Data Mining
Cluster Analysis: Basic Concepts and Algorithms

Lecture Notes for Chapter 7

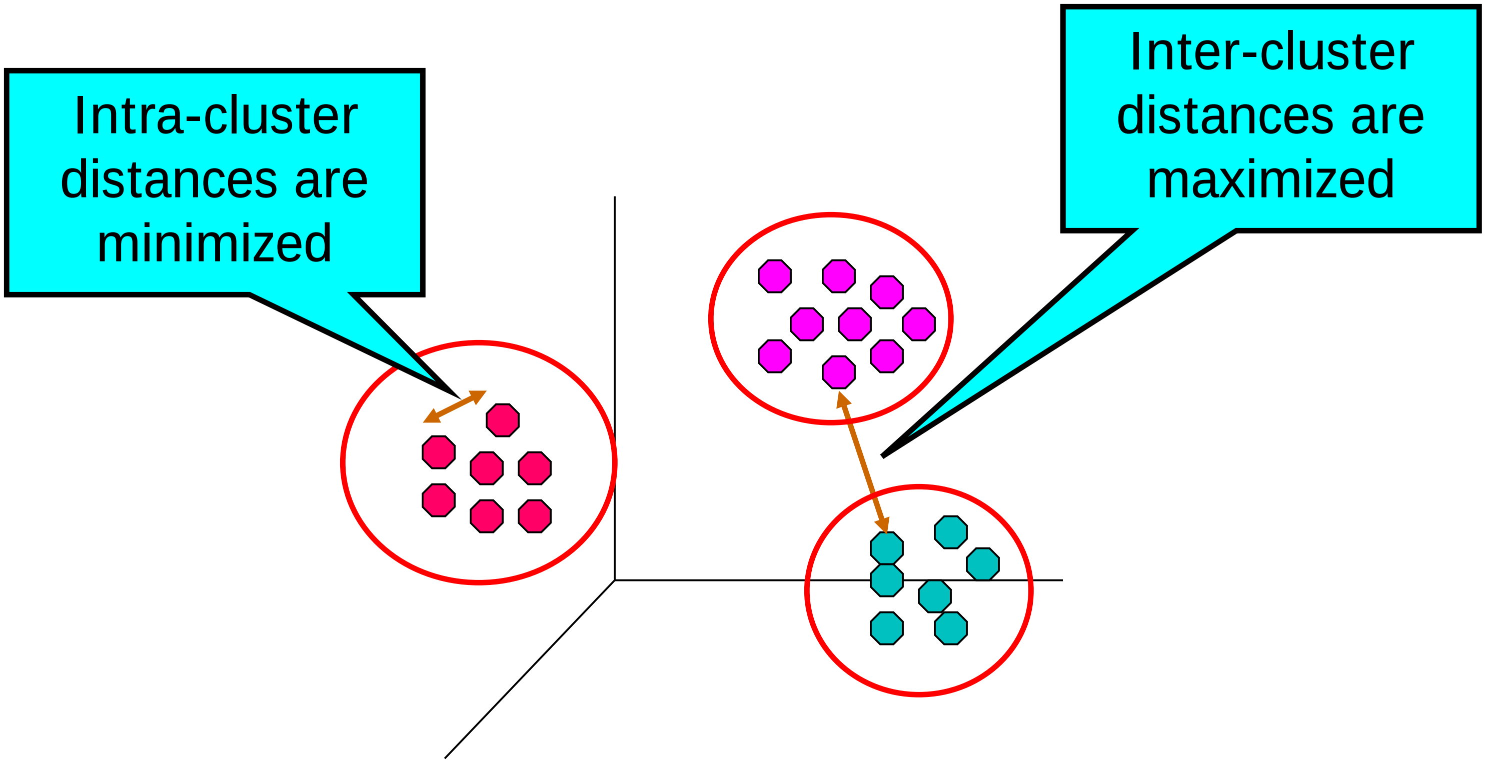
Introduction to Data Mining

by

Tan, Steinbach, Kumar

What is Cluster Analysis?

- Given a set of objects, place them in groups such that the objects in a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups

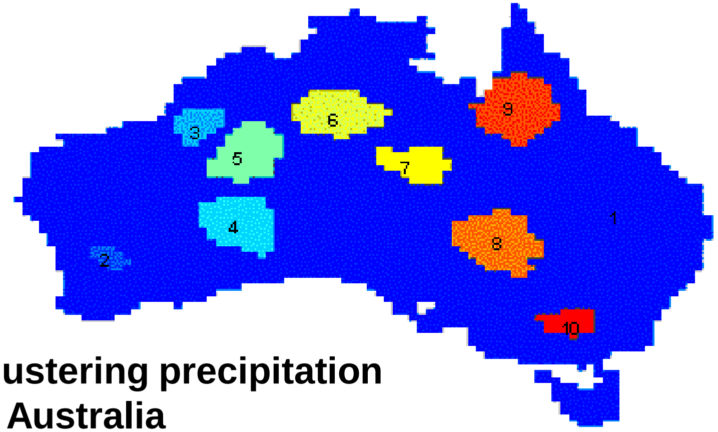


Applications of Cluster Analysis

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

	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

- **Summarization**
  - Reduce the size of large data sets



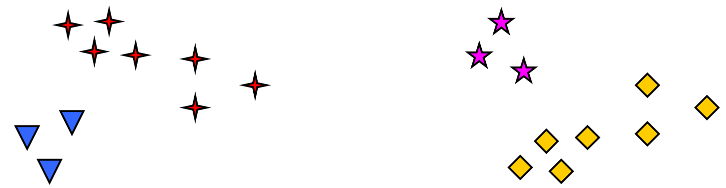
Clustering precipitation in Australia

Notion of a Cluster can be Ambiguous

How many clusters?

Six Clusters

Two Clusters

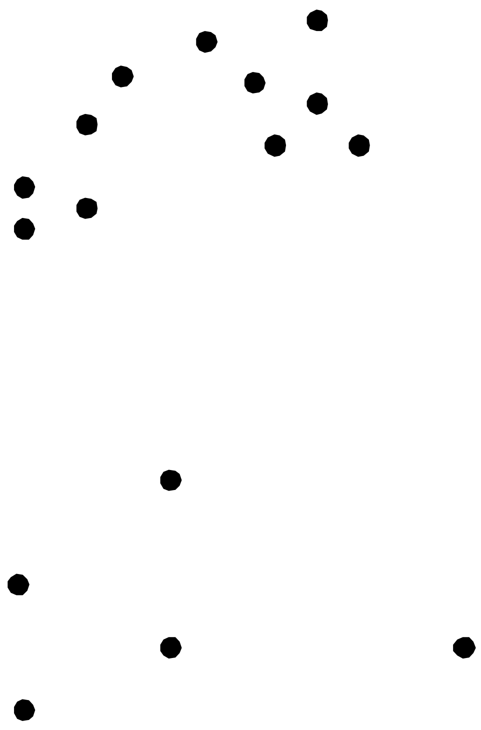


Four Clusters

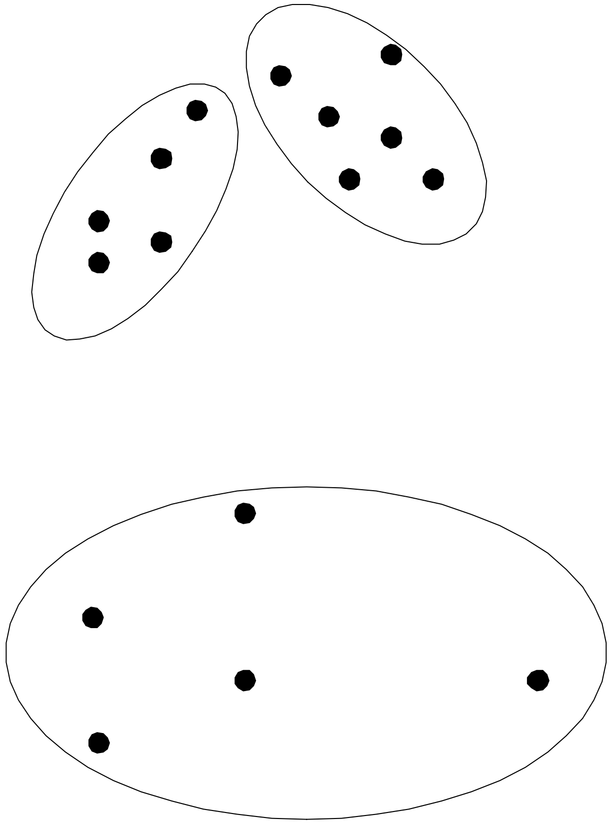
Types of Clusterings

- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
  - Partitional Clustering
    - A division of data objects into non-overlapping subsets (clusters)
  - Hierarchical clustering
    - A set of nested clusters organized as a hierarchical tree

Partitional Clustering

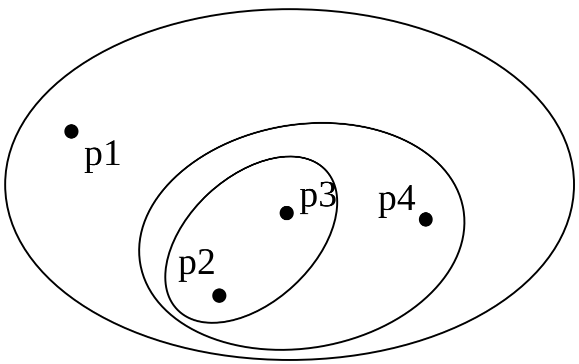


Original Points

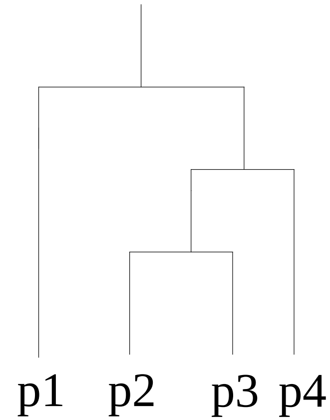


A Partitional Clustering

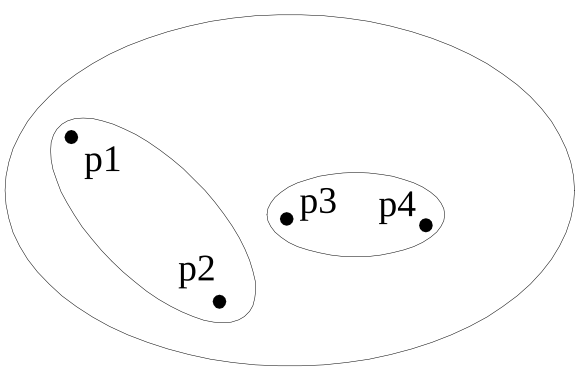
Hierarchical Clustering



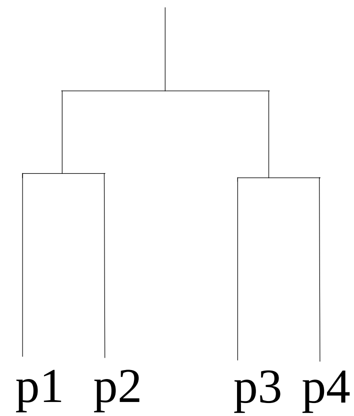
Traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Hierarchical Clustering



Non-traditional Dendrogram

Other Distinctions Between Sets of Clusters

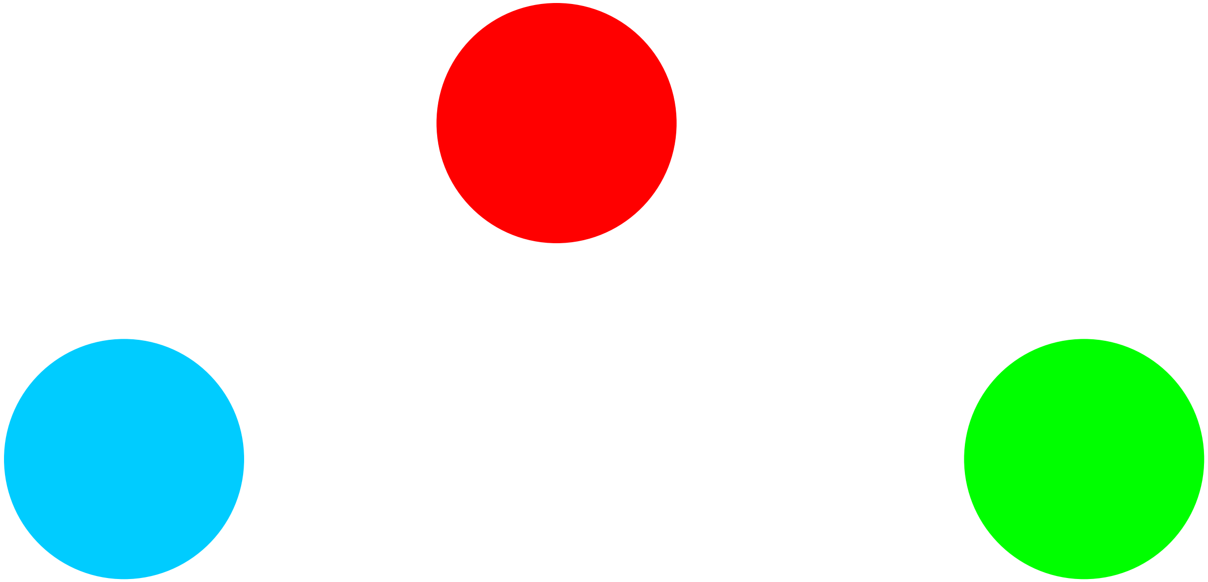
- Exclusive versus non-exclusive
  - In non-exclusive clusterings, points may belong to multiple clusters.
    - Can belong to multiple classes or could be ‘border’ points
  - Fuzzy clustering (one type of non-exclusive)
    - In fuzzy clustering, a point belongs to every cluster with some weight between 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

Types of Clusters

- Well-separated clusters
- Prototype-based clusters
- Contiguity-based clusters
- Density-based clusters
- Described by an Objective Function

Types of Clusters: Well-Separated

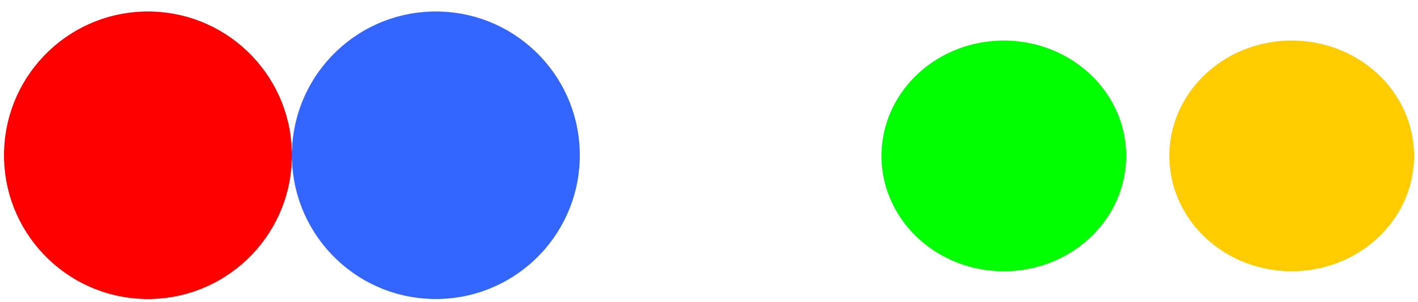
- Well-Separated Clusters:
  - A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.



3 well-separated clusters

Types of Clusters: Prototype-Based

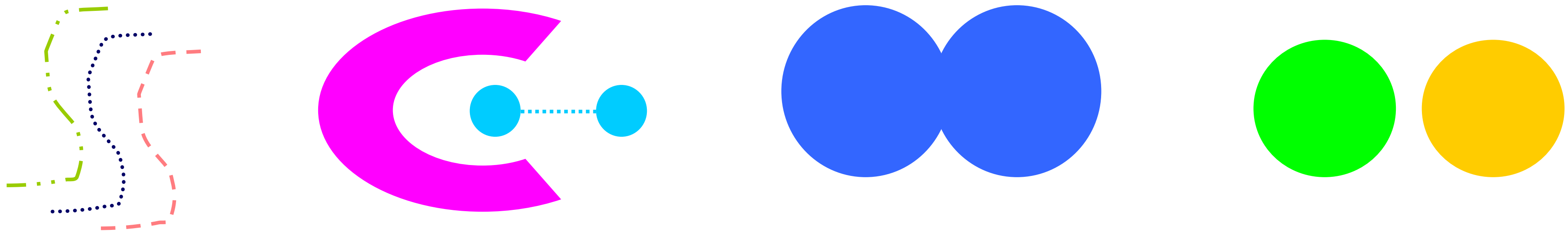
- Prototype-based
  - A cluster is a set of objects such that an object in a cluster is closer (more similar) to the prototype or "center" of a cluster, than to the center of any other cluster
  - The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most "representative" point of a cluster



4 center-based clusters

Types of Clusters: Contiguity-Based

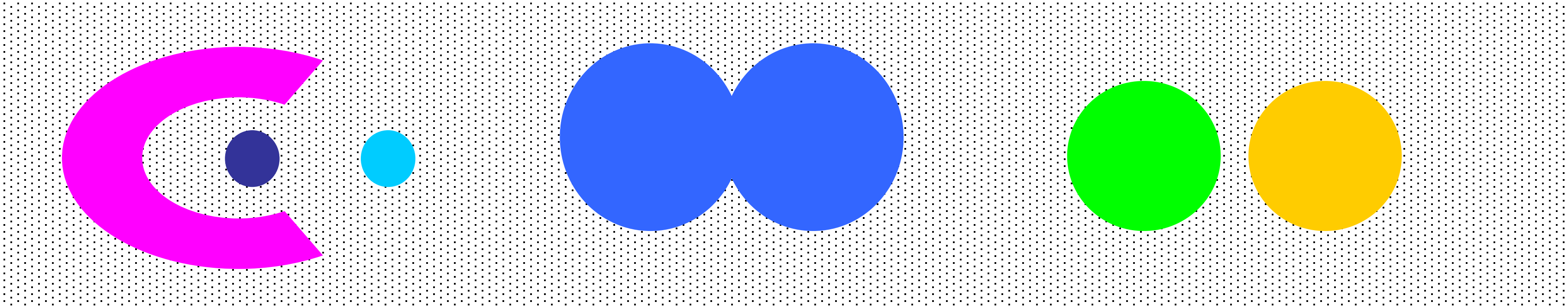
- Contiguous Cluster (Nearest neighbor or Transitive)
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.



8 contiguous clusters

Types of Clusters: Density-Based

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



6 density-based clusters

Types of Clusters: Objective Function

- Clusters Defined by an Objective Function
  - Finds clusters that minimize or maximize an objective function.
  - Enumerate all possible ways of dividing the points into clusters and evaluate the `goodness' of each potential set of clusters by using the given objective function. (NP Hard)
  - Can have global or local objectives.
    - Hierarchical clustering algorithms typically have local objectives
    - Partitional algorithms typically have global objectives
  - A variation of the global objective function approach is to fit the data to a parameterized model.
    - Parameters for the model are determined from the data.
    - Mixture models assume that the data is a ‘mixture' of a number of statistical distributions.

Characteristics of the Input Data Are Important

- Type of proximity or density measure
  - Central to clustering
  - Depends on data and application

- Data characteristics that affect proximity and/or density are
  - Dimensionality
    - Sparseness
  - Attribute type
  - Special relationships in the data
    - For example, autocorrelation
  - Distribution of the data

- Noise and Outliers
  - Often interfere with the operation of the clustering algorithm
- Clusters of differing sizes, densities, and shapes

Clustering Algorithms

- K-means and its variants

- Hierarchical clustering

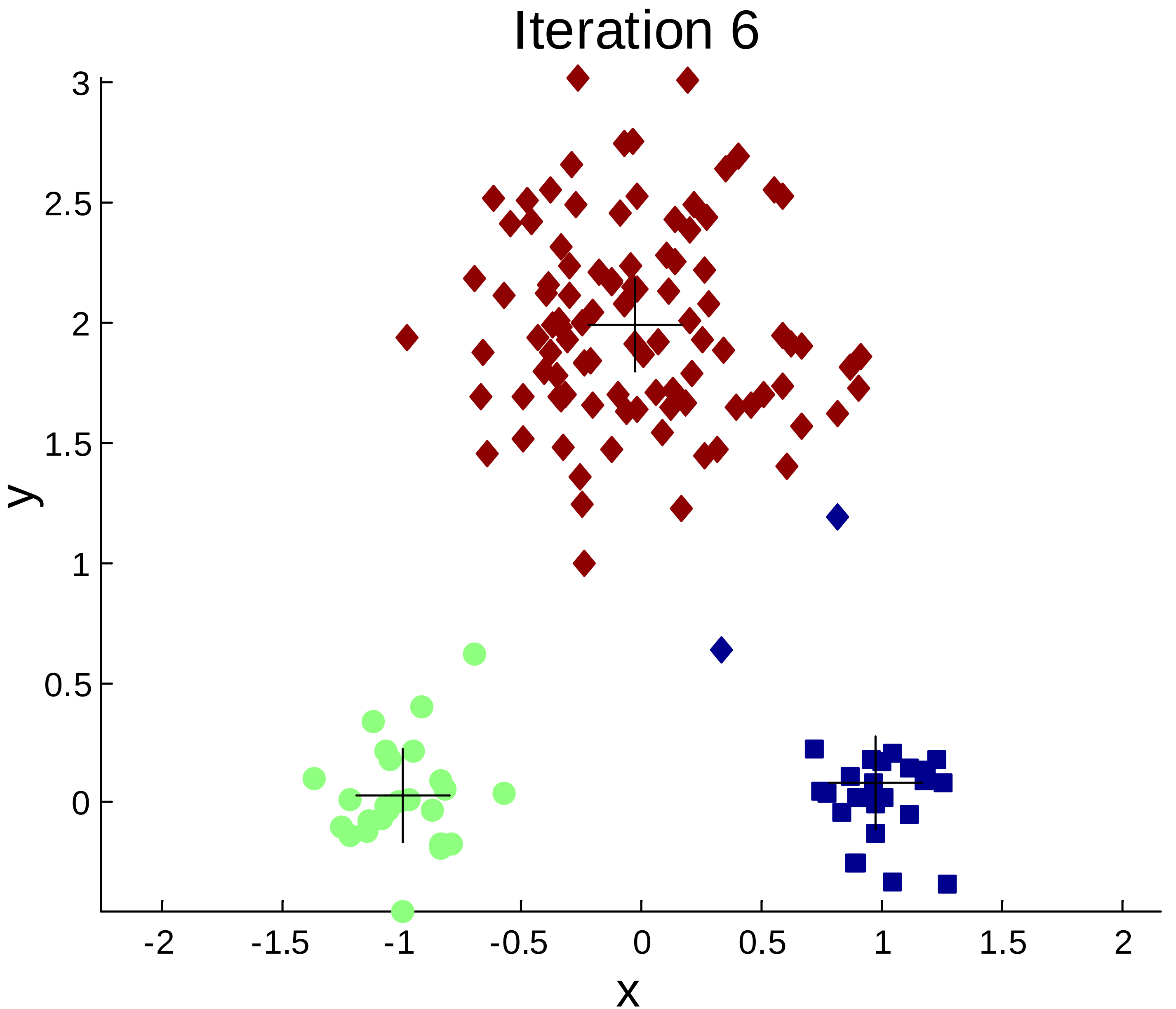
- Density-based clustering

K-means Clustering

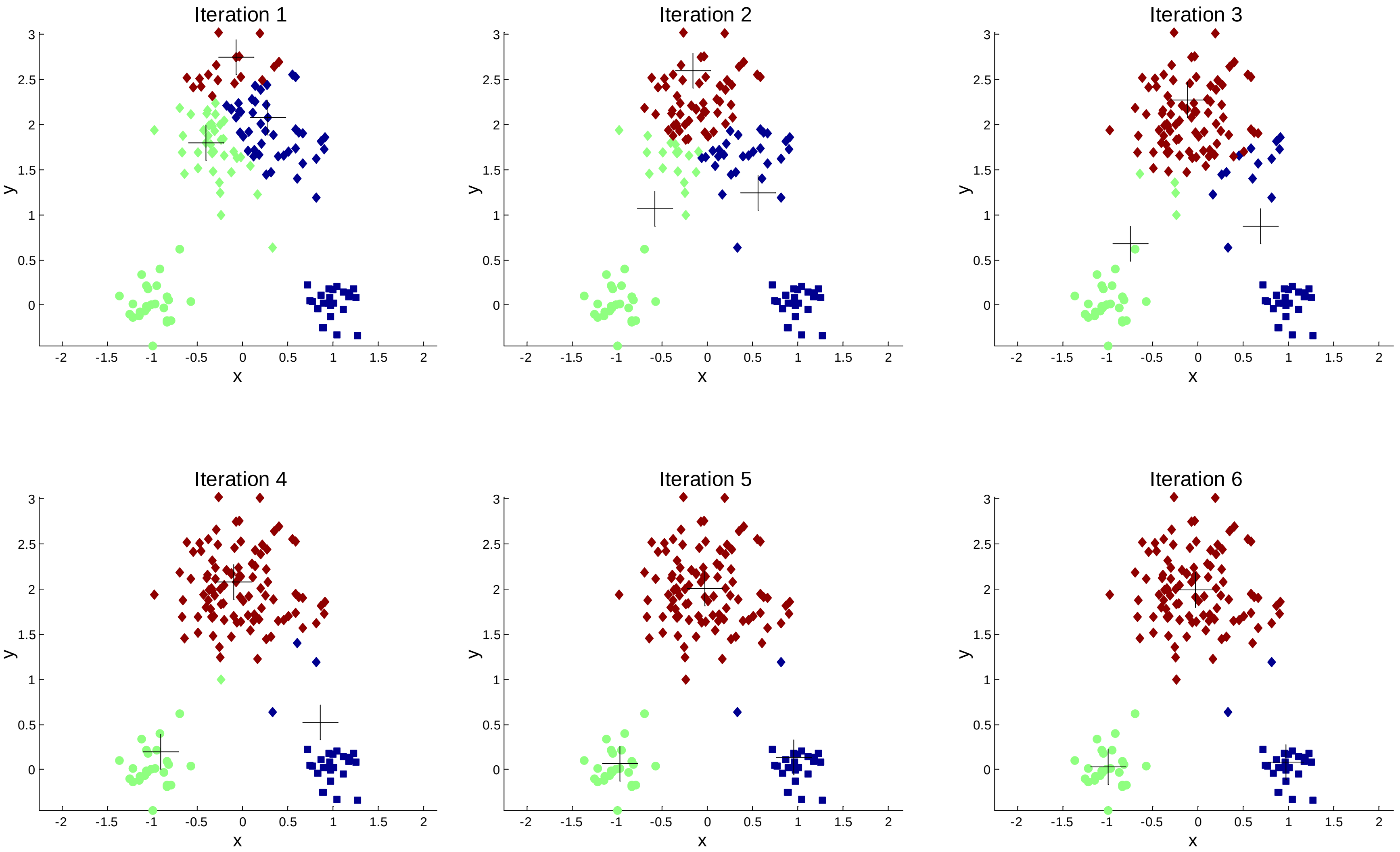
- Partitional clustering approach
- Number of clusters, K, must be specified
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- The basic algorithm is very simple

```
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
```

Example of K-means Clustering



Example of K-means Clustering



K-means Clustering – Details

- Simple iterative algorithm.
  - Choose initial centroids;
  - repeat {assign each point to a nearest centroid; re-compute cluster centroids}
  - until centroids stop changing.
- 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, but other definitions are possible (see Table 7.2).
- K-means will converge for common proximity measures with appropriately defined centroid (see Table 7.2)
- 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

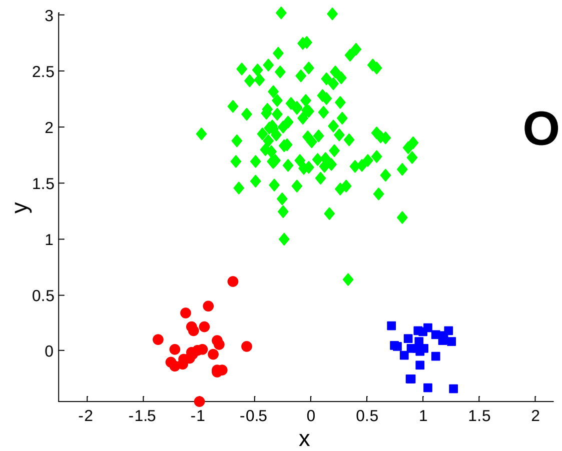
K-means Objective Function

- A common objective function (used with Euclidean distance measure) is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster center
  - To get SSE, we square these errors and sum them.

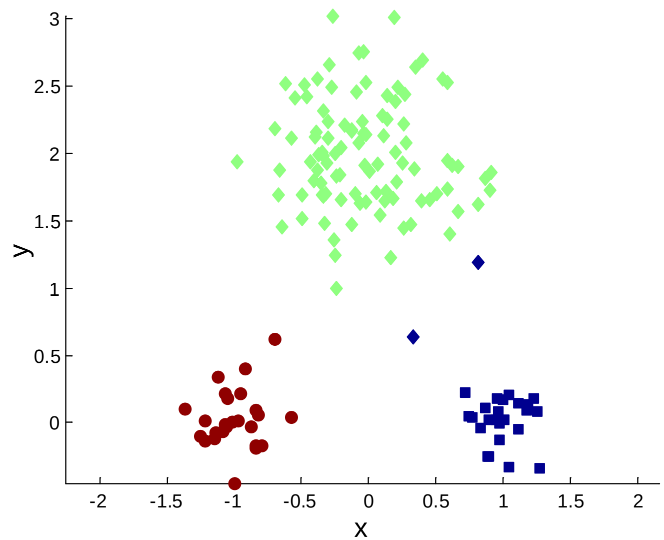
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

  - x is a data point in cluster C_i and m_i is the centroid (mean) for cluster C_i
  - SSE improves in each iteration of K-means until it reaches a local or global minima.

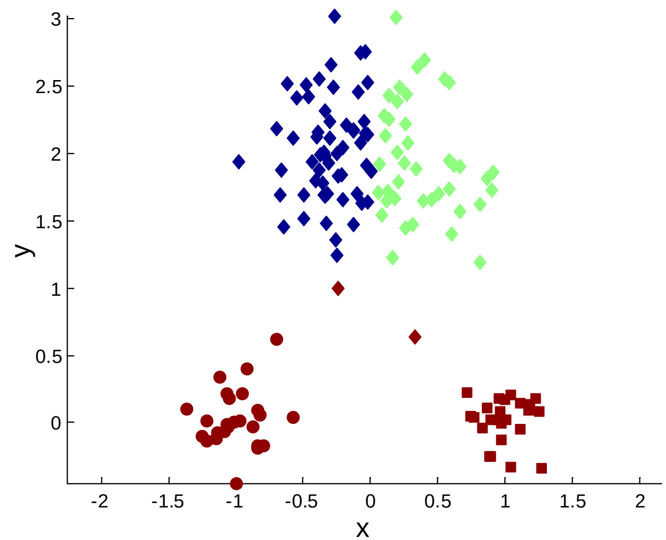
Two different K-means Clusterings



Original Points

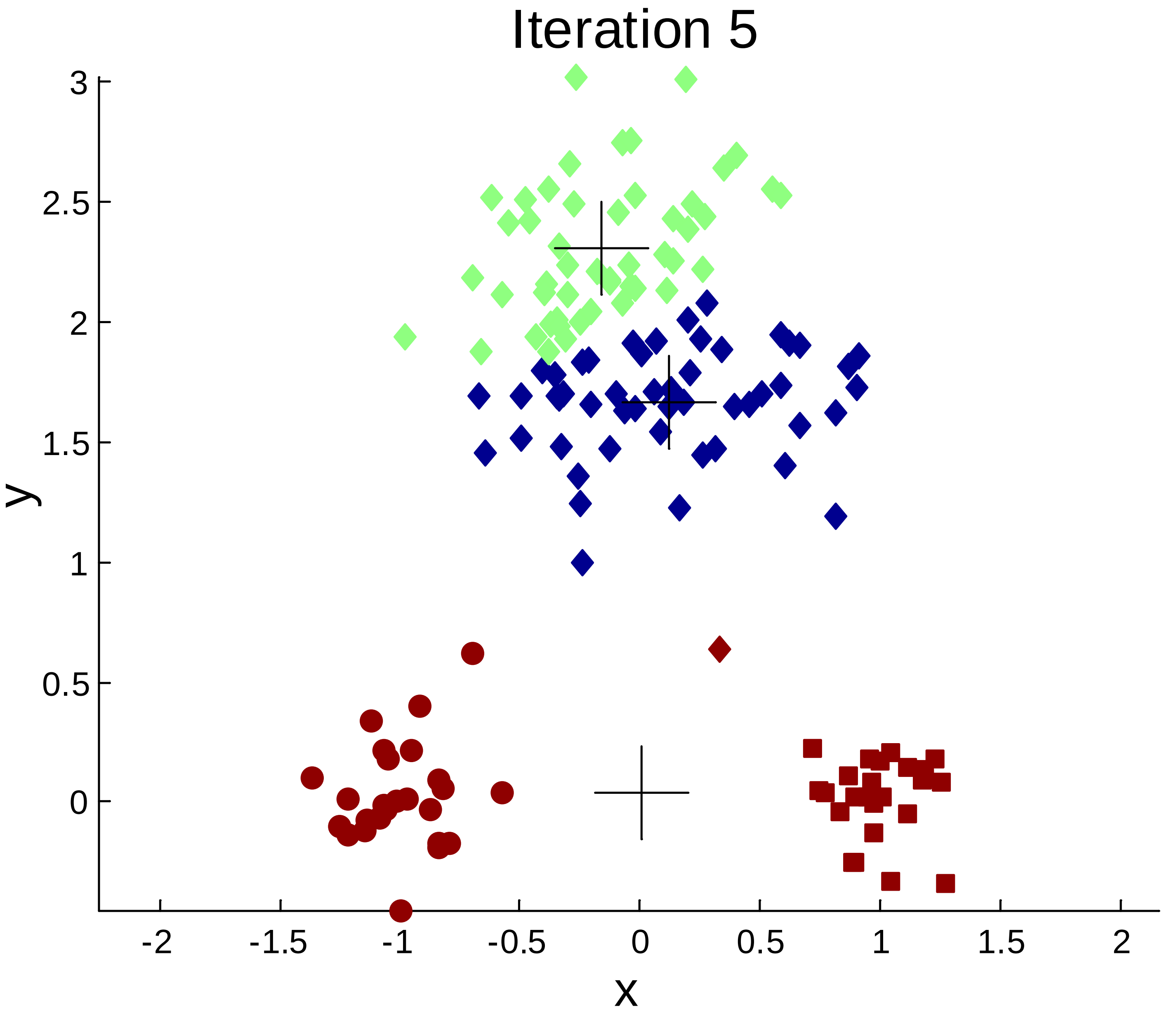


Optimal Clustering

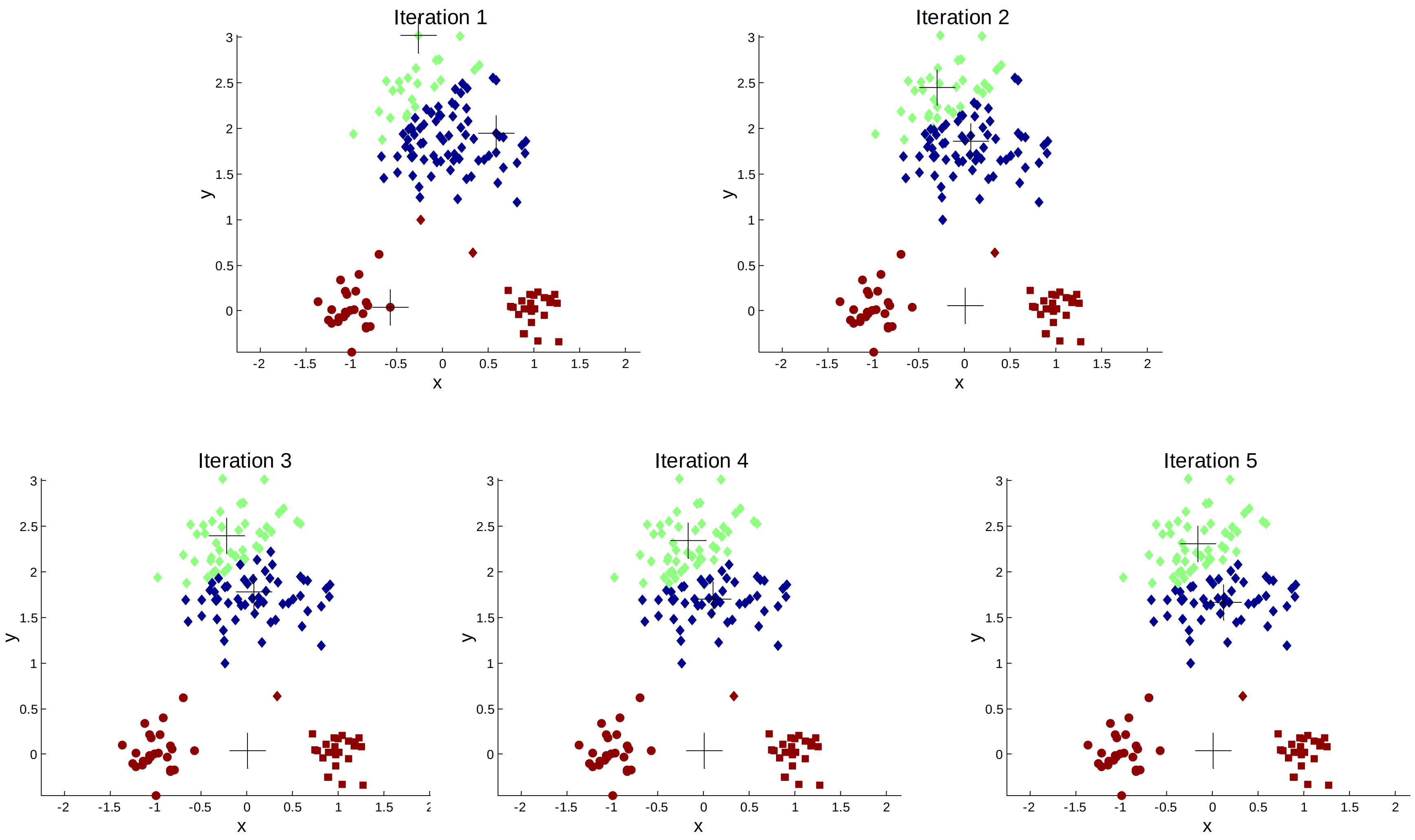


Sub-optimal Clustering

Importance of Choosing Initial Centroids …



Importance of Choosing Initial Centroids …



Importance of Choosing Intial Centroids

- Depending on the choice of initial centroids, B and C may get merged or remain separate

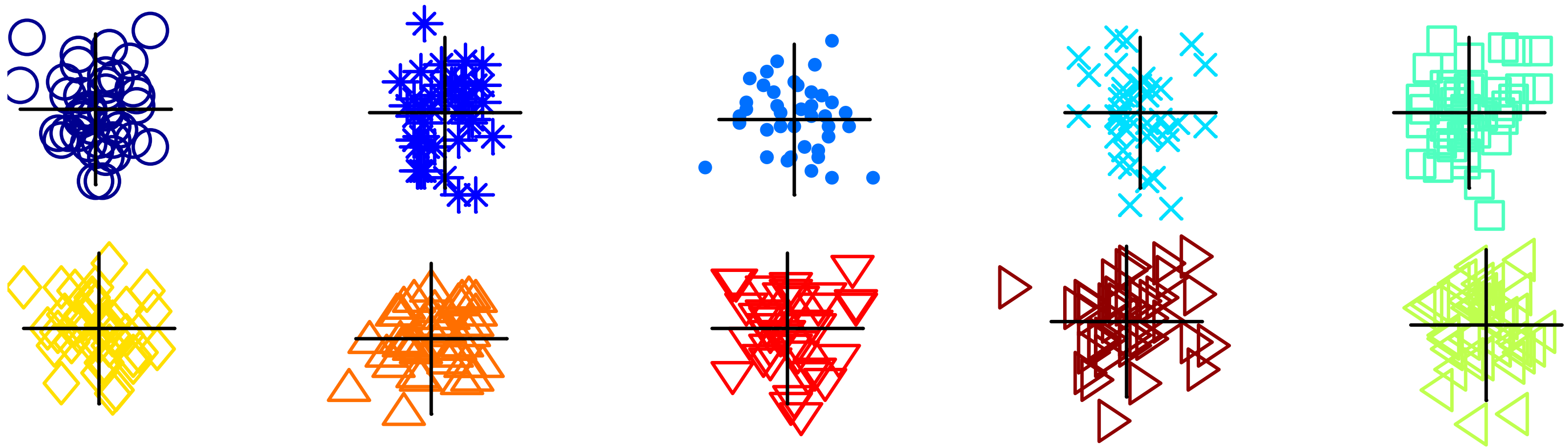
Problems with Selecting Initial Points

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

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

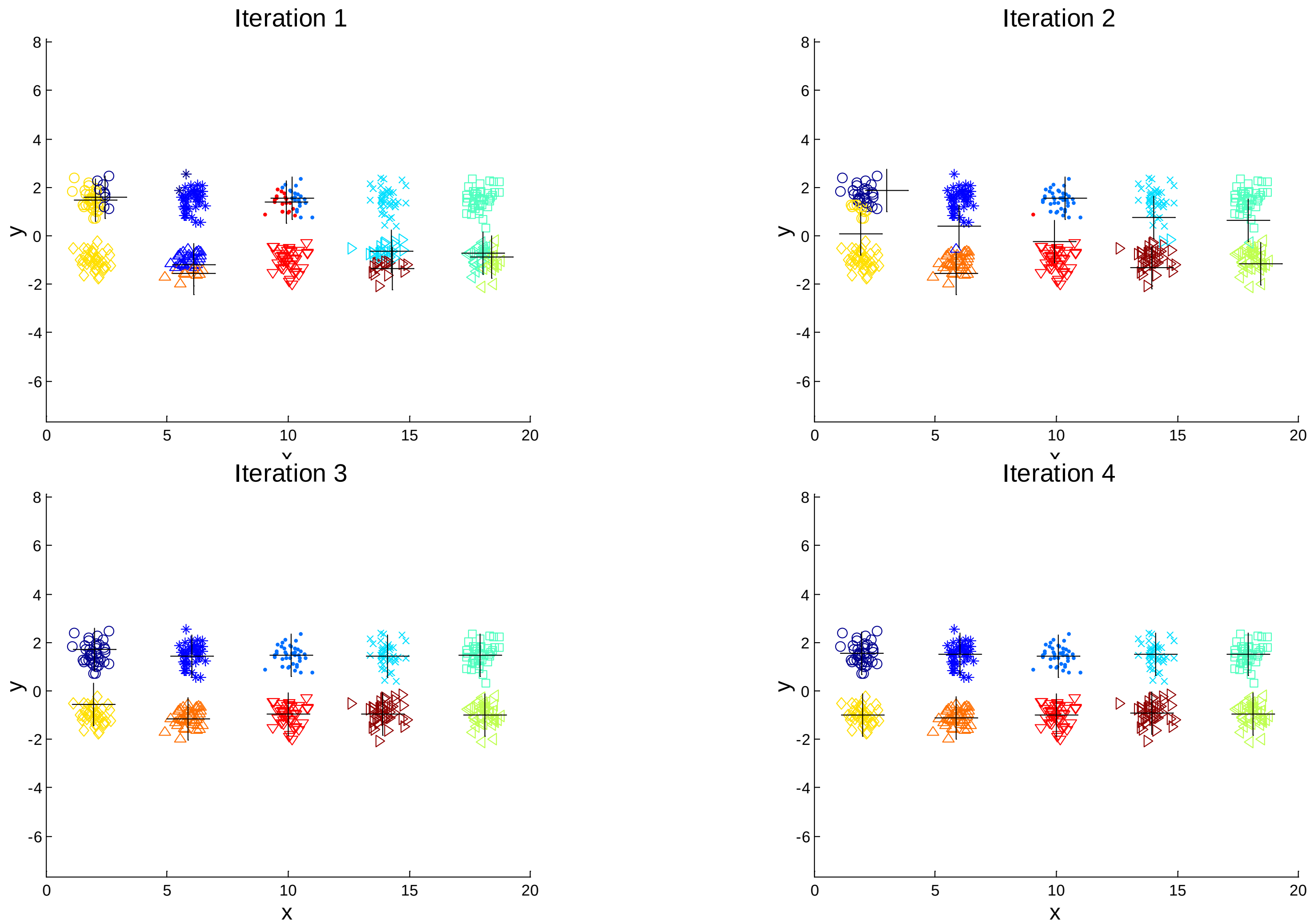
  - For example, if K = 10, then probability = $10!/10^{10}$ = 0.00036
  - Sometimes the initial centroids will readjust themselves in 'right' way, and sometimes they don't
  - Consider an example of five pairs of clusters

10 Clusters Example



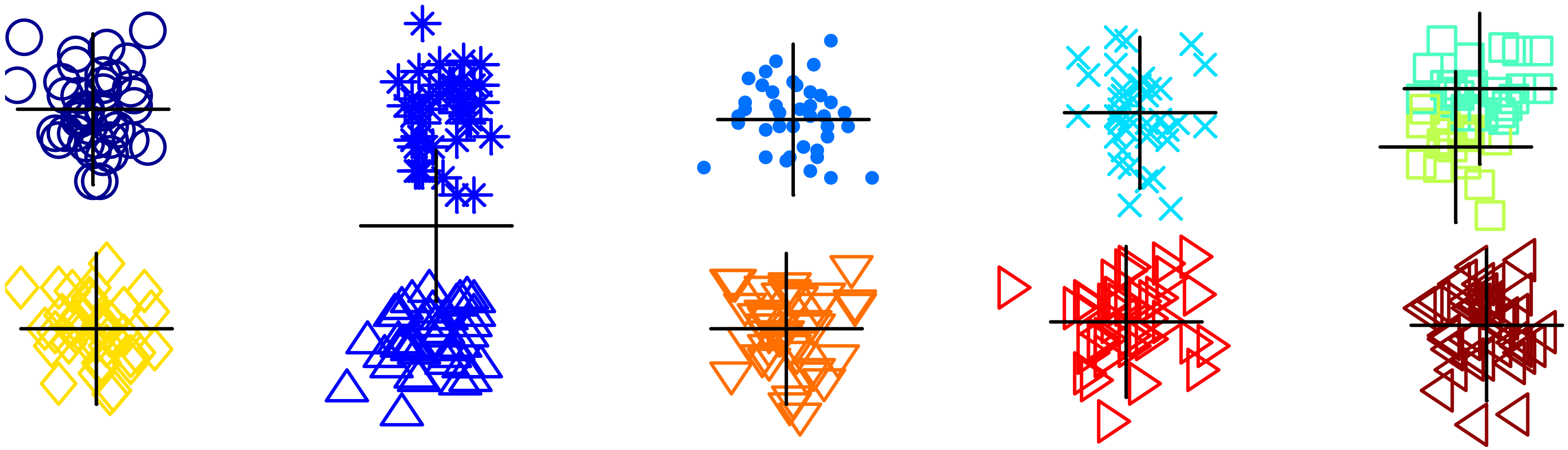
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example



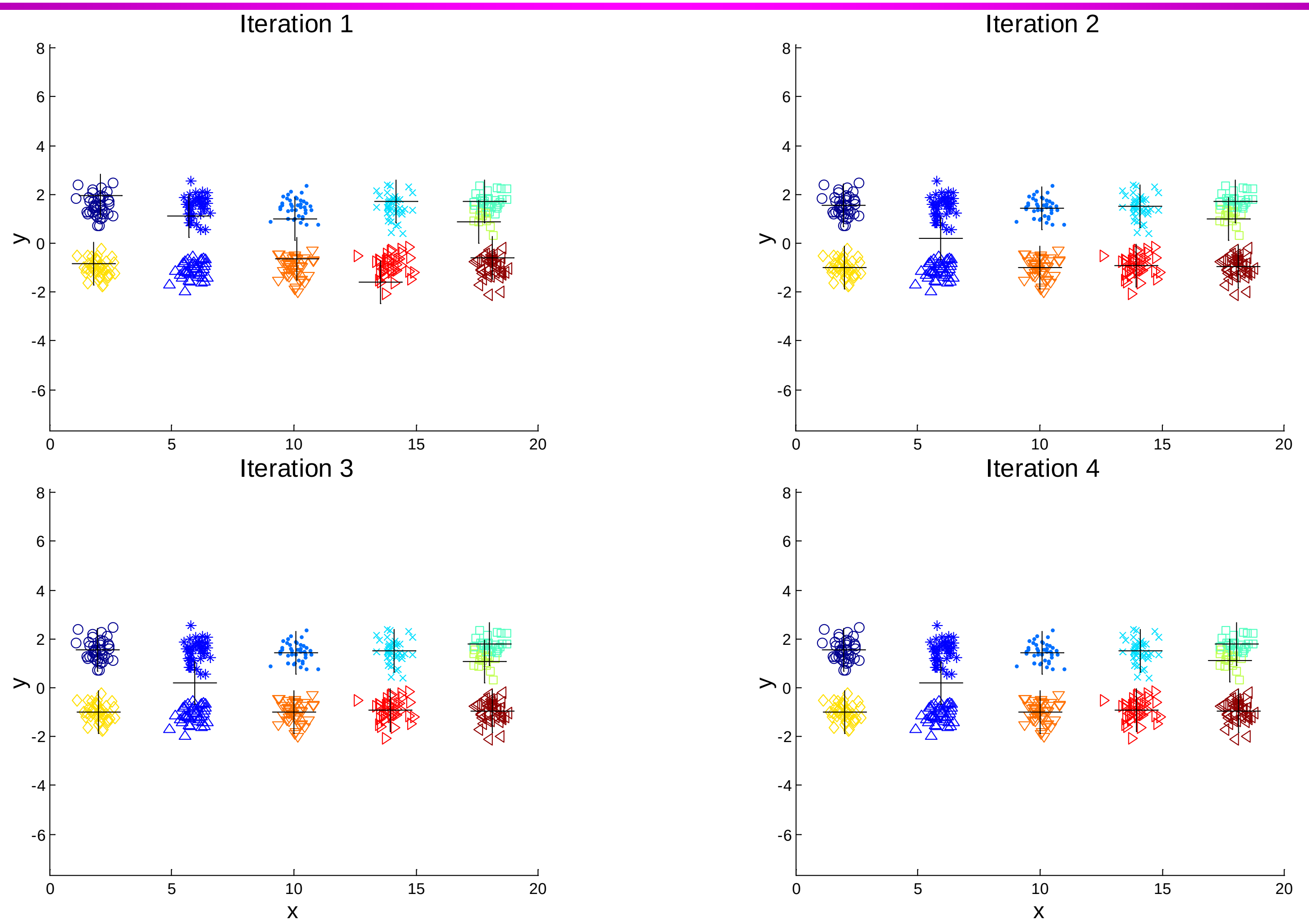
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example



Starting with some pairs of clusters having three initial centroids, while other have only one.

10 Clusters Example



Starting with some pairs of clusters having three initial centroids, while other have only one.

Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side
- Use some strategy to select the k initial centroids and then select among these initial centroids
  - Select most widely separated
    - K-means++ is a robust way of doing this selection
  - Use hierarchical clustering to determine initial centroids
- Bisecting K-means
  - Not as susceptible to initialization issues

K-means++

- This approach can be slower than random initialization, but very consistently produces better results in terms of SSE
  - The k-means++ algorithm guarantees an approximation ratio O(log k) in expectation, where k is the number of centers
- To select a set of initial centroids, *C*, perform the following

1. Select an initial point at random to be the first centroid
2. For k – 1 steps
3. For each of the N points, x_i, $1 \le i \le N$, find the minimum squared distance to the currently selected centroids, $C_1, \ldots, C_j$, $1 \le j < k$, i.e., $\min_j d^2(C_j, x_i)$
4. Randomly select a new centroid by choosing a point with probability proportional to $\frac{\min_j d^2(C_j, x_i)}{\sum_i \min_j d^2(C_j, x_i)}$ is
5. End For

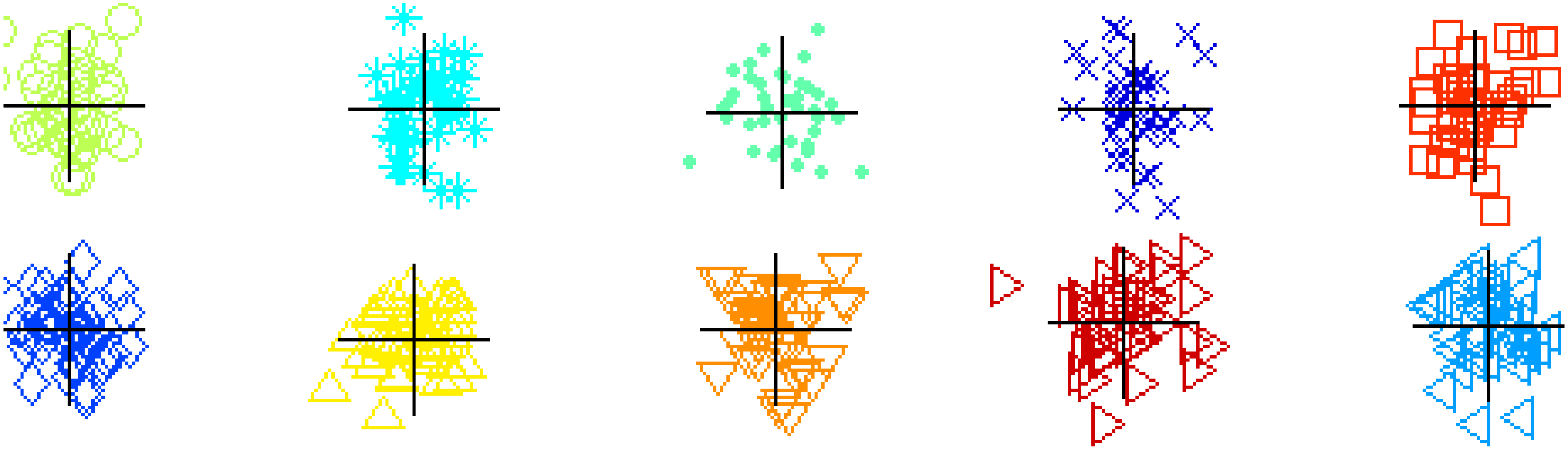
Bisecting K-means

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

```
Initialize the list of clusters to contain the cluster containing all points.
repeat
    Select a cluster from the list of clusters
    for i = 1 to number_of_iterations do
        Bisect the selected cluster using basic K-means
    end for
    Add the two clusters from the bisection with the lowest SSE to the list of clusters.
until Until the list of clusters contains K clusters
```

CLUTO: http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview

Bisecting K-means Example

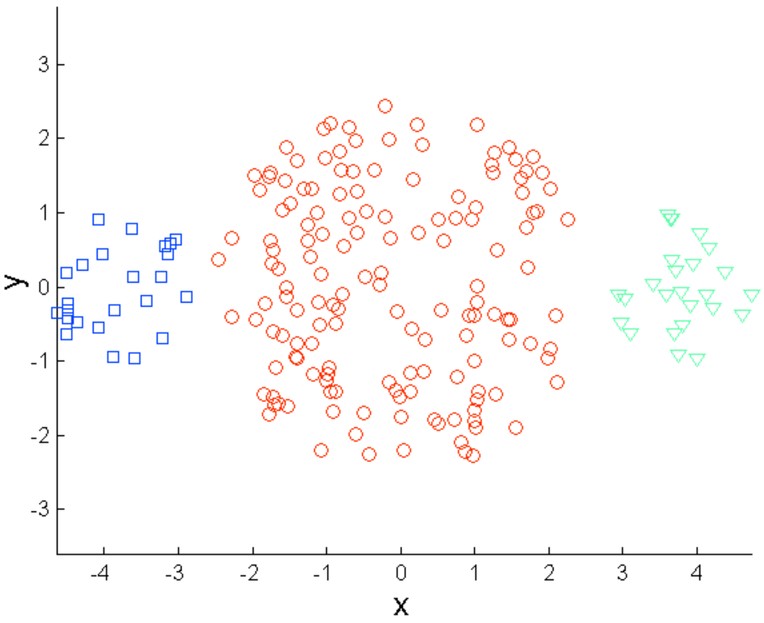


Limitations of K-means

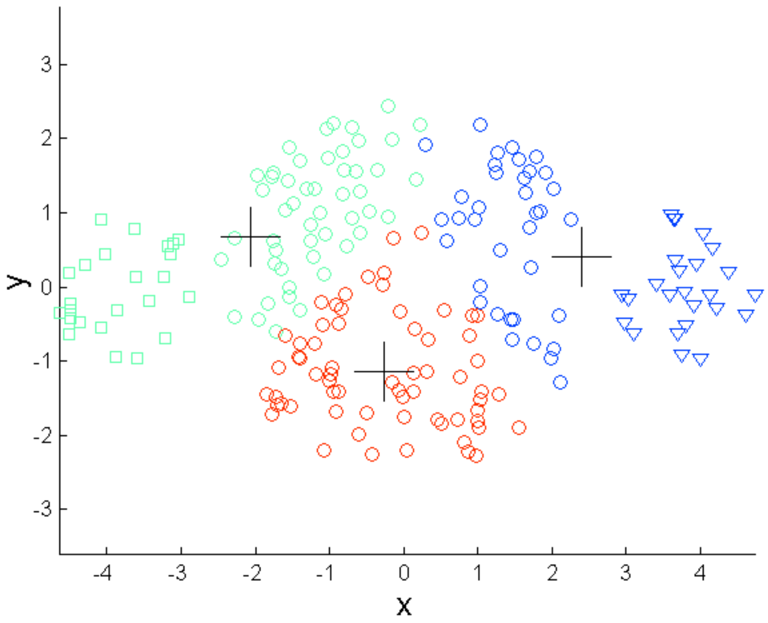
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes

- K-means has problems when the data contains outliers.
  - One possible solution is to remove outliers before clustering

Limitations of K-means: Differing Sizes

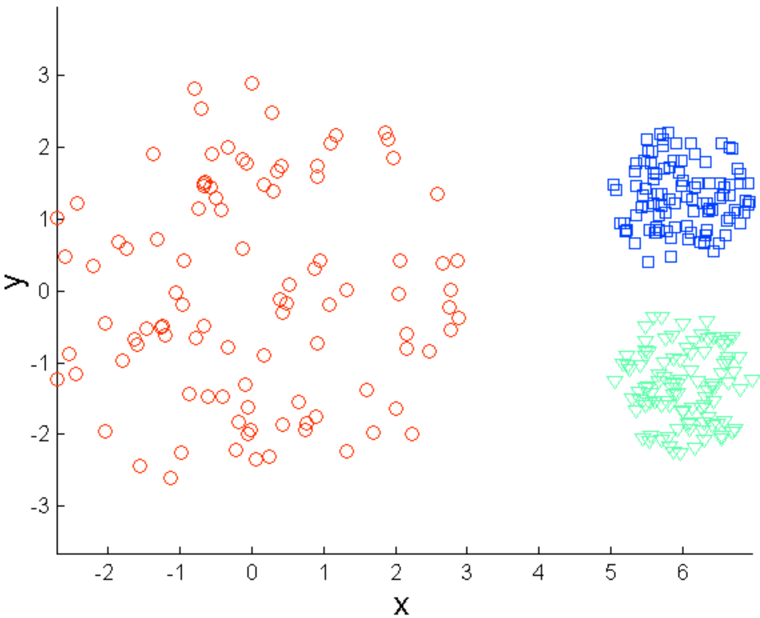


Original Points

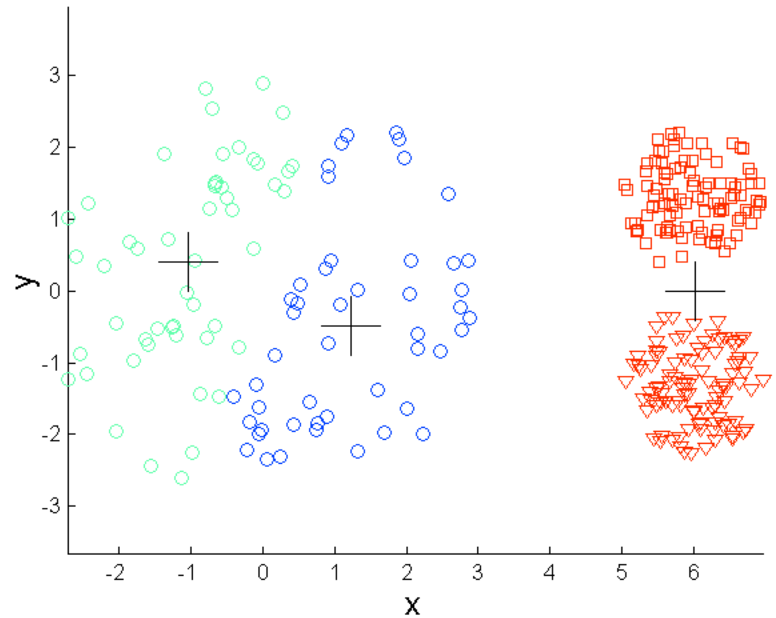


K-means (3 Clusters)

Limitations of K-means: Differing Density

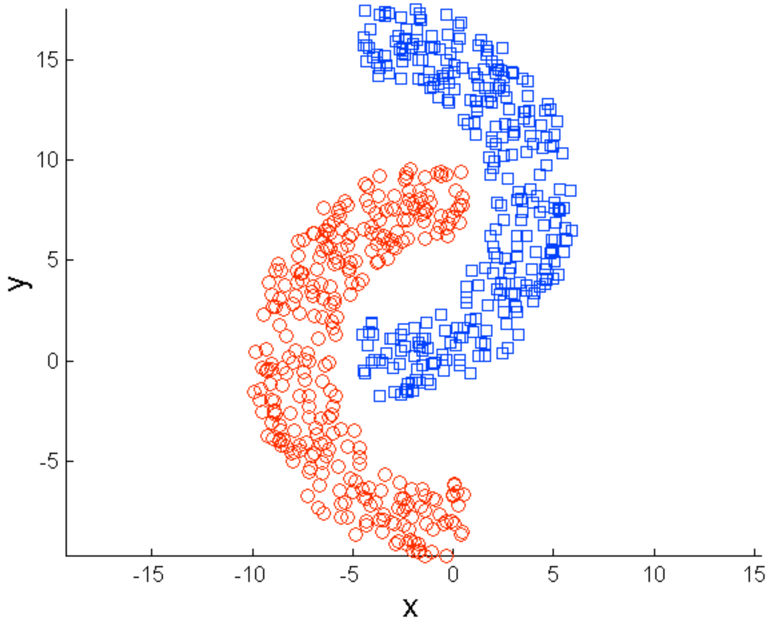


Original Points

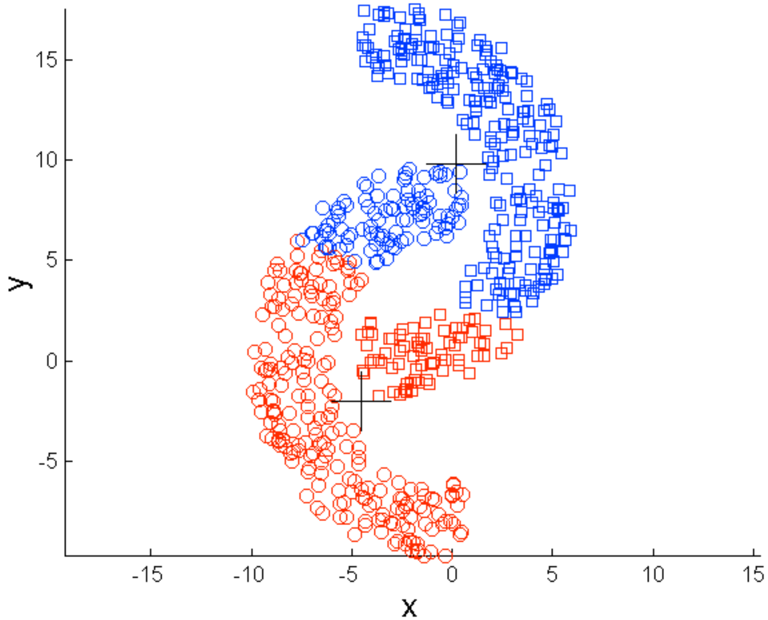


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

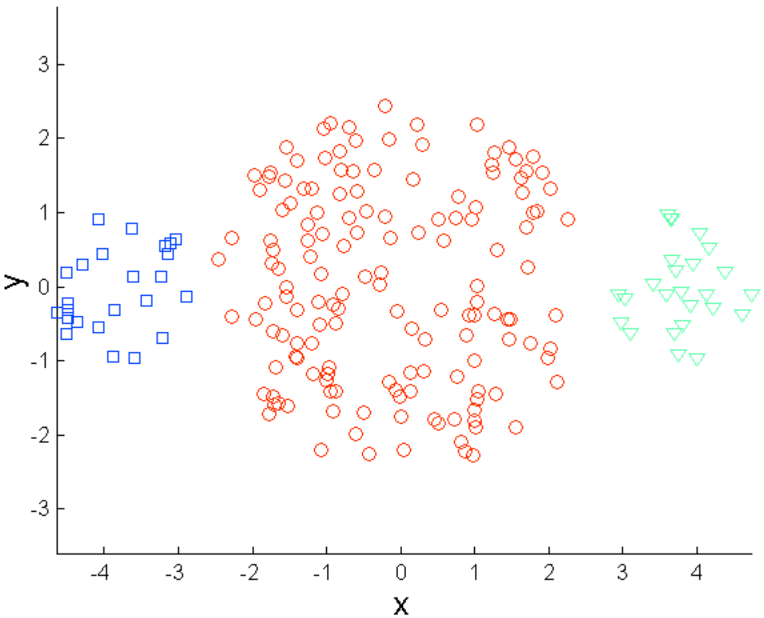


Original Points

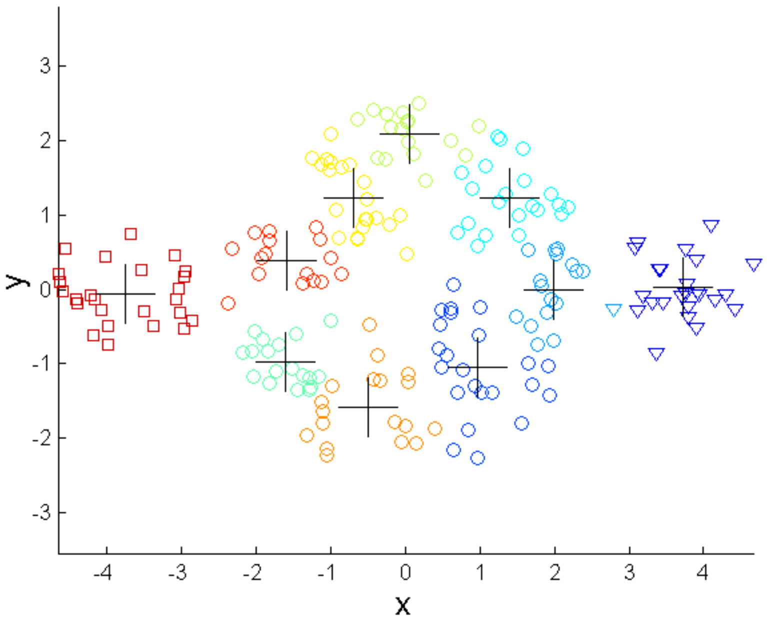


K-means (2 Clusters)

Overcoming K-means Limitations



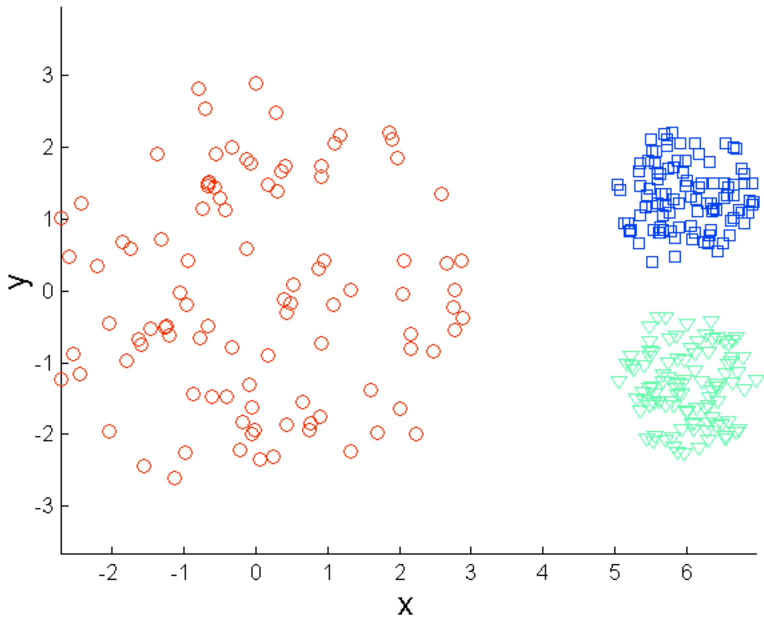
Original Points



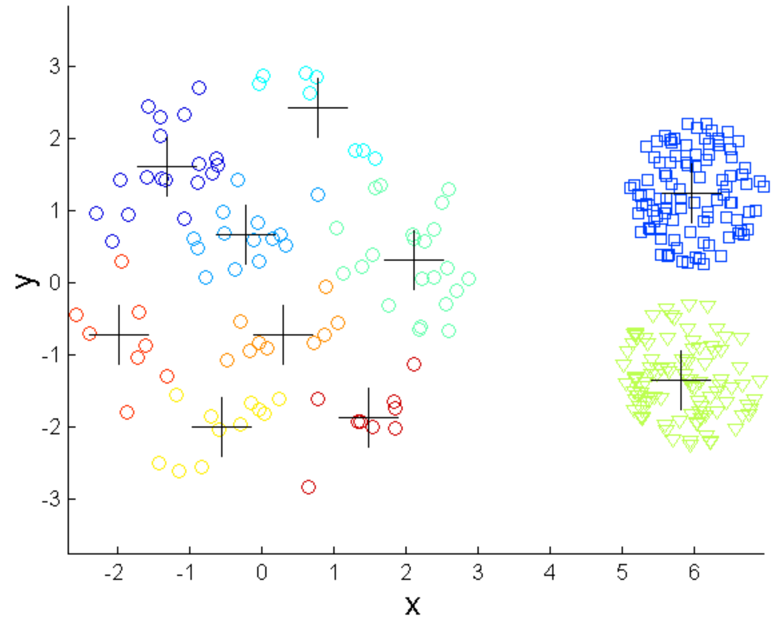
K-means Clusters

One solution is to find a large number of clusters such that each of them represents a part of a natural cluster. But these small clusters need to be put together in a post-processing step.

Overcoming K-means Limitations



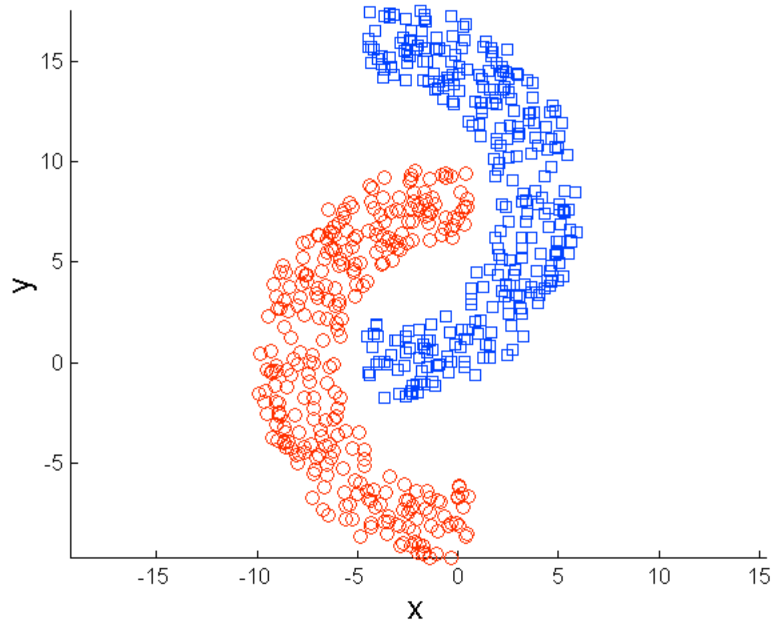
Original Points



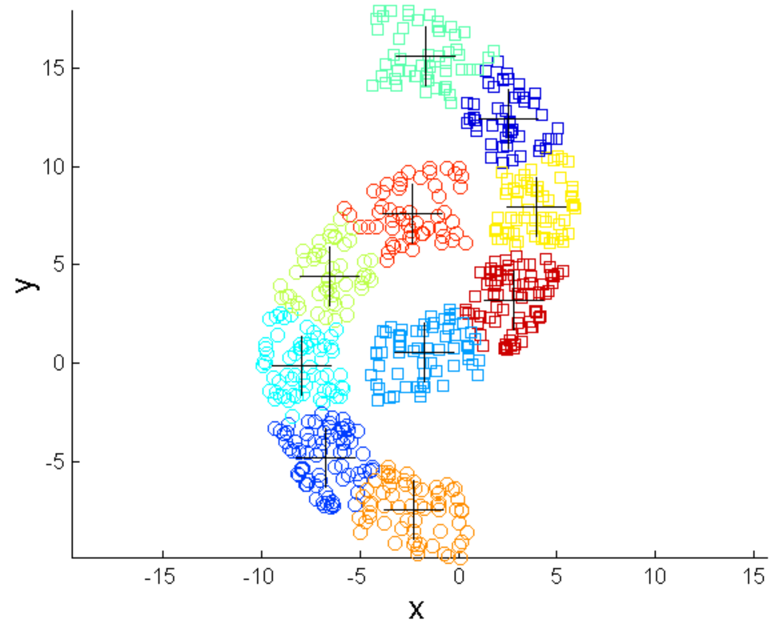
K-means Clusters

One solution is to find a large number of clusters such that each of them represents a part of a natural cluster. But these small clusters need to be put together in a post-processing step.

Overcoming K-means Limitations



Original Points

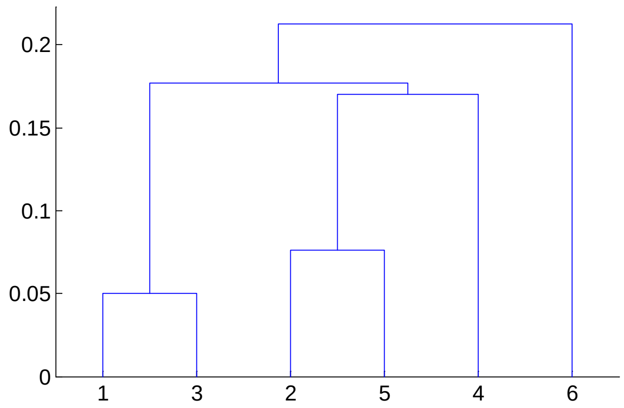


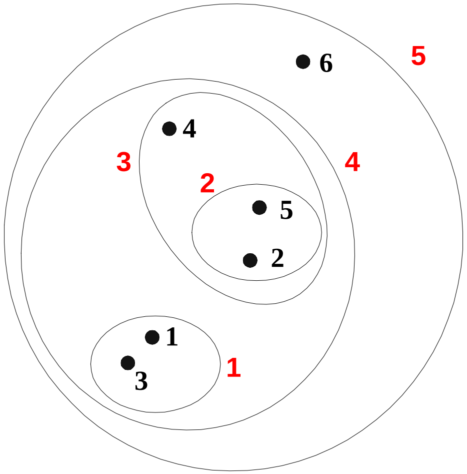
K-means Clusters

One solution is to find a large number of clusters such that each of them represents a part of a natural cluster. But these small clusters need to be put together in a post-processing step.

Hierarchical Clustering

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





Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level

- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

Hierarchical Clustering

- Two main types of hierarchical clustering
  - Agglomerative:
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - Divisive:
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains an individual 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

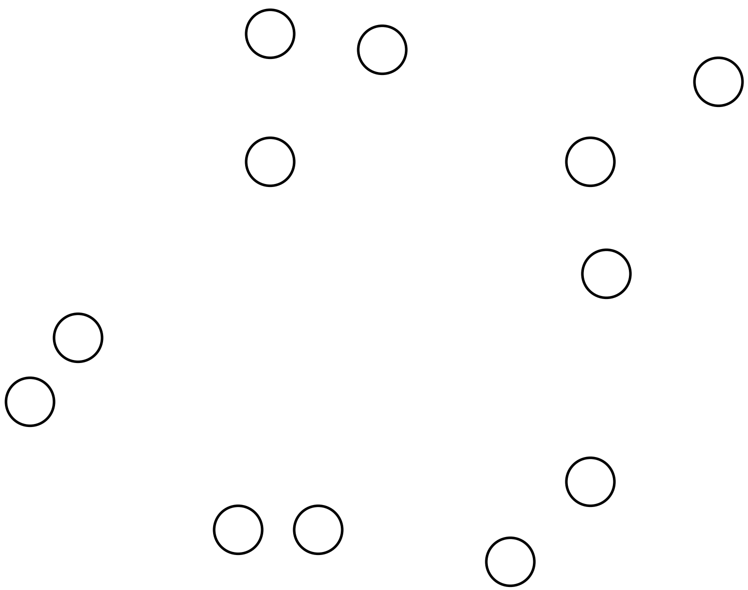
- **Key Idea: Successively merge closest clusters**

- Basic algorithm
  1. Compute the proximity matrix
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the proximity matrix
  6. **Until** only a single cluster remains

- Key operation is the computation of the proximity of two clusters
  – Different approaches to defining the distance between clusters distinguish the different algorithms

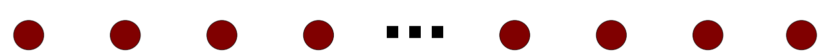
Steps 1 and 2

- Start with clusters of individual points and a proximity matrix



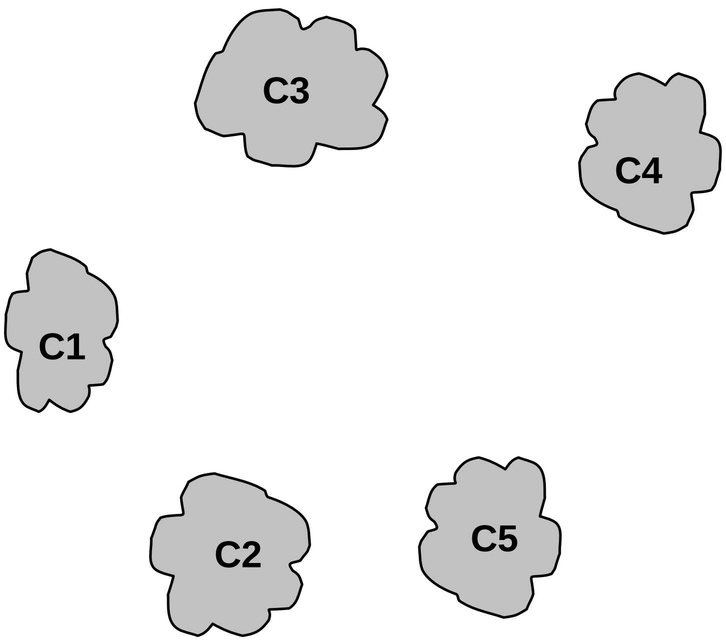
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix



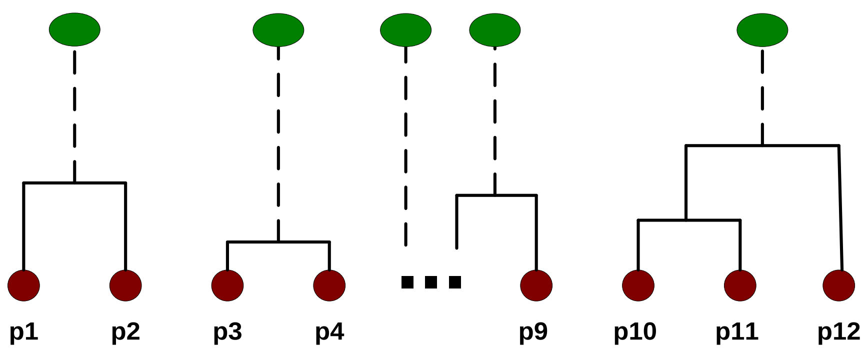
Intermediate Situation

- After some merging steps, we have some clusters



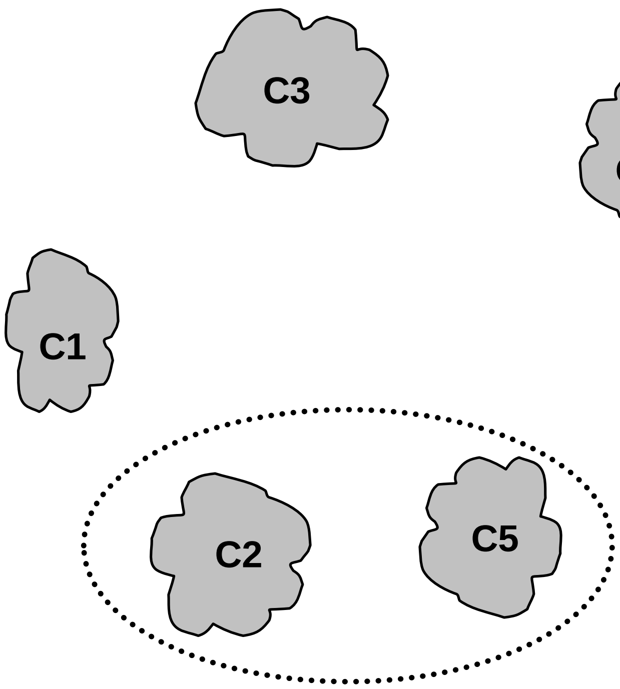
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

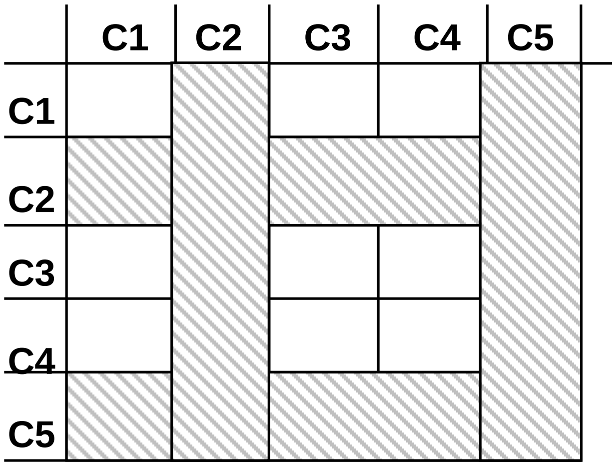
Proximity Matrix



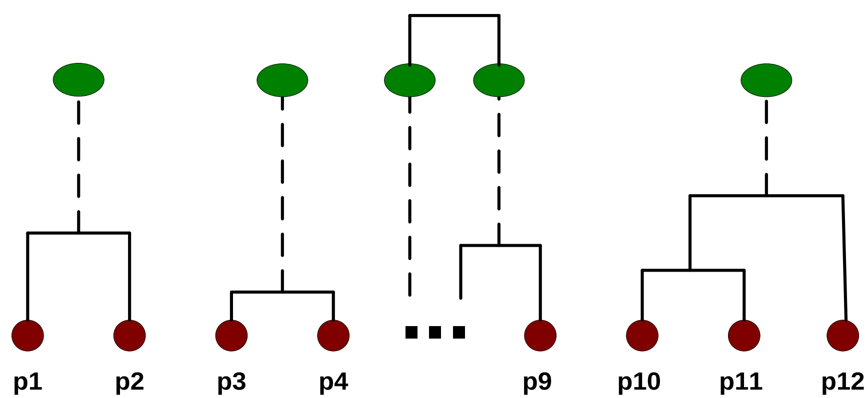
Step 4

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



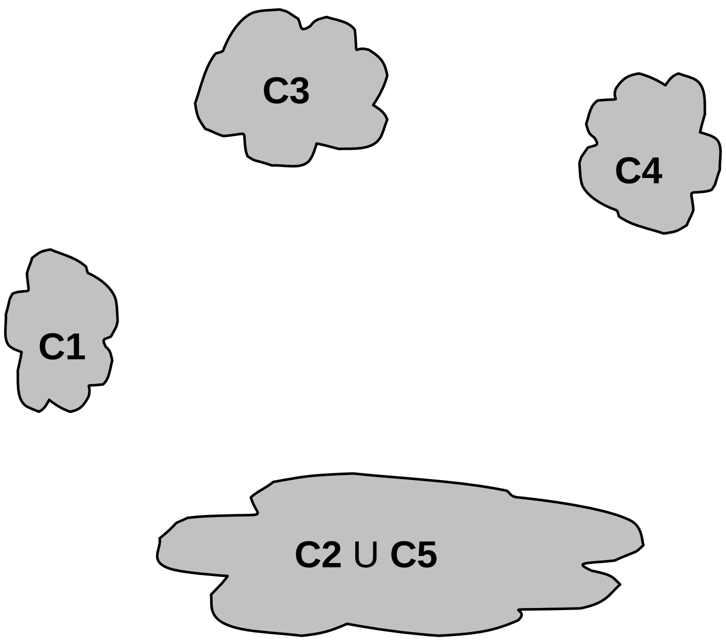


Proximity Matrix



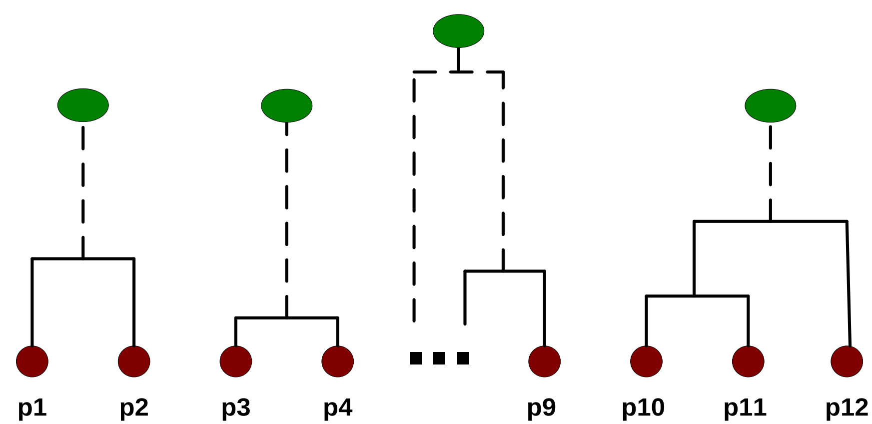
Step 5

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

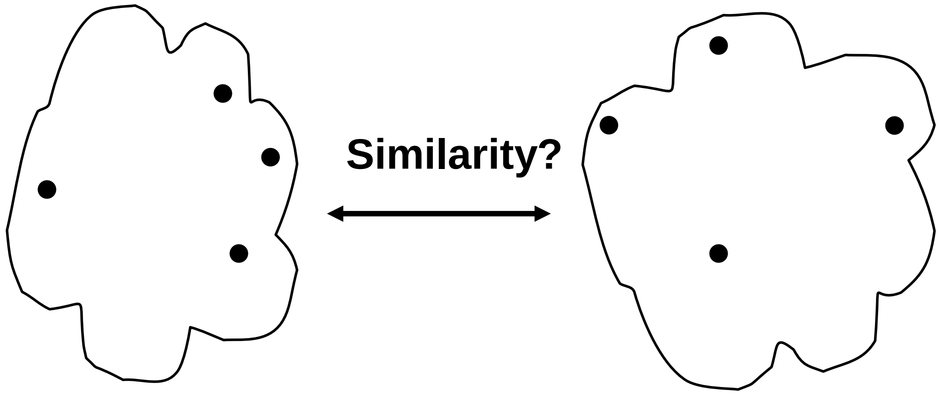


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Distance

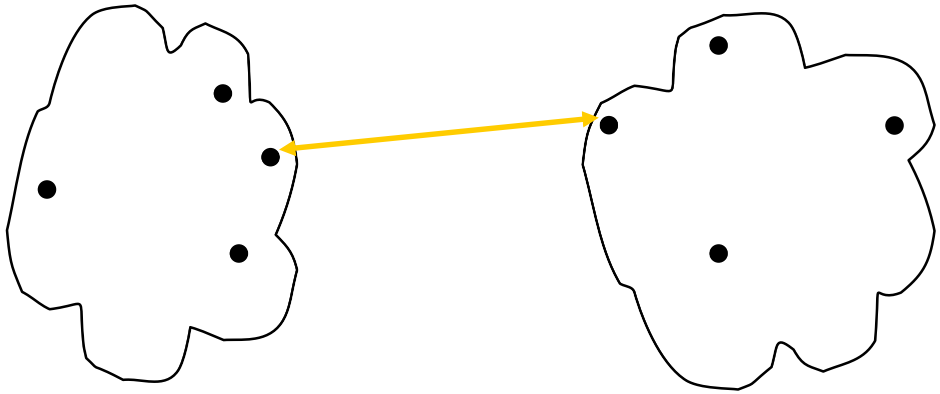


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

How to Define Inter-Cluster Similarity

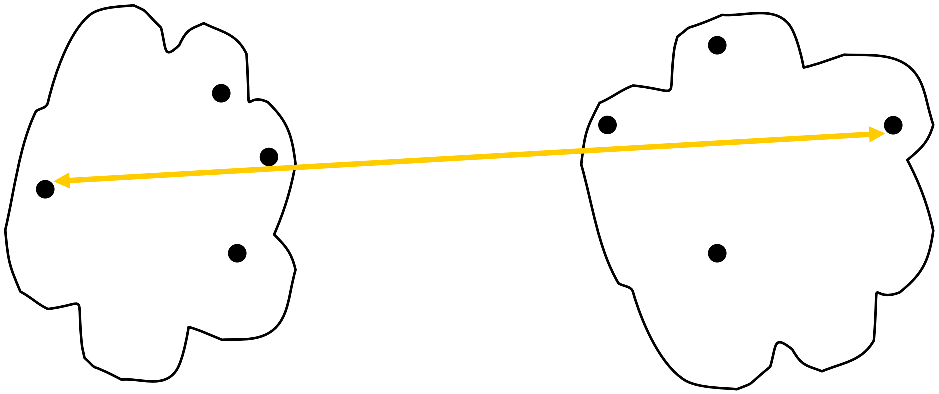


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

How to Define Inter-Cluster Similarity

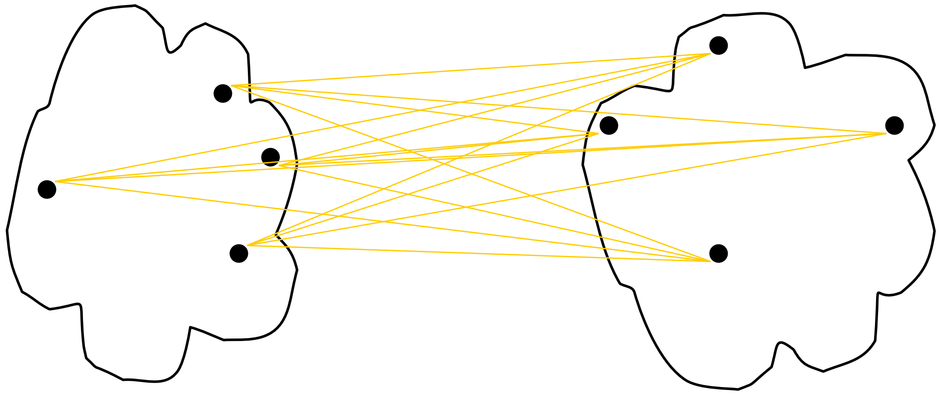


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

How to Define Inter-Cluster Similarity

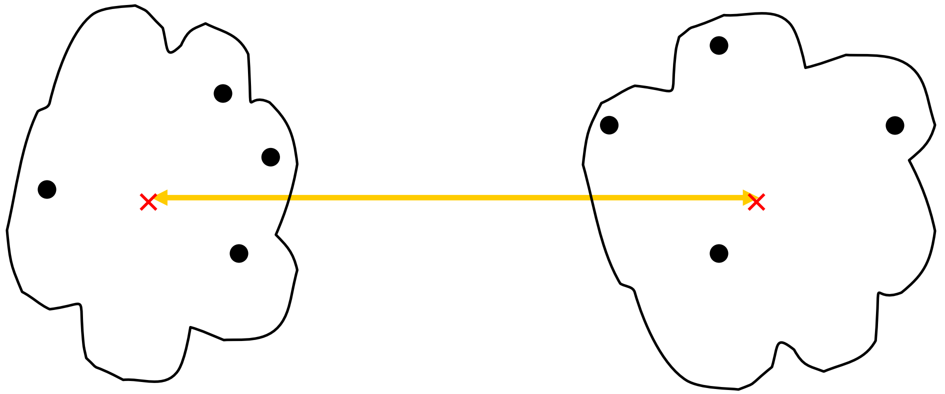


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

How to Define Inter-Cluster Similarity



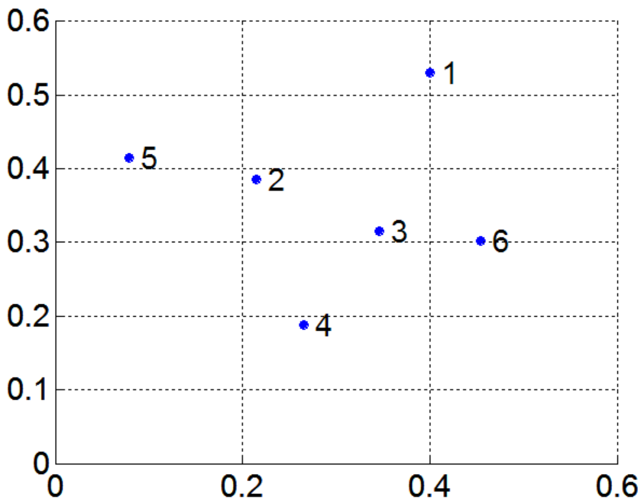
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

MIN or Single Link

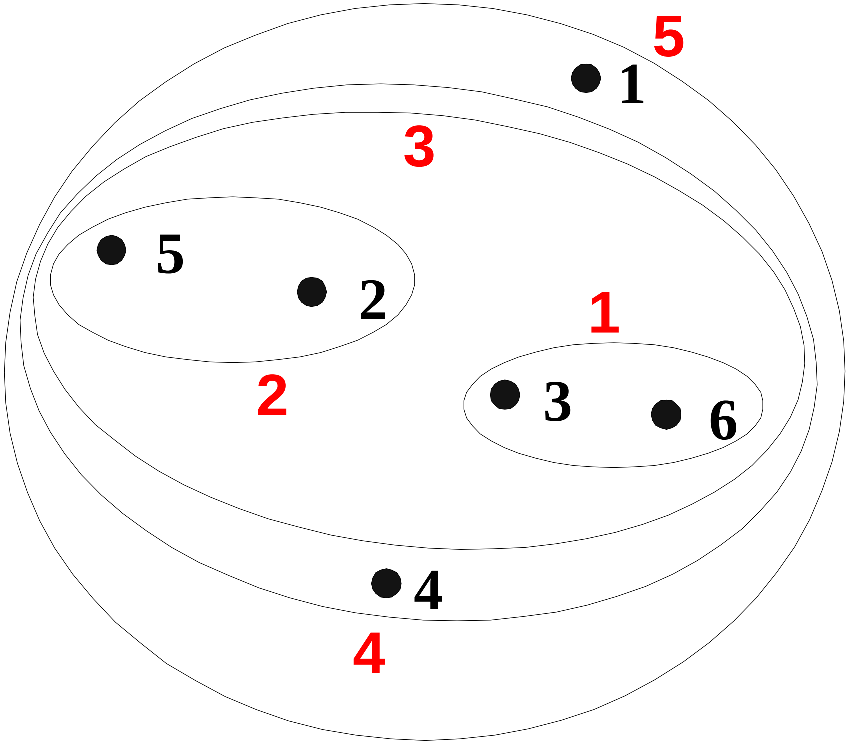
- Proximity of two clusters is based on the two closest points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph
- Example:



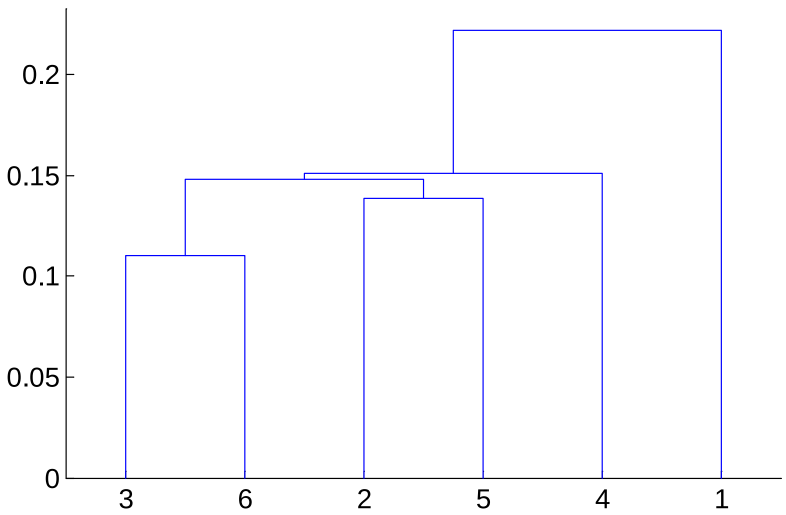
Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Hierarchical Clustering: MIN

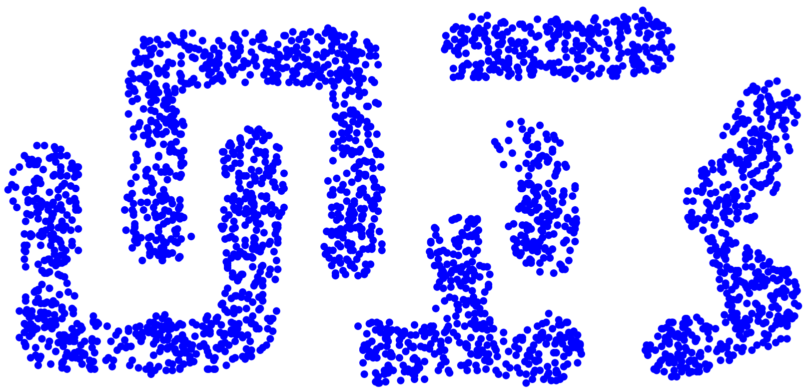


Nested Clusters

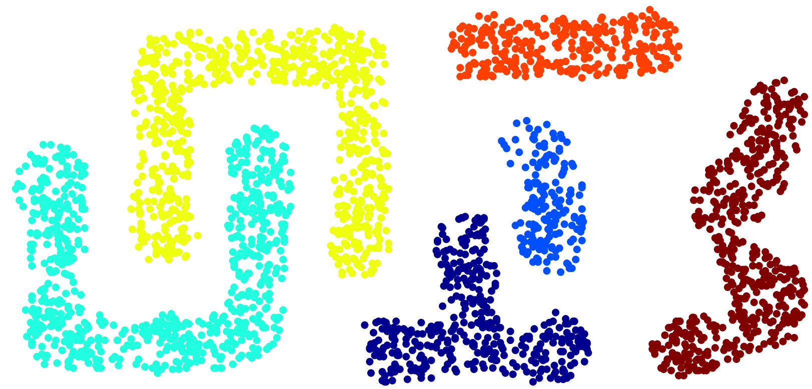


Dendrogram

Strength of MIN



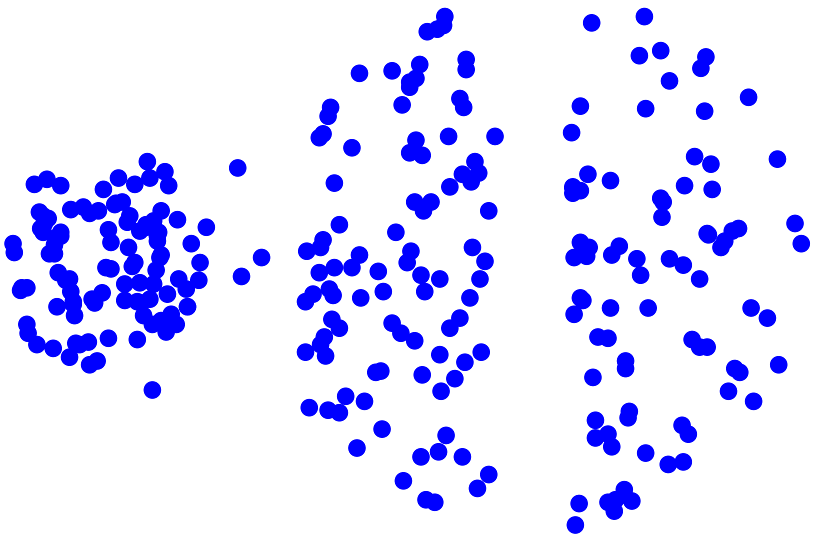
Original Points



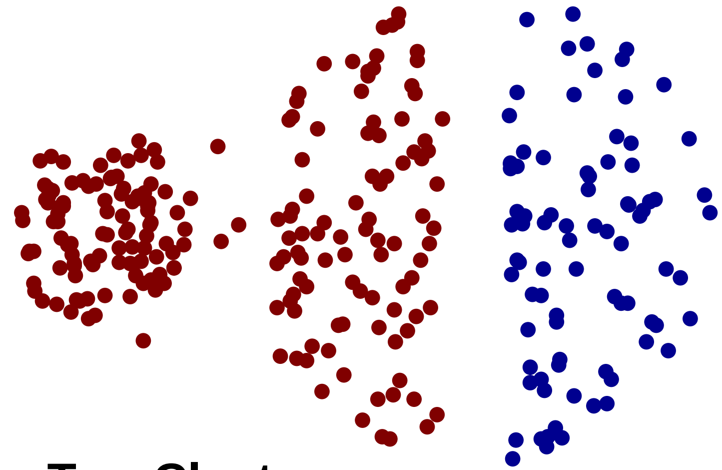
Six Clusters

• **Can handle non-elliptical shapes**

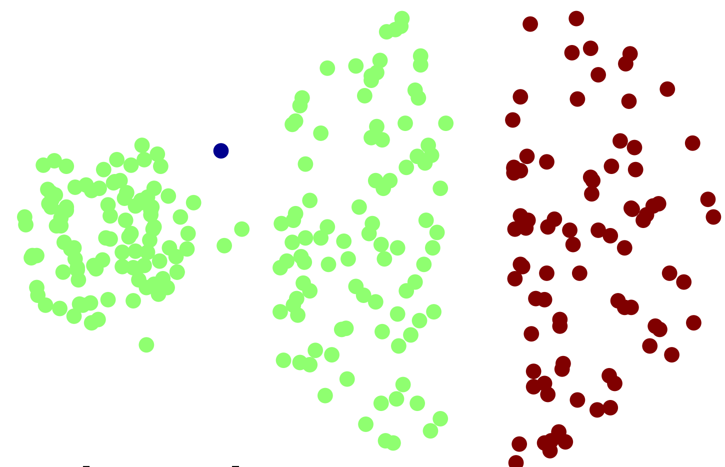
Limitations of MIN



Original Points



Two Clusters

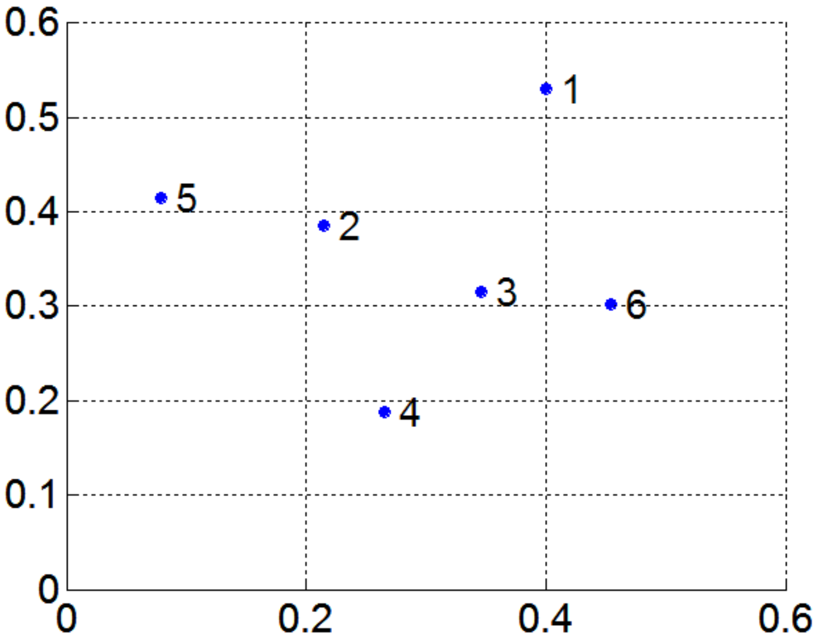


Three Clusters

- **Sensitive to noise**

MAX or Complete Linkage

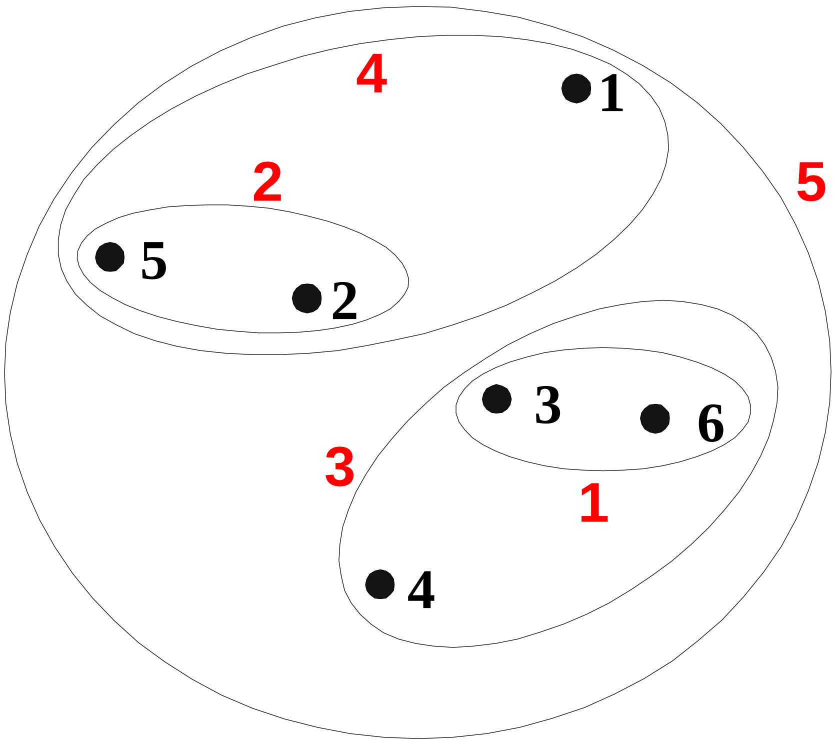
- Proximity of two clusters is based on the two most distant points in the different clusters
  - Determined by all pairs of points in the two clusters



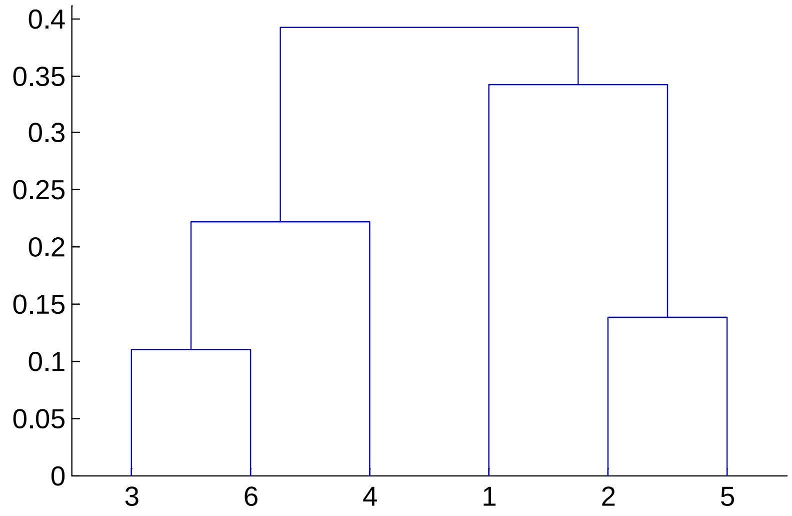
Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Hierarchical Clustering: MAX

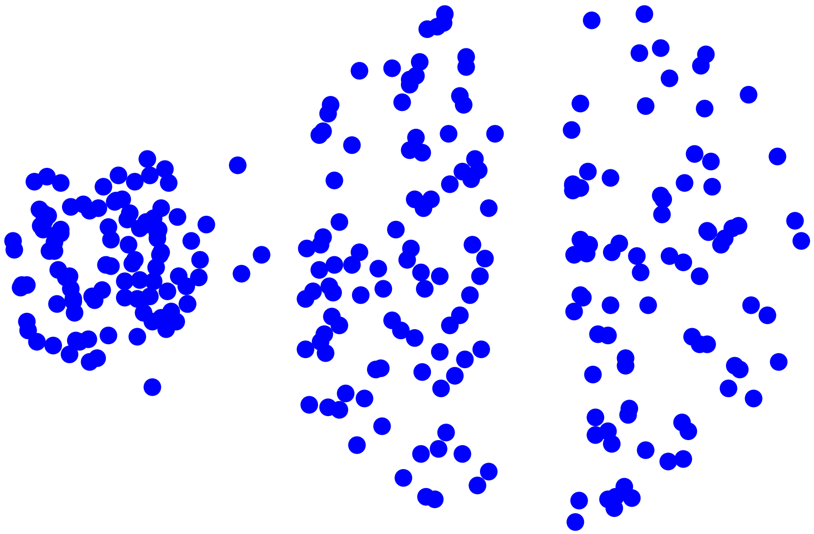


Nested Clusters

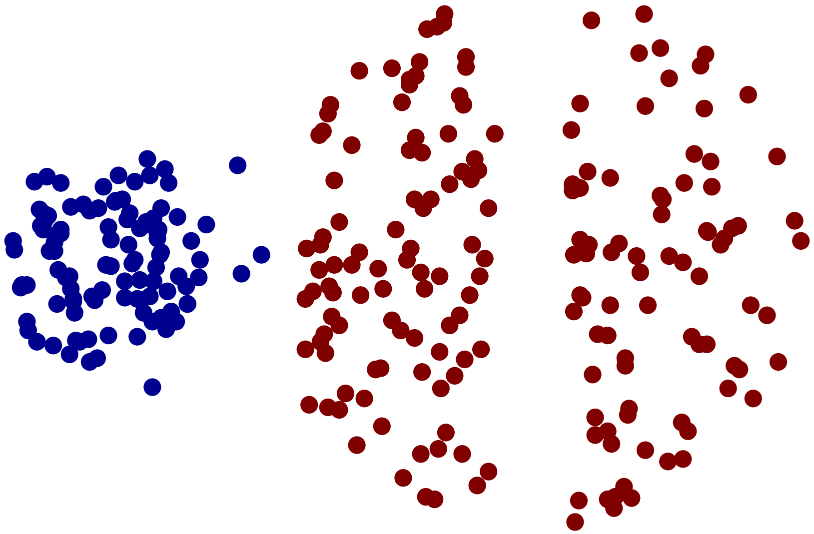


Dendrogram

Strength of MAX



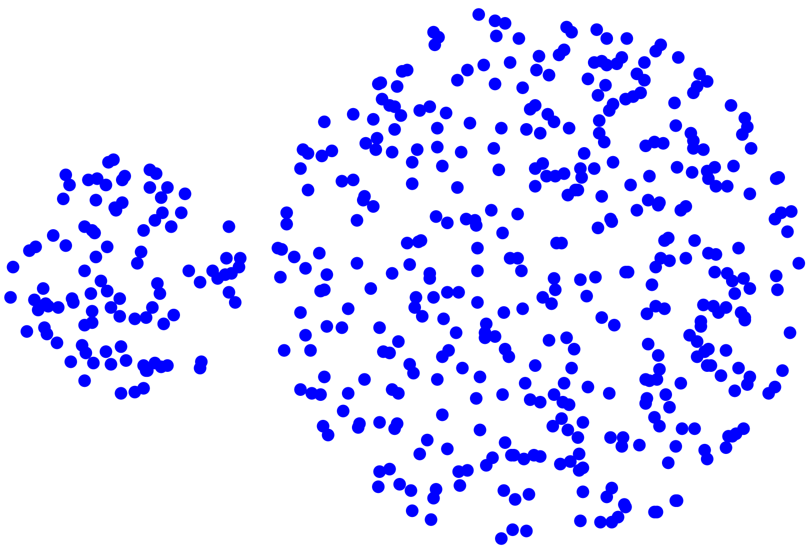
Original Points



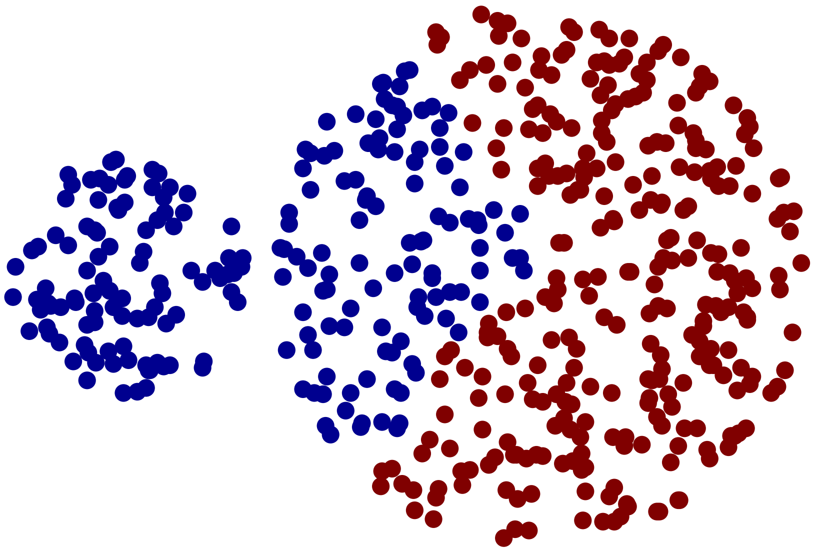
Two Clusters

- Less susceptible to noise

Limitations of MAX



Original Points



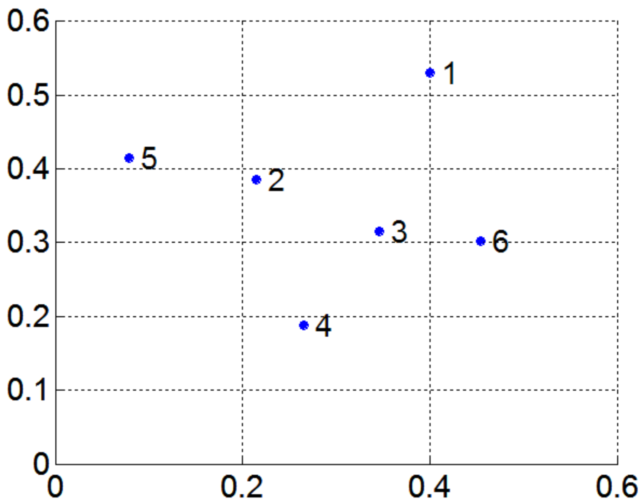
Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

Group Average

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

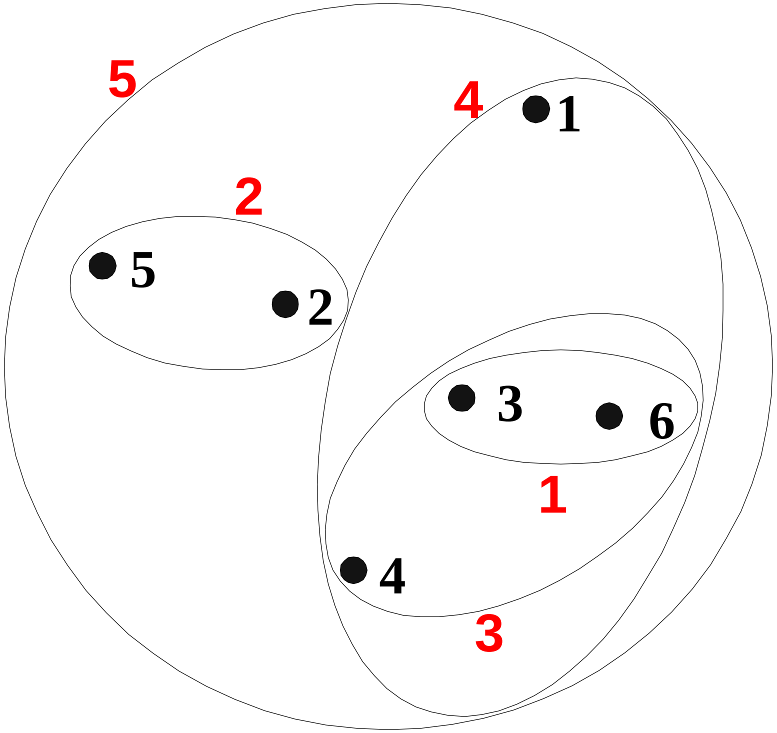
$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\displaystyle\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| \times |\text{Cluster}_j|}$$

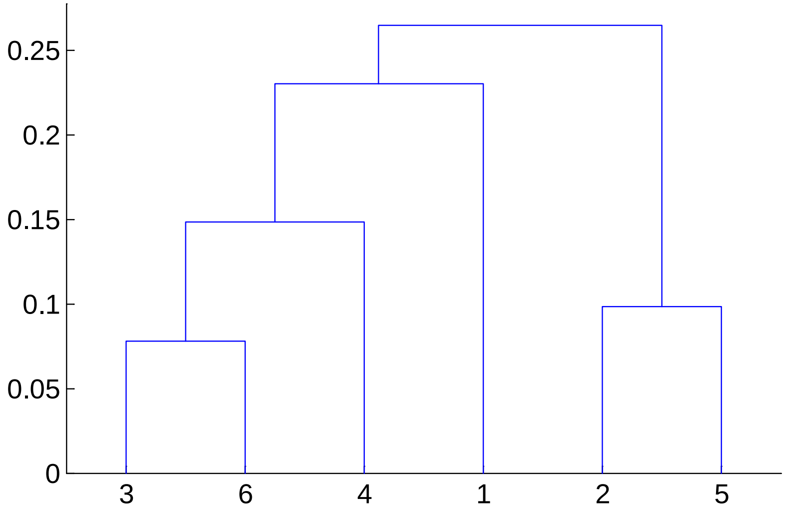


Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Hierarchical Clustering: Group Average





Nested Clusters **Dendrogram**

Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link

- Strengths
  – Less susceptible to noise

- Limitations
  – Biased towards globular clusters

Cluster Similarity: Ward's Method

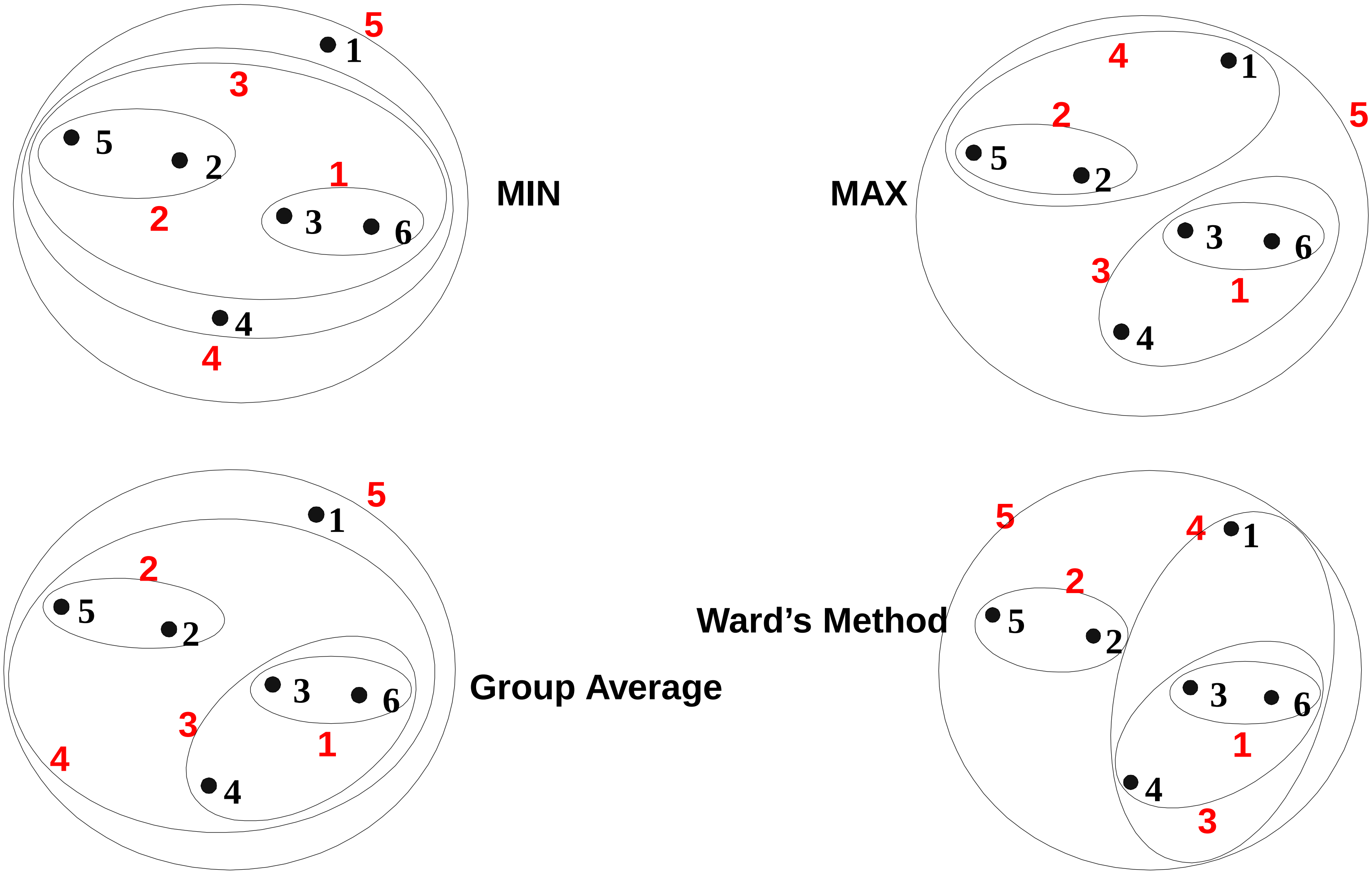
- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared

- Less susceptible to noise

- Biased towards globular clusters

- Hierarchical analogue of K-means
  - Can be used to initialize K-means

Hierarchical Clustering: Comparison



Hierarchical Clustering: Time and Space requirements

- $O(N^2)$ space since it uses the proximity matrix.
  - N is the number of points.

- $O(N^3)$ time in many cases
  - There are N steps and at each step the size, N^2, proximity matrix must be updated and searched
  - Complexity can be reduced to $O(N^2 \log(N))$ time with some cleverness

Hierarchical Clustering: Problems and Limitations

- Once a decision is made to combine two clusters, it cannot be undone

- No global objective function is directly minimized

- Different schemes have problems with one or more of the following:
  - Sensitivity to noise
  - Difficulty handling clusters of different sizes and non-globular shapes
  - Breaking large clusters

Density Based Clustering

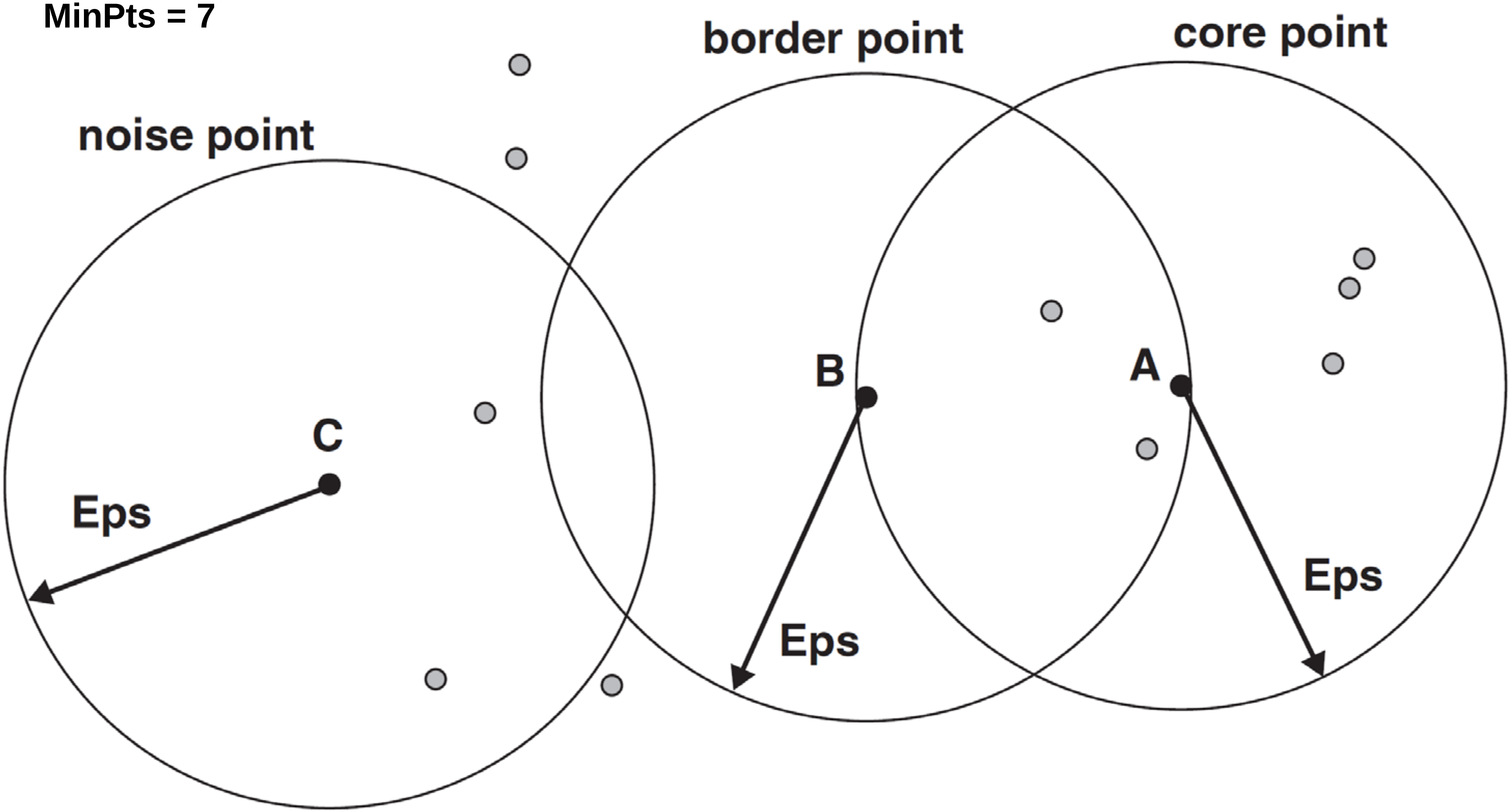
- Clusters are regions of high density that are separated from one another by regions on low density.

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 at least a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
    - Counts the point itself
  - A border point is not a core point, 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

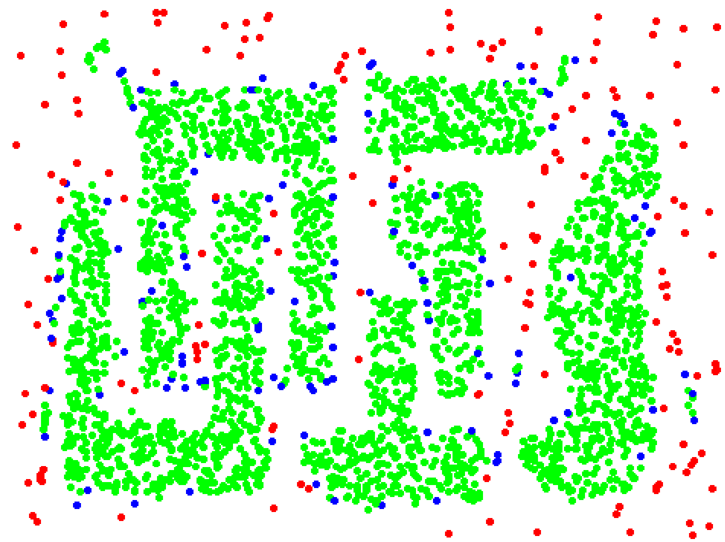
DBSCAN: Core, Border, and Noise Points

MinPts = 7



DBSCAN: Core, Border and Noise Points

Original Points



Point types: core, border and noise

Eps = 10, MinPts = 4

DBSCAN Algorithm

- Form clusters using core points, and assign border points to one of its neighboring clusters

1: Label all points as core, border, or noise points.

2: Eliminate noise points.

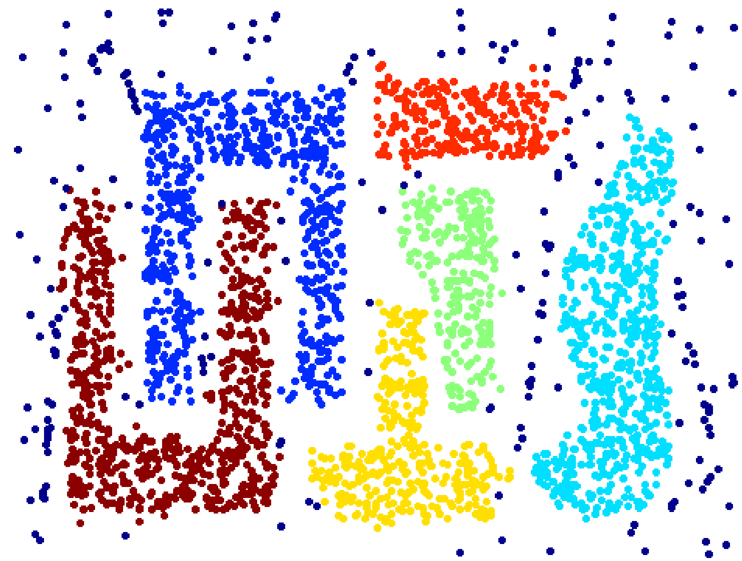
3: Put an edge between all core points within a distance *Eps* of each other.

4: Make each group of connected core points into a separate cluster.

5: Assign each border point to one of the clusters of its associated core points

When DBSCAN Works Well

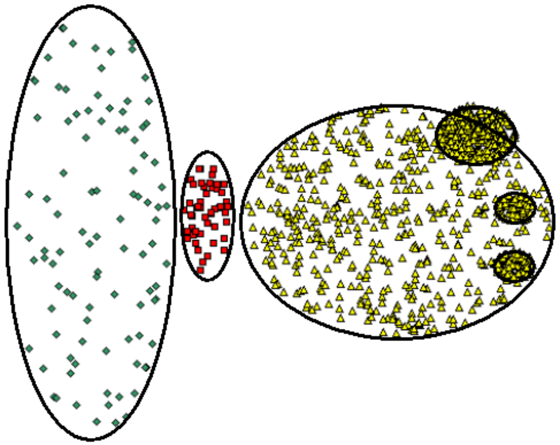
Original Points



Clusters (dark blue points indicate noise)

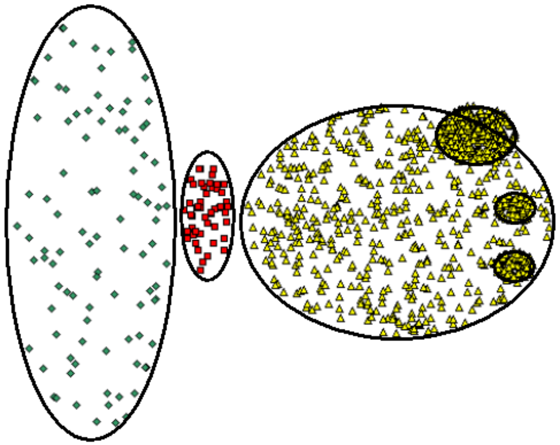
- **Can handle clusters of different shapes and sizes**
- **Resistant to noise**

When DBSCAN Does NOT Work Well

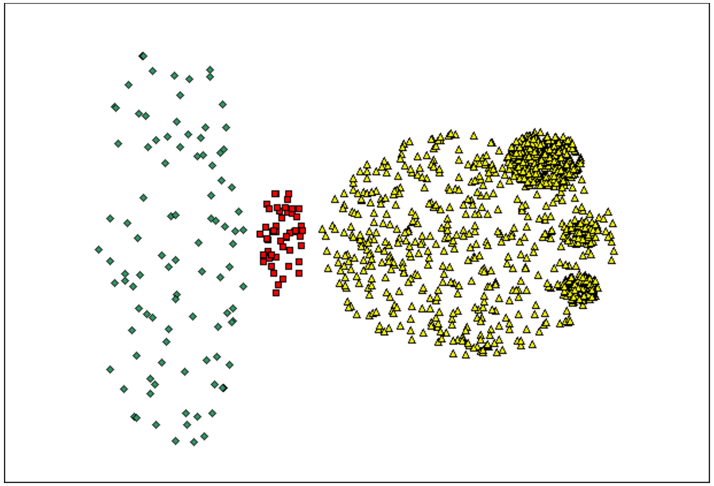


Original Points

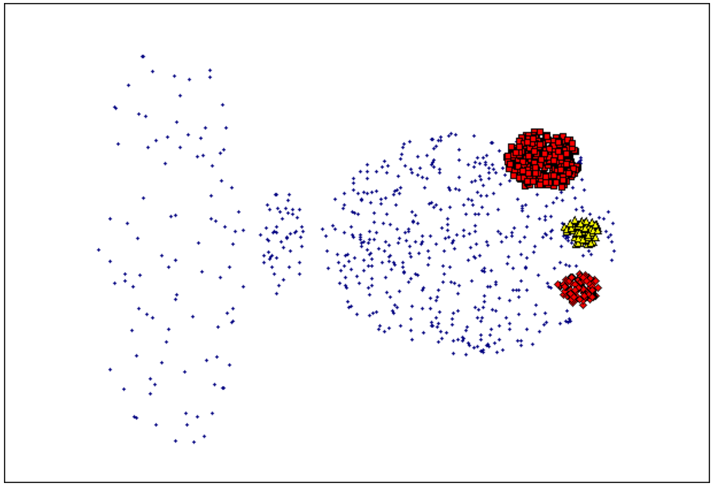
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.92).

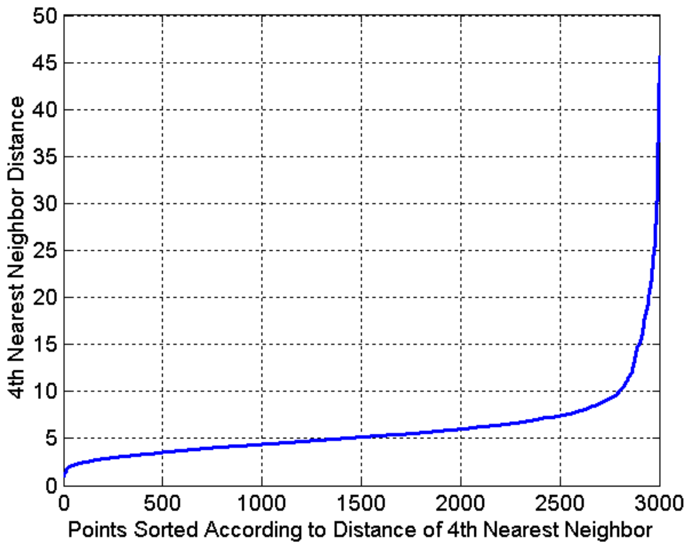


(MinPts=4, Eps=9.75)

- **Varying densities**
- **High-dimensional data**

DBSCAN: Determining EPS and MinPts

- Idea is that for points in a cluster, their k^{th} nearest neighbors are at close 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



Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is
  – Accuracy, precision, recall

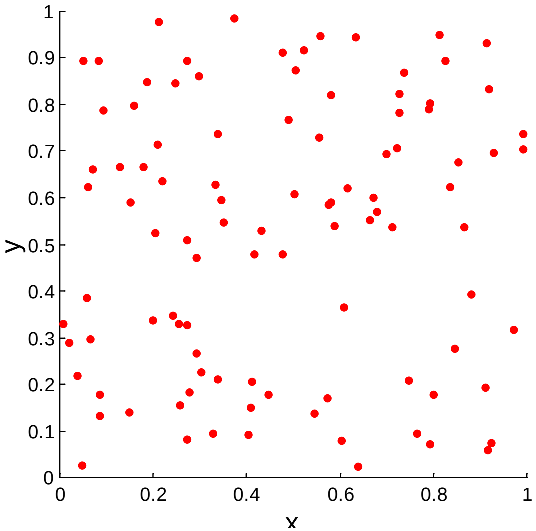
- For cluster analysis, the analogous question is how to evaluate the “goodness” of the resulting clusters?

- But “clusters are in the eye of the beholder”!
  – In practice the clusters we find are defined by the clustering algorithm

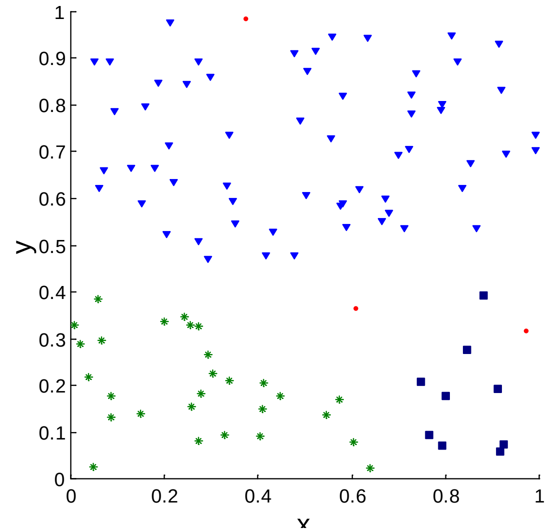
- Then why do we want to evaluate them?
  – To avoid finding patterns in noise
  – To compare clustering algorithms
  – To compare two sets of clusters
  – To compare two clusters

Clusters found in Random Data

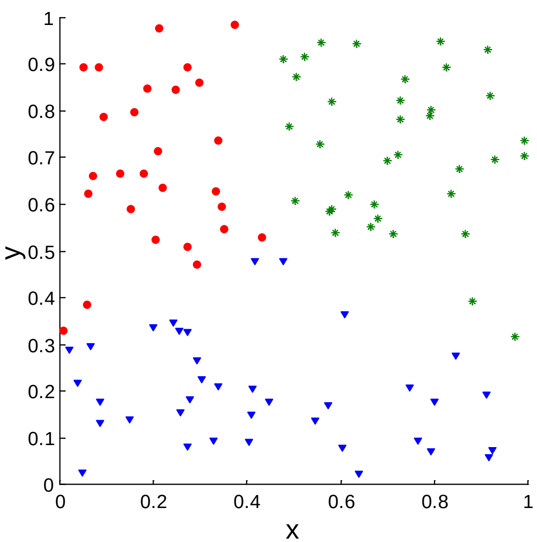
Random Points



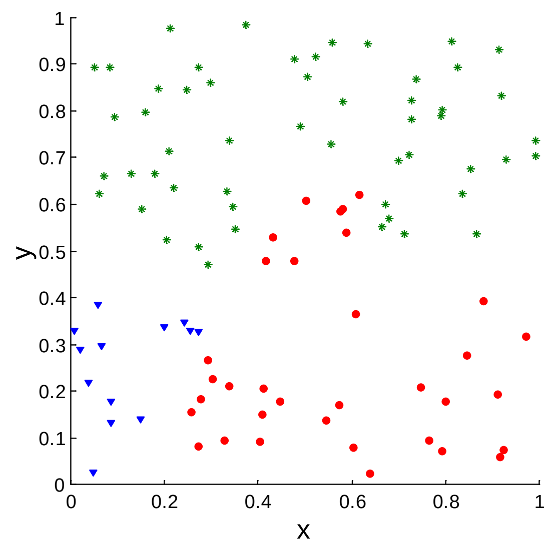
DBSCAN



K-means



Complete Link



Measures of Cluster Validity

- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following two types.
  - Supervised: Used to measure the extent to which cluster labels match externally supplied class labels.
    - Entropy
    - Often called *external indices* because they use information external to the data
  - Unsupervised: Used to measure the goodness of a clustering structure *without* respect to external information.
    - Sum of Squared Error (SSE)
    - Often called *internal indices* because they only use information in the data

- You can use supervised or unsupervised measures to compare clusters or clusterings

Unsupervised Measures: Cohesion and Separation

- Cluster Cohesion: Measures how closely related are objects in a cluster
  - Example: SSE
- Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters
- Example: Squared Error
  - Cohesion is measured by the within cluster sum of squares (SSE)

$$SSE = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

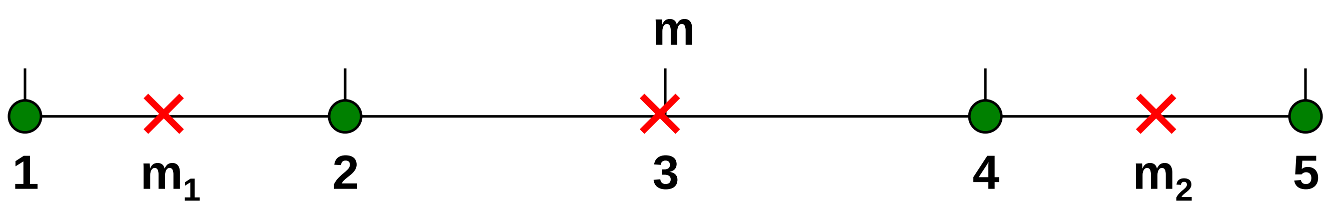
  - Separation is measured by the between cluster sum of squares

$$SSB = \sum_i |C_i| (m - m_i)^2$$

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

Unsupervised Measures: Cohesion and Separation

- Example: SSE
  - SSB + SSE = constant



K=1 cluster:

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

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

$$Total = 10 + 0 = 10$$

K=2 clusters:

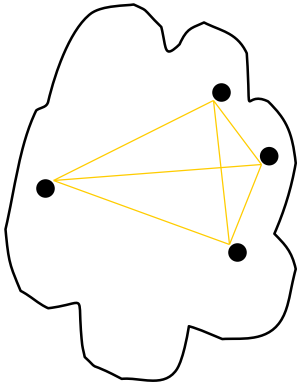
$$SSE = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$

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

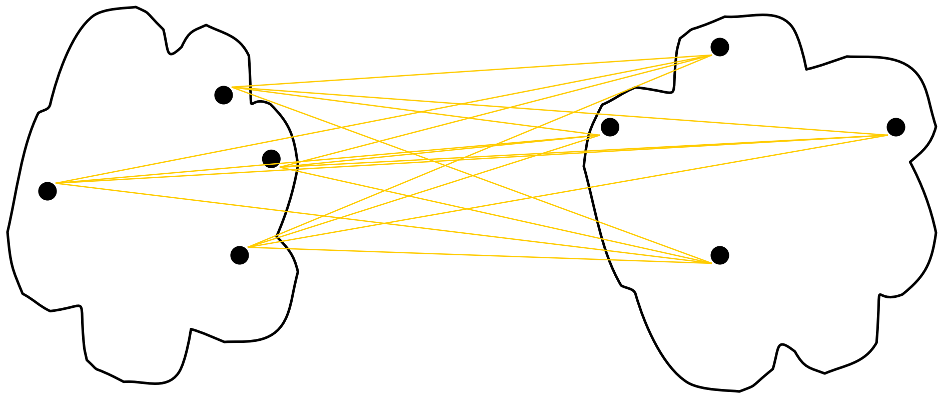
$$Total = 1 + 9 = 10$$

Unsupervised Measures: Cohesion and Separation

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



cohesion

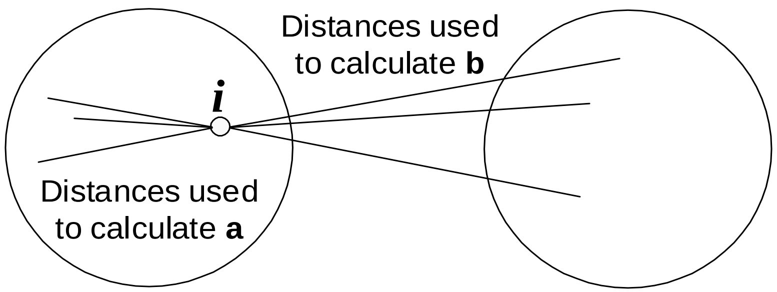


separation

Unsupervised Measures: Silhouette Coefficient

- Silhouette coefficient combines ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- For an individual point, ***i***
  - Calculate ***a*** = average distance of ***i*** to the points in its cluster
  - Calculate ***b*** = min (average distance of ***i*** to points in another cluster)
  - The silhouette coefficient for a point is then given by

    s = (b – a) / max(a,b)



  - Value can vary between -1 and 1
  - Typically ranges between 0 and 1.
  - The closer to 1 the better.
- Can calculate the average silhouette coefficient for a cluster or a clustering