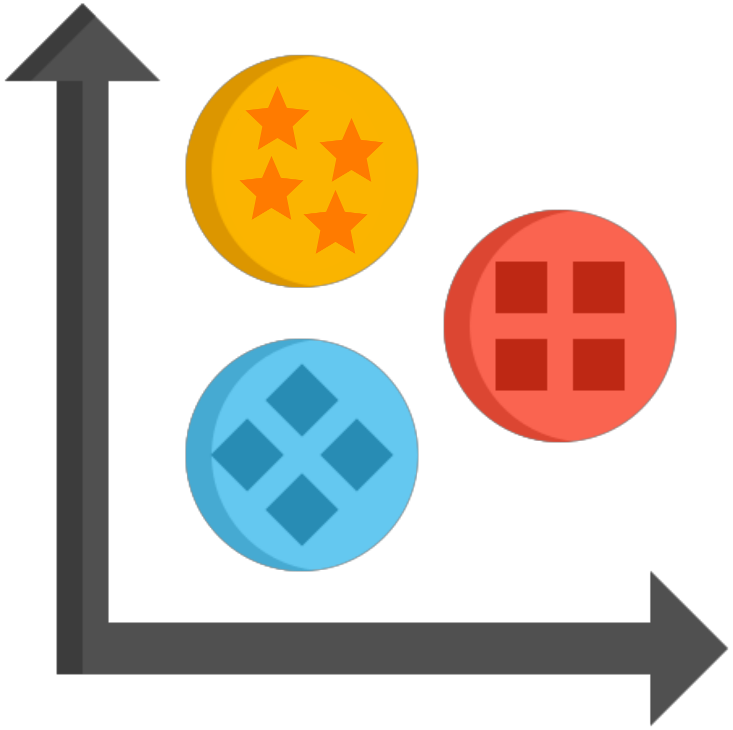
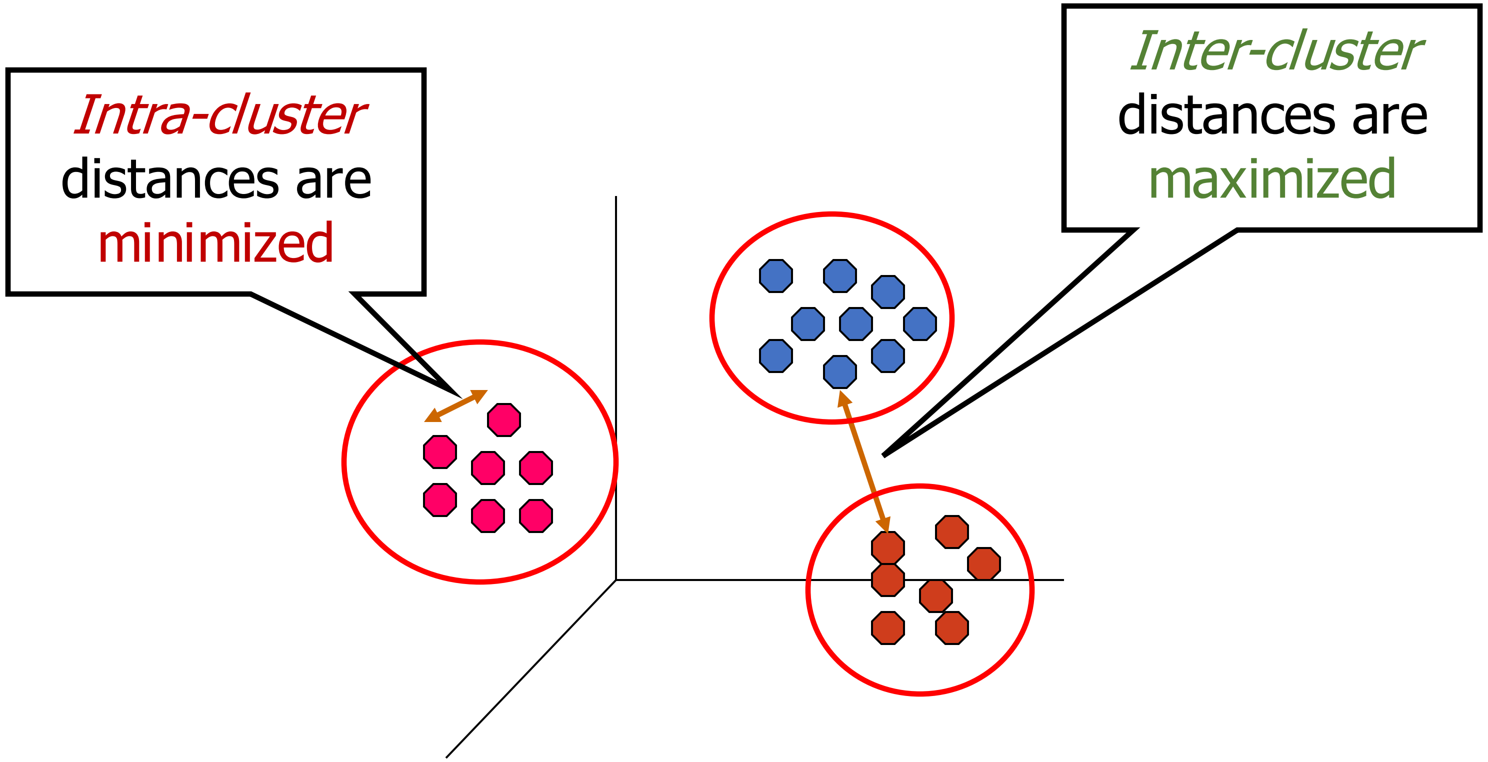
UNIVERSITÀ DEL SALENTO

Clustering



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**



Notion of a Cluster can be Ambiguous

How many clusters?

Six Clusters

Two Clusters

Four Clusters

Types of Clusterings

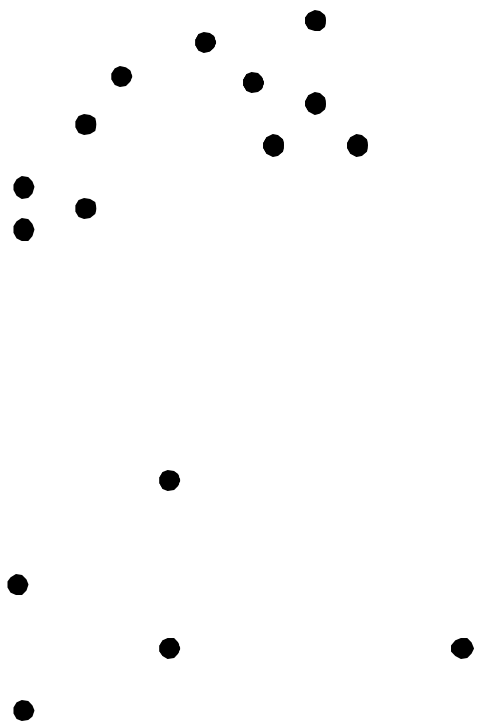
A clustering is a set of clusters

Important distinction between hierarchical and partitional sets of clusters

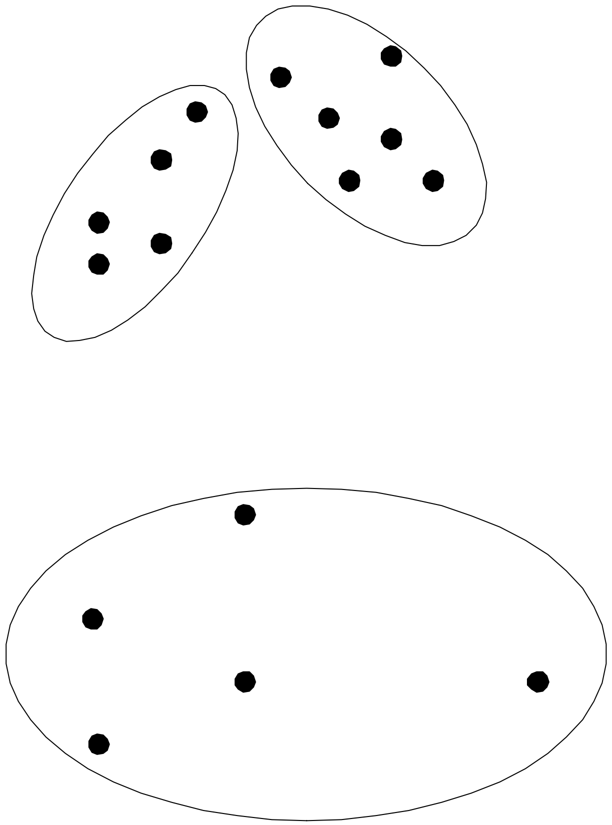
- **Partitional Clustering**: A division of data objects into non-overlapping subsets (clusters)

- **Hierarchical clustering**: A set of nested clusters organized as a hierarchical tree

Partitional Clustering

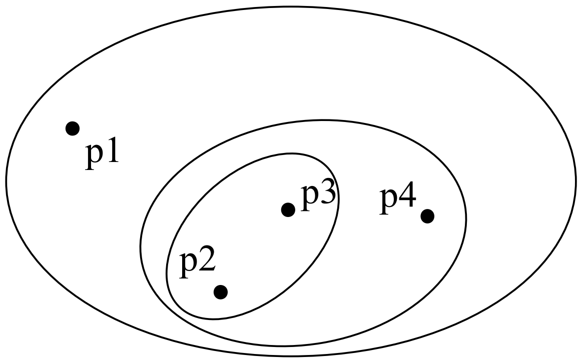


Original Points

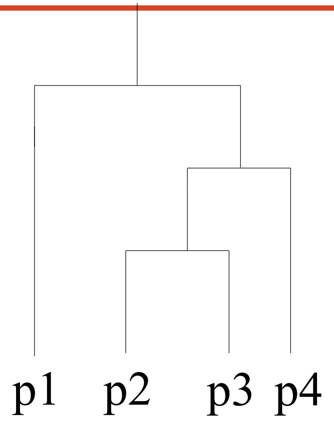


A Partitional Clustering

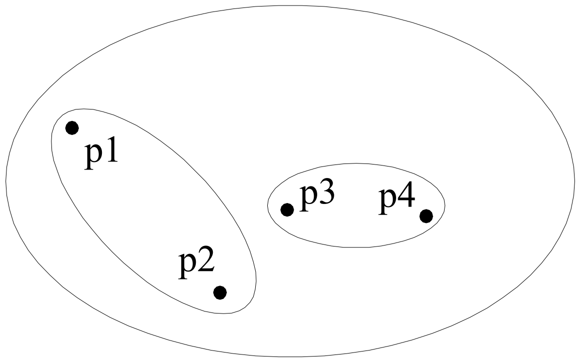
Hierarchical Clustering



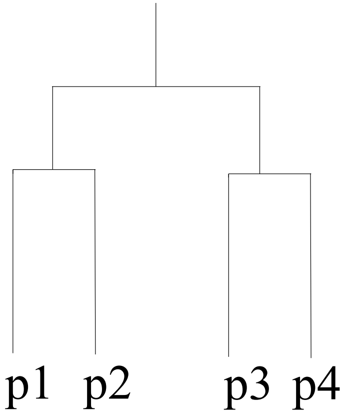
Traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Hierarchical Clustering



Non-traditional Dendrogram

Other Distinctions Between Sets of Clusters

Exclusive versus **non-exclusive** versus **Fuzzy**

- **Overlapping** or 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

a) Well-separated clusters

b) Prototype-based clusters

c) Contiguity-based clusters

d) Density-based clusters

e) Conceptual clusters

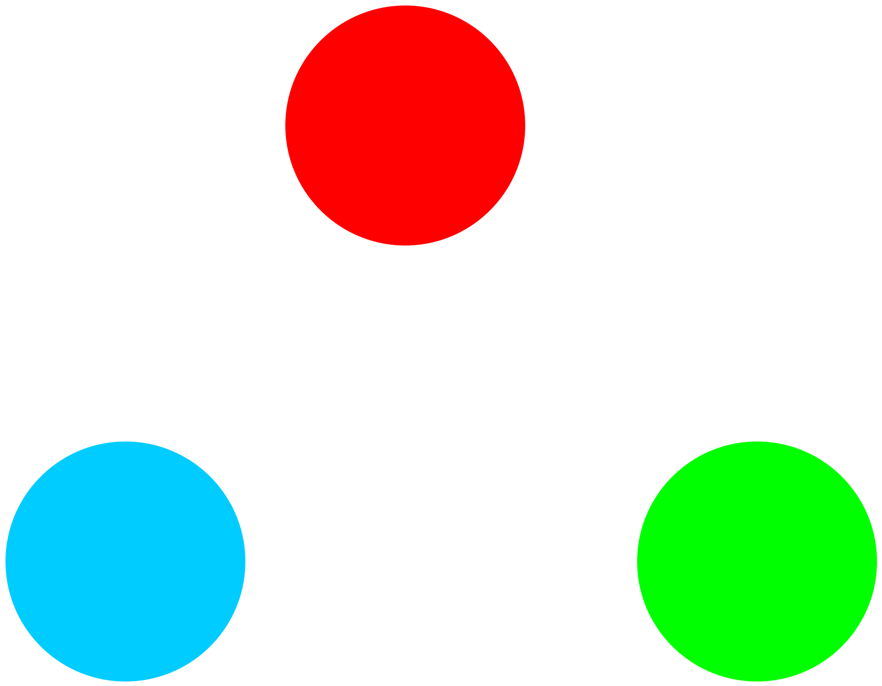
Types of Clusters: Well-Separated

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.

Clusters that are quite far from each other.

The distance between any two points in different groups is larger than the distance between any two points within a group.

Well-separated clusters do not need to be **globular**, but can have any shape.

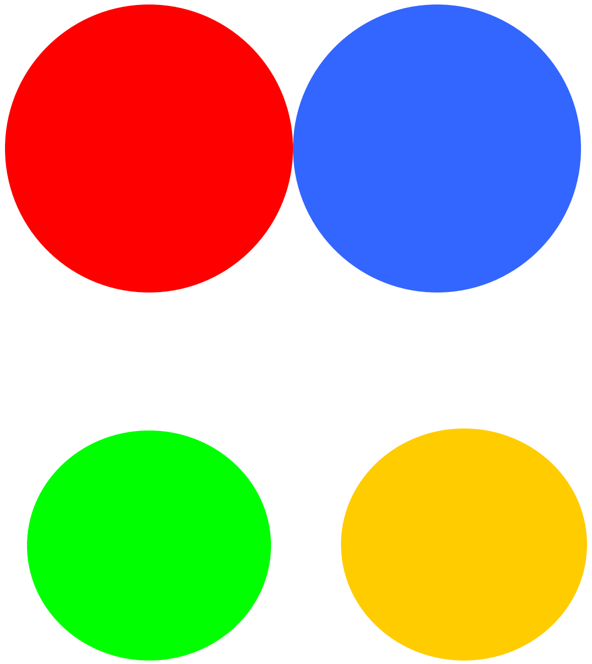


3 well-separated clusters

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

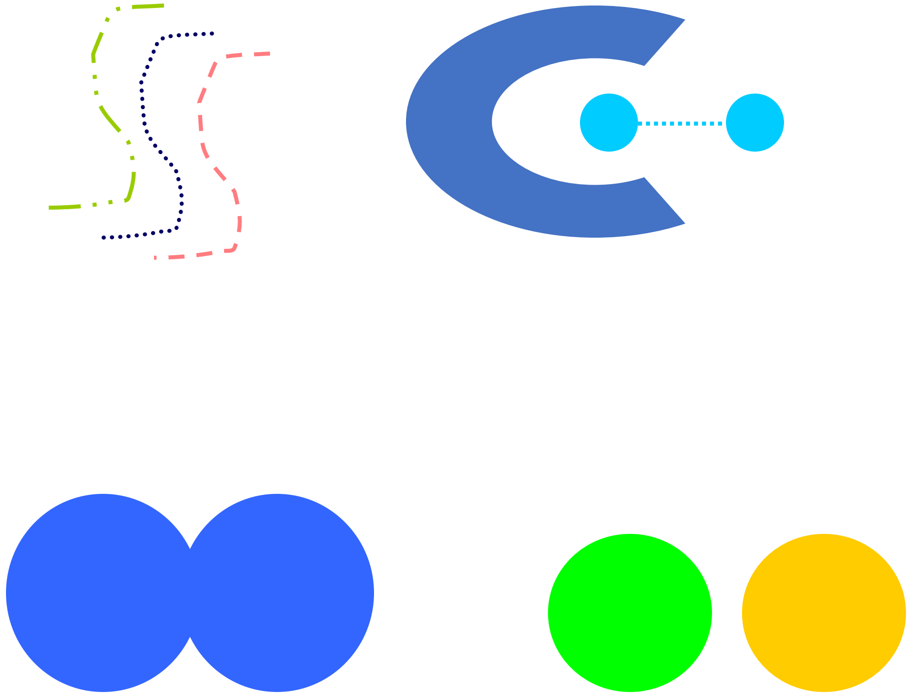
The center of a cluster is often a **centroid**, the average of all the points in the cluster, or a **medoid**, the most "representative" point of a cluster



4 center-based clusters

Types of Clusters: Contiguity-Based

Contiguous Cluster (Nearest neighbor or **Graph based**): 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.

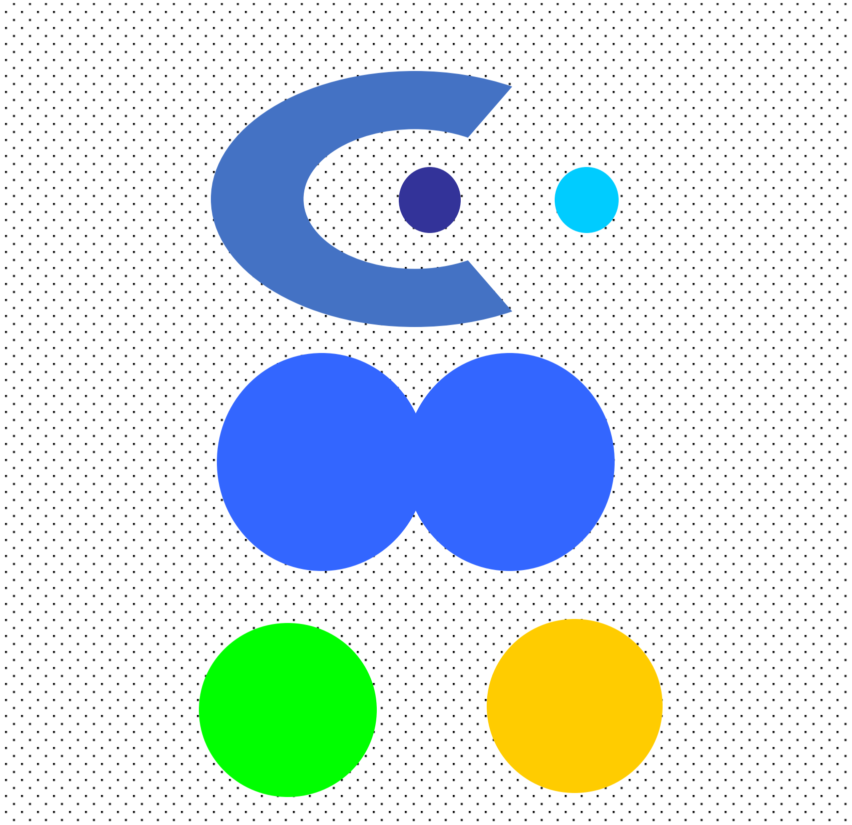


8 contiguous clusters

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.

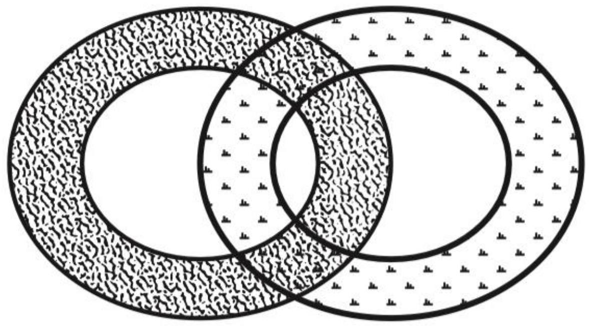
Used when the clusters are irregular or intertwined, and when noise and outliers are present.



6 density-based clusters

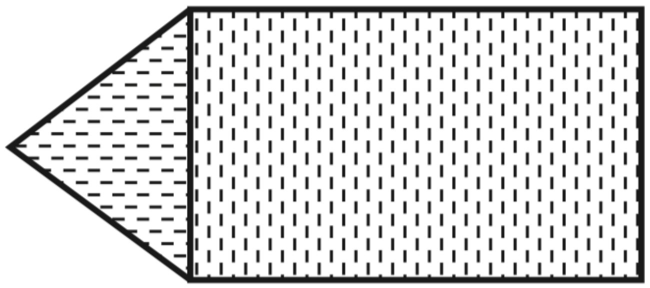
Types of Clusters: Conceptual cluster

Conceptual cluster: More generally, we can define a cluster as a set of objects that share some property.



This definition encompasses all the previous definitions of a cluster

New types of clusters --> clusters shown in figure: a triangular area is adjacent to a rectangular one



A clustering algorithm would need a very specific concept of a cluster to successfully detect these clusters.

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 (autocorrelation)

Noise and Outliers

Clustering Algorithms

K-means

Hierarchical clustering

Density-based clustering

K-means Clustering

Partitional clustering approach

Number of clusters, K, must be specified

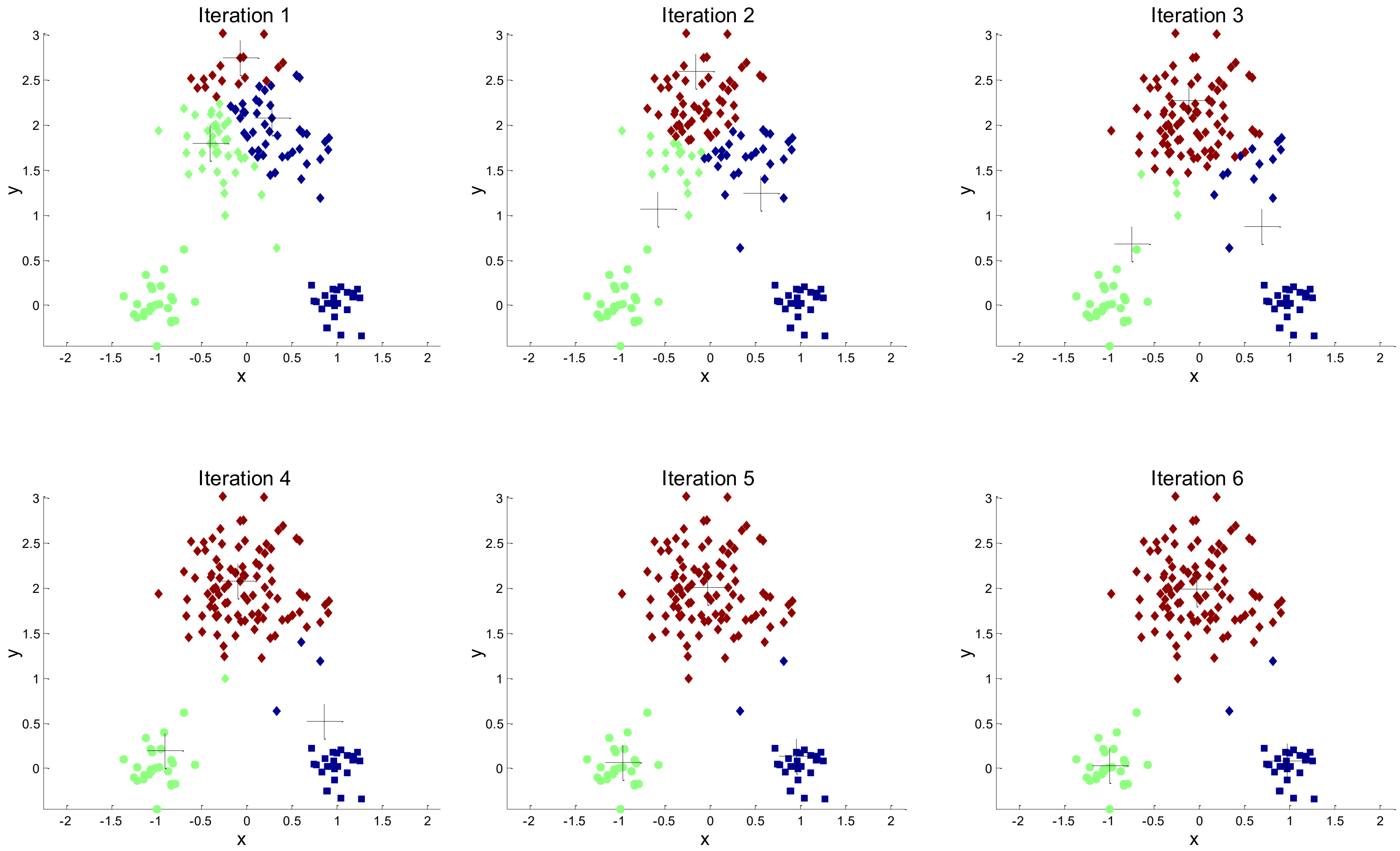
Each cluster is associated with a centroid (center point)

Each point is assigned to the cluster with the closest centroid

The basic algorithm is very simple

```
Select K points as the initial centroids.
repeat
    Form K clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change
```

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

K-means will converge for common proximity measures with appropriately defined centroid

Most of the convergence happens in the first few iterations.

- Often the stopping condition is changed to 'Until relatively few points change clusters'

K-means Objective Function

A common **objective function** (used with Euclidean distance measure) is

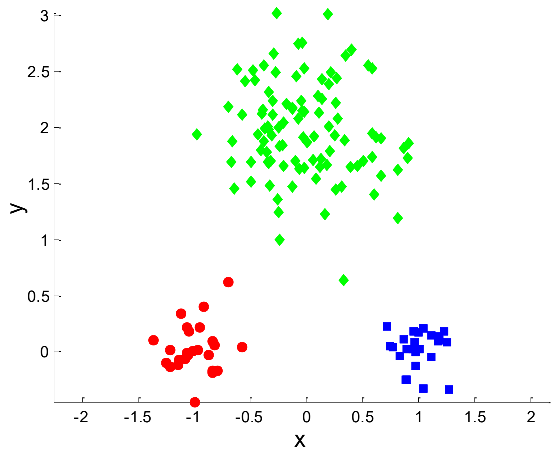
Sum of Squared Error **(SSE)**

- For each point, the error is the distance to the nearest cluster center
- To get SSE, we square these errors and sum them.

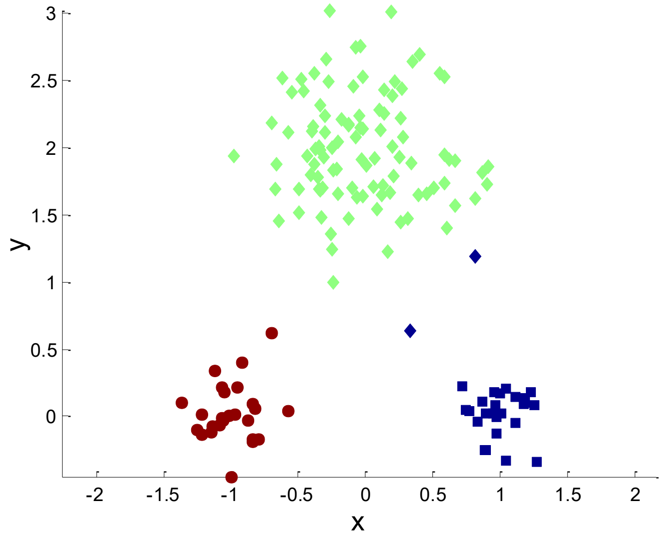
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

- *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.

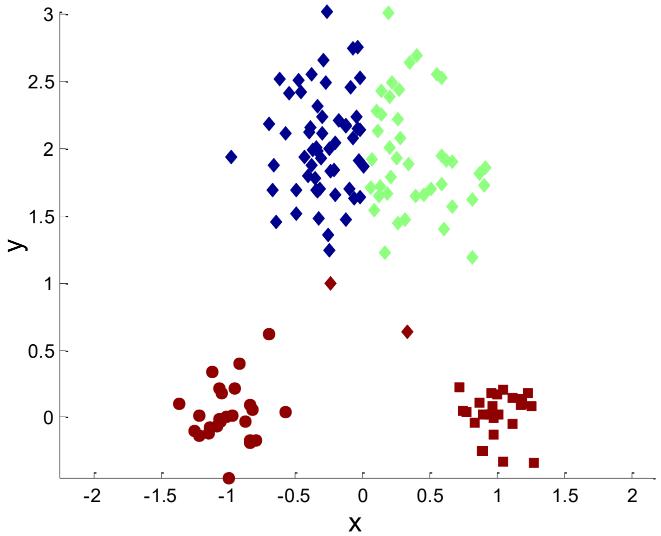
Two different K-means Clusterings



Original Points

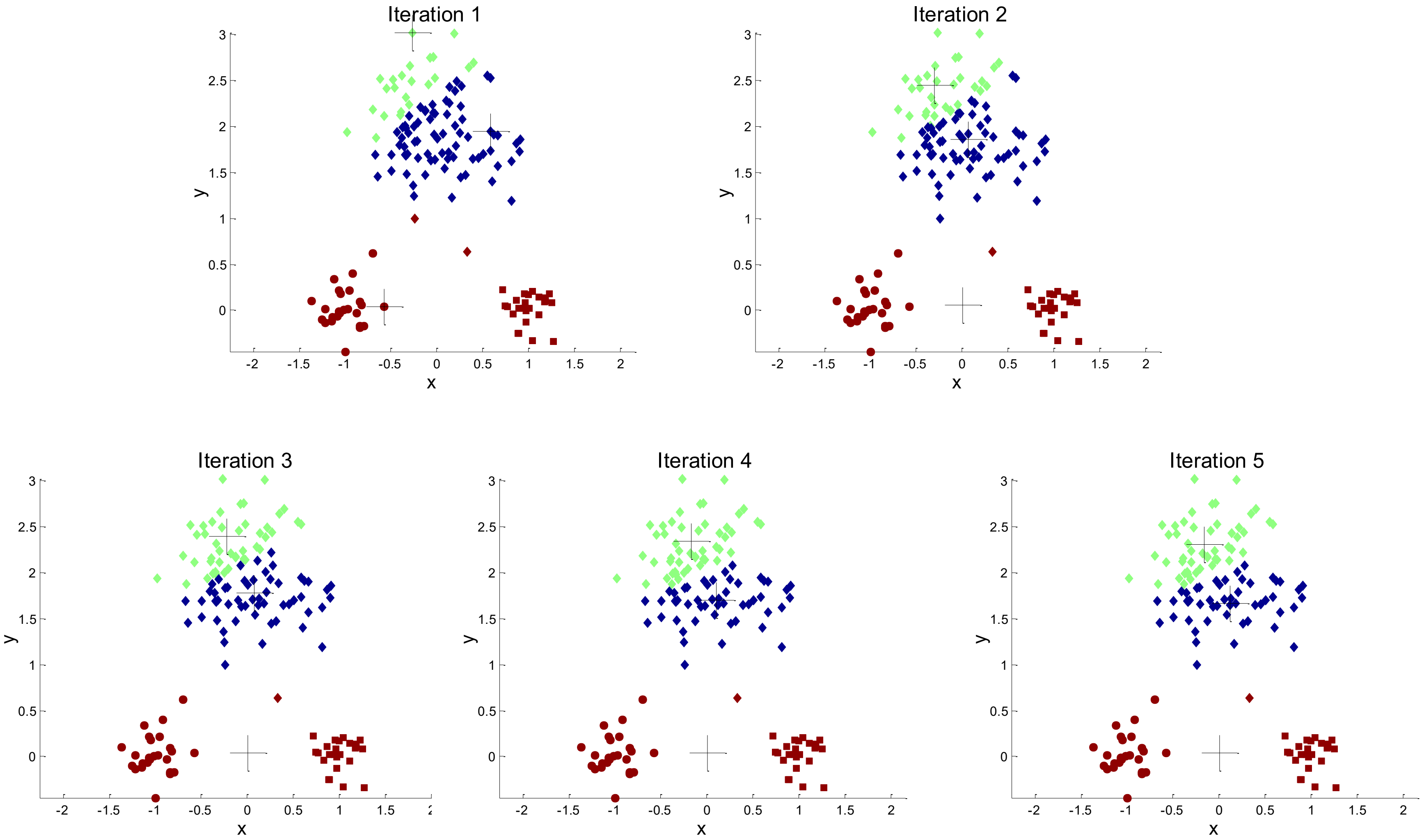


Optimal Clustering



Sub-optimal Clustering

Importance of Choosing Initial Centroids …



Problems with Selecting Initial Points

If there are K 'real' clusters then the chance of selecting one centroid from each cluster is small.

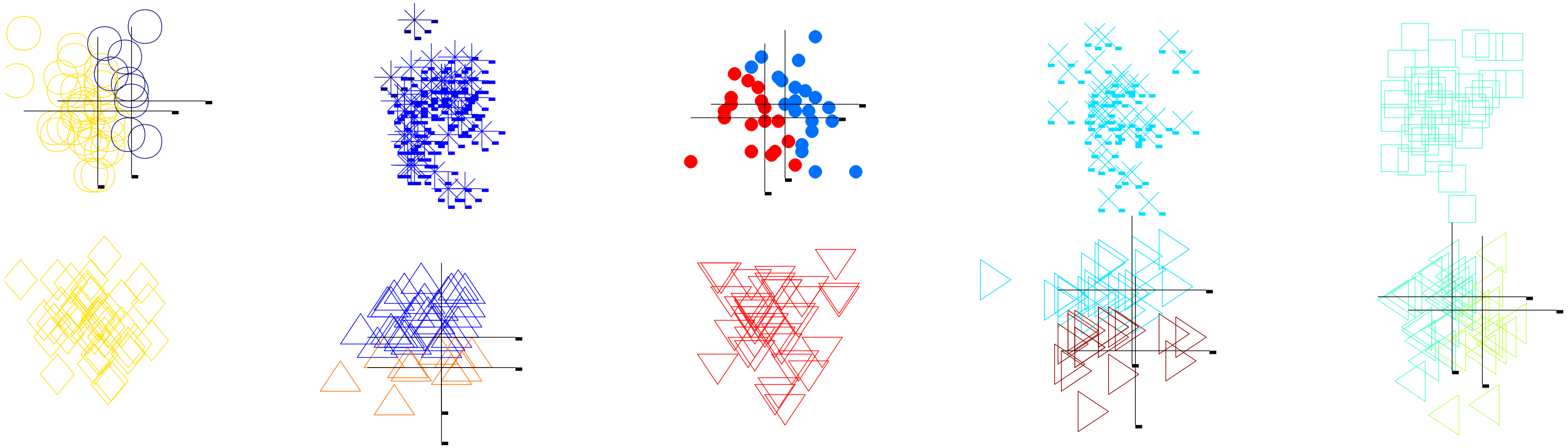
- Chance is relatively small when K is large
- If clusters are the same size, n, then

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

- 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

Limits in random initialization: 10 Clusters Example

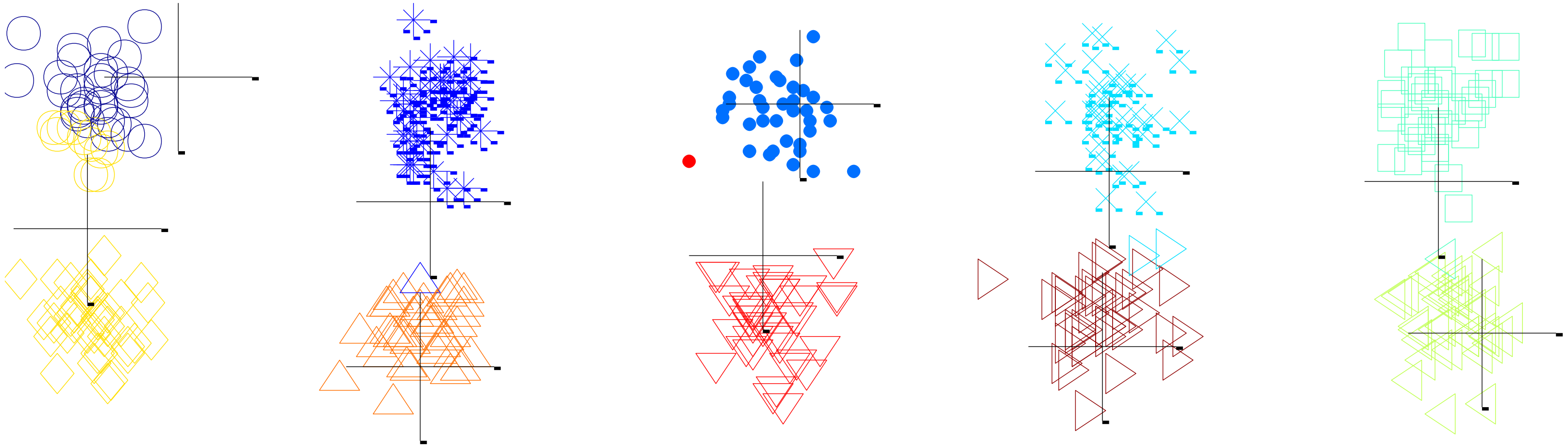
The data consists of 5 pairs of clusters, where the clusters in each (top-bottom) pair are closer to each other than to the clusters in the other pair.



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

Limits in random initialization: 10 Clusters Example

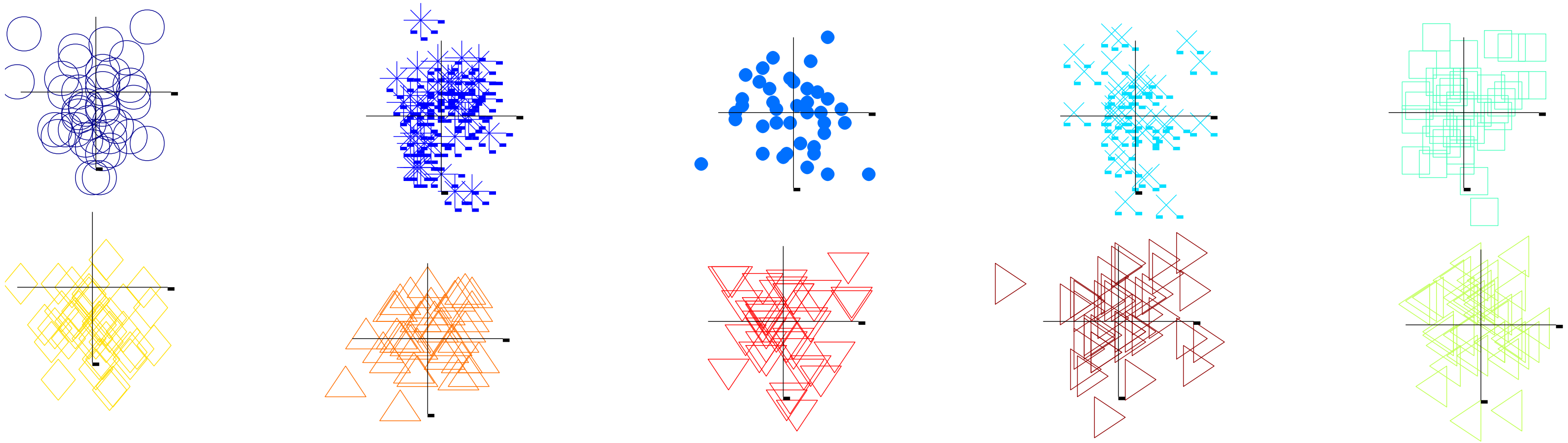
The data consists of 5 pairs of clusters, where the clusters in each (top-bottom) pair are closer to each other than to the clusters in the other pair.



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

Limits in random initialization: 10 Clusters Example

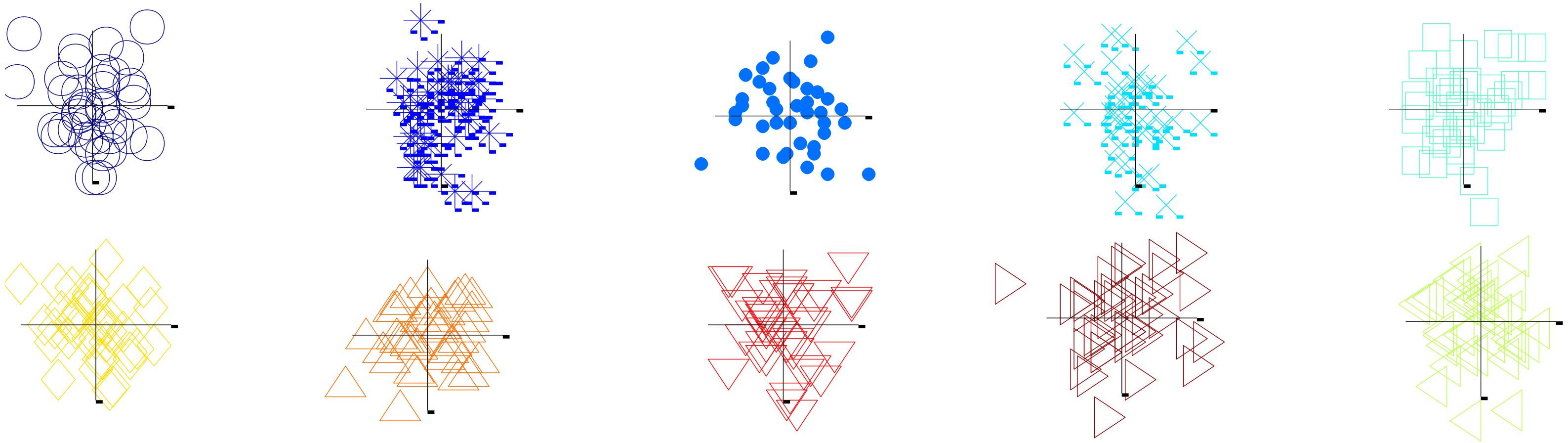
The data consists of 5 pairs of clusters, where the clusters in each (top-bottom) pair are closer to each other than to the clusters in the other pair.



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

Limits in random initialization: 10 Clusters Example

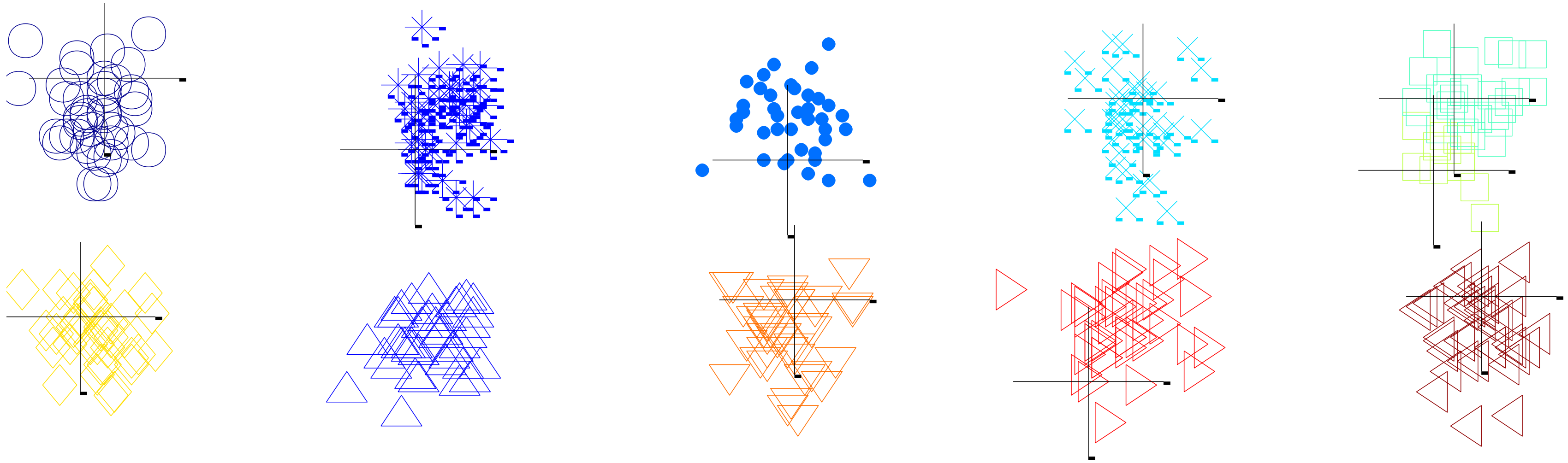
The data consists of 5 pairs of clusters, where the clusters in each (top-bottom) pair are closer to each other than to the clusters in the other pair.



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

Limits in random initialization: 10 Clusters Example

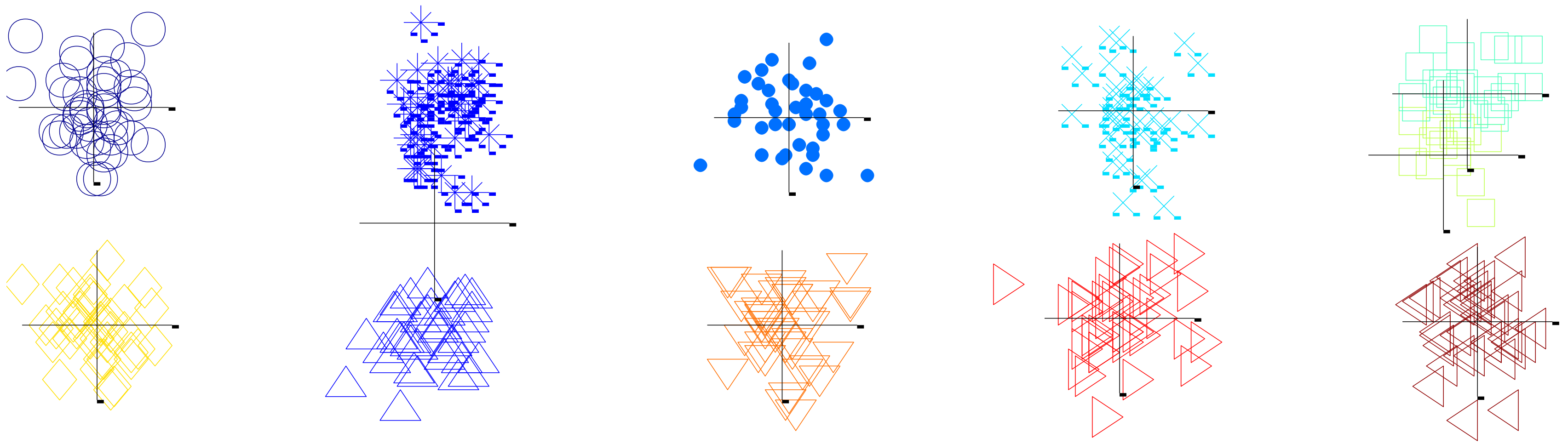
The data consists of 5 pairs of clusters, where the clusters in each (top-bottom) pair are closer to each other than to the clusters in the other pair.



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

Limits in random initialization: 10 Clusters Example

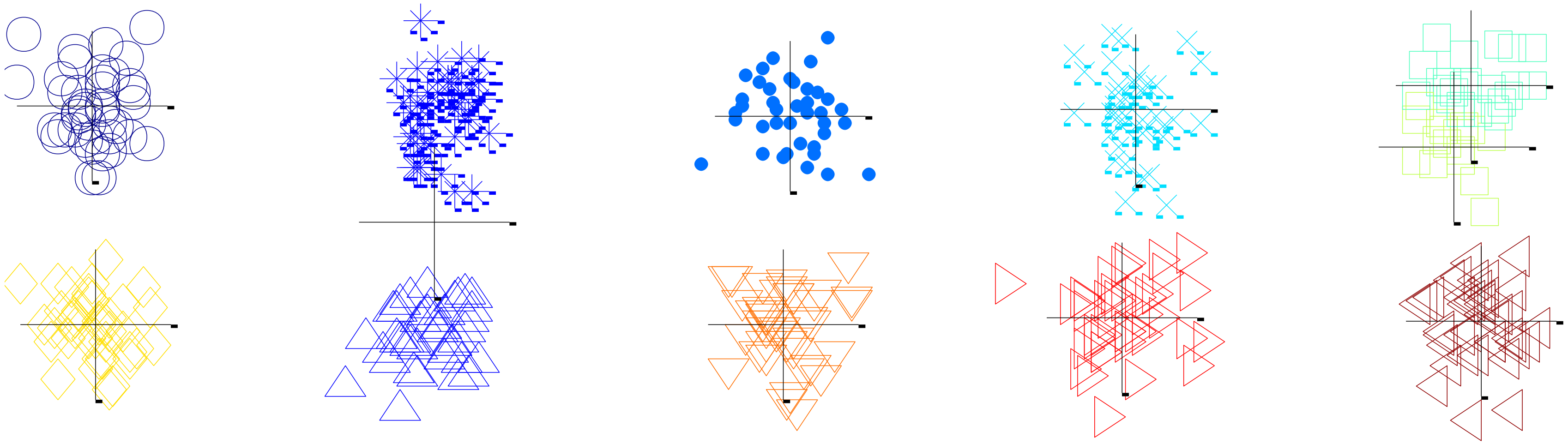
The data consists of 5 pairs of clusters, where the clusters in each (top-bottom) pair are closer to each other than to the clusters in the other pair.



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

Limits in random initialization: 10 Clusters Example

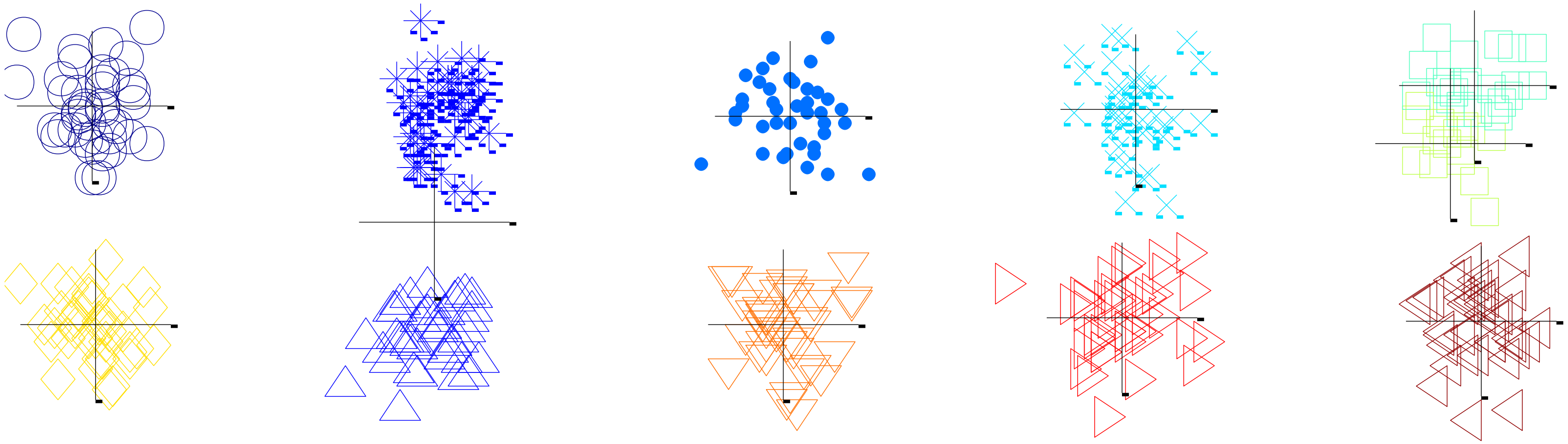
The data consists of 5 pairs of clusters, where the clusters in each (top-bottom) pair are closer to each other than to the clusters in the other pair.



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

Limits in random initialization: 10 Clusters Example

The data consists of 5 pairs of clusters, where the clusters in each (top-bottom) pair are closer to each other than to the clusters in the other pair.



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

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, $1 \leq i \leq N$, find the minimum squared distance to the currently selected centroids, $C_1, \ldots, C_j$, $1 \leq j < k$, 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

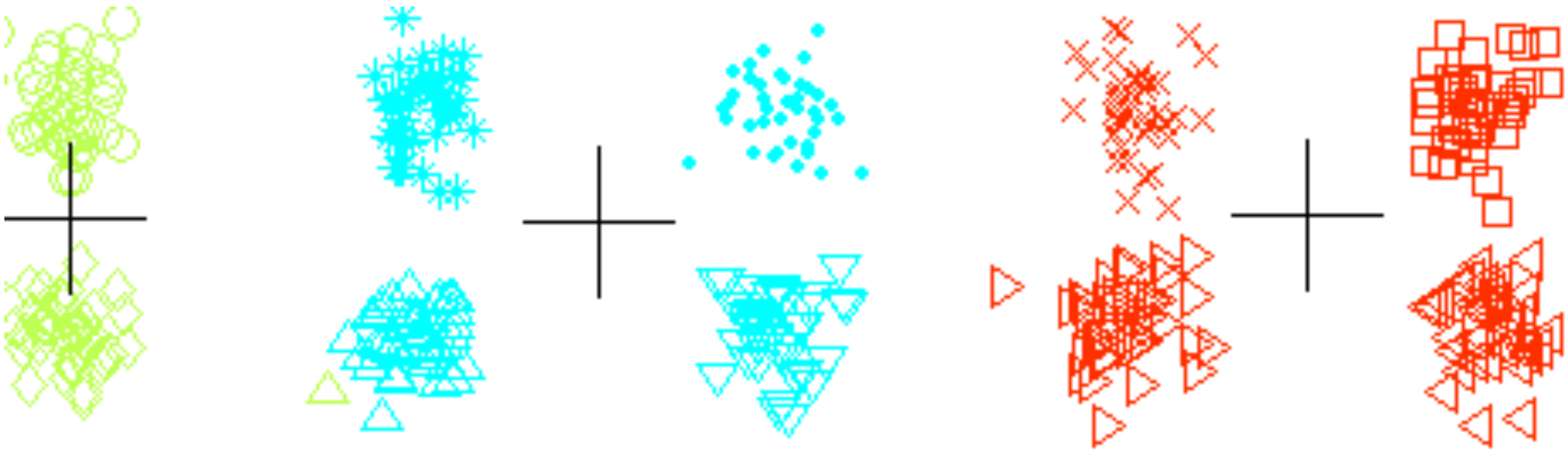
Variant of K-means that can produce a partitional or a hierarchical clustering

```
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
```

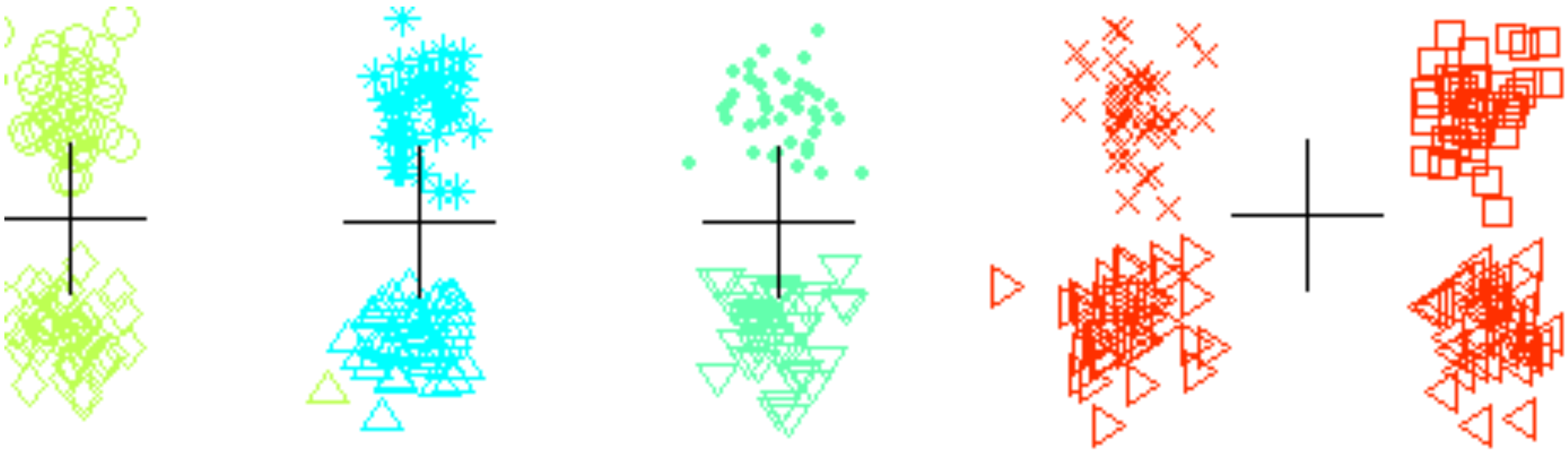
CLUTO: http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview

Bisecting K-means Example

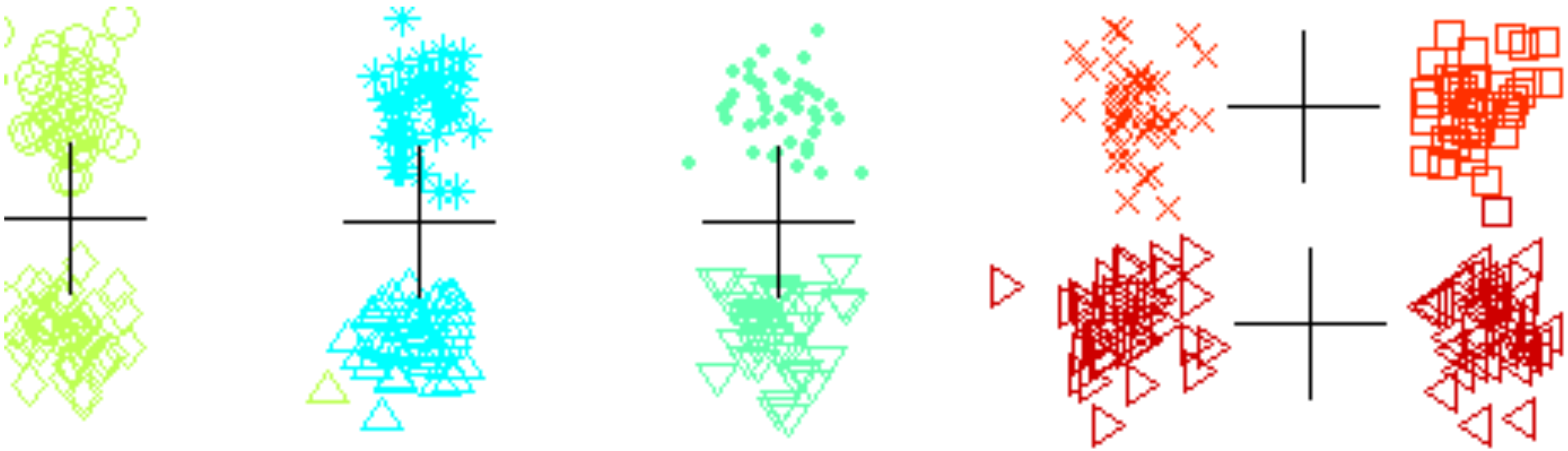
Bisecting K-means Example



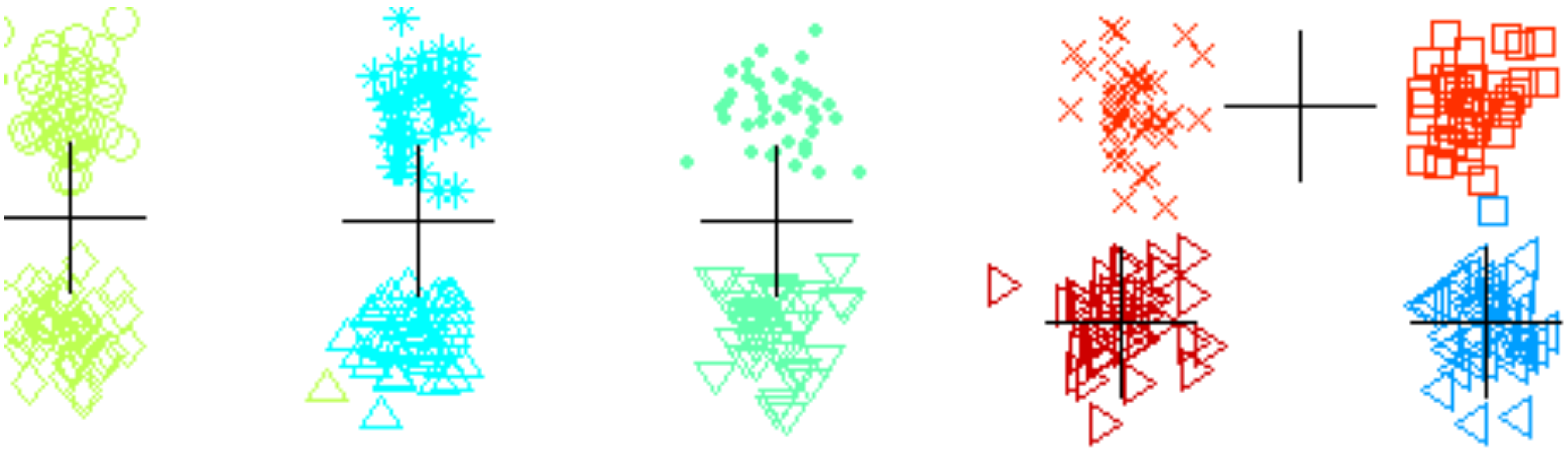
Bisecting K-means Example



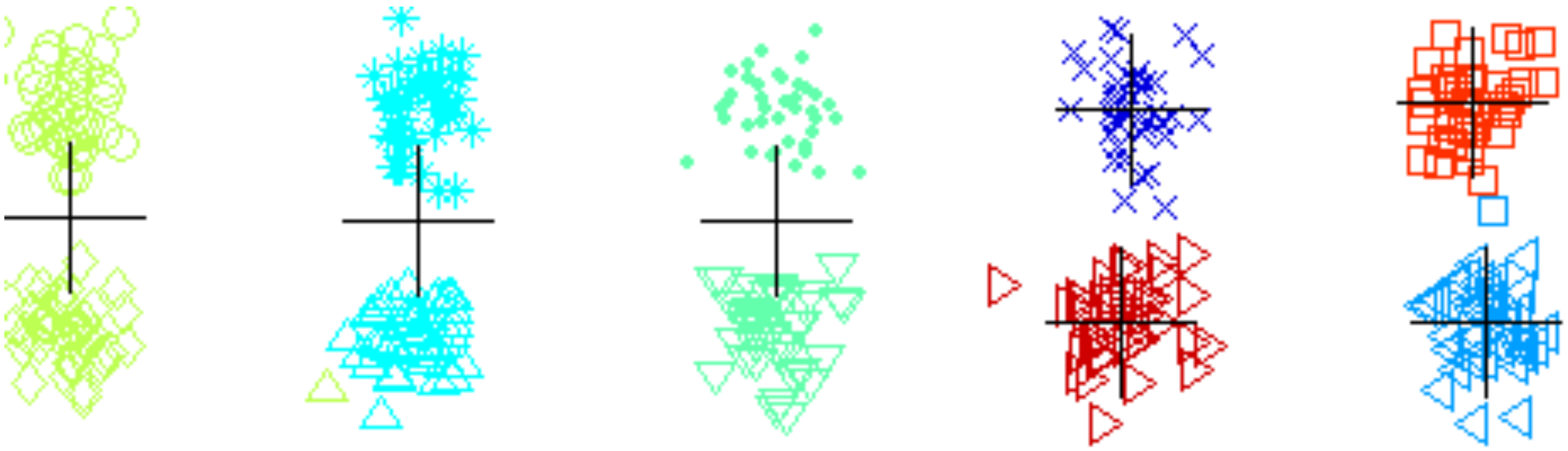
Bisecting K-means Example



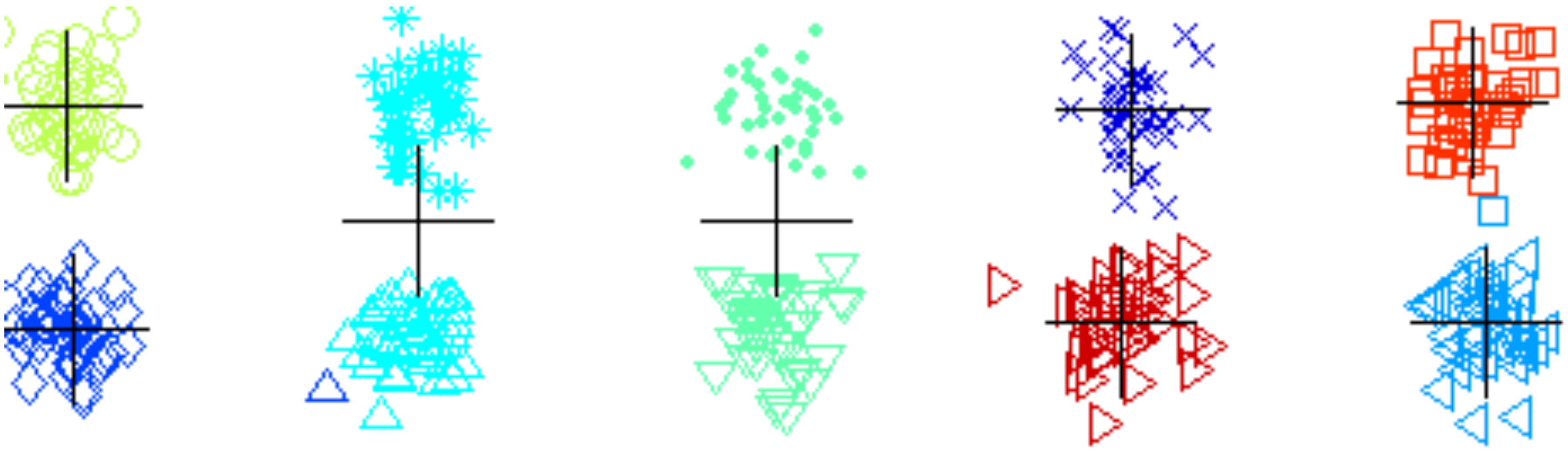
Bisecting K-means Example



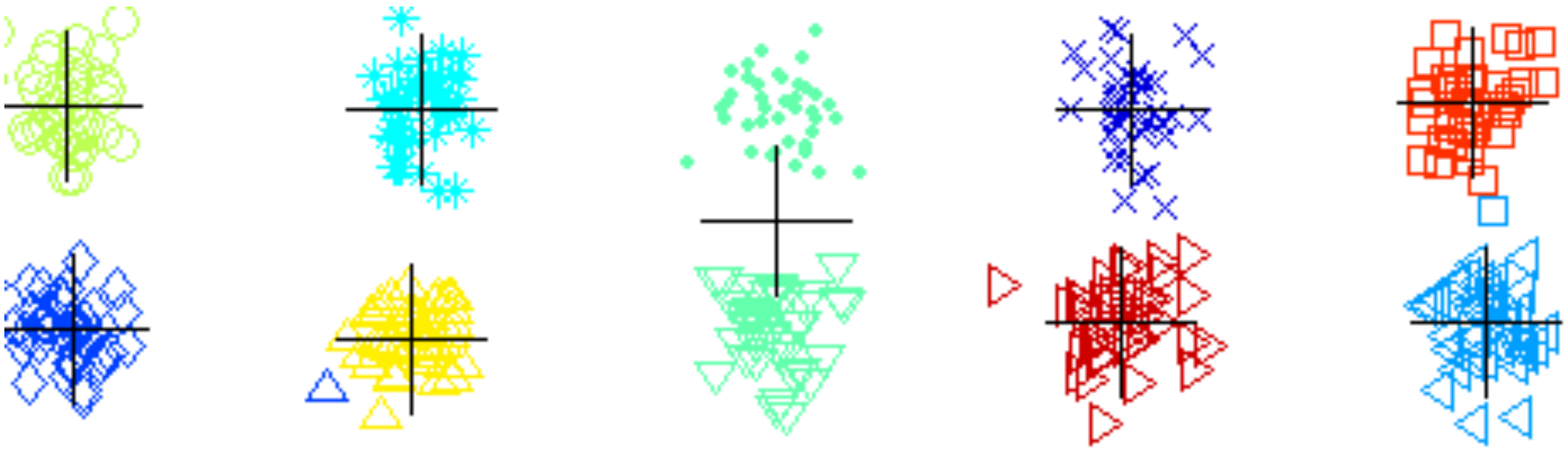
Bisecting K-means Example



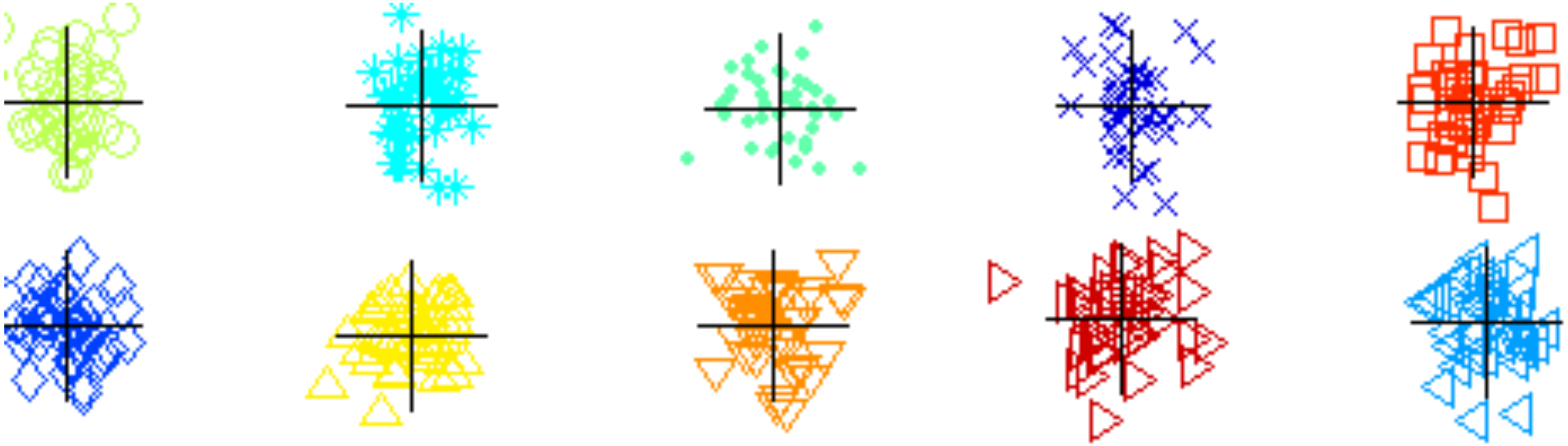
Bisecting K-means Example



Bisecting K-means Example



Bisecting K-means Example



Limitations of K-means

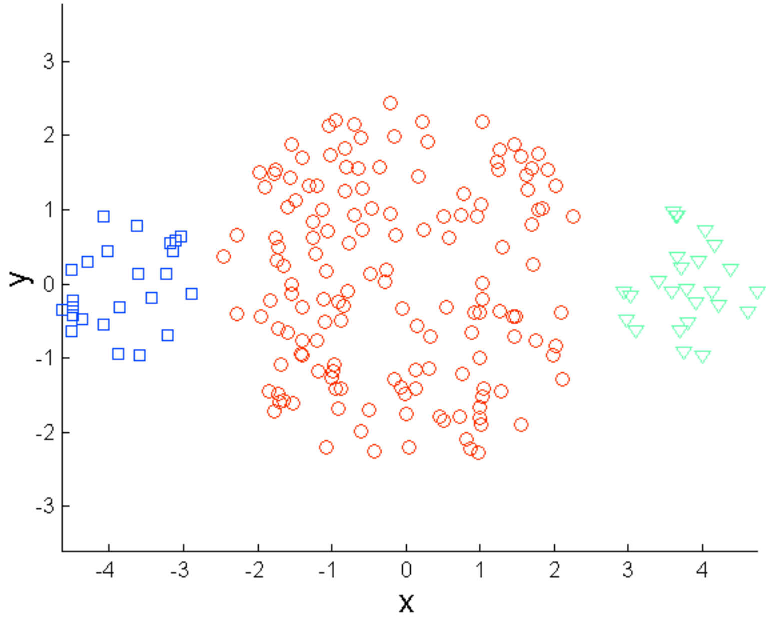
K-means has problems when clusters are of differing

- Sizes
- Densities
- Non-globular shapes

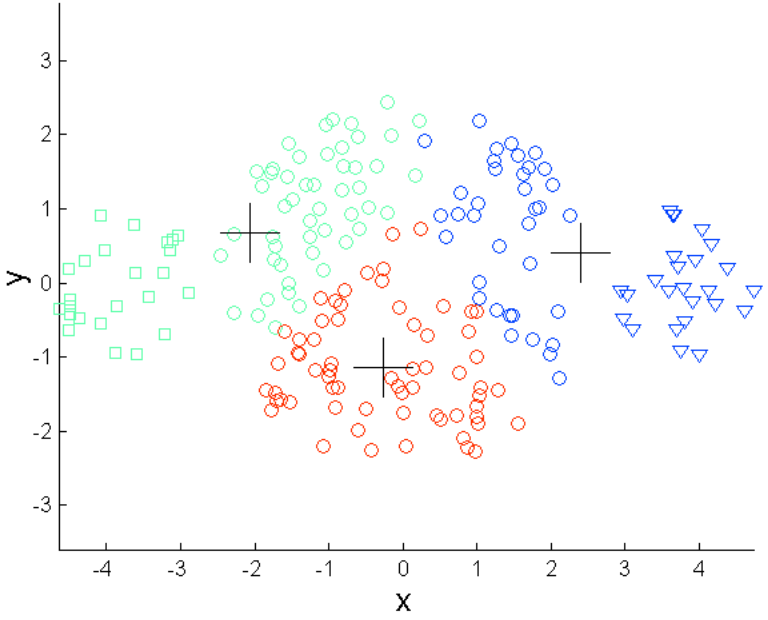
K-means has problems when the data contains outliers.

- One possible solution is to remove outliers before clustering

Limitations of K-means: Differing Sizes

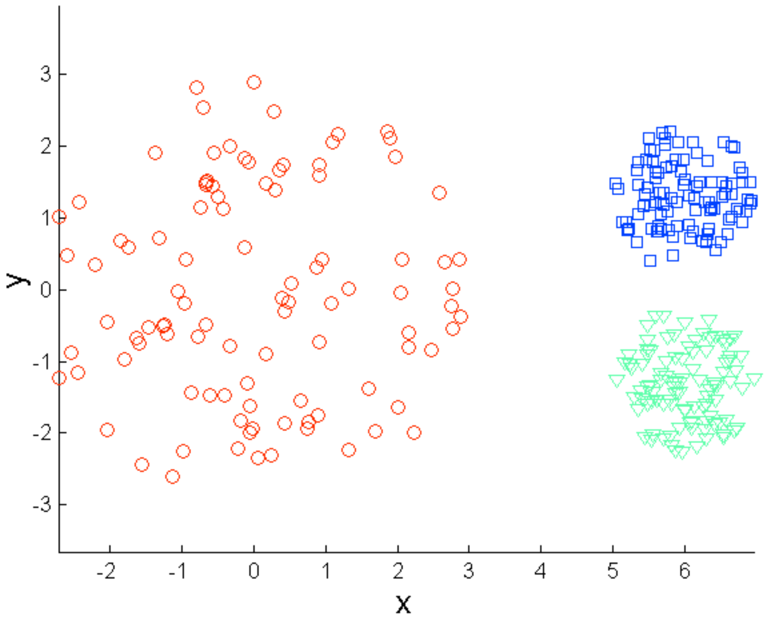


Original Points

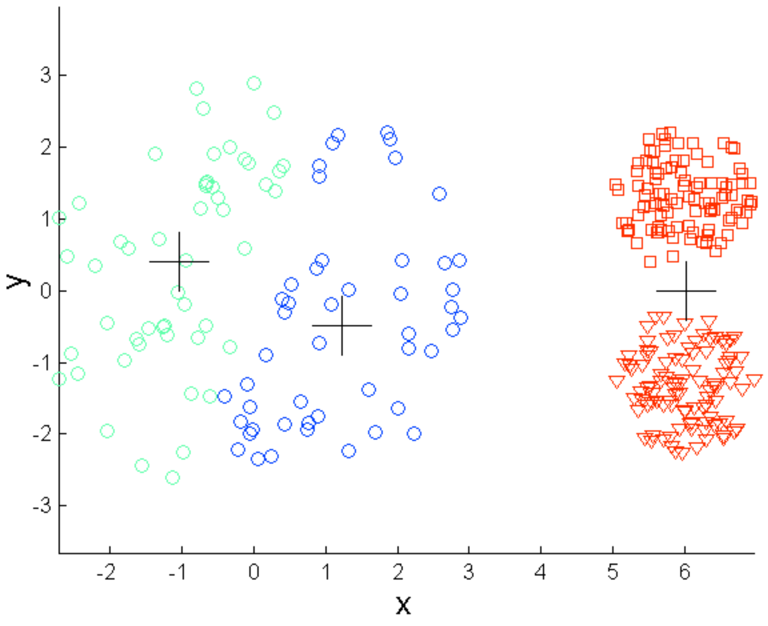


K-means (3 Clusters)

Limitations of K-means: Differing Density

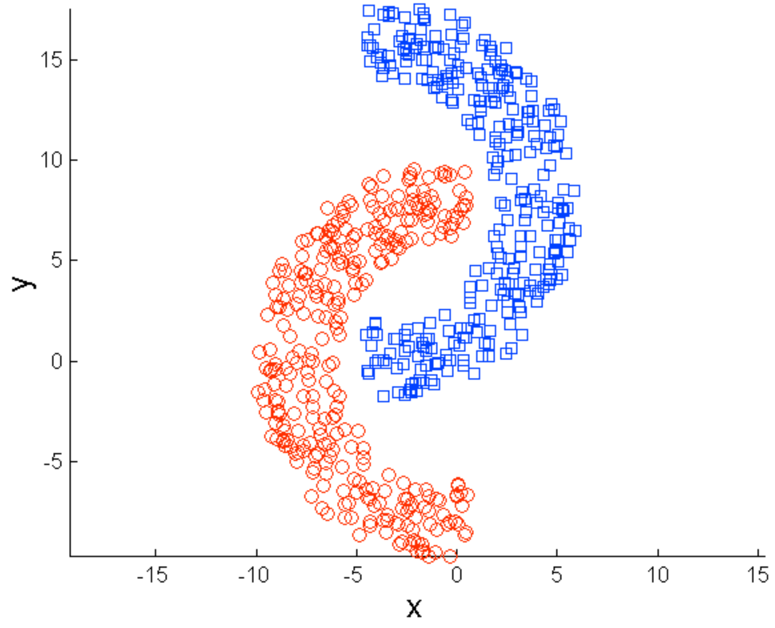


Original Points

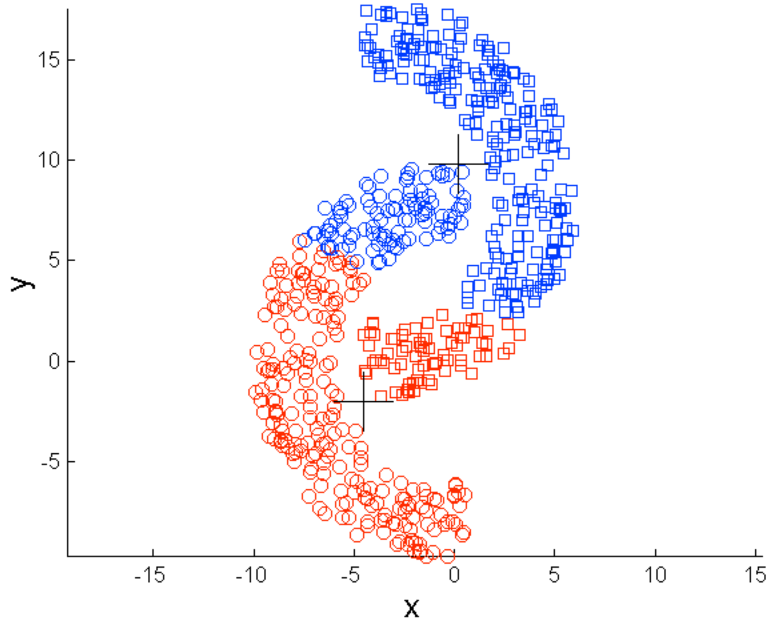


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes



Original Points

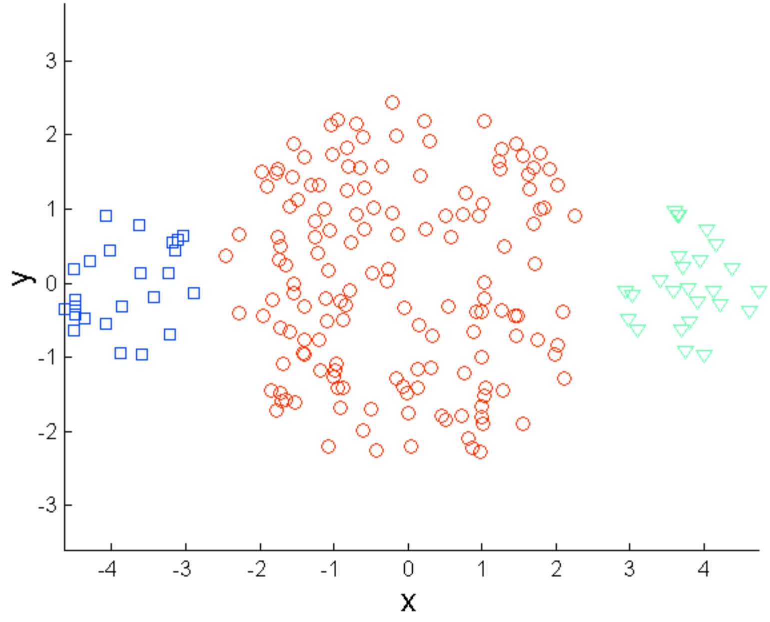


K-means (2 Clusters)

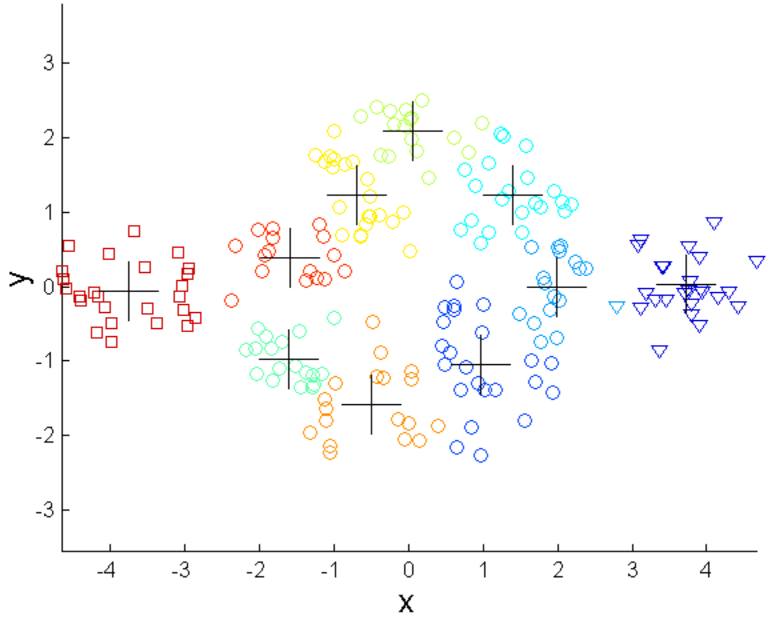
Overcoming K-means Limitations

One solution is to find a **large number of clusters** such that each of them represents a part of a natural cluster.

Small clusters need to be put together in a **post-processing** step.



Original Points

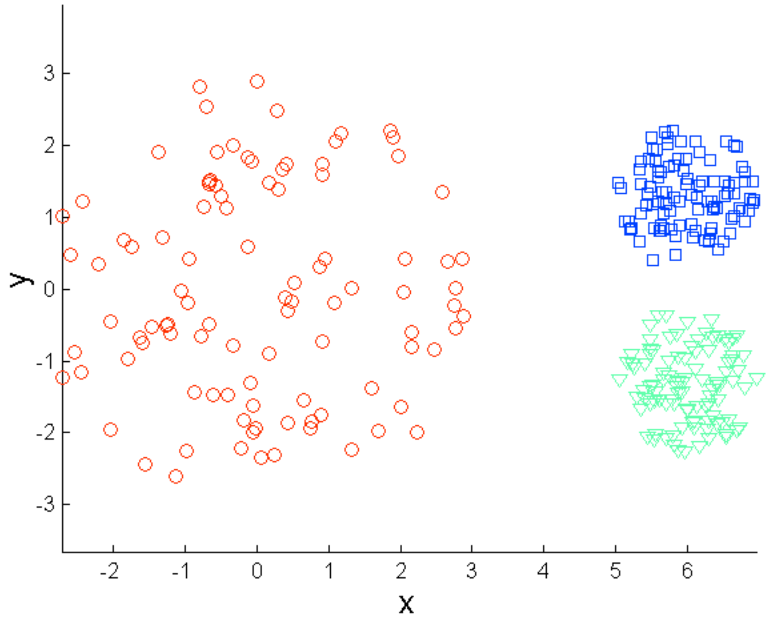


K-means Clusters

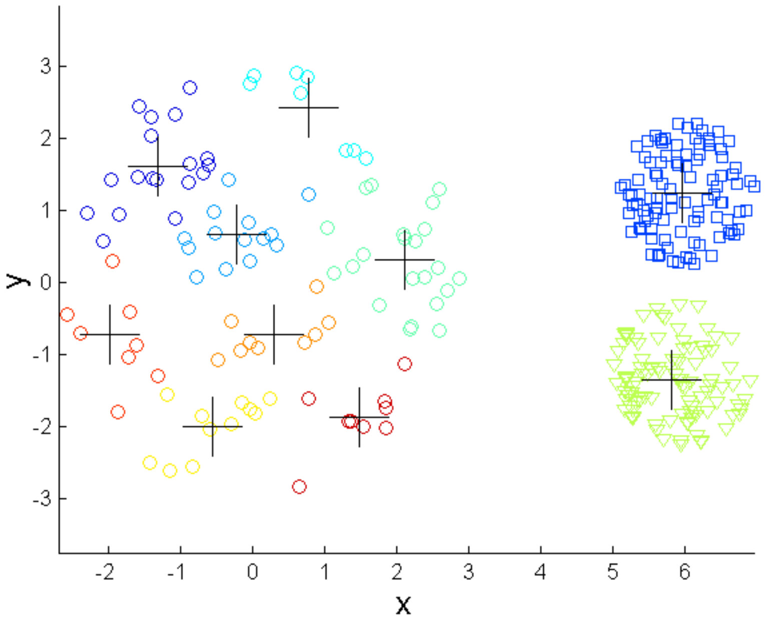
Overcoming K-means Limitations

One solution is to find a **large number of clusters** such that each of them represents a part of a natural cluster.

Small clusters need to be put together in a **post-processing** step.



Original Points

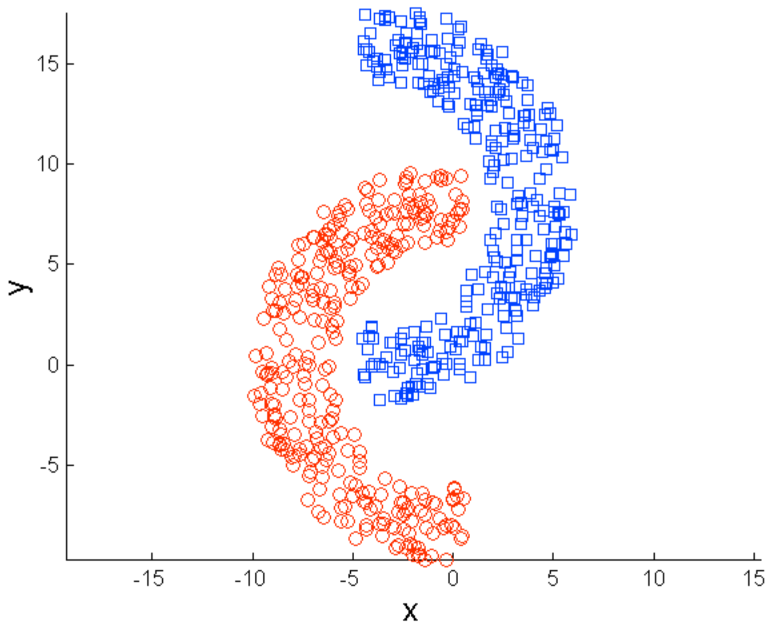


K-means Clusters

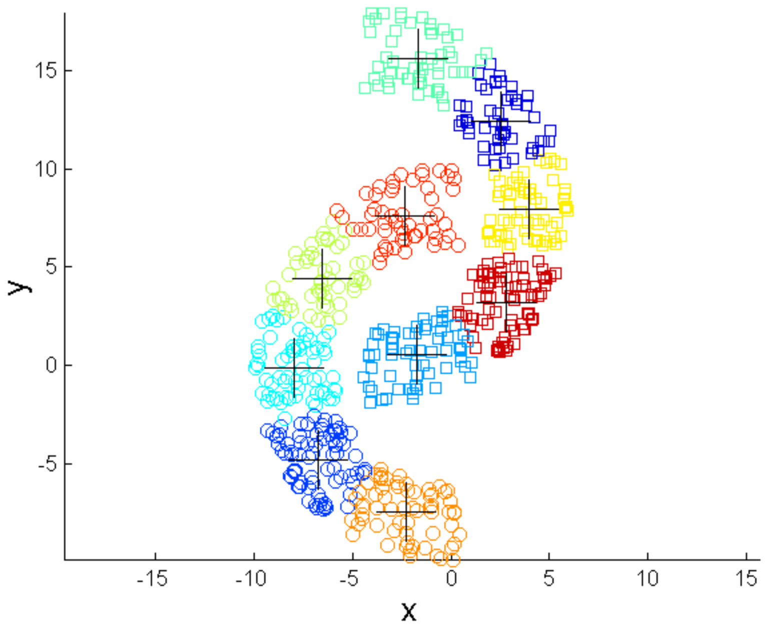
Overcoming K-means Limitations

One solution is to find a **large number of clusters** such that each of them represents a part of a natural cluster.

Small clusters need to be put together in a **post-processing** step.



Original Points

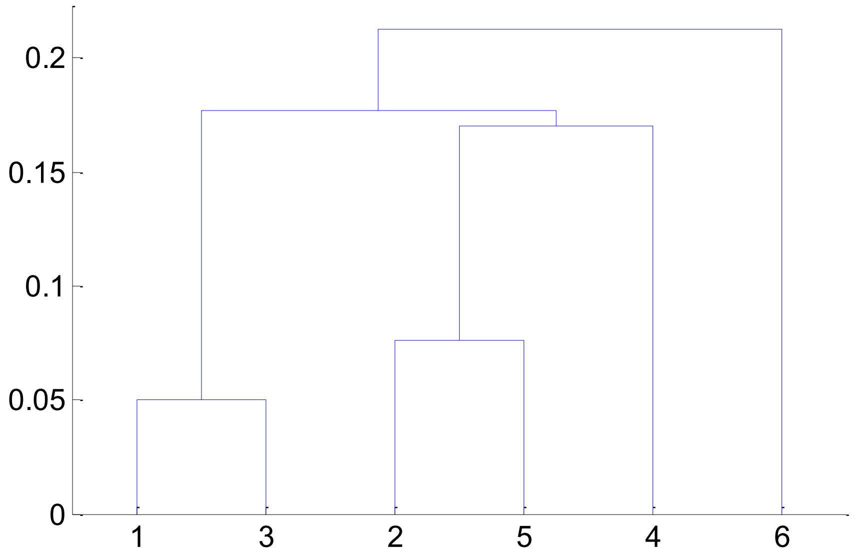


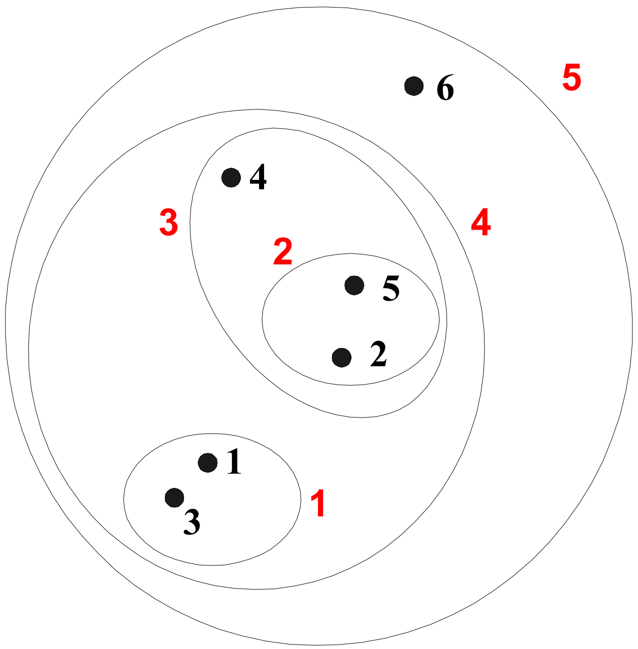
K-means Clusters

Hierarchical Clustering

Produces a set of nested clusters organized as a hierarchical tree (**dendogram**)

- 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 cluster

Basic algorithm

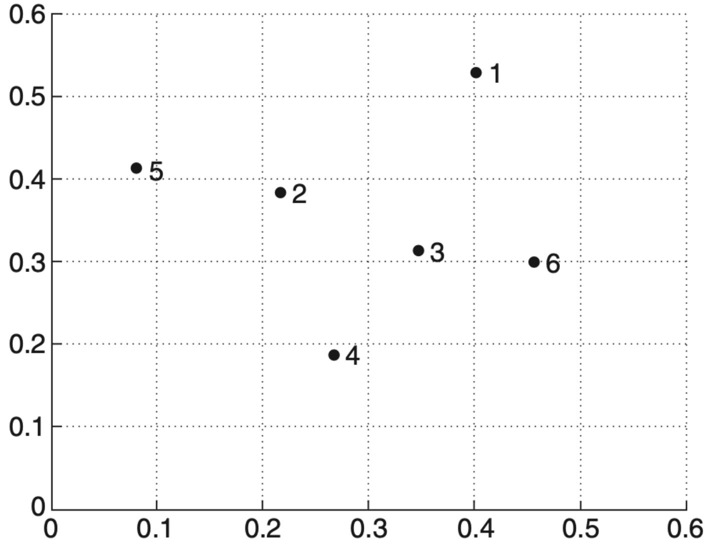
1. Compute the proximity matrix
2. Let each data point be a cluster
3. **Repeat**
4. Merge the two closest clusters
5. Update the proximity matrix
6. **Until** only a single cluster remains

Key operation is the computation of the proximity of two clusters

- Different approaches to defining the distance between clusters distinguish the different algorithms

Proximity matrix

Start with clusters of individual points and a proximity matrix

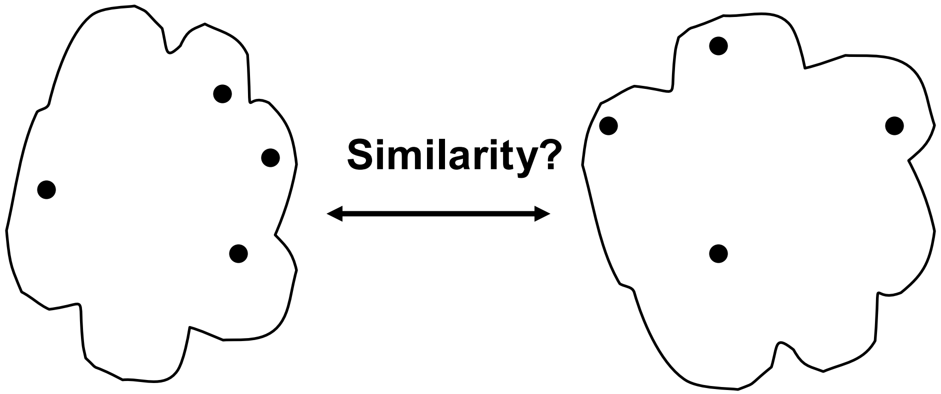


Point	x Coordinate	y Coordinate
p1	0.4005	0.5306
p2	0.2148	0.3854
p3	0.3457	0.3156
p4	0.2652	0.1875
p5	0.0789	0.4139
p6	0.4548	0.3022

	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

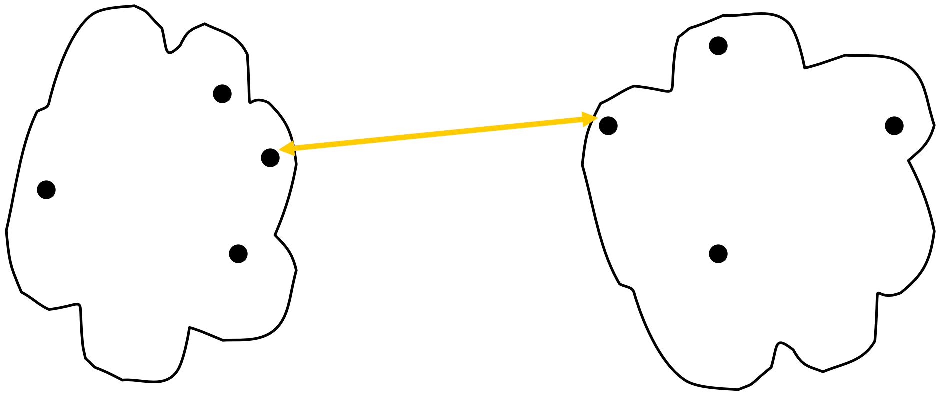
Type of similarity

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



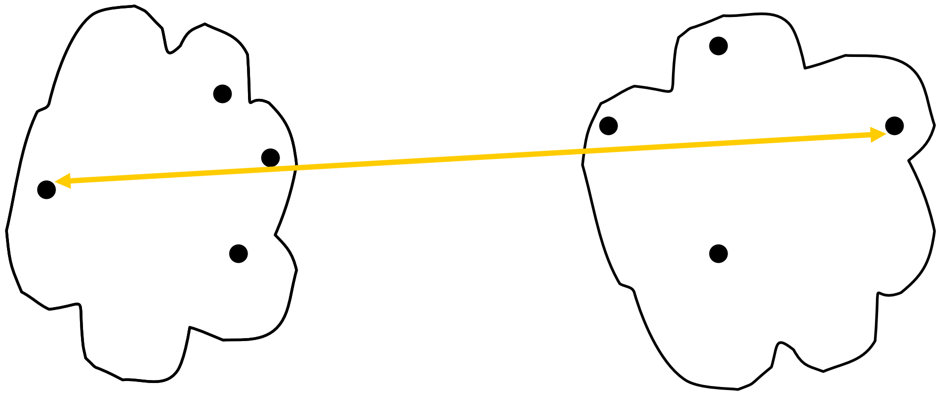
Type of similarity

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



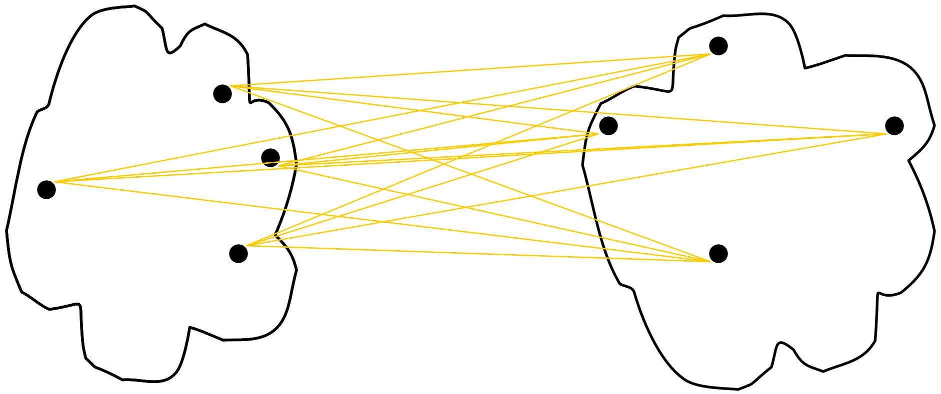
Type of similarity

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



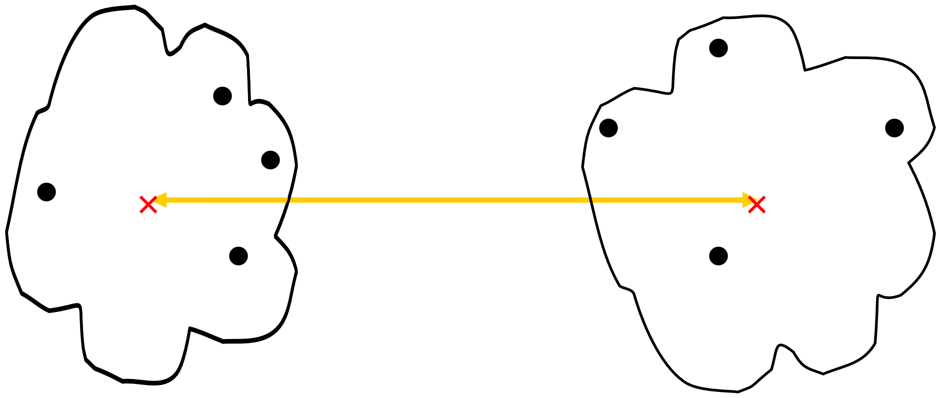
Type of similarity

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



Type of similarity

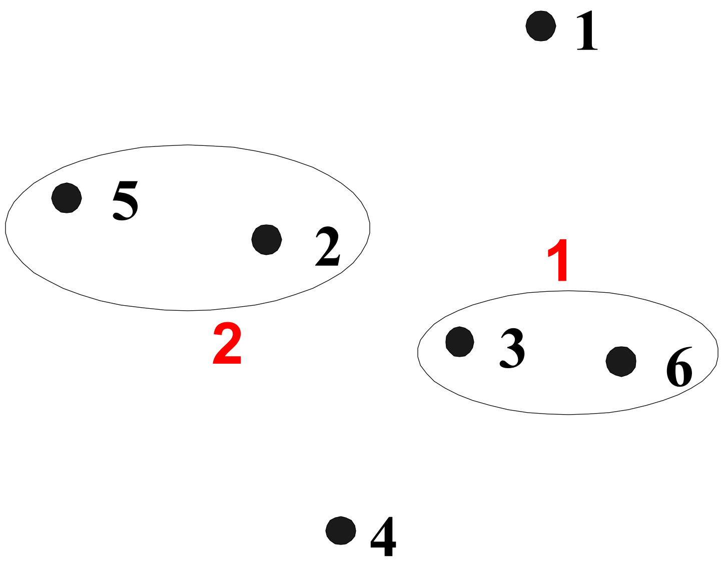
- MIN
- MAX
- Group Average
- Distance Between Centroids
- Methods driven by an objective function (Ward's Method uses squared error)



MIN or Single Link

Proximity of two clusters is defined as the minimum of the distance (maximum of the similarity) between any two points in the two different clusters

$$
\begin{aligned}
dist(\{3,6\},\{2,5\}) &= \min(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\
&= \min(0.15, 0.25, 0.28, 0.39) \\
&= 0.15.
\end{aligned}
$$



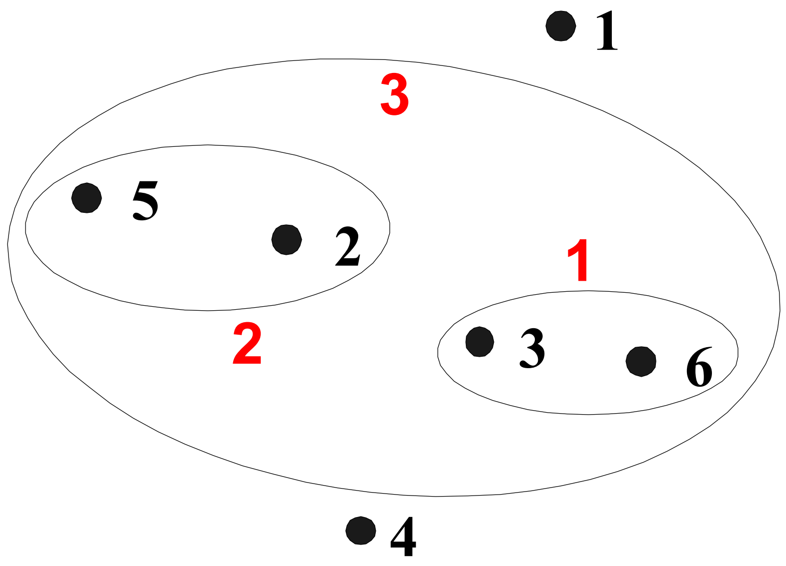
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Distance Matrix:

MIN or Single Link

Proximity of two clusters is defined as the minimum of the distance (maximum of the similarity) between any two points in the two different clusters

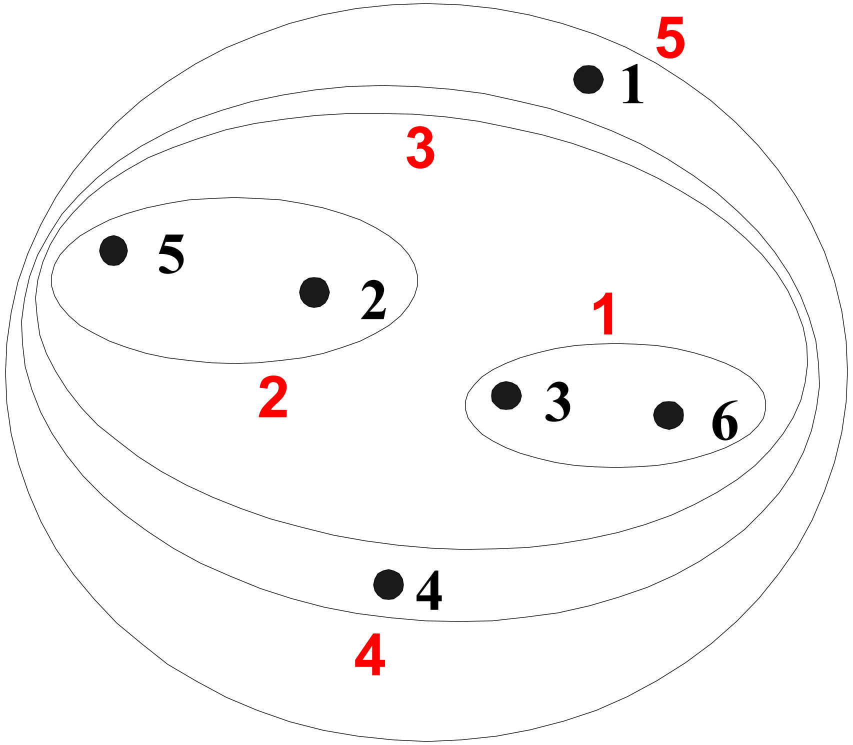
$$
\begin{aligned}
dist(\{3,6\},\{2,5\}) &= \min(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\
&= \min(0.15, 0.25, 0.28, 0.39) \\
&= 0.15.
\end{aligned}
$$



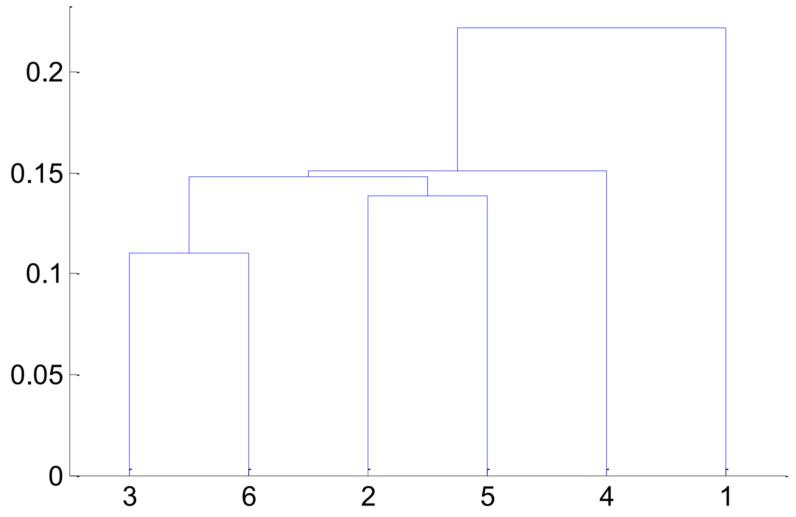
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Distance Matrix:

MIN or Single Link



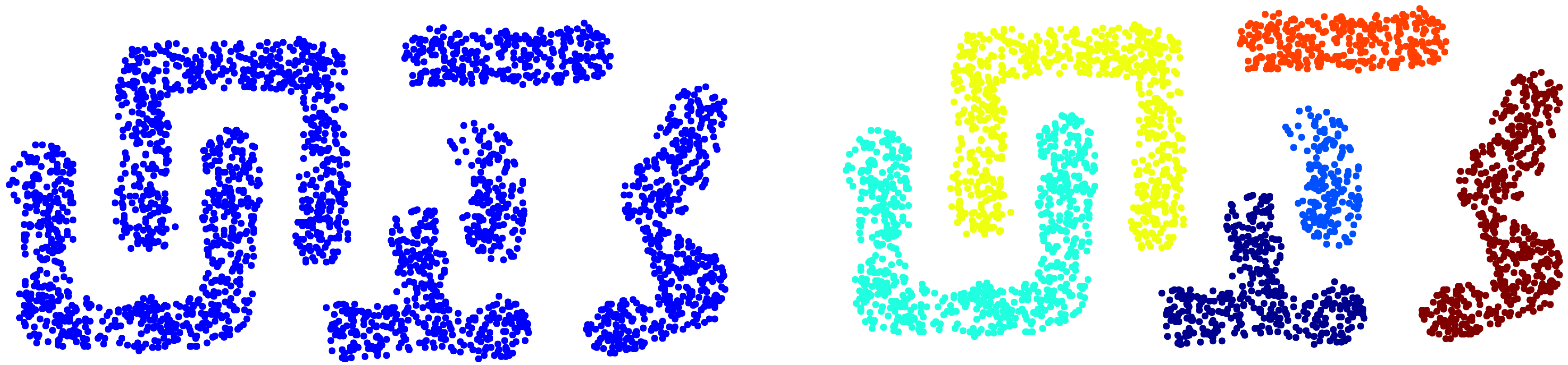
Nested Clusters



Dendrogram

MIN or Single Link: Strength

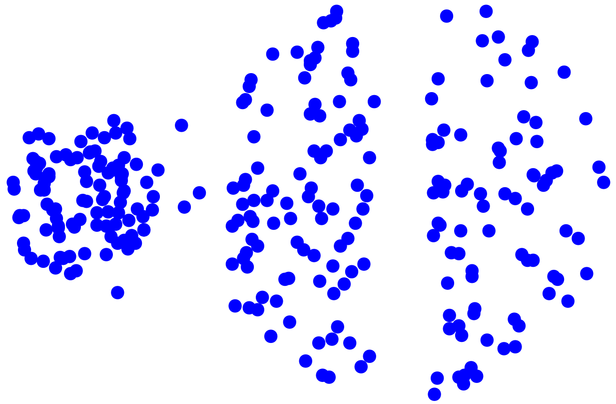
- **Can handle non-elliptical shapes**



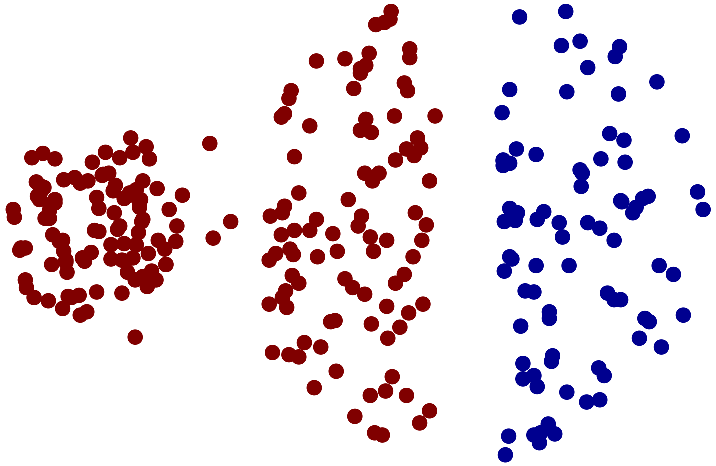
Original Points **Six Clusters**

MIN or Single Link: Limitations

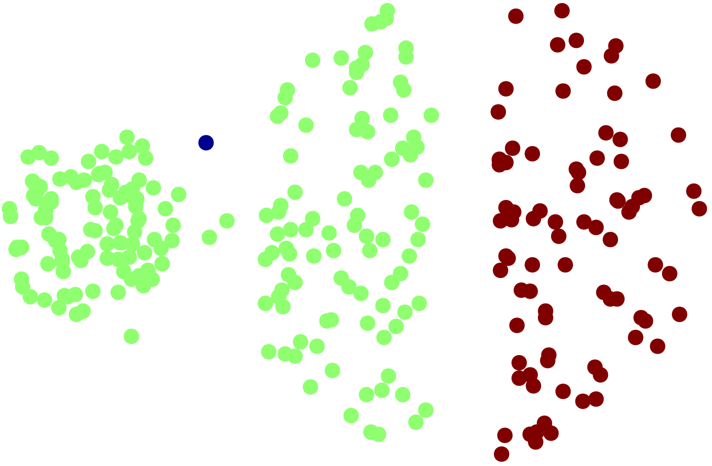
- **Sensitive to noise**



Original Points



Two Clusters



Three Clusters

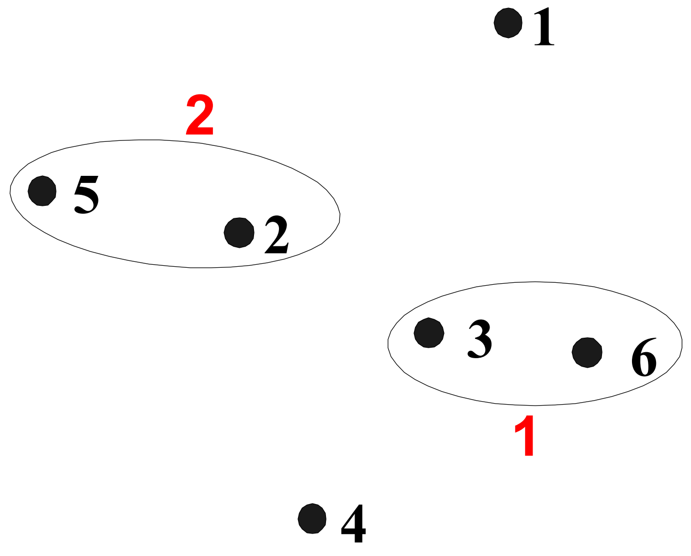
MAX or Complete Linkage

Proximity of two clusters is defined as the maximum of the distance (minimum of the similarity) between any two points in the two different clusters

$$
\begin{aligned}
dist(\{3,6\},\{4\}) &= \max(dist(3,4), dist(6,4)) \\
&= \max(0.15, 0.22) \\
&= 0.22. \\
dist(\{3,6\},\{2,5\}) &= \max(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\
&= \max(0.15, 0.25, 0.28, 0.39) \\
&= 0.39. \\
dist(\{3,6\},\{1\}) &= \max(dist(3,1), dist(6,1)) \\
&= \max(0.22, 0.23) \\
&= 0.23.
\end{aligned}
$$

	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

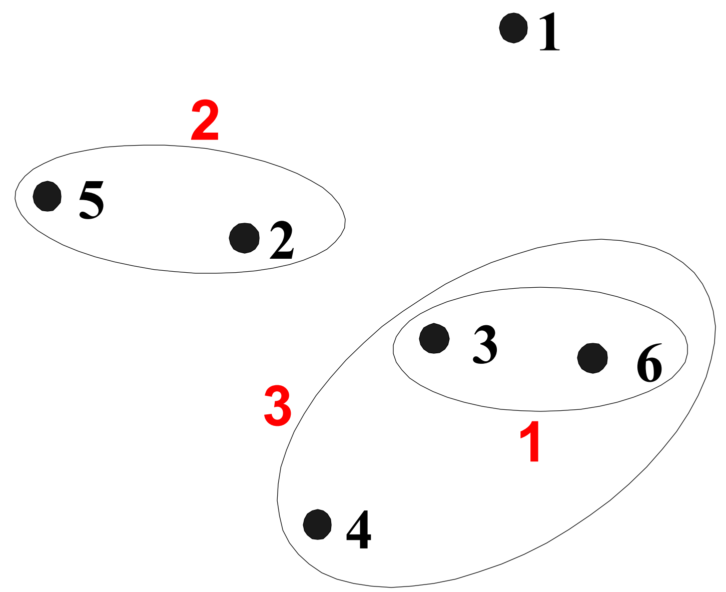
Distance Matrix:



MAX or Complete Linkage

Proximity of two clusters is defined as the maximum of the distance (minimum of the similarity) between any two points in the two different clusters

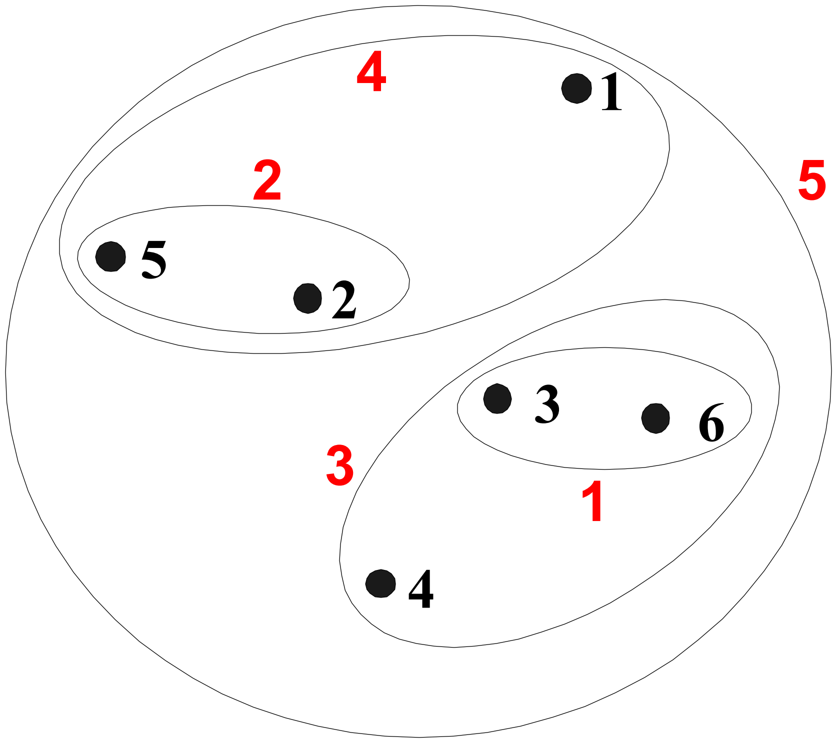
$$
\begin{aligned}
dist(\{3,6\},\{4\}) &= \max(dist(3,4), dist(6,4)) \\
&= \max(0.15, 0.22) \\
&= 0.22. \\
dist(\{3,6\},\{2,5\}) &= \max(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\
&= \max(0.15, 0.25, 0.28, 0.39) \\
&= 0.39. \\
dist(\{3,6\},\{1\}) &= \max(dist(3,1), dist(6,1)) \\
&= \max(0.22, 0.23) \\
&= 0.23.
\end{aligned}
$$



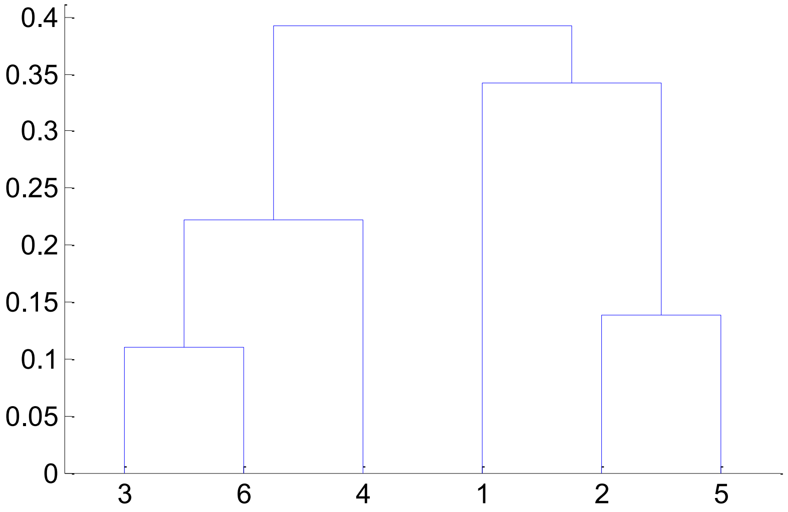
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Distance Matrix:

MAX or Complete Linkage



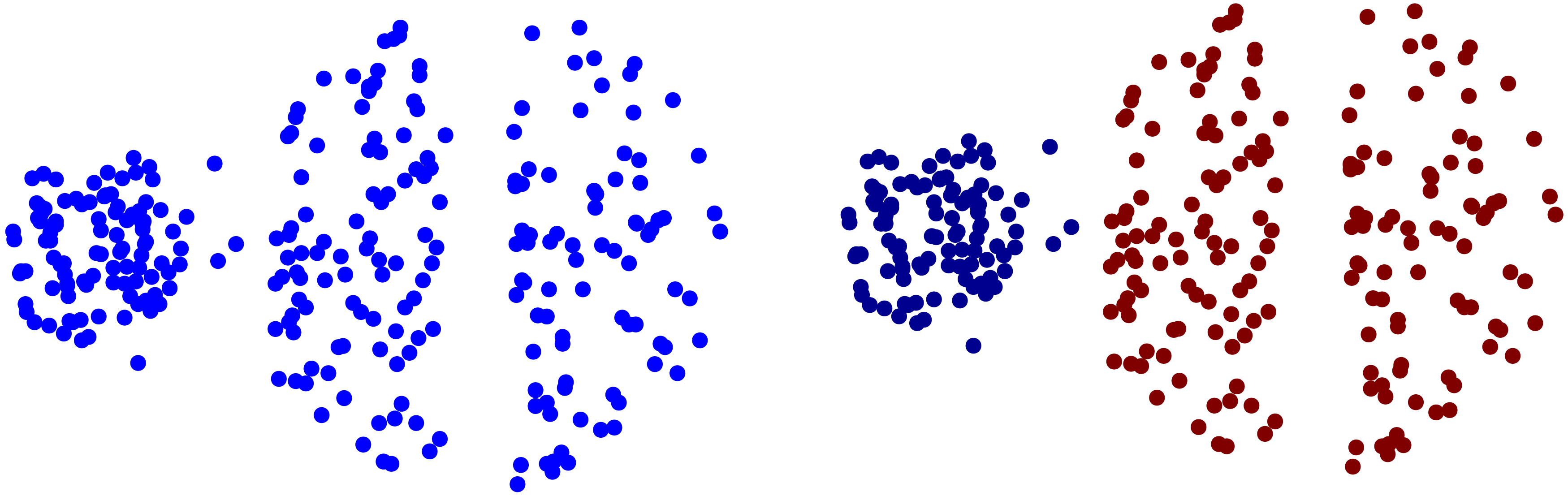
Nested Clusters



Dendrogram

MAX or Complete Linkage: Strength

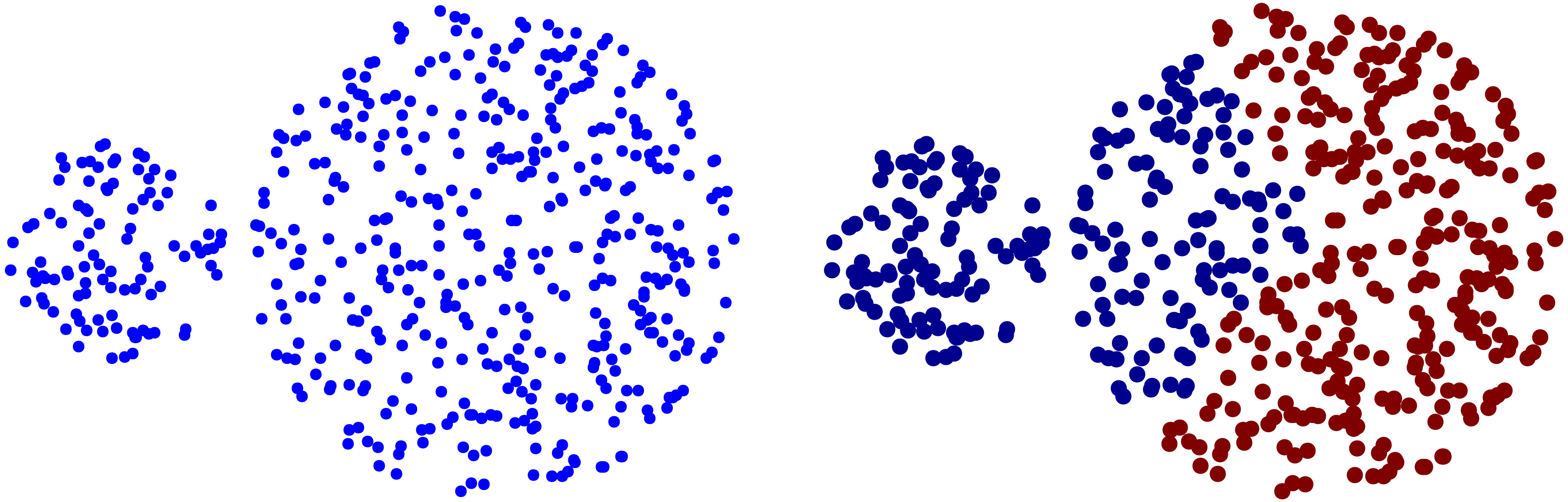
- **Less susceptible to noise**



Original Points **Two Clusters**

MAX or Complete Linkage: Limitations

- **Tends to break large clusters**
- **Biased towards globular clusters**



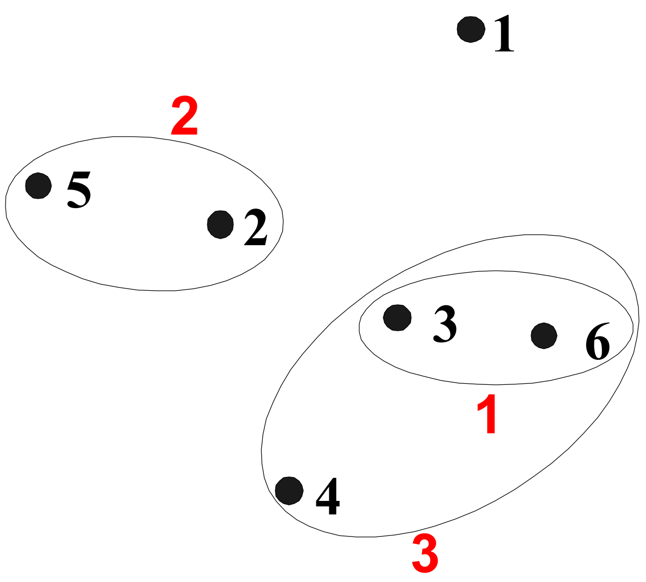
Original Points **Two Clusters**

Group Average

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

$$proximity(C_i, C_j) = \frac{\sum_{\substack{\mathbf{x}\in C_i \\ \mathbf{y}\in C_j}} proximity(\mathbf{x}, \mathbf{y})}{m_i \times m_j}.$$

$$\begin{aligned}
dist(\{3,6,4\},\{1\}) &= (0.22 + 0.37 + 0.23)/(3 \times 1) \\
&= 0.28 \\
dist(\{2,5\},\{1\}) &= (0.24 + 0.34)/(2 \times 1) \\
&= 0.29 \\
dist(\{3,6,4\},\{2,5\}) &= (0.15 + 0.28 + 0.25 + 0.39 + 0.20 + 0.29)/(3 \times 2) \\
&= 0.26
\end{aligned}$$



	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

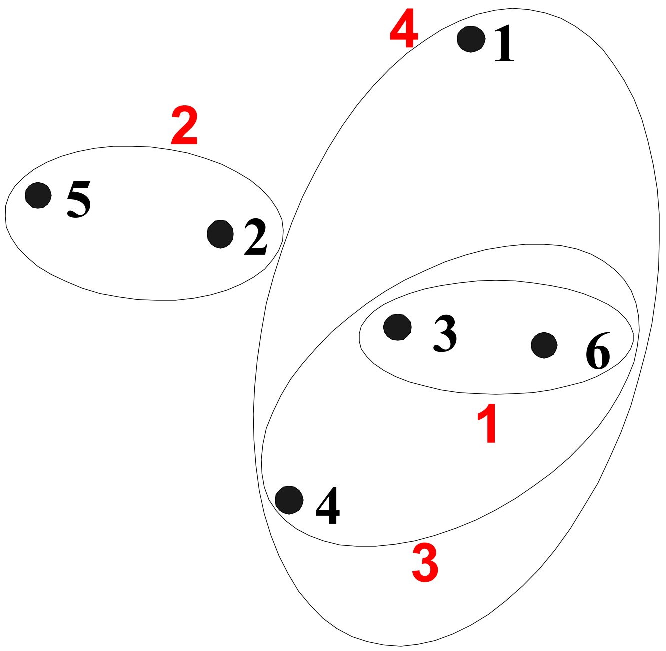
Distance Matrix:

Group Average

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

$$proximity(C_i, C_j) = \frac{\sum_{\substack{\mathbf{x}\in C_i \\ \mathbf{y}\in C_j}} proximity(\mathbf{x}, \mathbf{y})}{m_i \times m_j}.$$

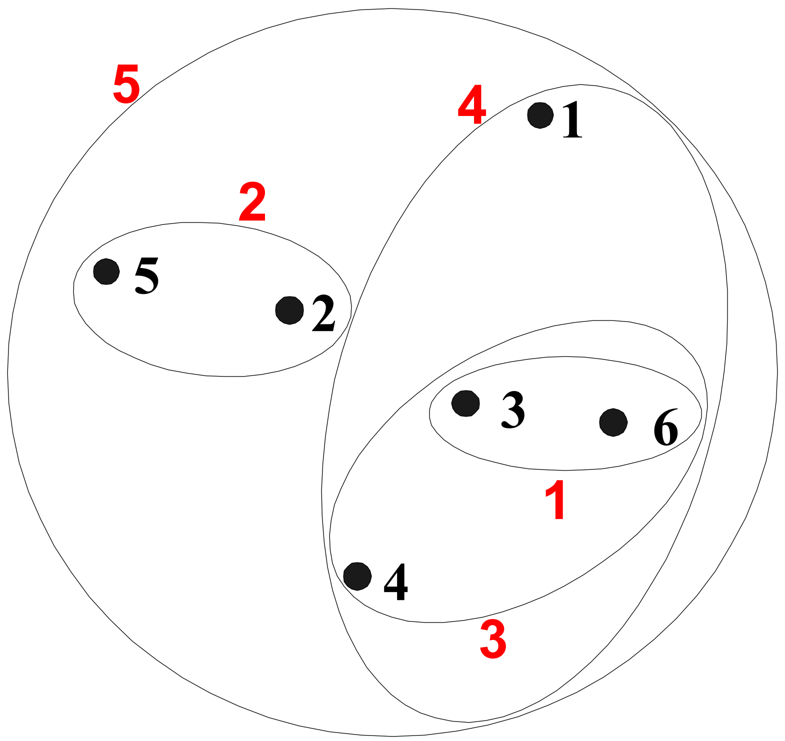
$$\begin{aligned}
dist(\{3,6,4\},\{1\}) &= (0.22 + 0.37 + 0.23)/(3 \times 1) \\
&= 0.28 \\
dist(\{2,5\},\{1\}) &= (0.24 + 0.34)/(2 \times 1) \\
&= 0.29 \\
dist(\{3,6,4\},\{2,5\}) &= (0.15 + 0.28 + 0.25 + 0.39 + 0.20 + 0.29)/(3 \times 2) \\
&= 0.26
\end{aligned}$$

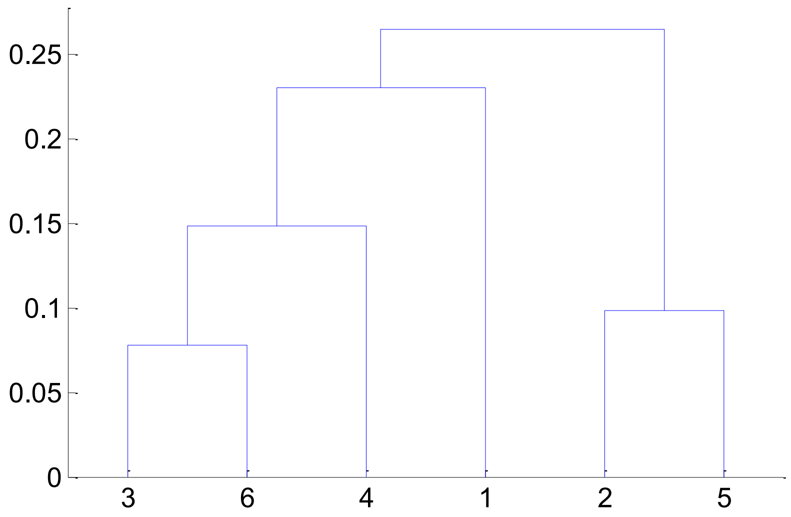


	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Distance Matrix:

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

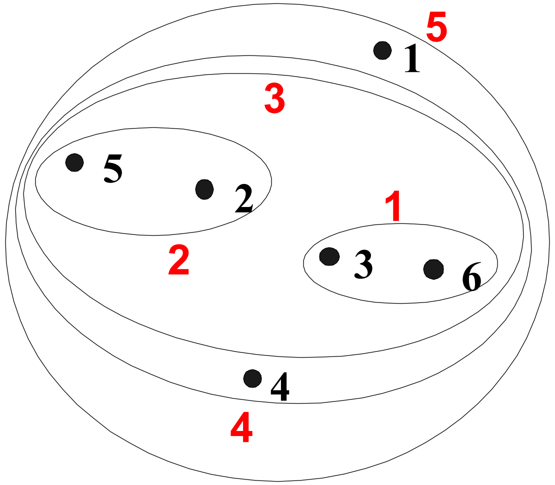
Less susceptible to noise

Biased towards globular clusters

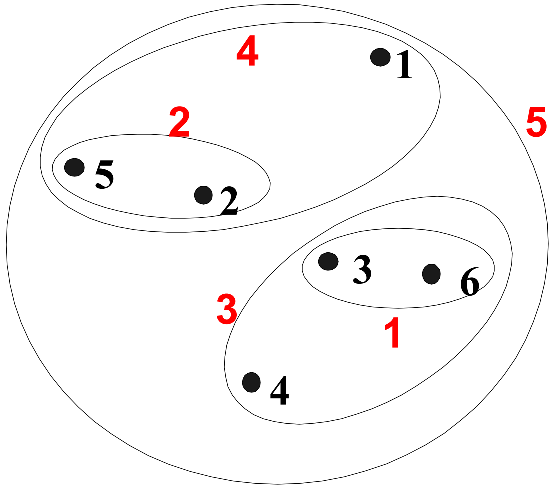
Hierarchical analogue of K-means

- Can be used to initialize K-means

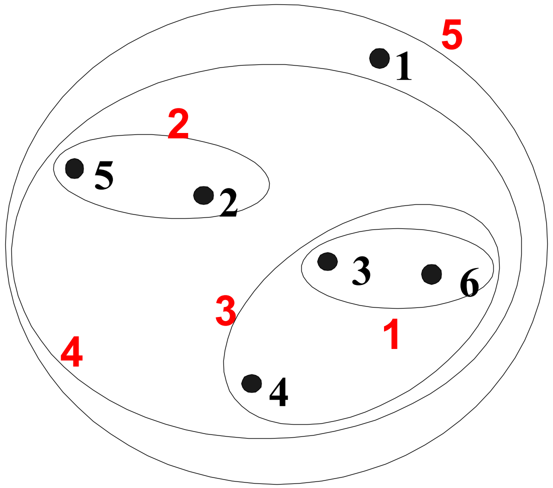
Hierarchical Clustering: Comparison



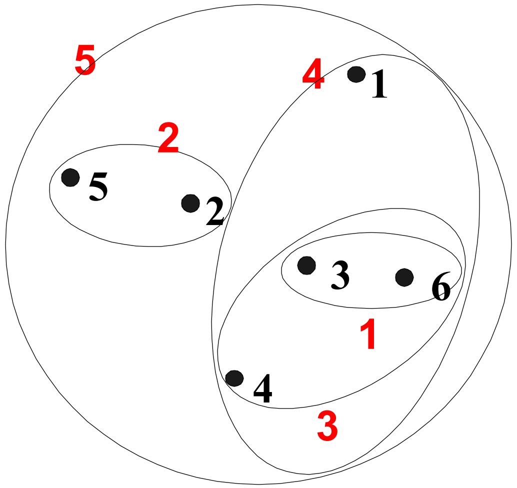
MIN



MAX



Group Average



Ward's Method

Hierarchical Clustering: Problems and Limitations

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

- K-means to create several small clusters -> Hierarchical clustering

No global objective function is directly minimized

Different schemes have problems with one or more of the following:

- Sensitivity to noise -> Outliers increase SSE in Ward's Rule
- 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 is a simple and effective density-based clustering algorithm

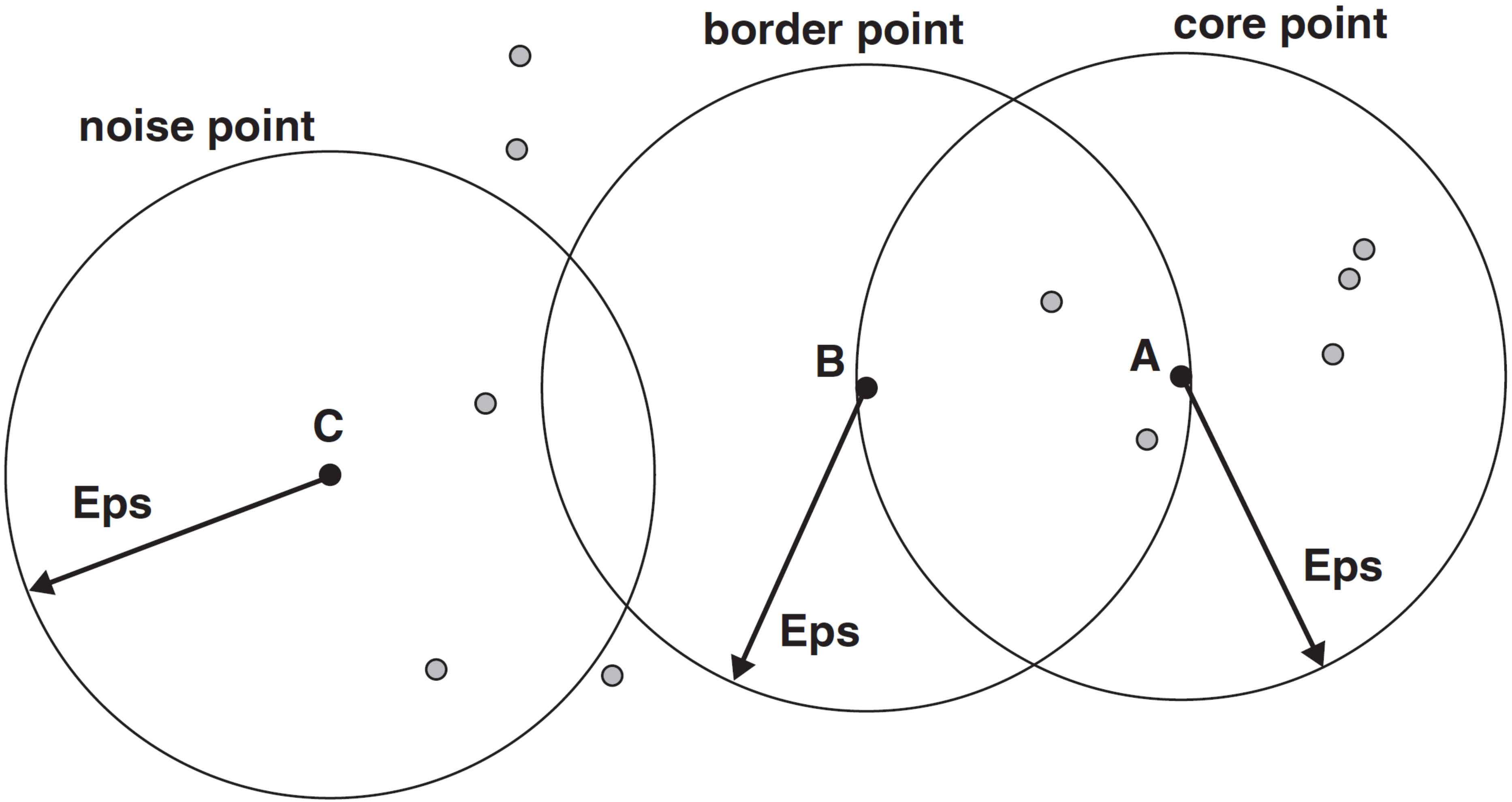
DBSCAN

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 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.

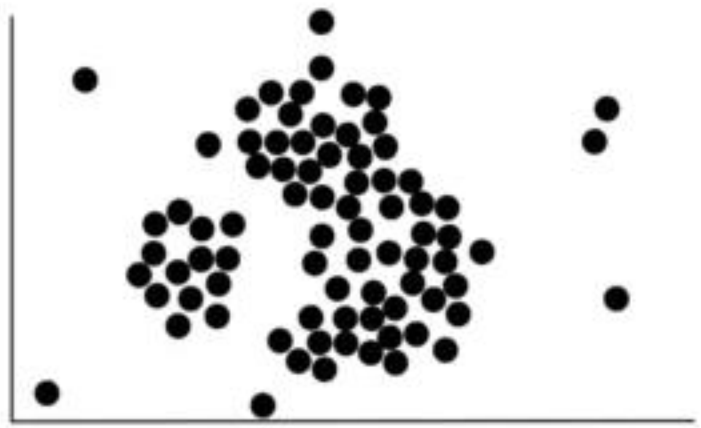
5: Assign each border point to one of the clusters of its associated core points

A simple DBSCAN example

Let's start with a set of points and define eps and minPoints

eps =

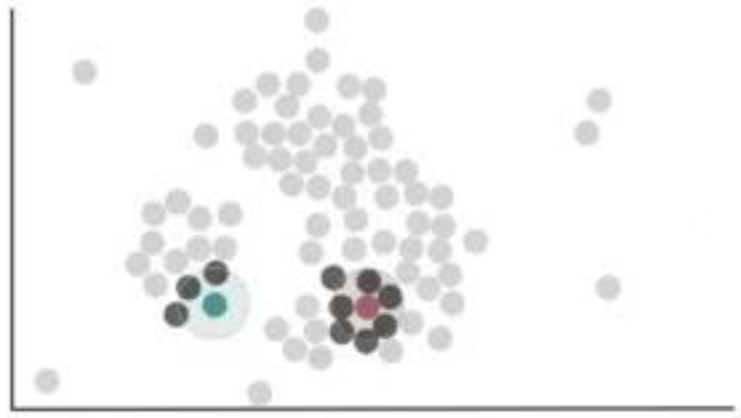
minPoints = 4



The first step is to find all the core points

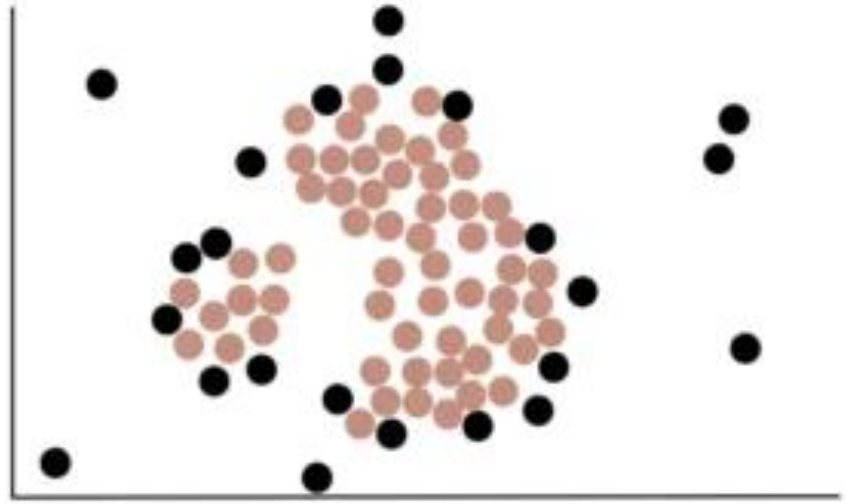
The red point is a core point (close to 7 points)

The green point is not a core point



A simple DBSCAN example

This way we can finally find all the core points

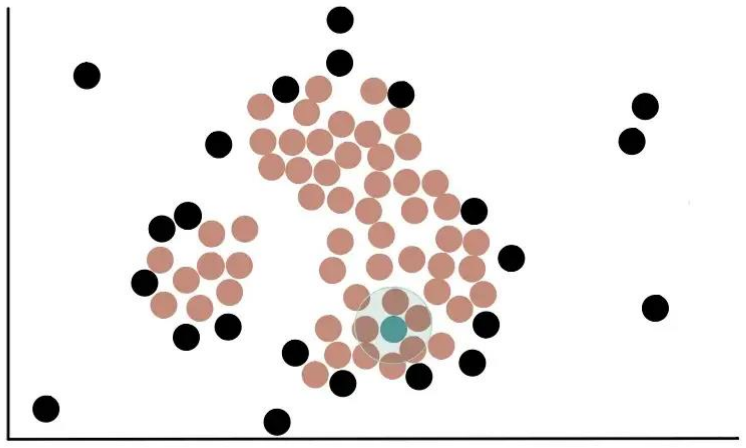


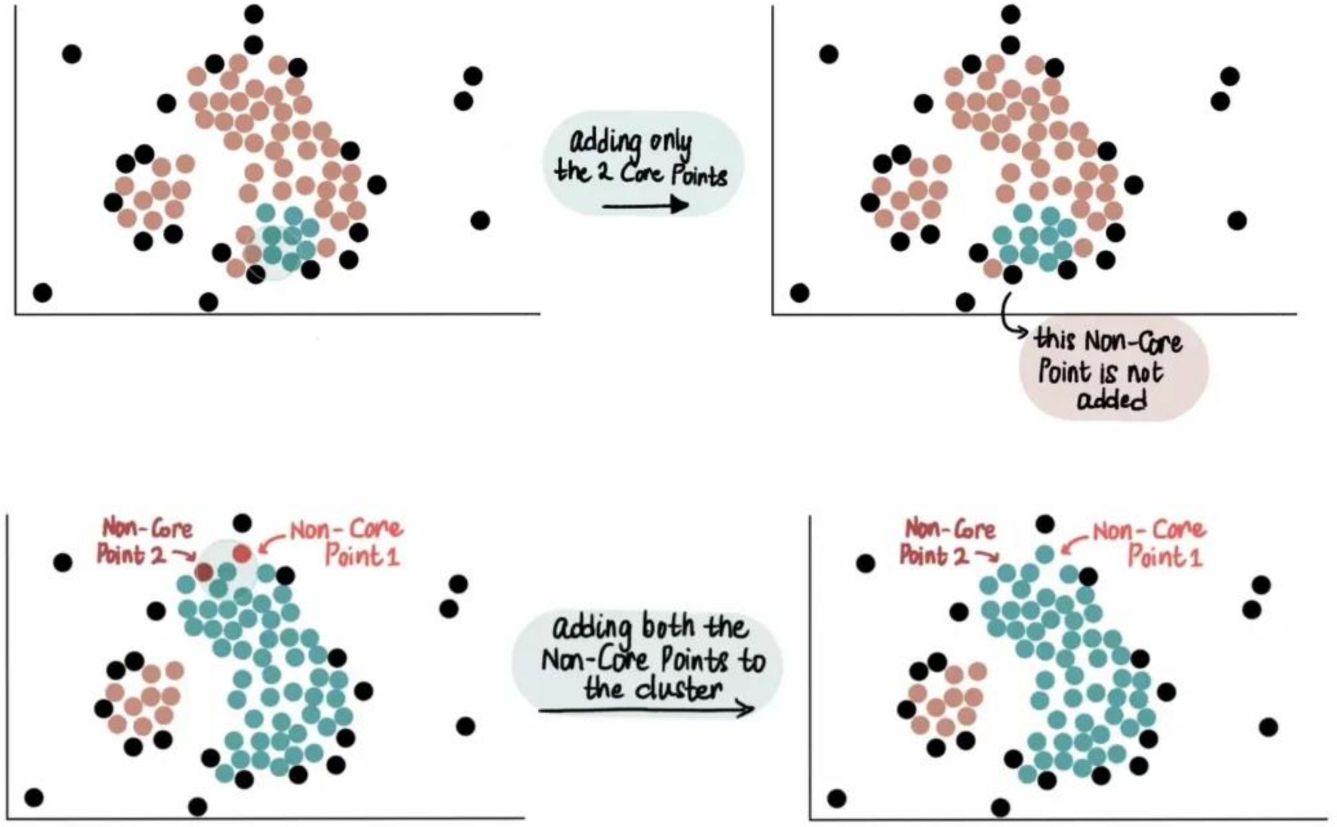
A simple DBSCAN example

Select a core point to ‘start’ a new cluster

Iteratively add core points within the eps of another core point to the cluster

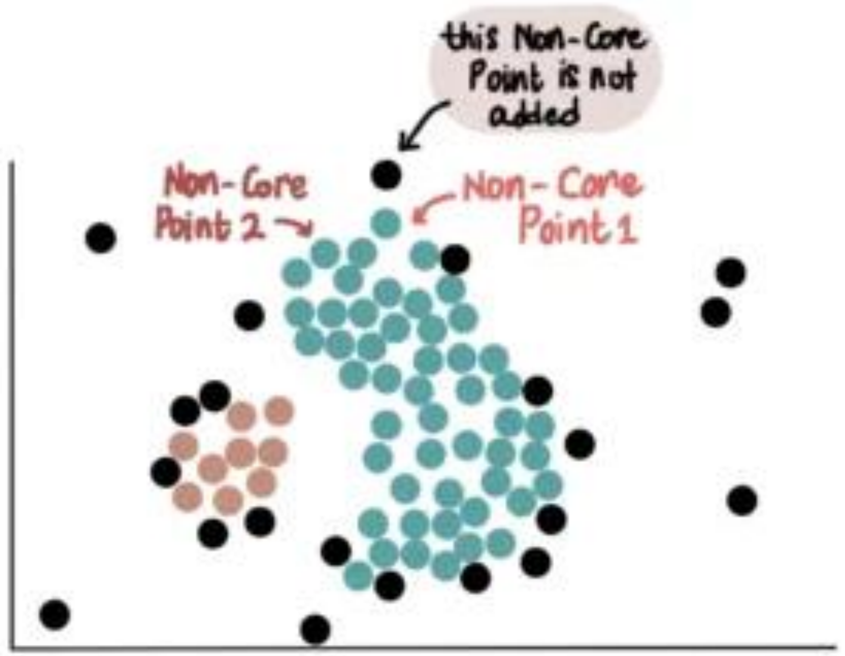
Finally, add the border points





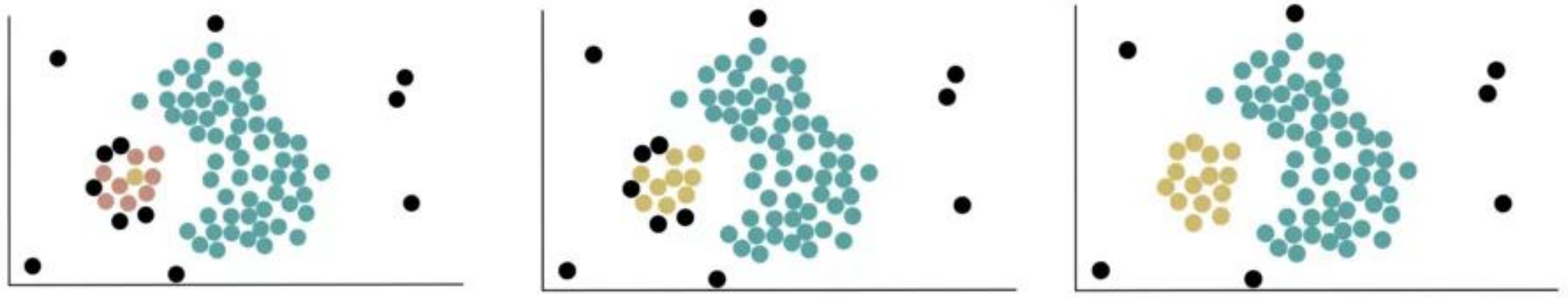
A simple DBSCAN example

Noise points are not added



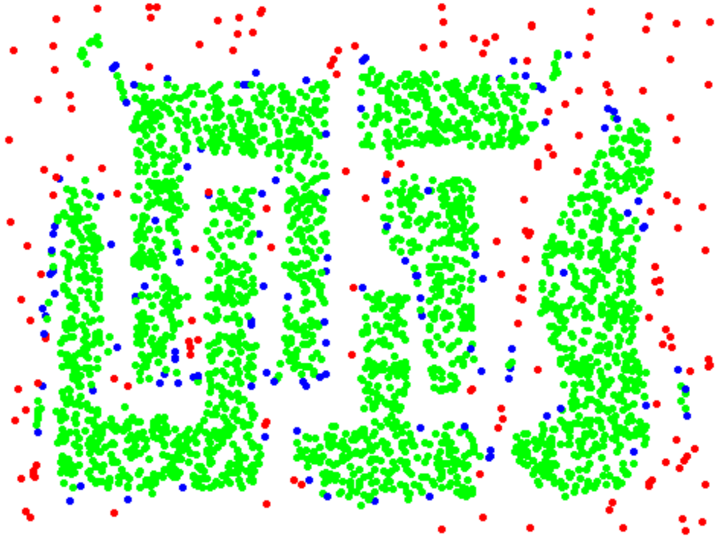
A simple DBSCAN example

Finally, we can repeat the procedure to generate other clusters



When DBSCAN Works Well

Original Points



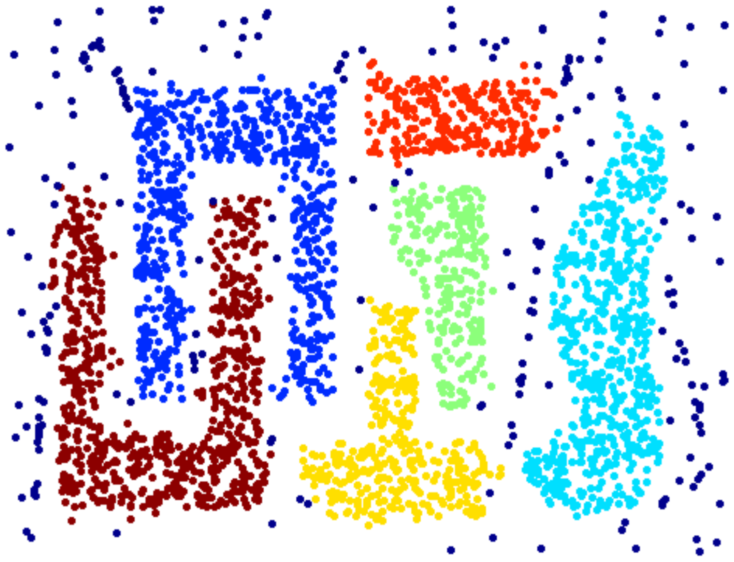
border and **noise**

Eps = 10, MinPts = 4

When DBSCAN Works Well

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

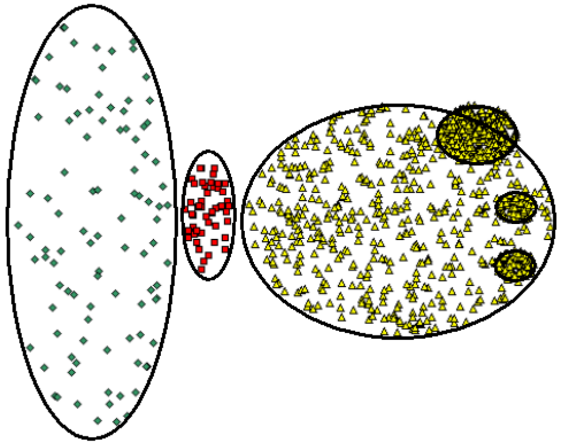
Original Points



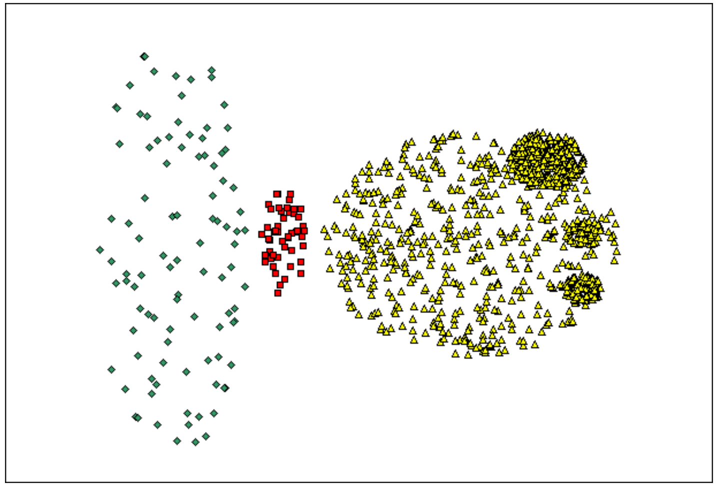
Clusters (dark blue points indicate noise)

When DBSCAN Does NOT Work Well

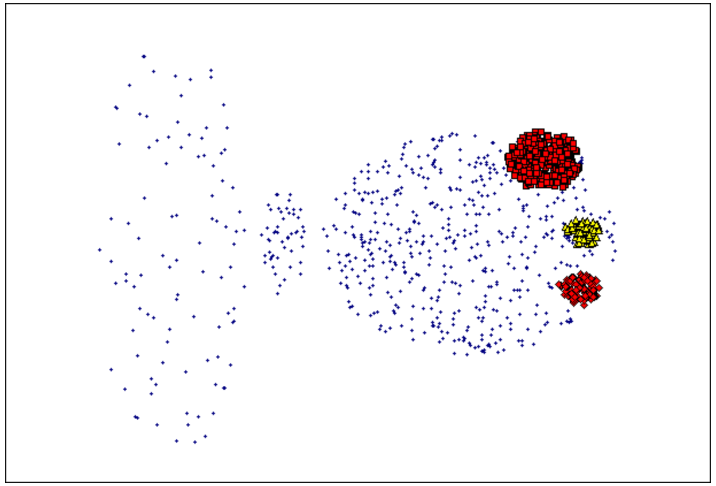
- **Varying densities**
- **High-dimensional data**



Original Points



(MinPts=4, Eps=9.92).



(MinPts=4, Eps=9.75)

DBSCAN: Determining EPS and MinPts

The 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 a farther distance

So, plot the sorted distance of every point to its k^{th} nearest neighbor