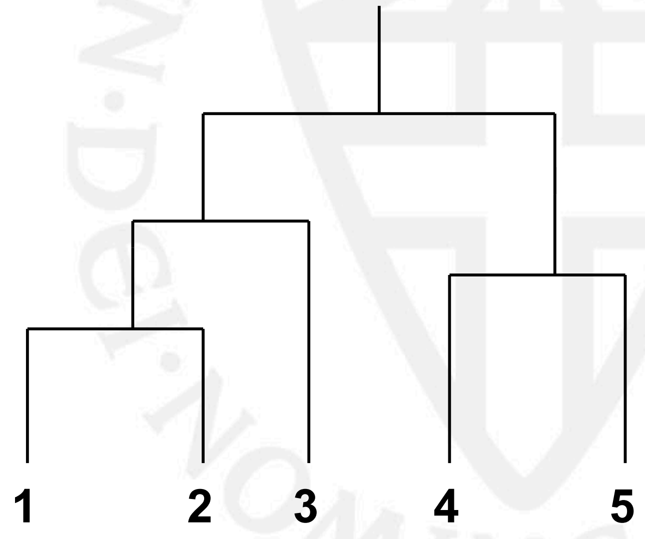
Proximity Matrix

- Min
- Max
- Group average
- Distance between centroids
- Other methods driven by an objective function
  - Ward's method uses squared error

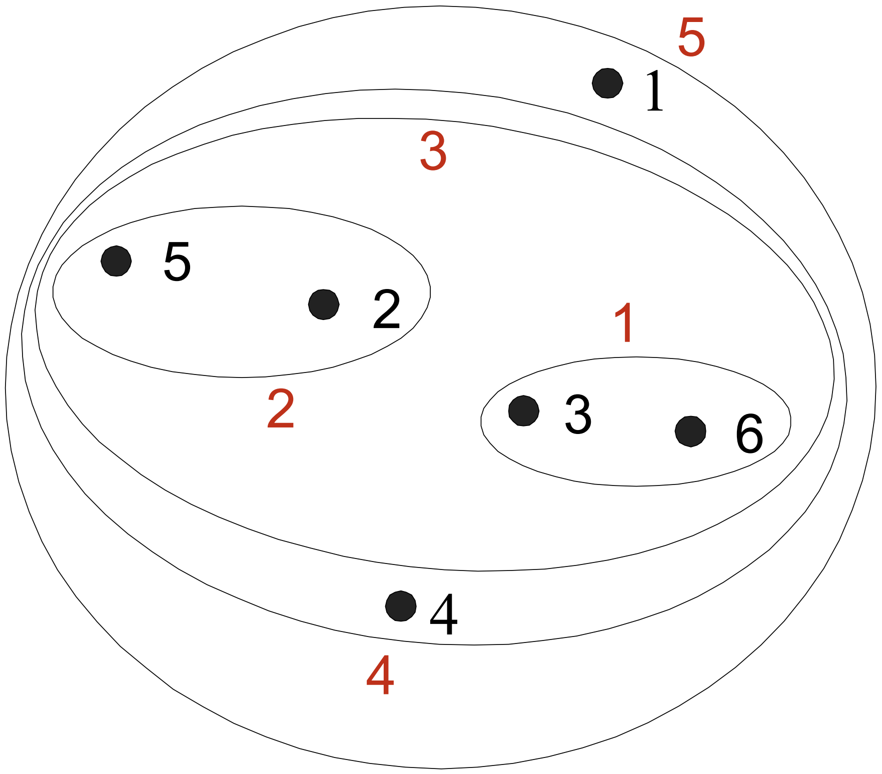
Cluster Similarity: Min or Single Link

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

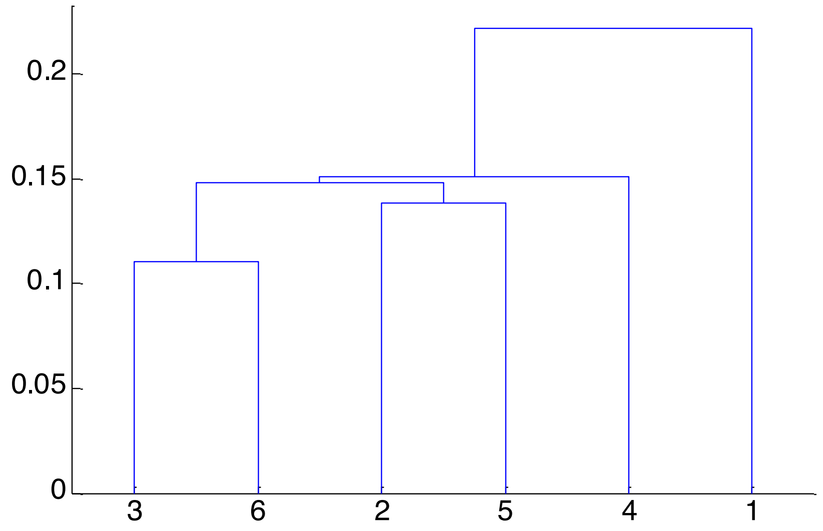
	p1	p2	p3	p4	p5
p1	1.00	0.90	0.10	0.65	0.20
p2	0.90	1.00	0.70	0.60	0.50
p3	0.10	0.70	1.00	0.40	0.30
p4	0.65	0.60	0.40	1.00	0.80
p5	0.20	0.50	0.30	0.80	1.00



Hierarchical Clustering: Min



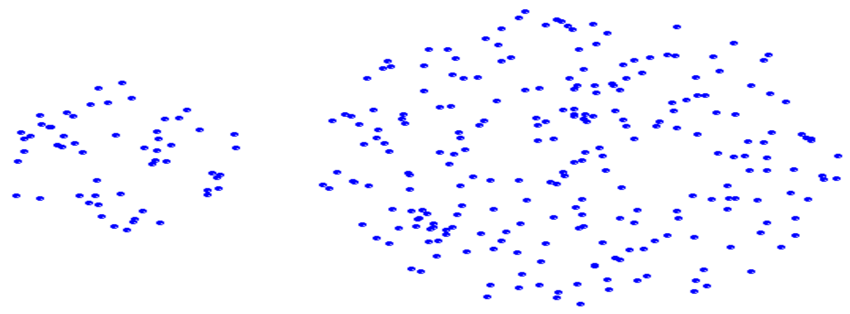
Nested Clusters



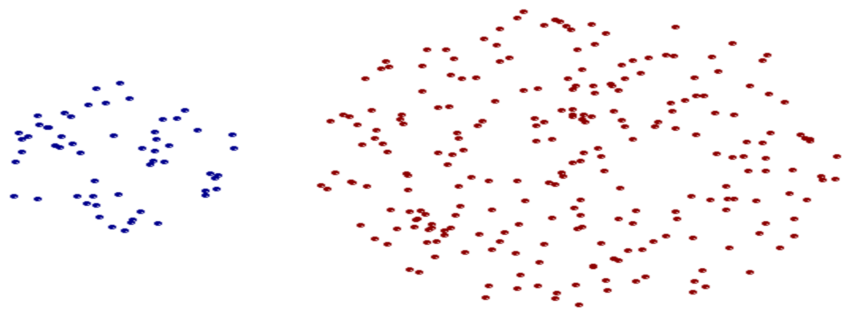
Dendrogram

Strength of Min

Can handle non-spherical shapes



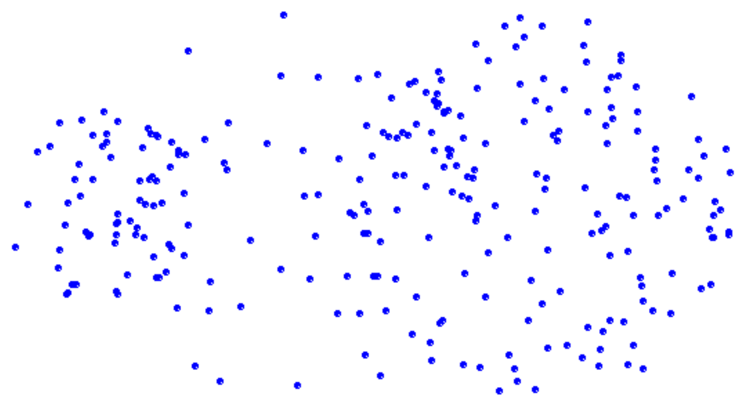
Original Points



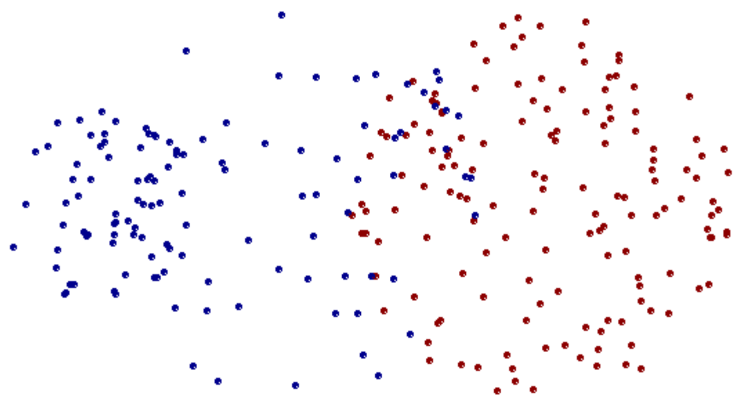
Two Clusters

Limitations of Min

Sensitive to noise and outliers



Original Points

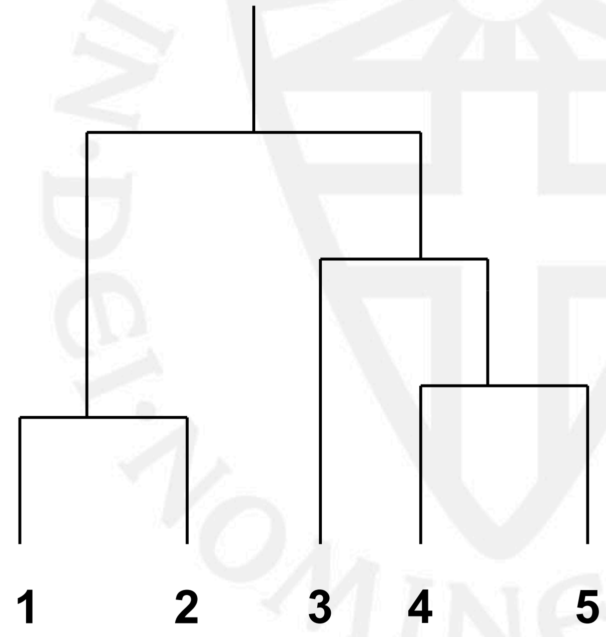


Two Clusters

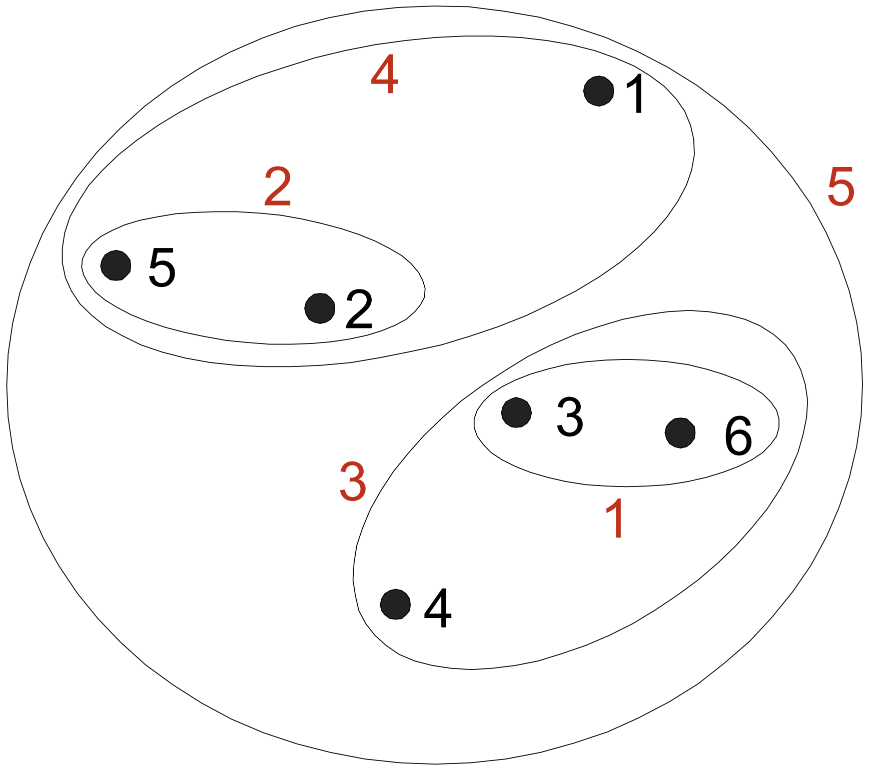
Cluster Similarity: Max or Complete Linkage

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

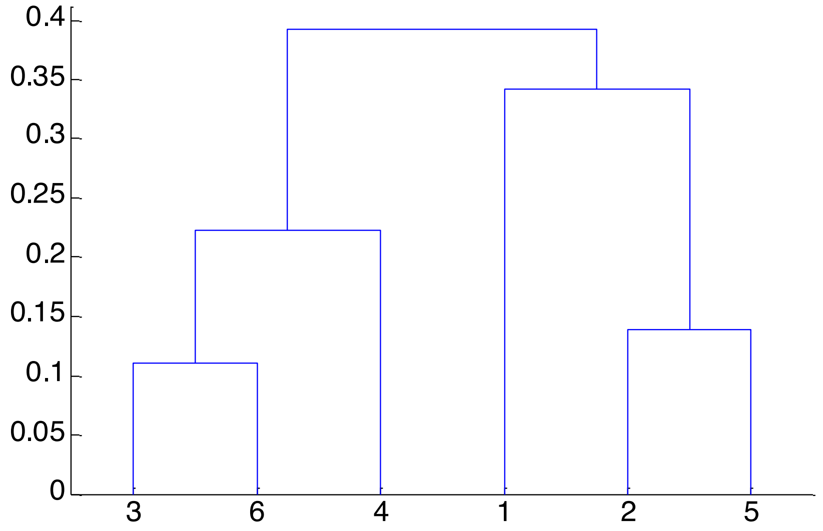
	p1	p2	p3	p4	p5
p1	1.00	0.90	0.10	0.65	0.20
p2	0.90	1.00	0.70	0.60	0.50
p3	0.10	0.70	1.00	0.40	0.30
p4	0.65	0.60	0.40	1.00	0.80
p5	0.20	0.50	0.30	0.80	1.00



Hierarchical Clustering: Max



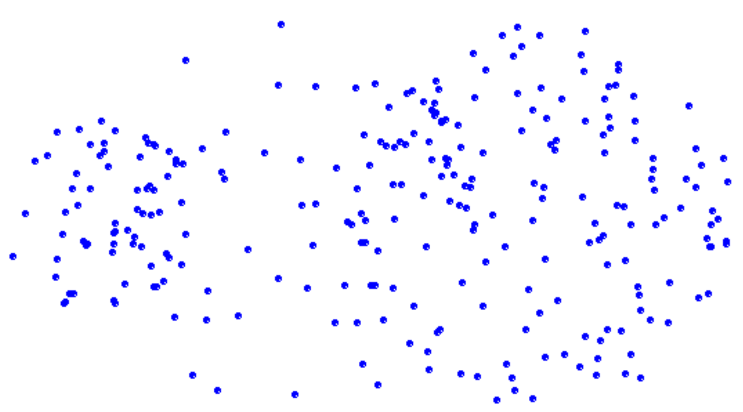
Nested Clusters



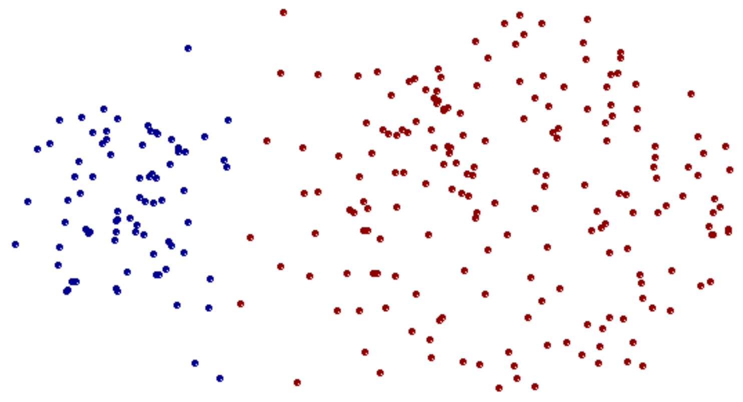
Dendrogram

Strength of Max

Less susceptible to noise and outliers



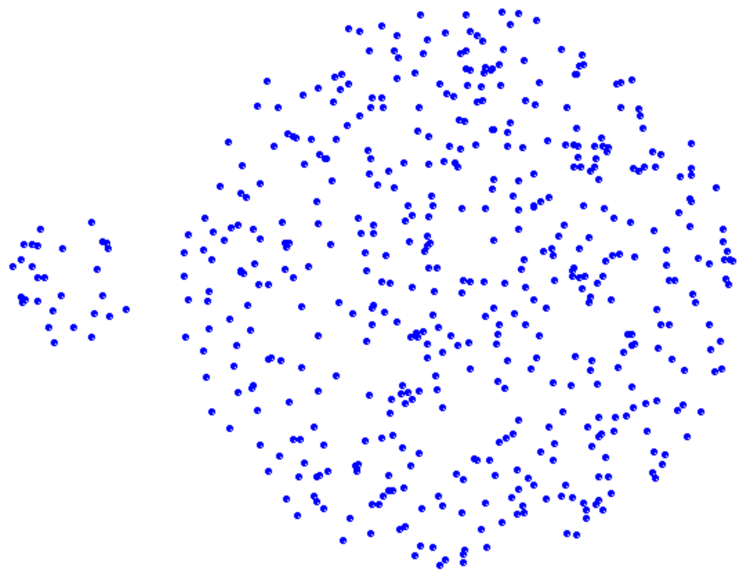
Original Points

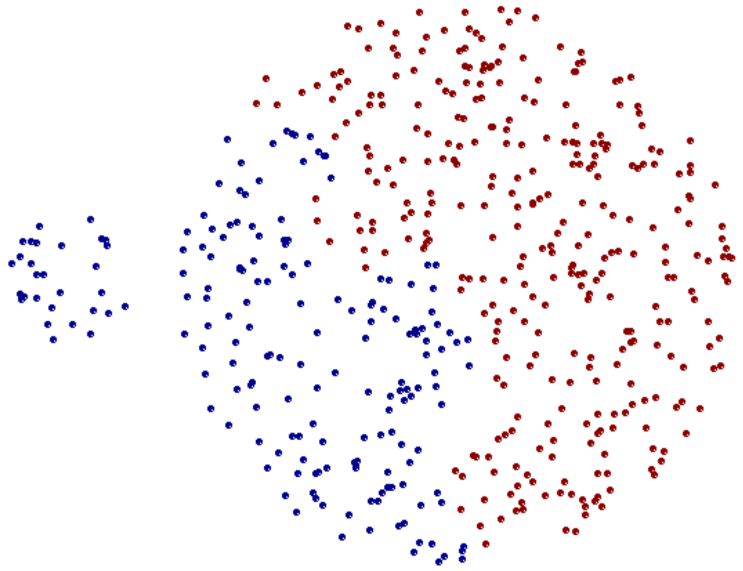


Two Clusters

Limitations of Max

- Tends to break large clusters
- Biased towards globular clusters


Original Points

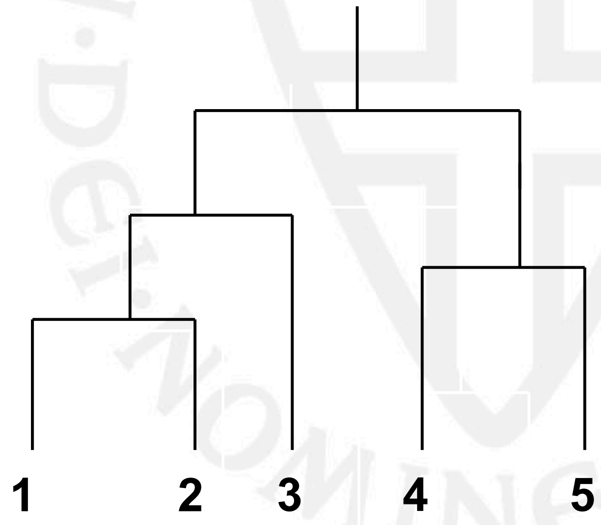

Two Clusters

Cluster Similarity: Group Average

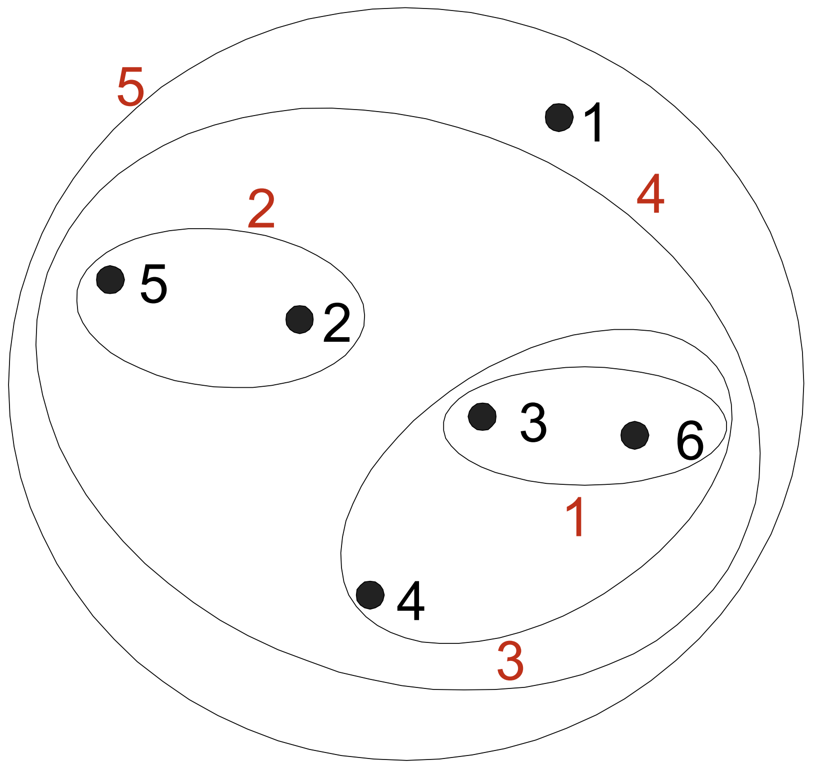
- Proximity of two clusters is the average of pairwise proximity between points in the two clusters:

$$\text{proximity}(C_i, C_j) = \frac{\sum_{p_i \in C_i;\, p_j \in C_j} \text{proximity}(p_i, p_j)}{|C_i| |C_j|}$$

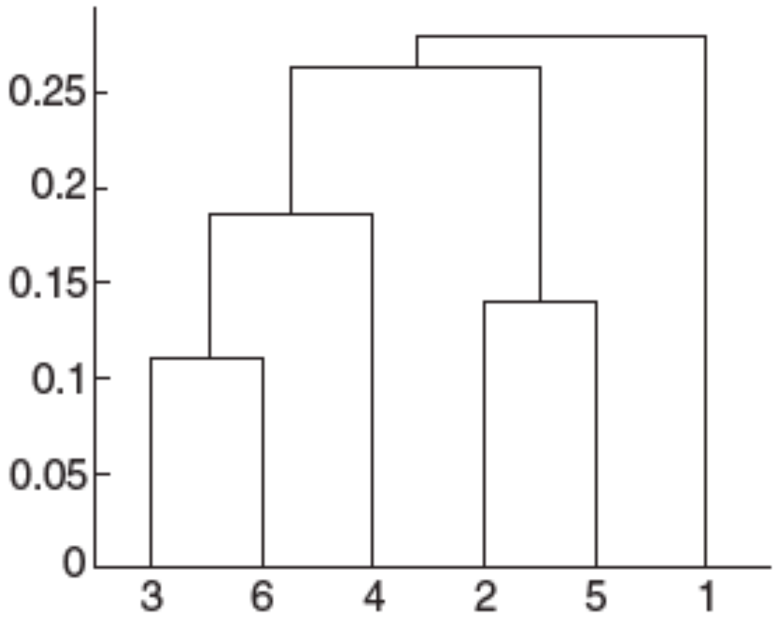
	p1	p2	p3	p4	p5
p1	1.00	0.90	0.10	0.65	0.20
p2	0.90	1.00	0.70	0.60	0.50
p3	0.10	0.70	1.00	0.40	0.30
p4	0.65	0.60	0.40	1.00	0.80
p5	0.20	0.50	0.30	0.80	1.00



Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram

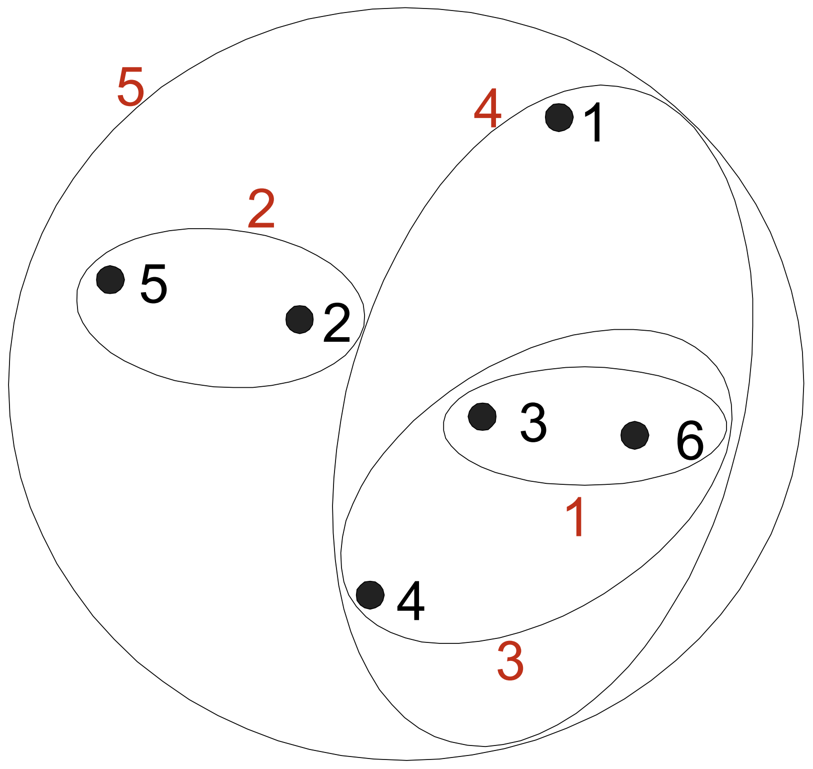
Hierarchical Clustering: Group Average

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

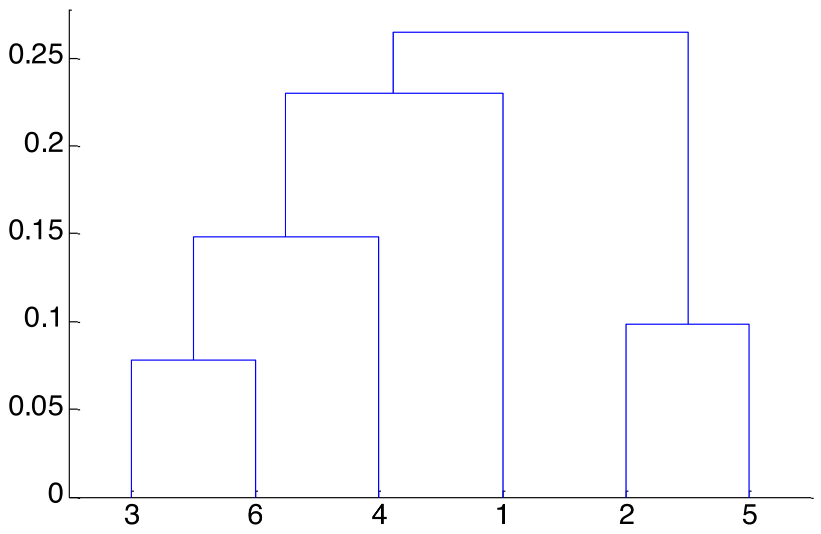
Cluster Similarity: Ward's Method

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

Hierarchical Clustering: Ward’s method

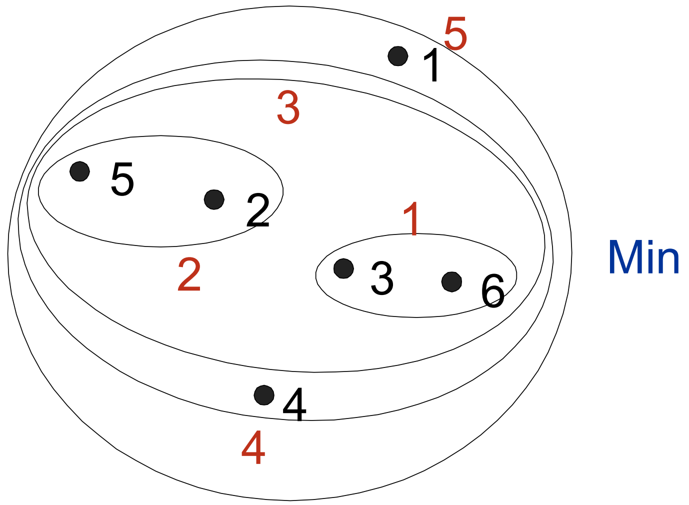


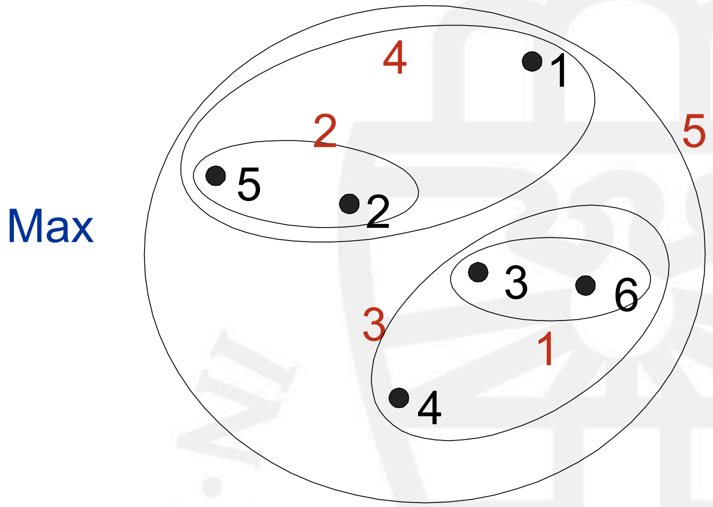
Nested Clusters

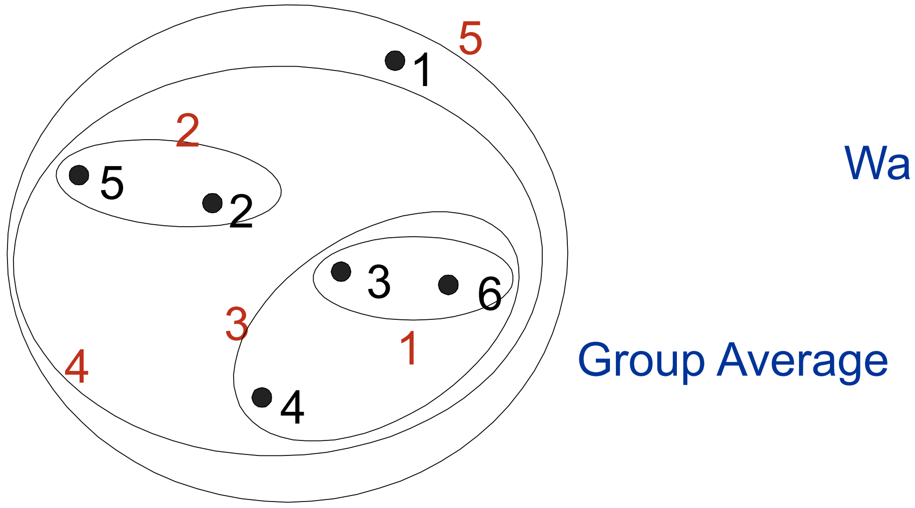


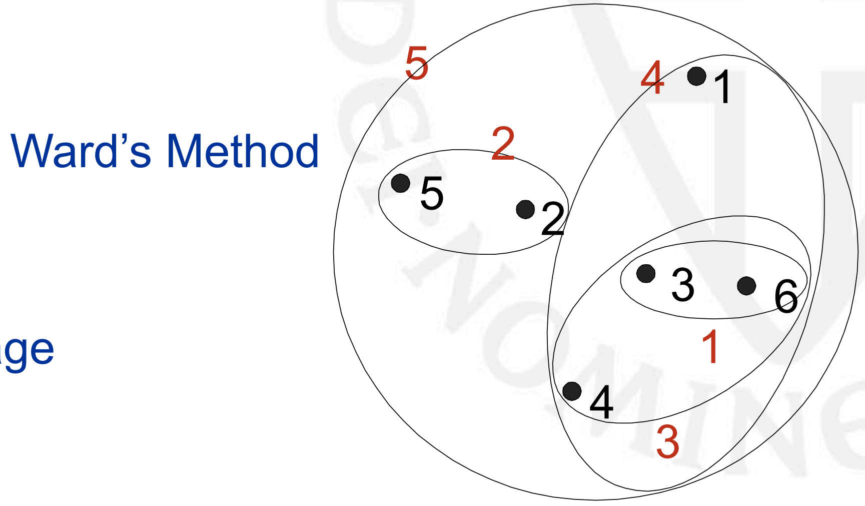
Dendrogram

Hierarchical Clustering: Comparison









Hierarchical Clustering: Time and Space Requirements

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

Hierarchical Clustering: Problems and Limitations

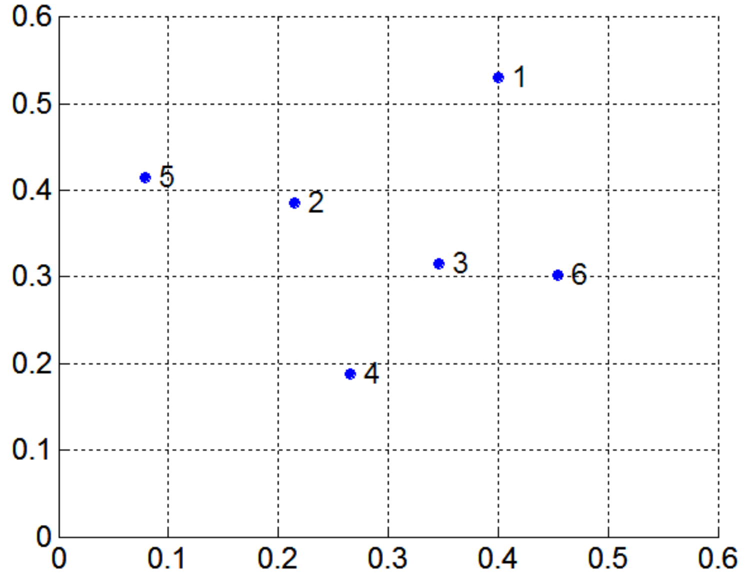
- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers
  - Difficulty handling different sized clusters and convex shapes
  - Breaking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

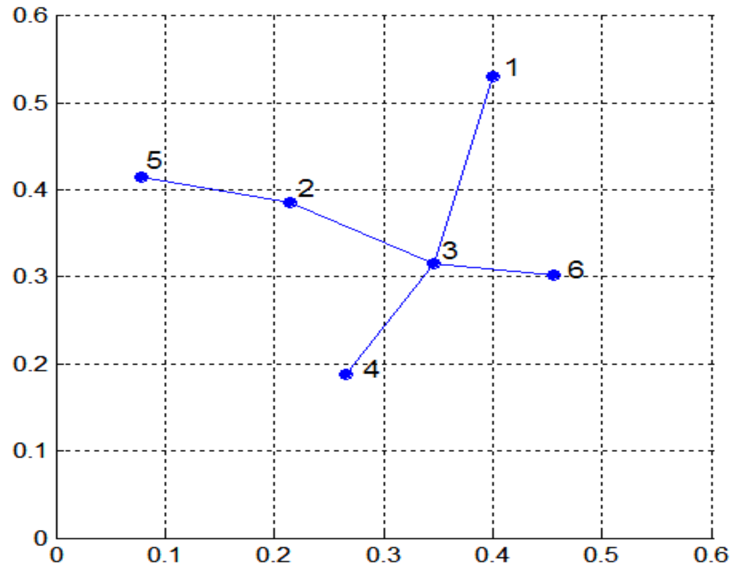
Hierarchical Clustering in MATLAB

```
% load iris data set; gives input in meas and
% cluster labels in species

load fisheriris

% Ward’s method with Euclidean distance as metric

Z = linkage(meas,'ward','euclidean');
dendrogram(Z)                        % plots dendrogram

c = cluster(Z,'maxclust',4);  % choose 4 clusters
crosstab(c,species)                  % confusion matrix
```

MST: Divisive Hierarchical Clustering (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
  - Add *q* to the tree and put an edge between *p* and *q*





MST: Divisive Hierarchical Clustering (2)

- Use MST for constructing hierarchy of clusters

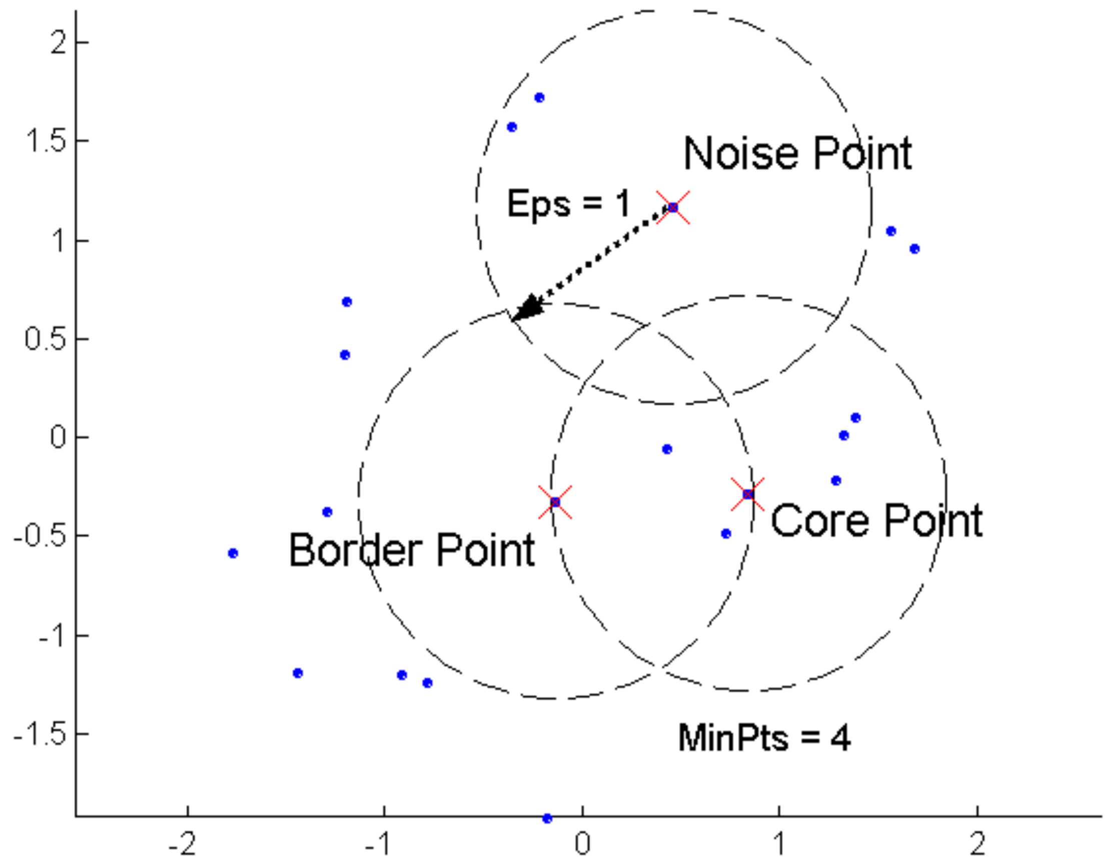
Algorithm 7.5 MST Divisive Hierarchical Clustering Algorithm

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

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 number of points (*MinPts*) within *Eps*
  - These are points that are at the interior of a cluster
- A border point has fewer than *MinPt*s 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: Core, Border, and Noise Points

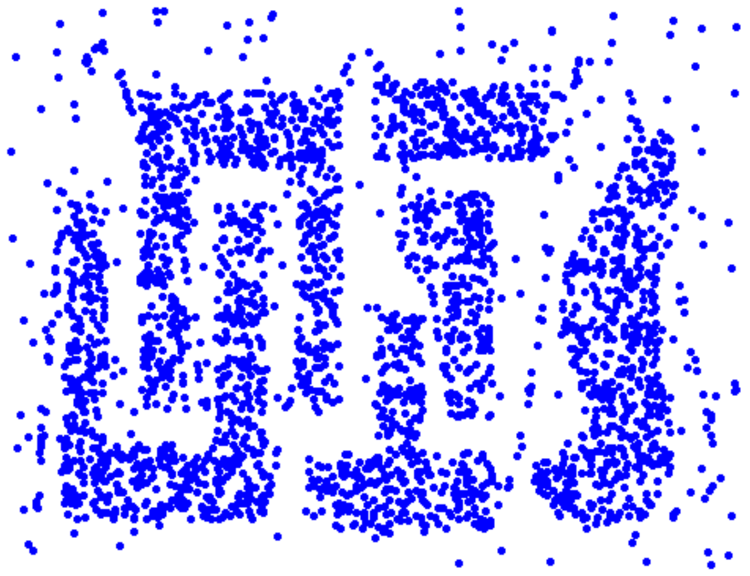


DBSCAN Algorithm (more efficient version than Alg.8.4)

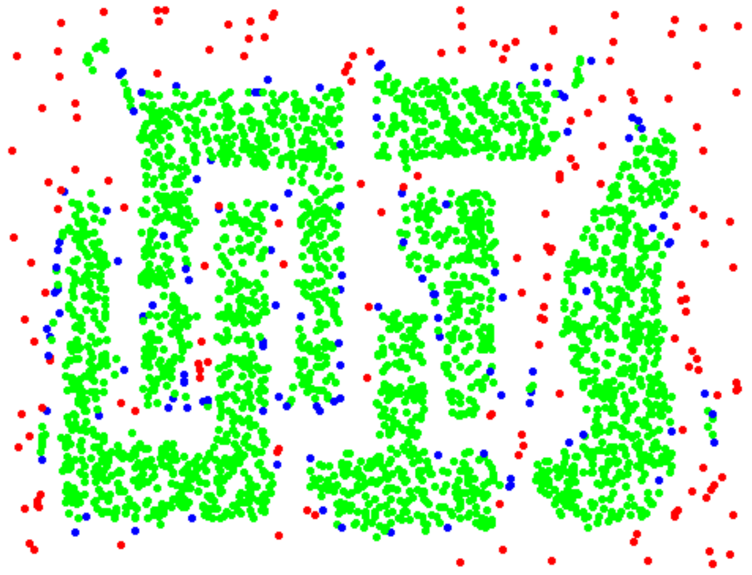
```
DBSCAN(DB, dist, eps, minPts) {
   C = 0
   for each point P in database DB {
      if label(P) ≠ undefined then continue
      Neighbors N = RangeQuery(DB, dist, P, eps)
      if |N| < minPts then {
         label(P) = Noise
         continue
      }
      C = C + 1
      label(P) = C
      Seed set S = N \ {P}
      for each point Q in S {
         if label(Q) = Noise then label(Q) = C
         if label(Q) ≠ undefined then continue
         label(Q) = C
         Neighbors N = RangeQuery(DB, dist, Q, eps)
         if |N| ≥ minPts then {
            S = S ∪ N
         }
      }
   }
}
```

DBSCAN: Core, Border and Noise Points

Eps = 10, *MinPts* = 4



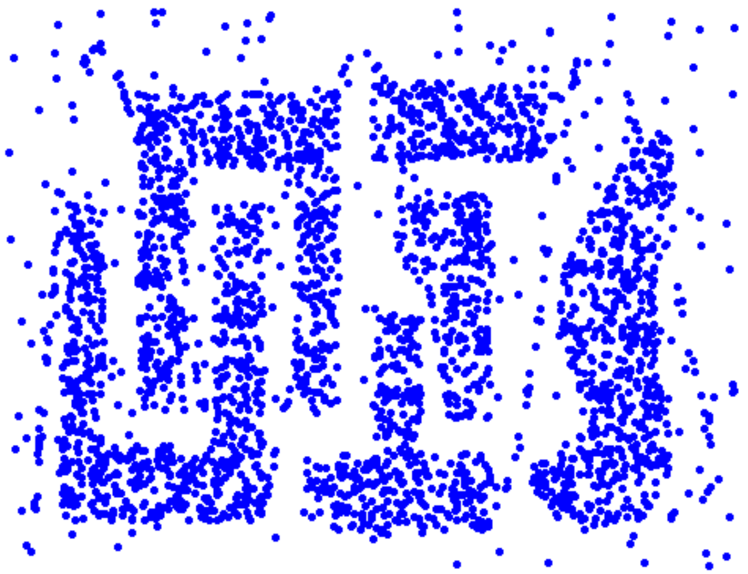
Original Points

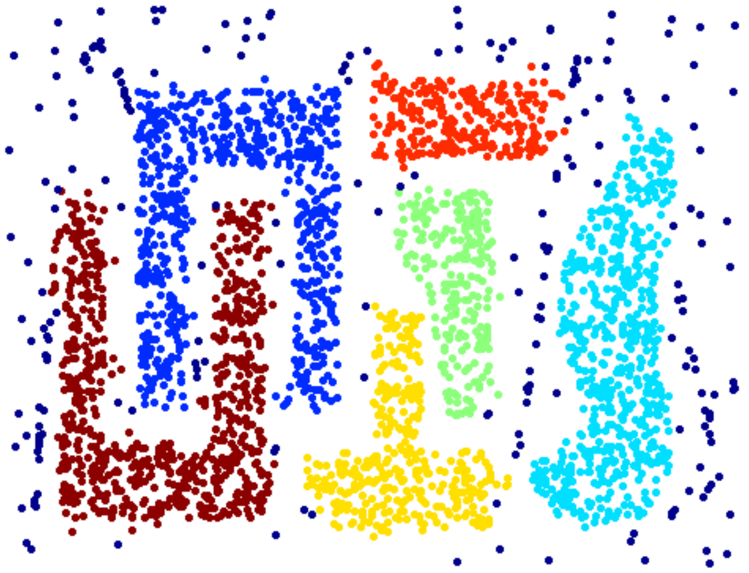


Point types: core, border and noise

When DBSCAN Works Well

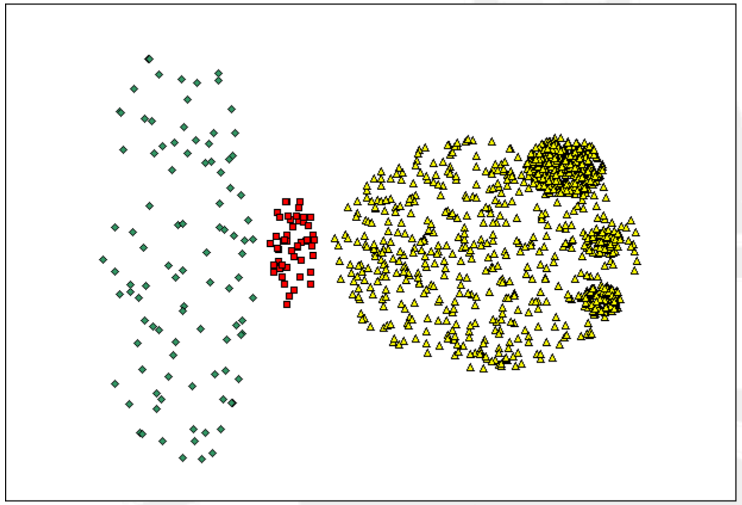
- Resistant to Noise
- Can handle clusters of different shapes and sizes


Original Points

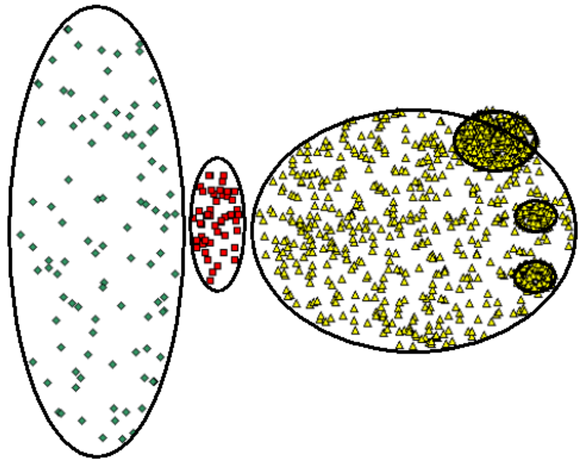

Clusters

When DBSCAN Does NOT Work Well

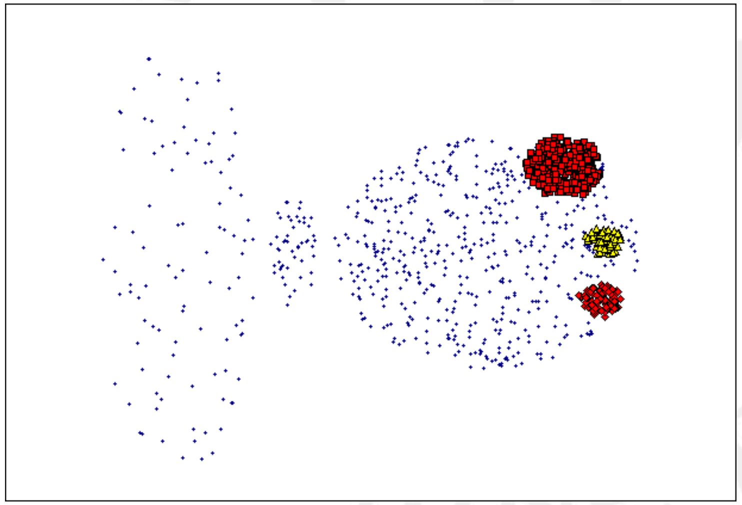
- Varying densities
- High-dimensional data



MinPts=4, *Eps*=9.75



Original Points



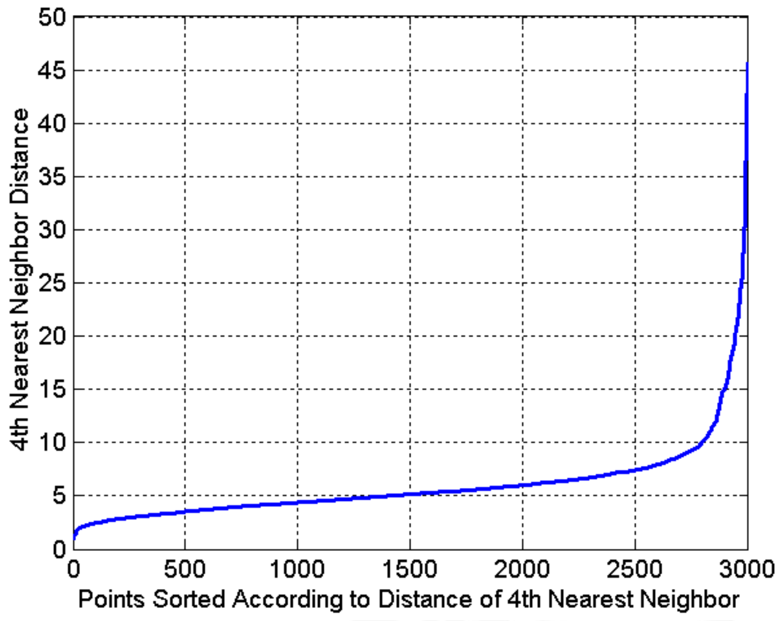
MinPts=4, *Eps*=9.92

DBSCAN: Determining *EPS* and *MinPts*

- Idea is that for points in a cluster, their *k*th nearest neighbors, with *k* equal to *MinPts,* 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 and look for sharp increase

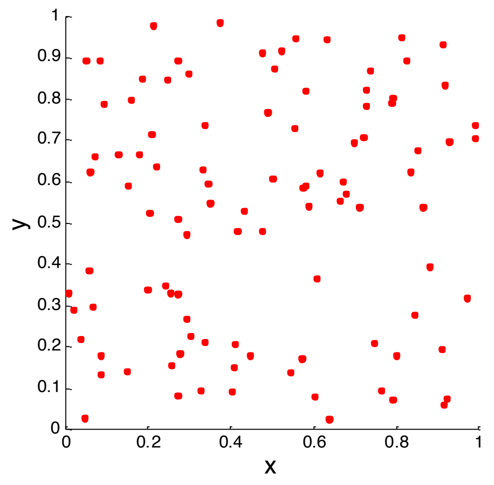


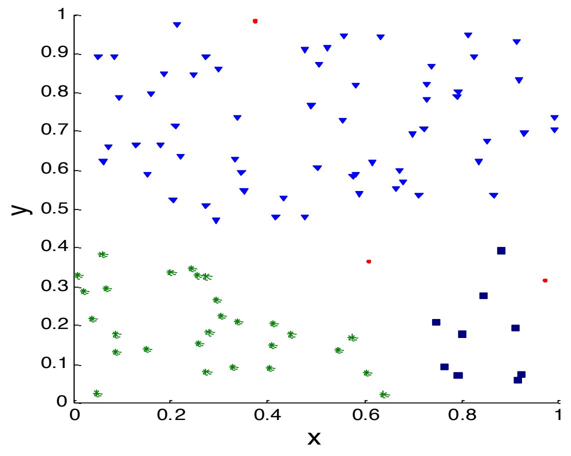
- In this example: choose *Eps* ≈10

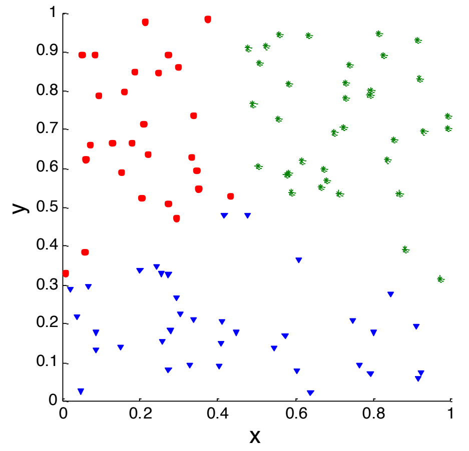
Cluster Validity

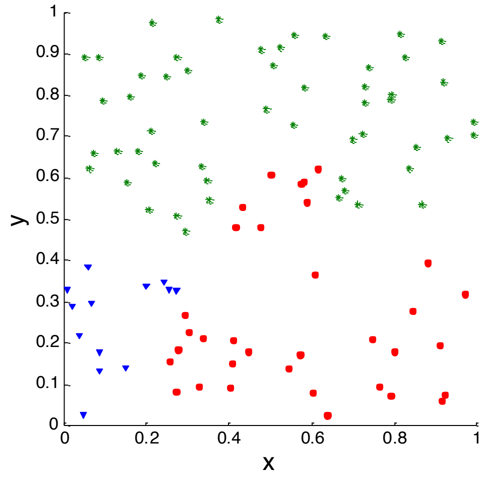
- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall, area under the curve, ...
- For cluster analysis, the analogous question is how to evaluate the “goodness” of the resulting clusters?
- But “clusters are in the eye of the beholder”!
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

Clusters found in Random Data









Different Aspects of Cluster Validation

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

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

Measures of Cluster Validity

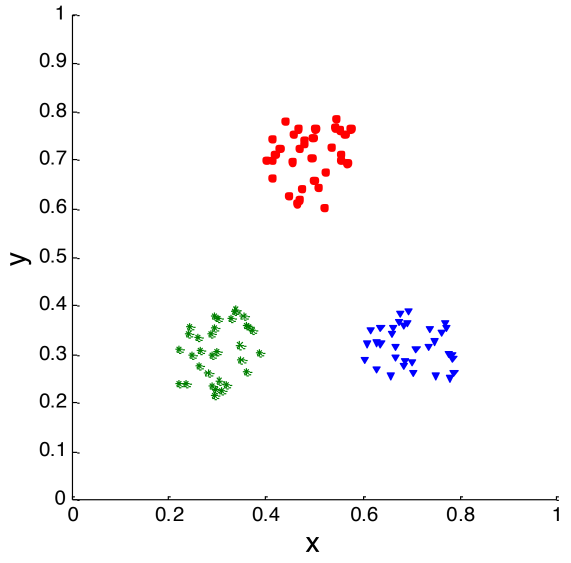
- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types
  - External Index: Used to measure the extent to which cluster labels match externally supplied class labels; example: entropy
  - Internal Index: Used to measure the goodness of a clustering structure *without* respect to external information; example: sum of squared errors (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

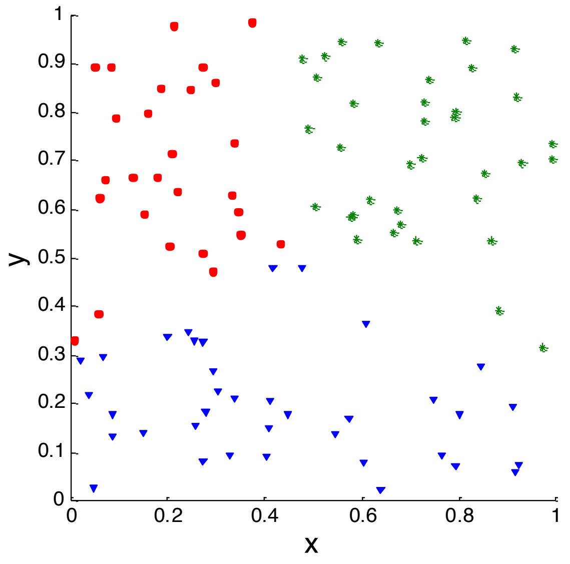
Measuring Cluster Validity Via Correlation

- Two matrices
  - Proximity Matrix (e.g., used as input to the clustering algorithm)
  - “Incidence” Matrix (computed from the clustering result)
- Incidence matrix:
  - One row and one column for each data point
  - An entry is 1 if the associated pair of points belong to the same cluster
  - An entry is 0 if the associated pair of points belongs to different clusters
- Compute the correlation between the two matrices
- Since the matrices are symmetric, only the correlation between $n(n-1)/2$ entries needs to be calculated
- High correlation (in absolute sense) indicates that points that belong to the same cluster are close to each other
- Not a good measure for some density or contiguity based clusters

Measuring Cluster Validity Via Correlation

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



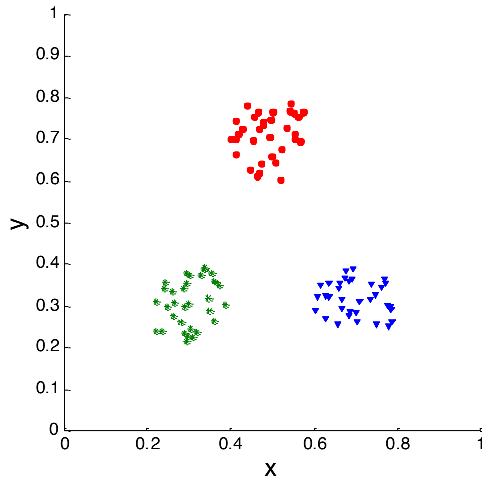


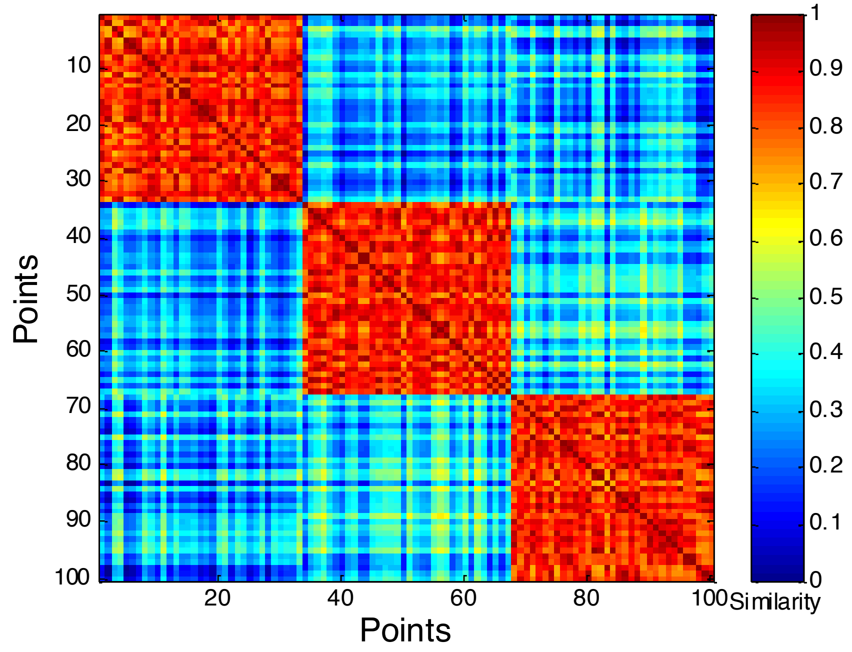
Correlation = -0.9235

Correlation = -0.5810

Using Similarity Matrix for Cluster Validation (1)

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

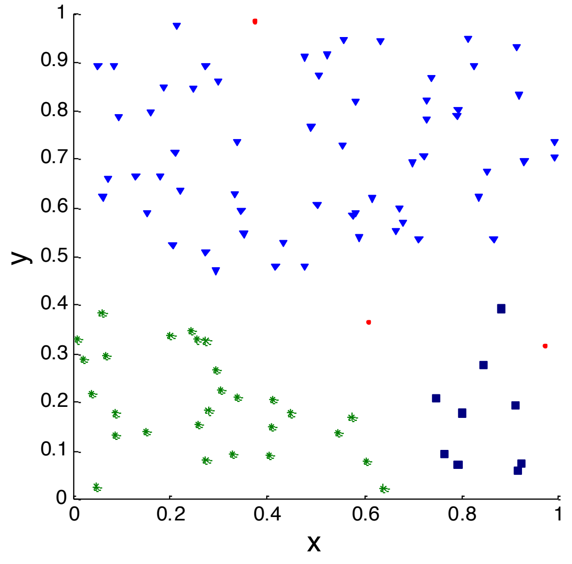


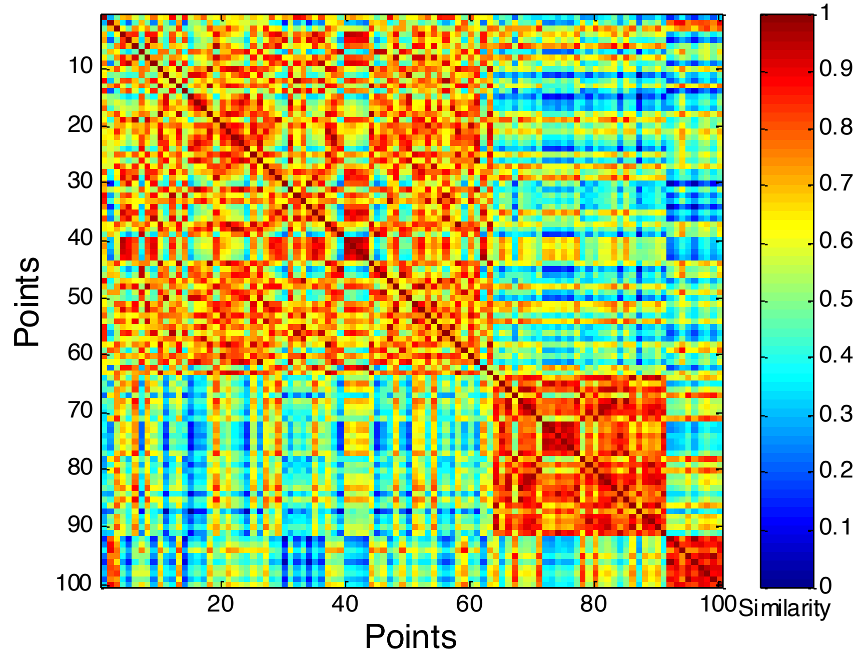


$$s_{ij} = 1 - \frac{d_{ij} - \min d}{\max d - \min d}$$

Using Similarity Matrix for Cluster Validation (2)

Clusters in random data are not so crisp

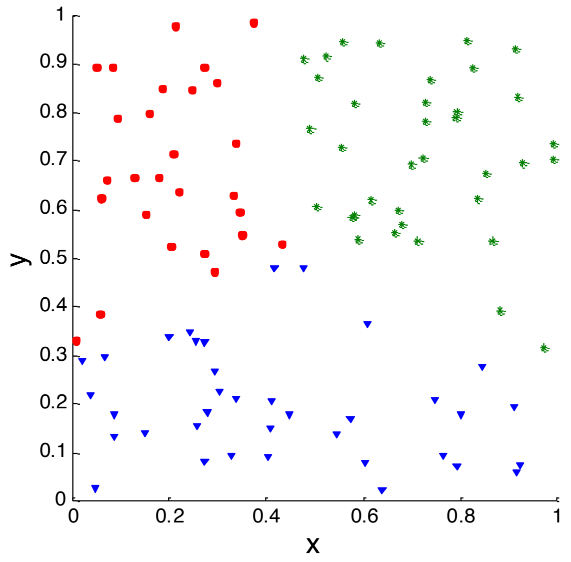


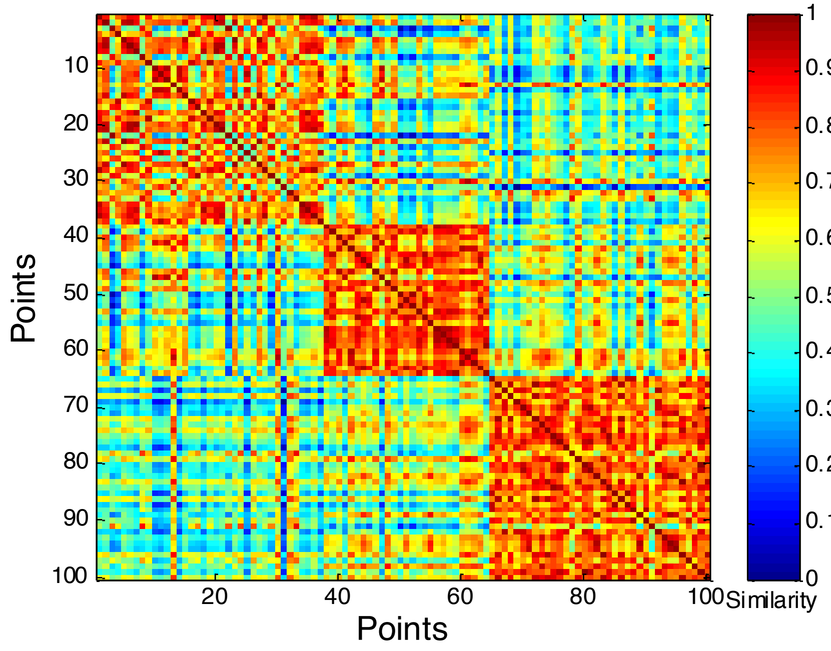


K-means

Using Similarity Matrix for Cluster Validation (3)

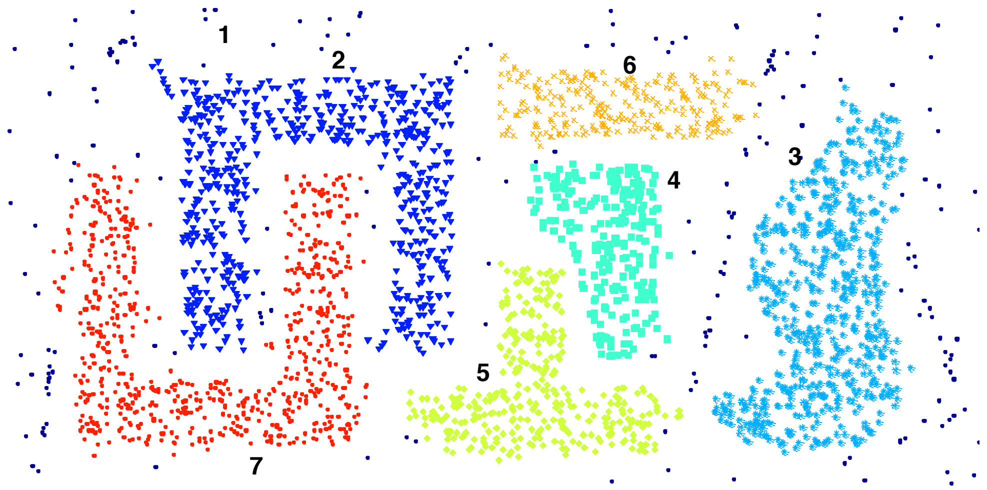
Clusters in random data are not so crisp

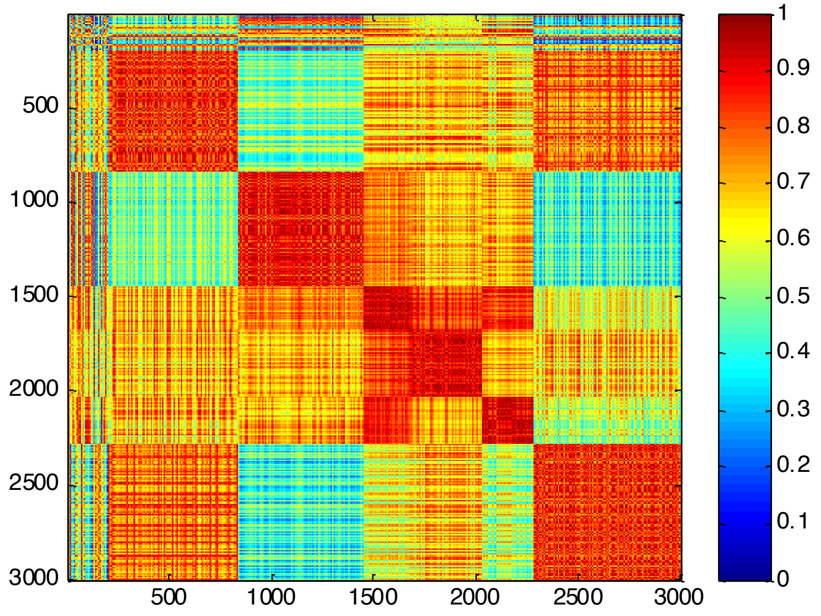




Complete linkage

Using Similarity Matrix for Cluster Validation (4)

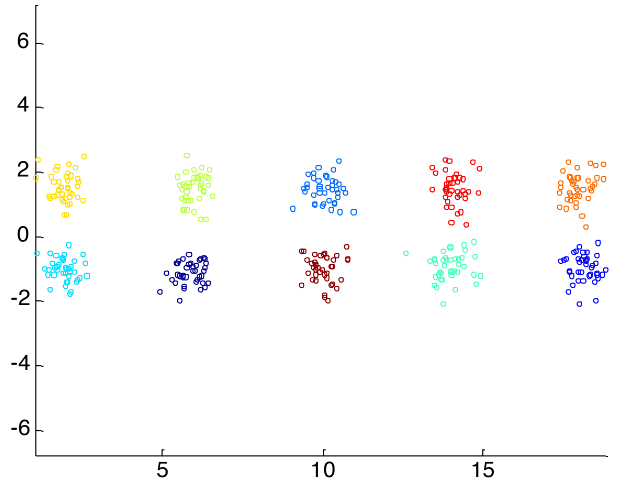


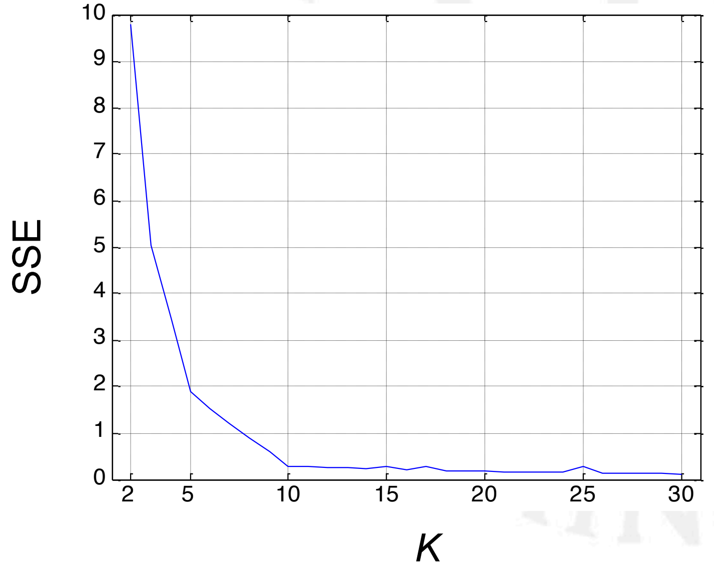


DBSCAN

Internal Measures: Sum of Squared Errors

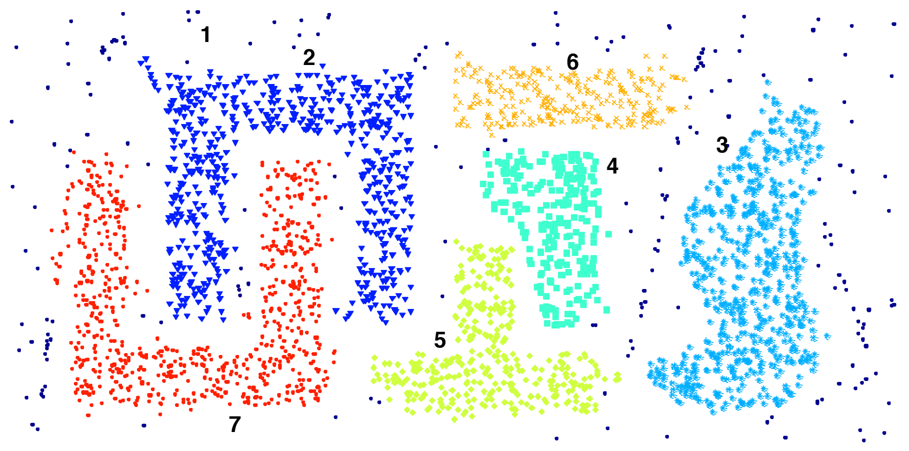
- Internal index: used to measure the goodness of a clustering structure without respect to external information
- SSE can be used for comparing two clusterings (e.g., based on different numbers of clusters) or two clusters (small SSE: tight; high SSE: loose)

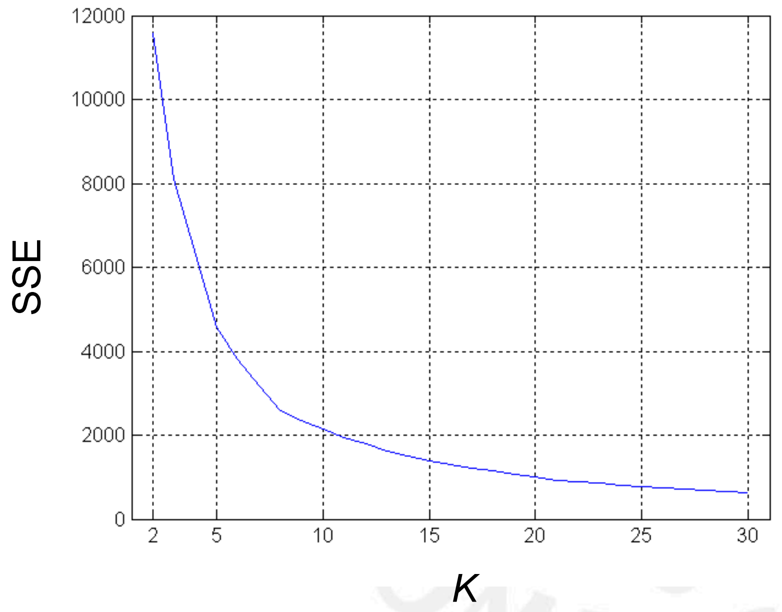




Internal Measures: Sum of Squared Errors

SSE curve for a more complicated data set and K-means



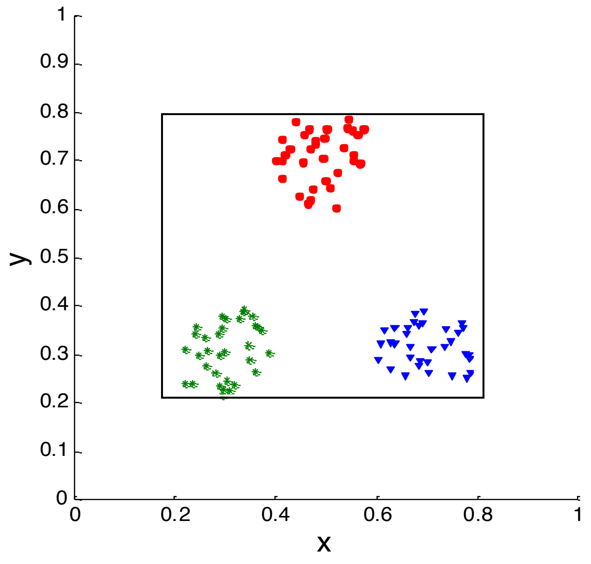


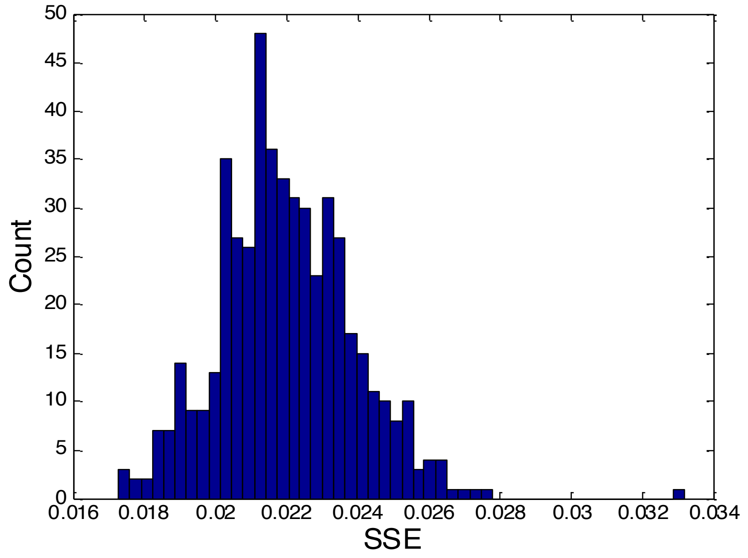
Framework for Cluster Validity

- Need a framework to interpret any measure.
  - For example, if our measure of evaluation has the value 10, is that good, fair, or poor?
- Statistics, in particular Monte Carlo sampling, provides a framework for cluster validity
  - The more “atypical” a clustering result is, the more likely it represents valid structure in the data
  - Can compare the values of an index obtained by clustering actual data with those obtained by clustering random data in the same range
  - If the value of the index is unlikely, then the cluster results are valid

Statistical Framework for SSE

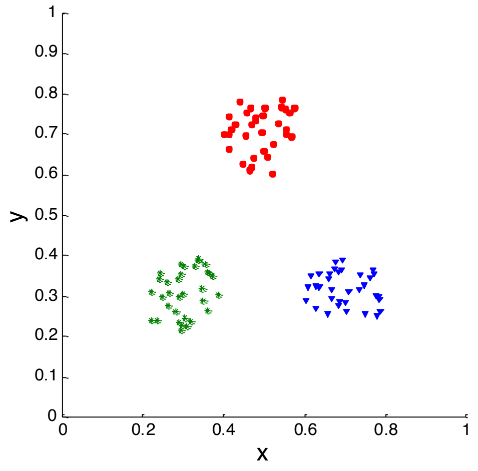
- Compare SSE of 0.005 against three clusters in random data
- Histogram shows SSE of three clusters in 500 sets of random data points of size 100 distributed over the range [0.2,0.8] for *x* and *y* values

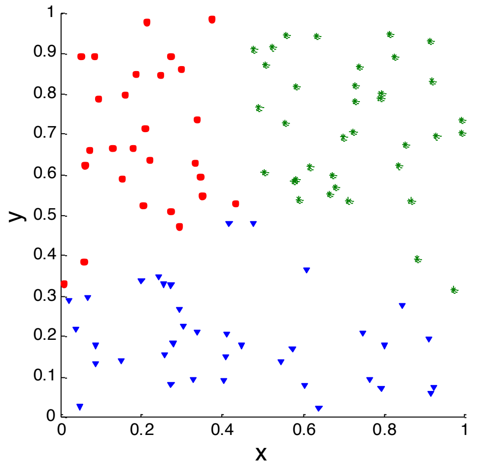


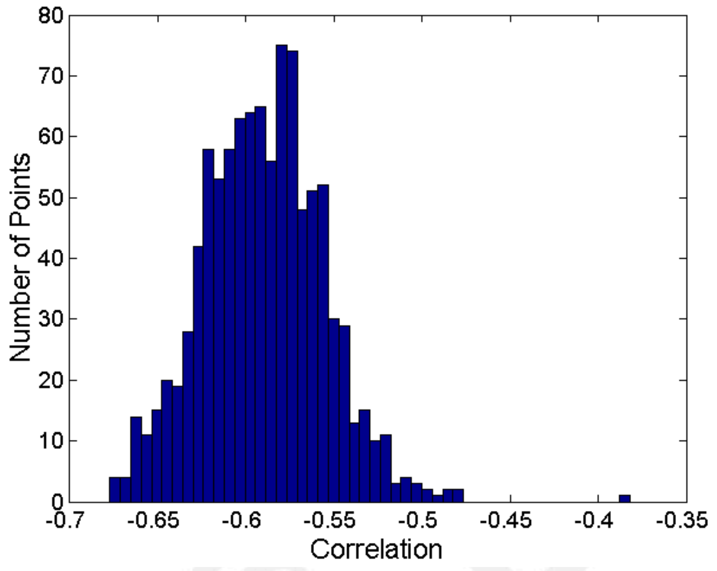


Statistical Framework for Correlation

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







Correlation = -0.9235

Correlation = -0.5810

Internal Measures: Cohesion and Separation (1)

- Cluster Cohesion measures how closely related are objects in a cluster, e.g., through the within cluster sum of squared errors:

$$WSS = \sum_i \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \mathbf{m}_i\|^2$$

- Cluster Separation measure how distinct or well-separated a cluster is from other clusters, e.g., through the between cluster sum of squares:

$$BSS = \sum_i |C_i| \|\mathbf{m} - \mathbf{m}_i\|^2$$

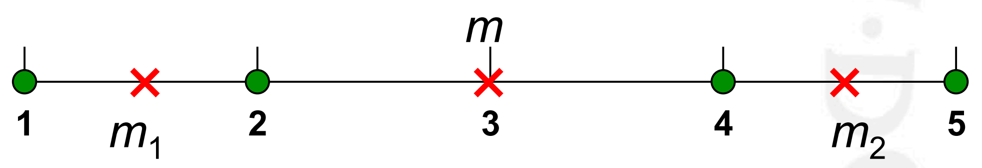
with $|C_i|$ is the size of (i.e., number of data points belonging to) cluster *i* and **m** the overall mean of the data

Internal Measures: Cohesion and Separation (2)

- It can be be shown that

$$WSS + BSS = \sum_i \|\mathbf{x}_i - \mathbf{m}\|^2 = \text{constant}$$

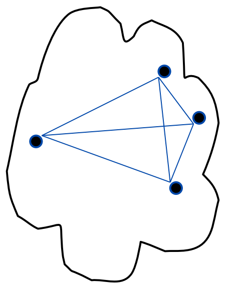
i.e., independent of the clustering

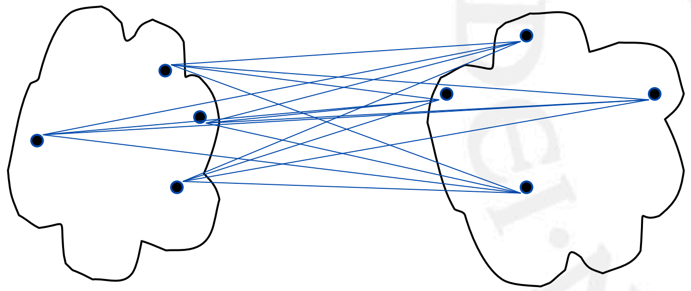


- K=1 cluster: WSS = $(3-1)^2+(3-2)^2+(4-3)^2+(5-3)^2=10$; BSS = 0
- K=2 clusters: WSS = $(1.5-1)^2+(2-1.5)^2+(4.5-4)^2+(5-4.5)^2 = 1$;
  BSS = $2\times(3-1.5)^2+2\times (4.5-3)^2 = 9$

Internal Measures: Cohesion and Separation (3)

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


cohesion


separation

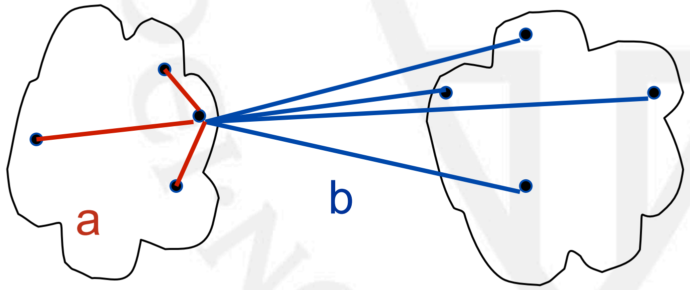
Internal Measures: Silhouette Coefficient

- The silhouette coefficient combines ideas of both cohesion and separation

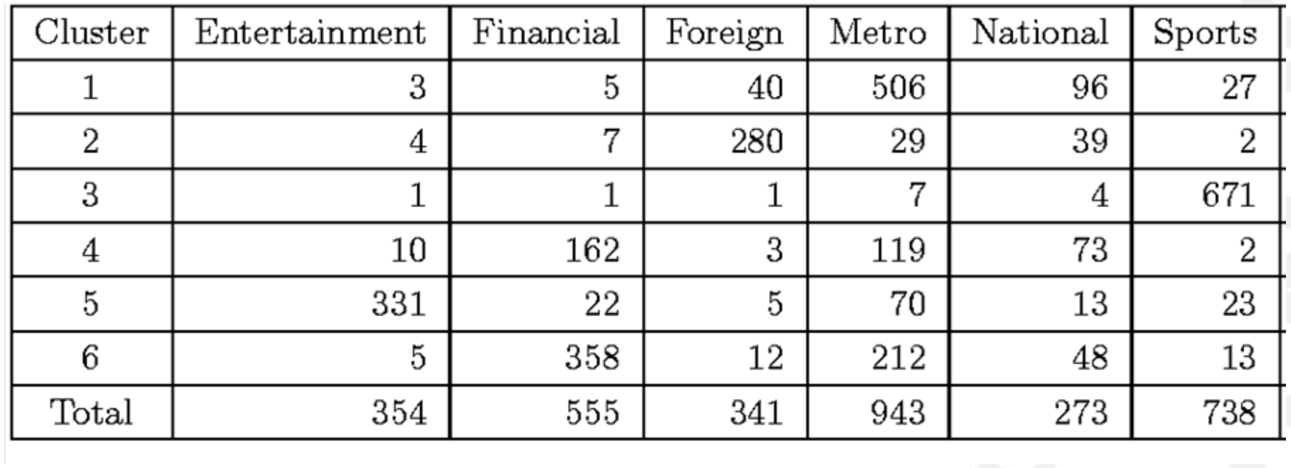
- Silhouette for data point *i*: $s_i = (b_i - a_i) / \max(a_i, b_i)$, where
  - a_i = average distance of *i* to the points in the same cluster
  - b_i = min (average distance of *i* to points in another cluster), where the minimum is taken over all other clusters

- Typically $b_i \geq a_i$ and then $0 \leq s_i \leq 1$; the higher the better

- Often also averaged over clusters and clusterings



External Measures of Cluster Validity



Cluster	Entertainment	Financial	Foreign	Metro	National	Sports
1	3	5	40	506	96	27
2	4	7	280	29	39	2
3	1	1	1	7	4	671
4	10	162	3	119	73	2
5	331	22	5	70	13	23
6	5	358	12	212	48	13
Total	354	555	341	943	273	738

- When given labels (e.g., afterwards), there are better/easier ways to evaluate clusterings using so-called external measures
- In the above clustering, which of the clusters performs best?

Purity and Entropy (1)

- "Probability" that a member of cluster *j* belongs to class *i*:

$$p_{ij} = \frac{N_{ij}}{N_j} \text{ with } N_j = \sum_i N_{ij}$$

  with N_{ij} the number of data points belonging to cluster *j* and class *i*

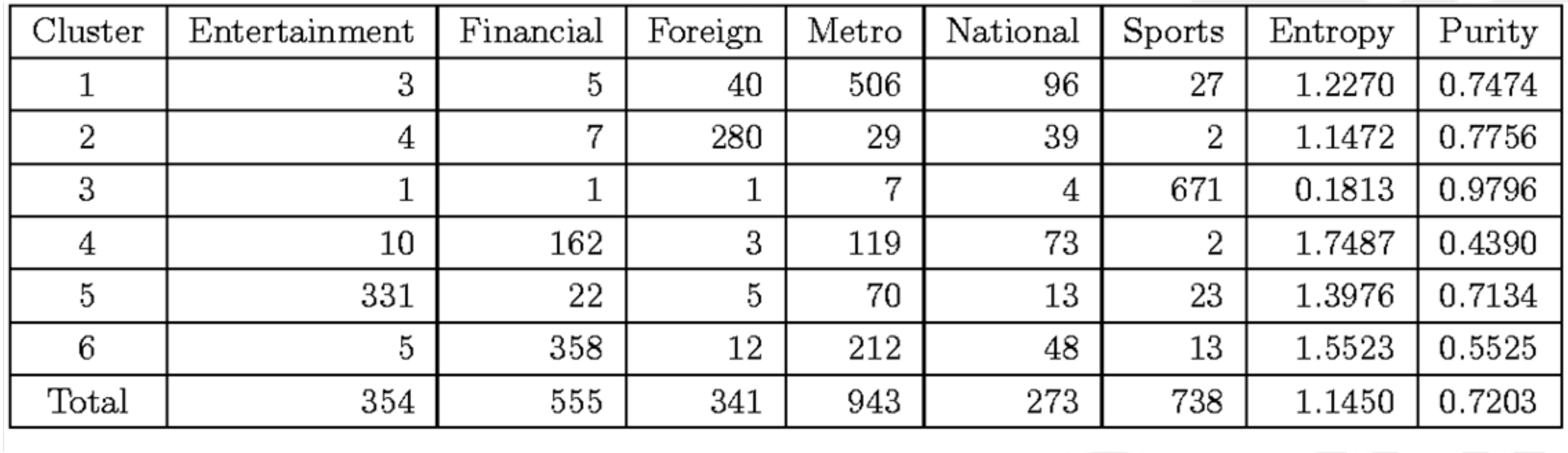
- Purity of a cluster: $purity_j = \max_i p_{ij}$
  - quality when you assign cluster *j* to the most likely corresponding class

- Entropy of a cluster: $entropy_j = \sum_i p_{ij} \, log_2 \, p_{ij}$
  - lower is better

- Total entropy/purity of a clustering:

$$\{entropy, purity\} = \sum_j \frac{N_j}{N} \{entropy_j, purity_j\}$$

Purity and Entropy (2)



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

Final Comment on Cluster Validity

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

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

Algorithms for Clustering Data, Jain and Dubes