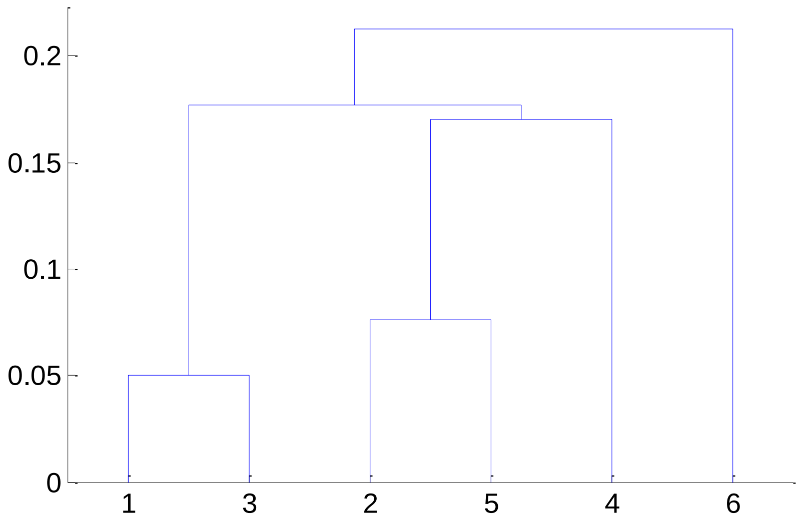
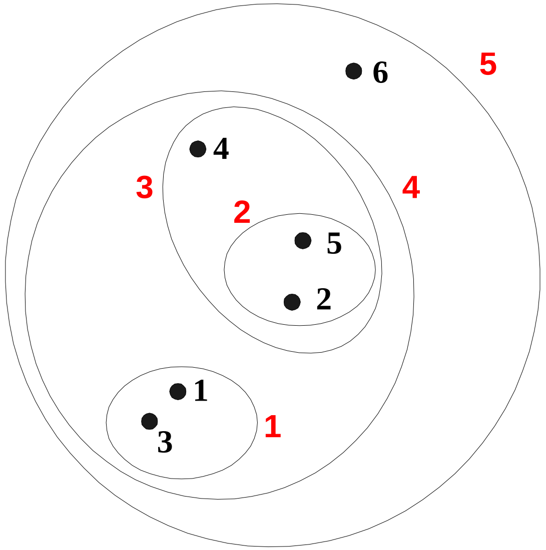
Cluster Analysis

Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree





Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
  - ‘cut’ the dendogram at the proper level to have a certain number of clusters

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

Hierarchical Clustering

Algorithm

Let each data point be a cluster

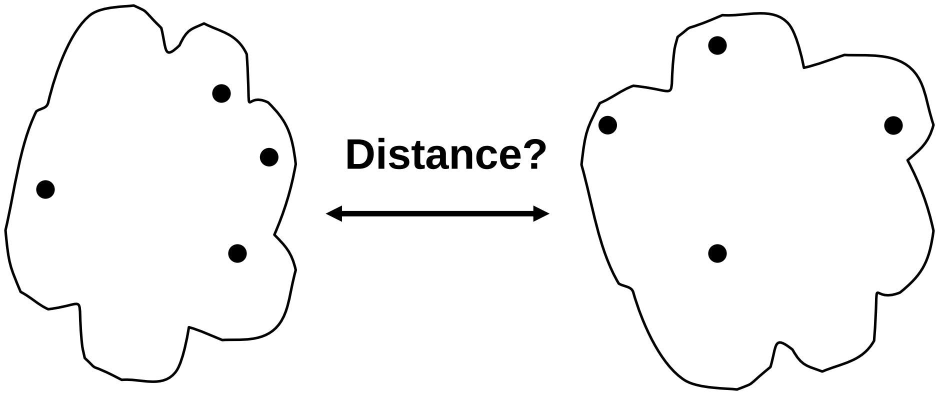
Repeat

  Merge **the two closest** clusters

Until only a single cluster remains

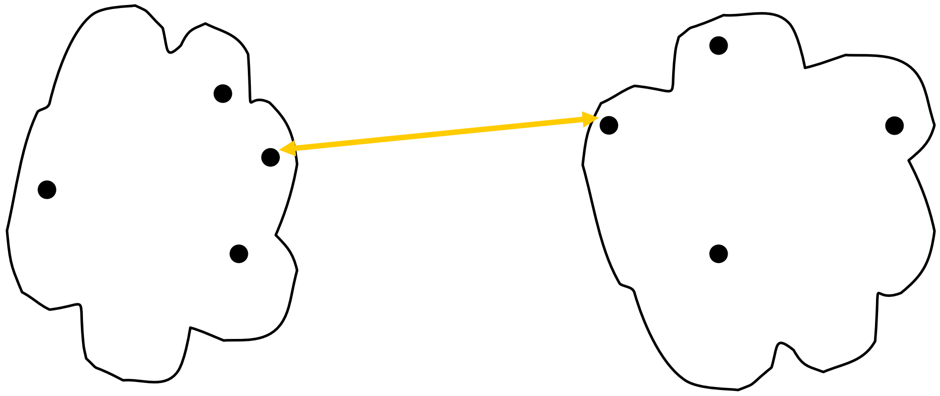
- Key operation is the computation of the proximity of two clusters.

First Define Inter-Cluster Similarity



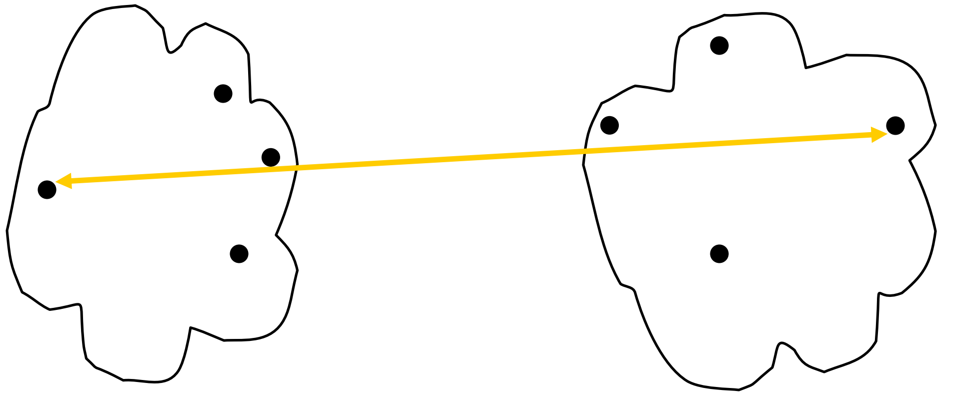
- MIN
- MAX
- Group Average

How to Define Inter-Cluster Similarity



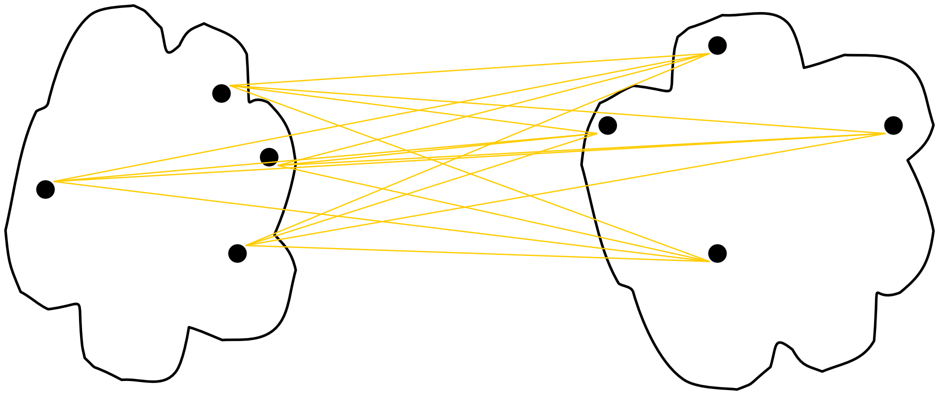
- MIN
- MAX
- Group Average

How to Define Inter-Cluster Similarity



- MIN
- MAX
- Group Average

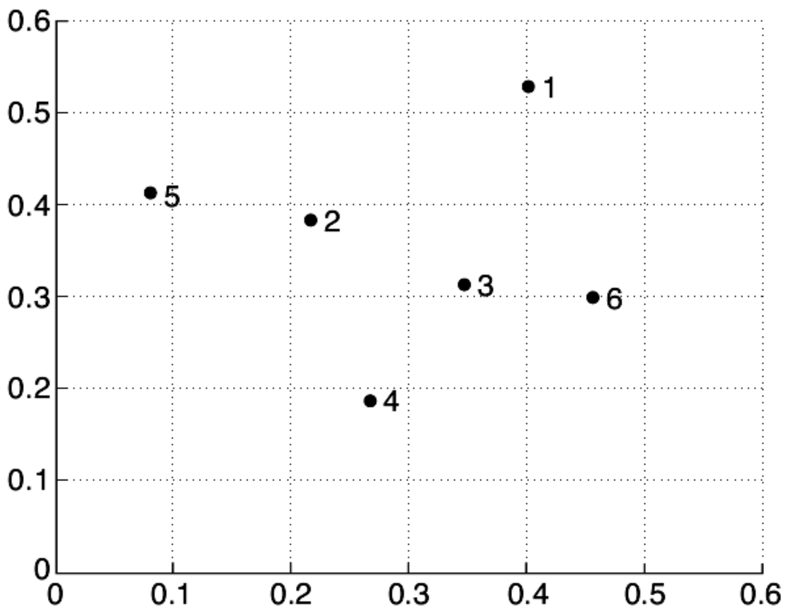
How to Define Inter-Cluster Similarity



- MIN
- MAX
- Group Average

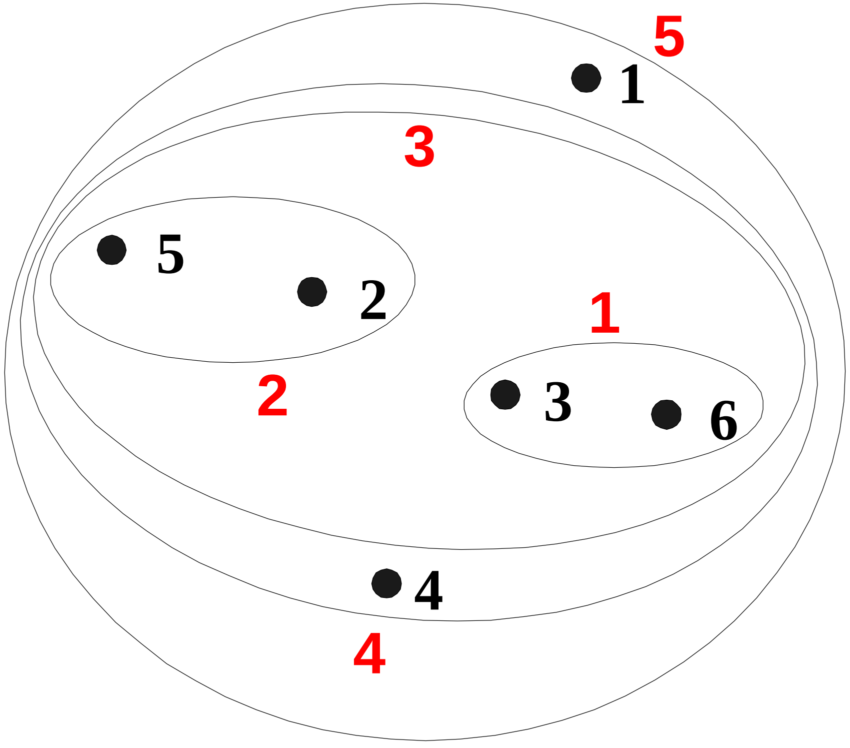
Cluster Similarity: MIN

- Similarity of two clusters is based on the two most similar (closest) points in the different clusters
  - Determined by one pair of points

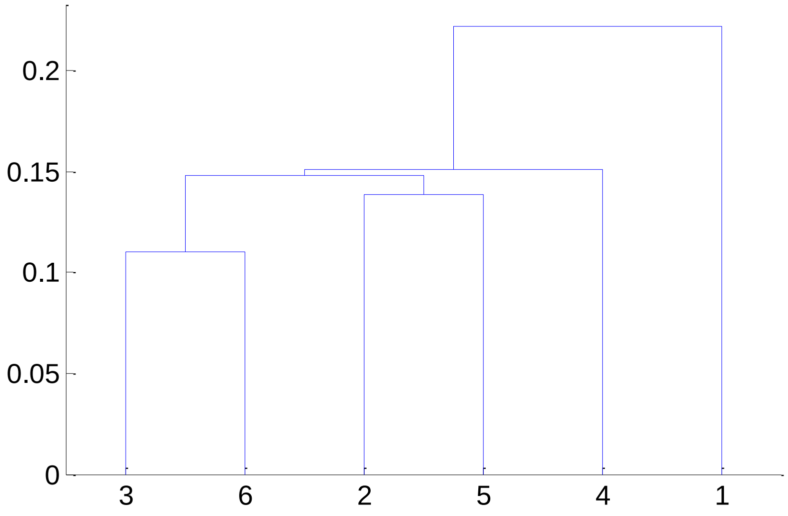


	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Hierarchical Clustering: MIN

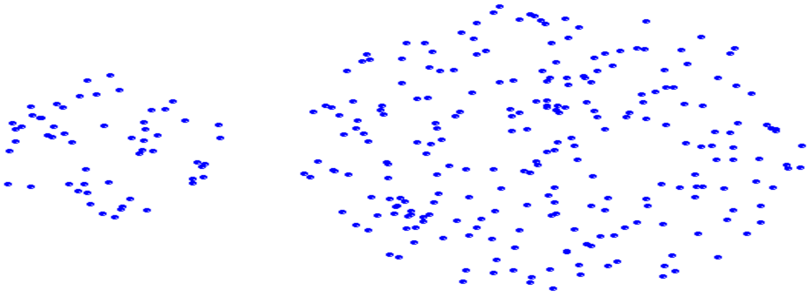


Nested Clusters

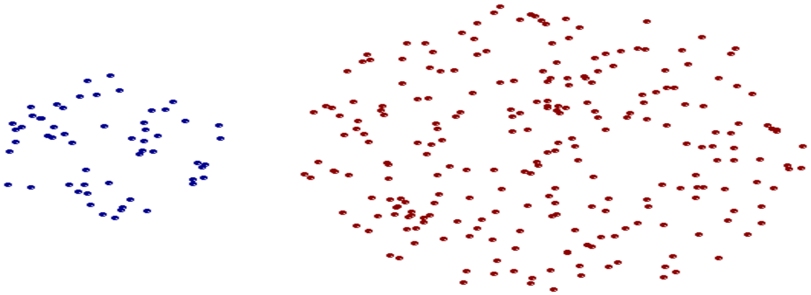


Dendrogram

Strength of MIN



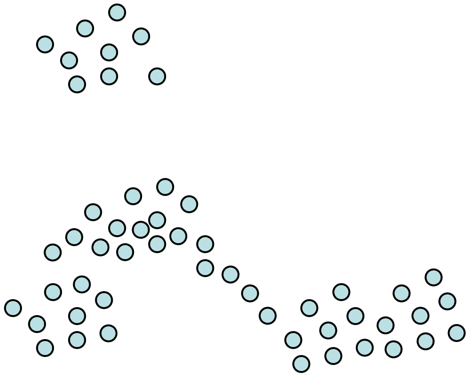
Original Points



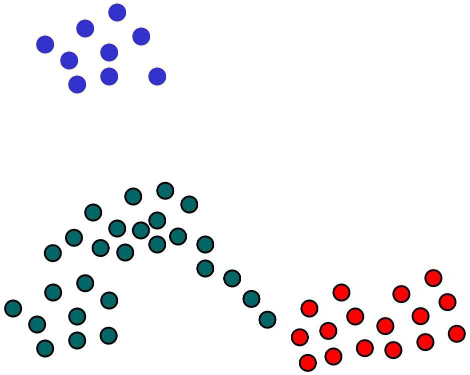
Two Clusters

Can handle non-globular shapes

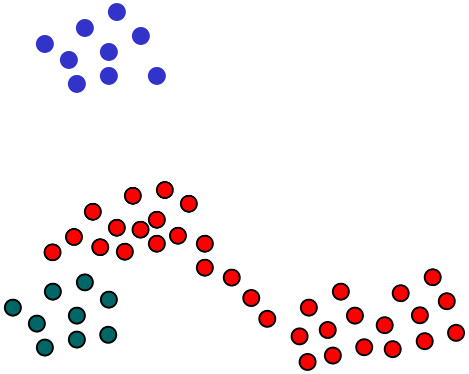
Limitations of MIN



Original Points



Real clusters



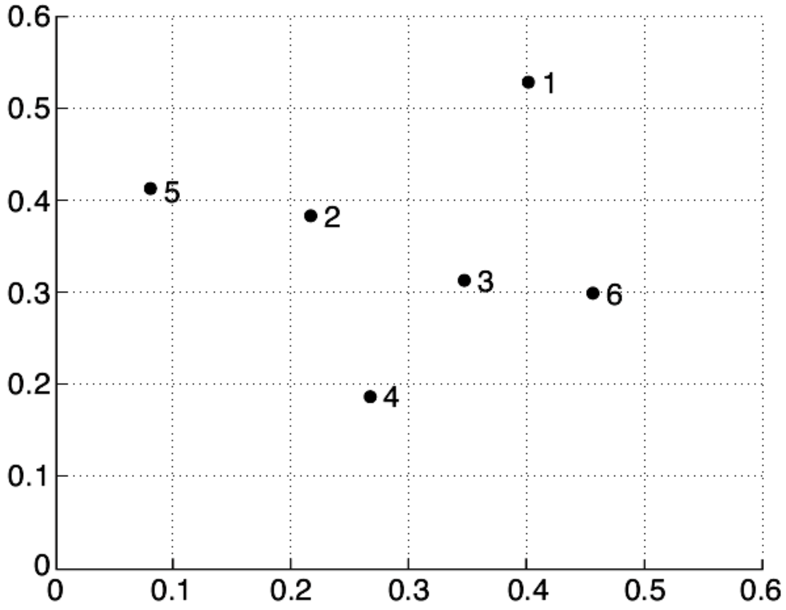
Three clusters computed by MIN:

The green points got wrongly merged with the red ones, as opposed to the green one.

Sensitive to noise and outliers

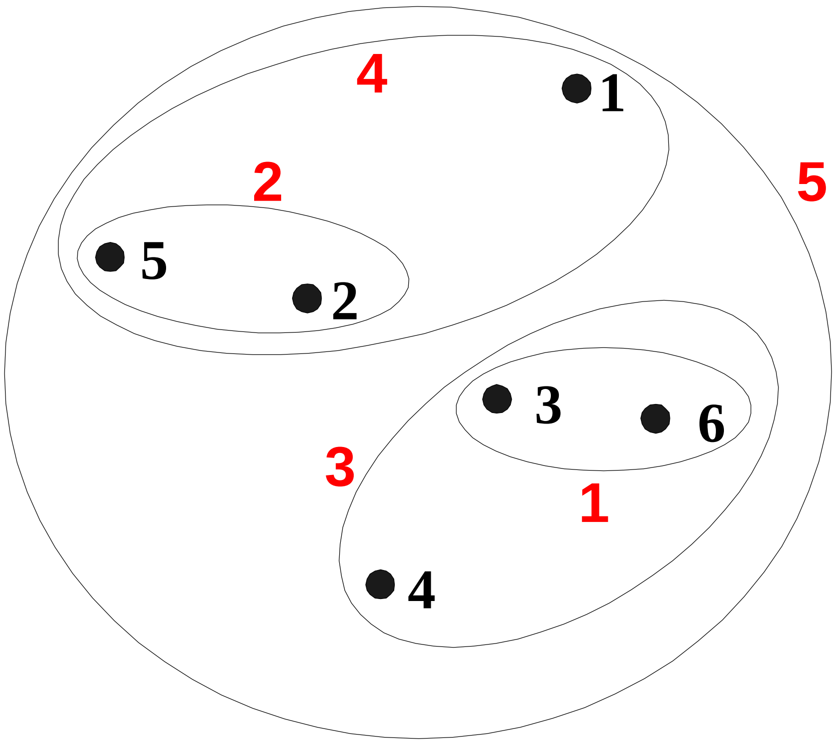
Cluster Similarity: MAX

- 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

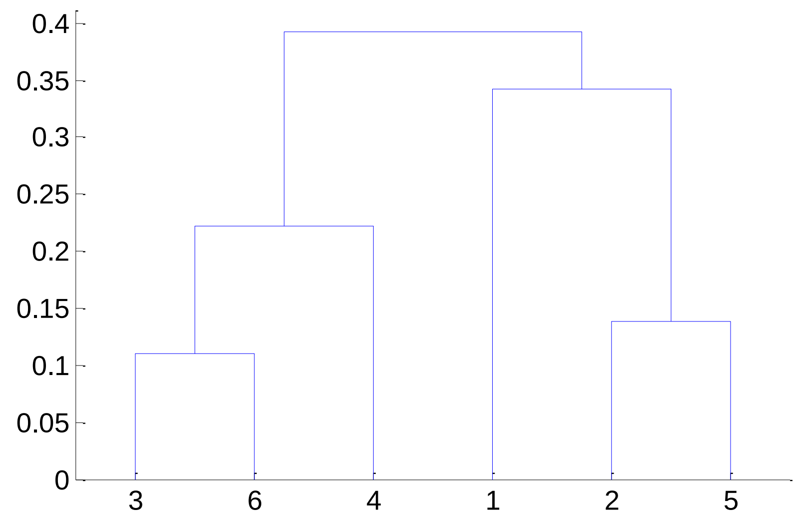


	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Hierarchical Clustering: MAX

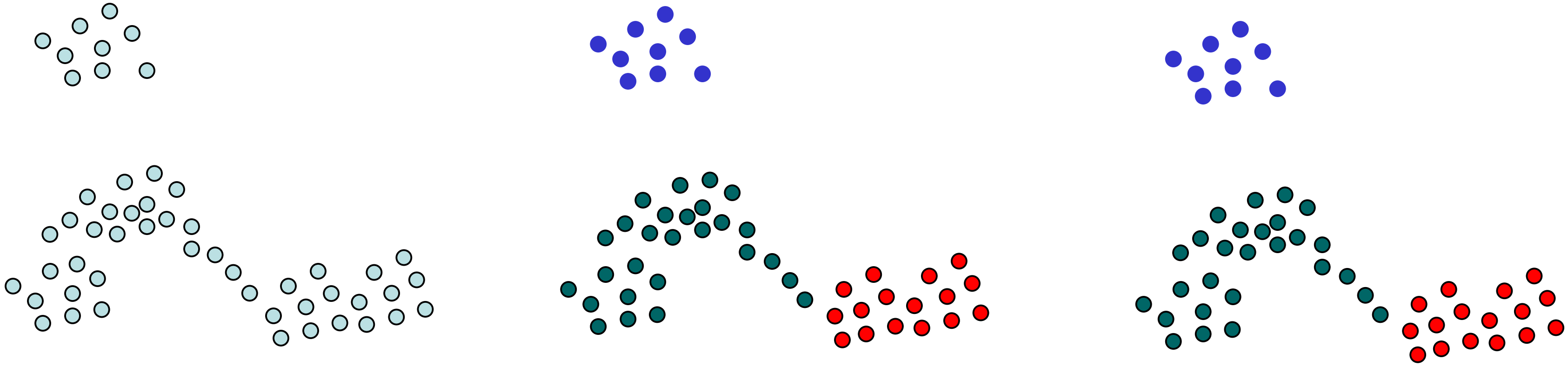


Nested Clusters



Dendrogram

Strengths of MAX



Original Points

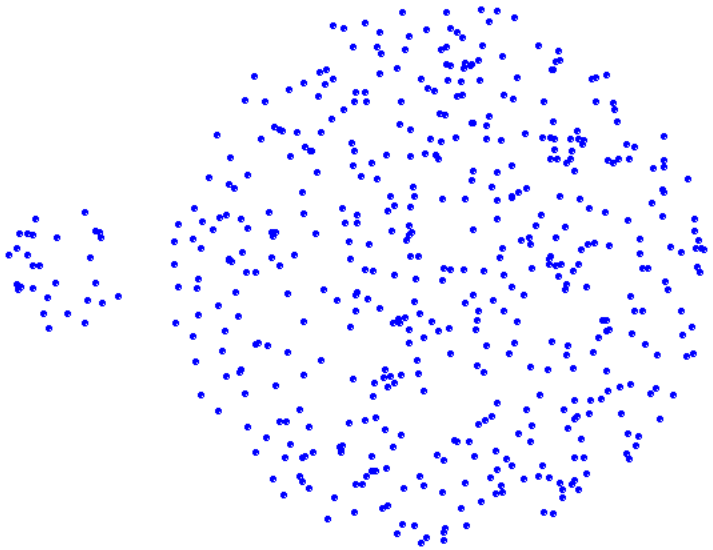
Real clusters

Three clusters computed by MAX:

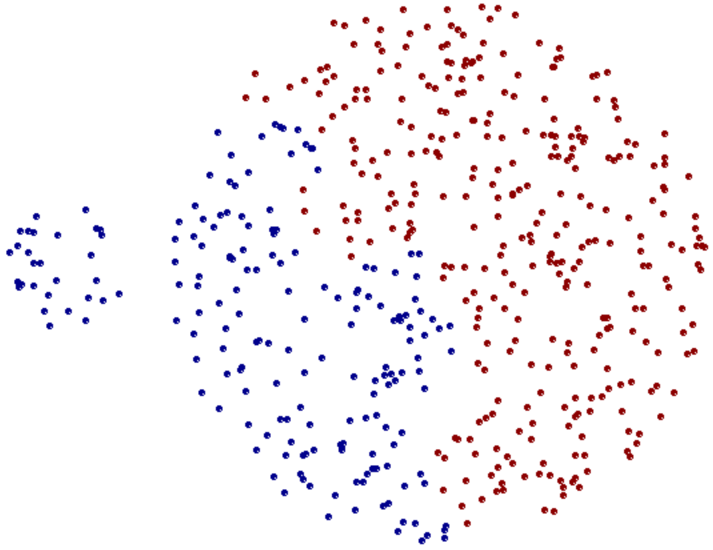
The upper green points get now merged with the other green one.

Less susceptible with respect to noise and outliers

Limitations of MAX



Original Points



Two Clusters

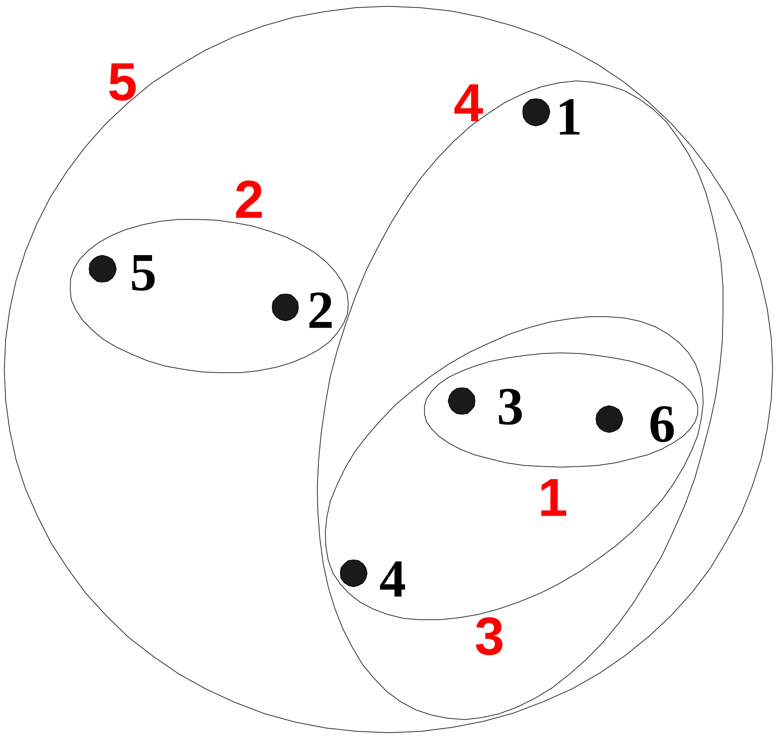
Tends to break large clusters

Cluster Similarity: Group Average

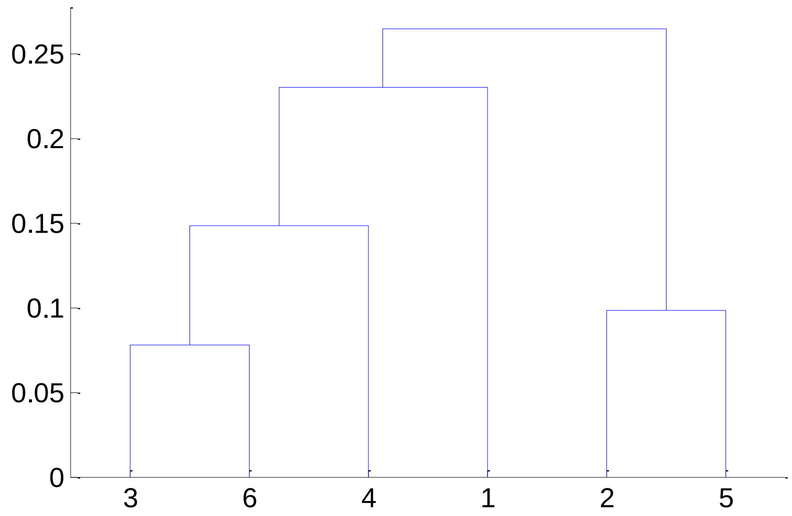
- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

$$\mathbf{proximity}(\mathbf{Cluster}_i, \mathbf{Cluster}_j) = \frac{\sum\limits_{\substack{p_i \in \mathbf{Cluster}_i \\ p_j \in \mathbf{Cluster}_j}} \mathbf{proximity}(p_i, p_j)}{|\mathbf{Cluster}_i| * |\mathbf{Cluster}_j|}$$

Hierarchical Clustering: Group Average



Nested Clusters



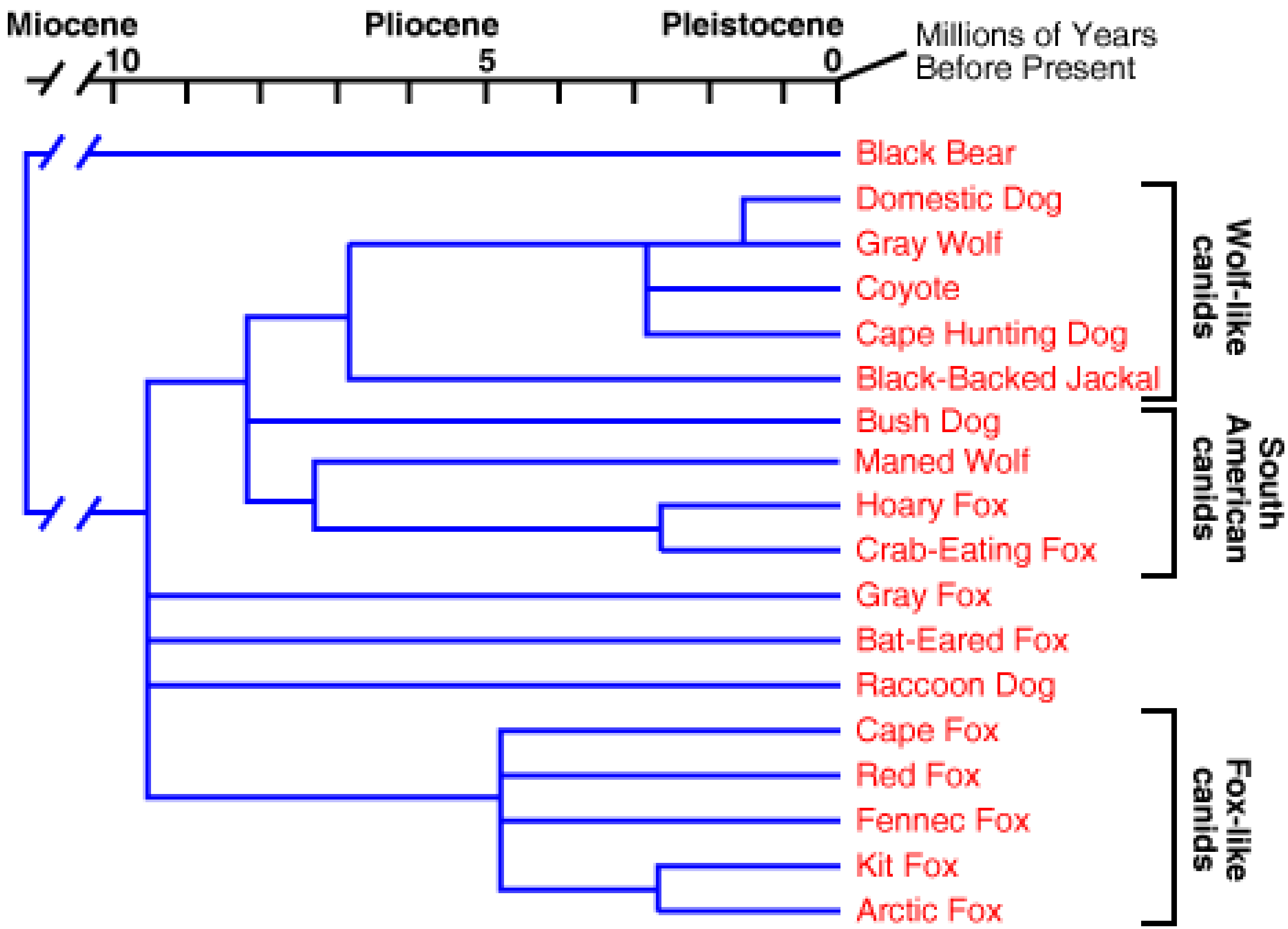
Dendrogram

Hierarchical Clustering: Time and Space

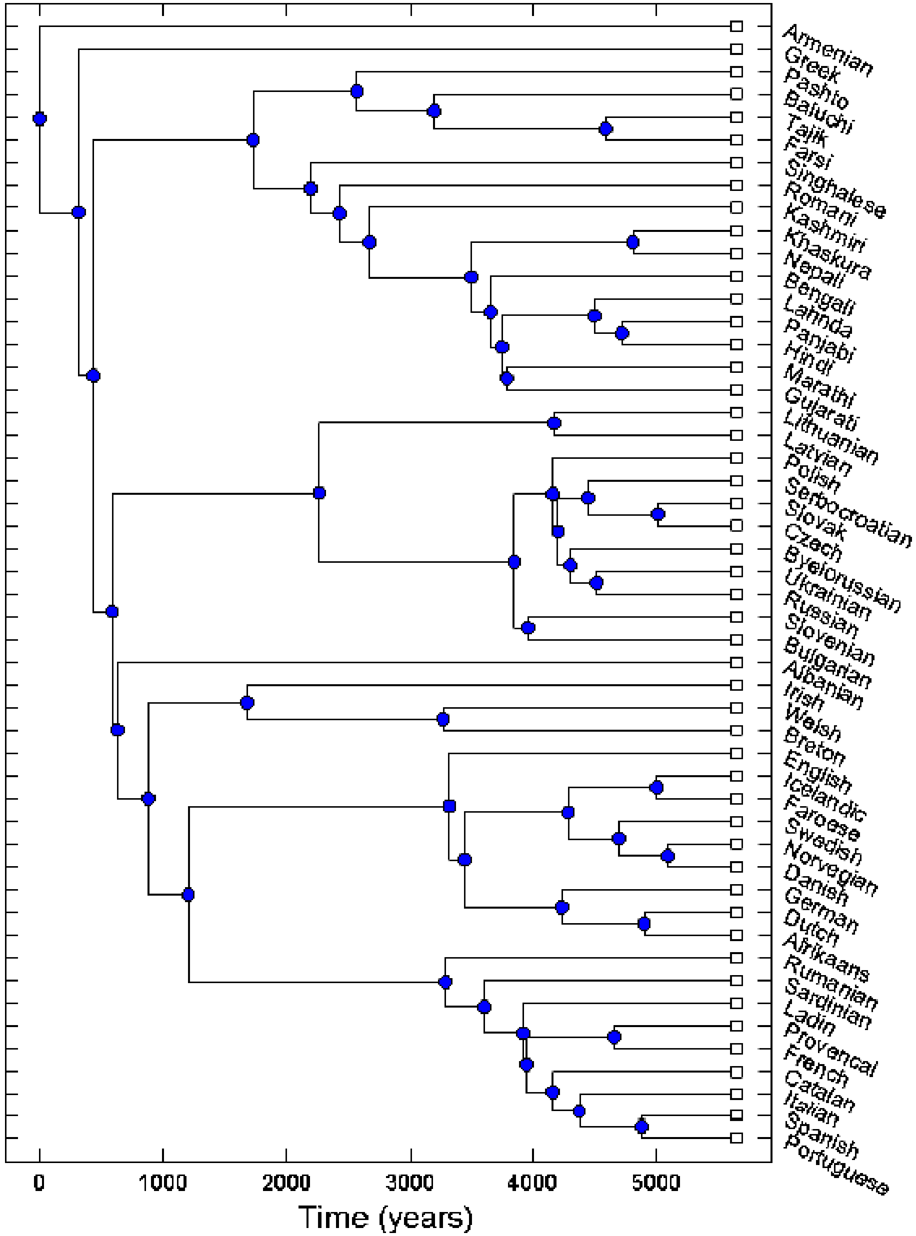
- $O(N^2)$ **space** since it uses the proximity matrix.
  - N is the number of points.

- $O(N^3)$ **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 Example



Hierarchical Clustering Example



From
“Indo-European languages tree by Levenshtein distance”
by M. Serva1 and F. Petroni

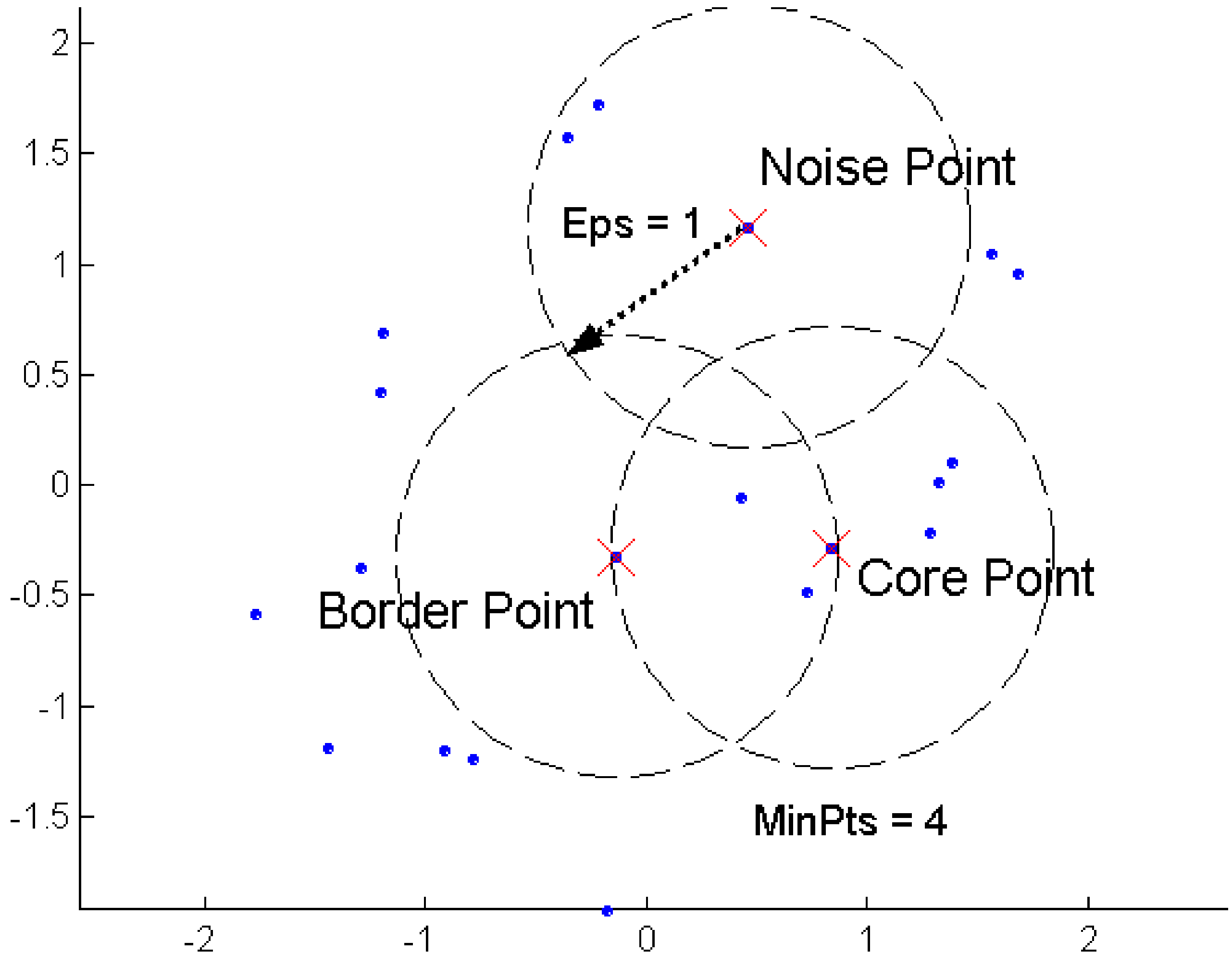
DBSCAN

DBSCAN is a density-based algorithm.

Locates regions of **high density** that are separated from one another by regions of **low density**.

- **Density** = number of points within a specified radius (Eps)
- A point is a core point if it has more than a specified number of points (MinPts) within Eps
  - These are points that are at the interior of a cluster
- A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point
- A noise point is any point that is neither a core point nor a border point.

DBSCAN: Core, Border, and Noise Points

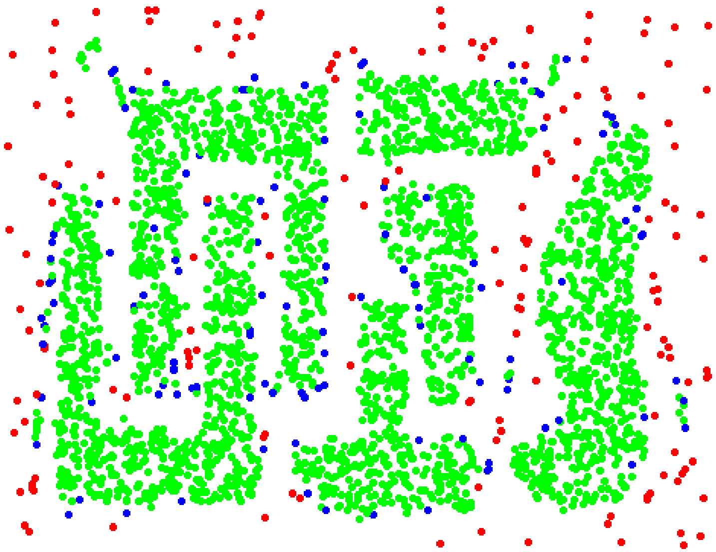


DBSCAN Algorithm

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

DBSCAN: Core, Border and Noise Points

Original Points

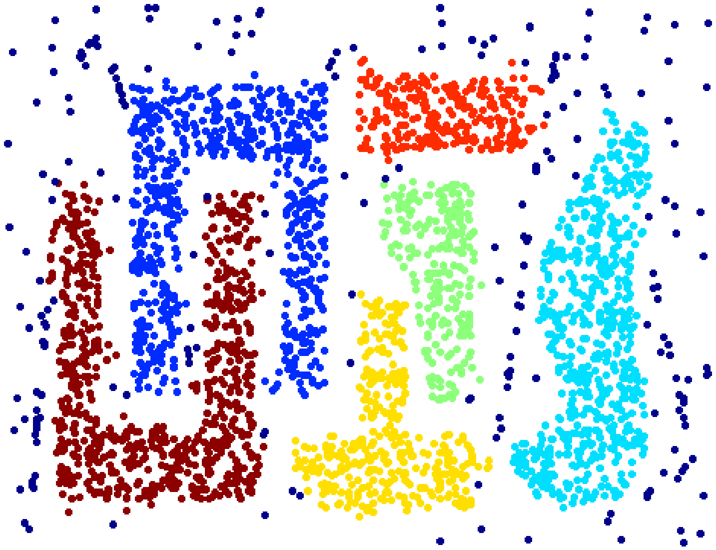


Point types: core, border and noise

Eps = 10, MinPts = 4

When DBSCAN Works Well

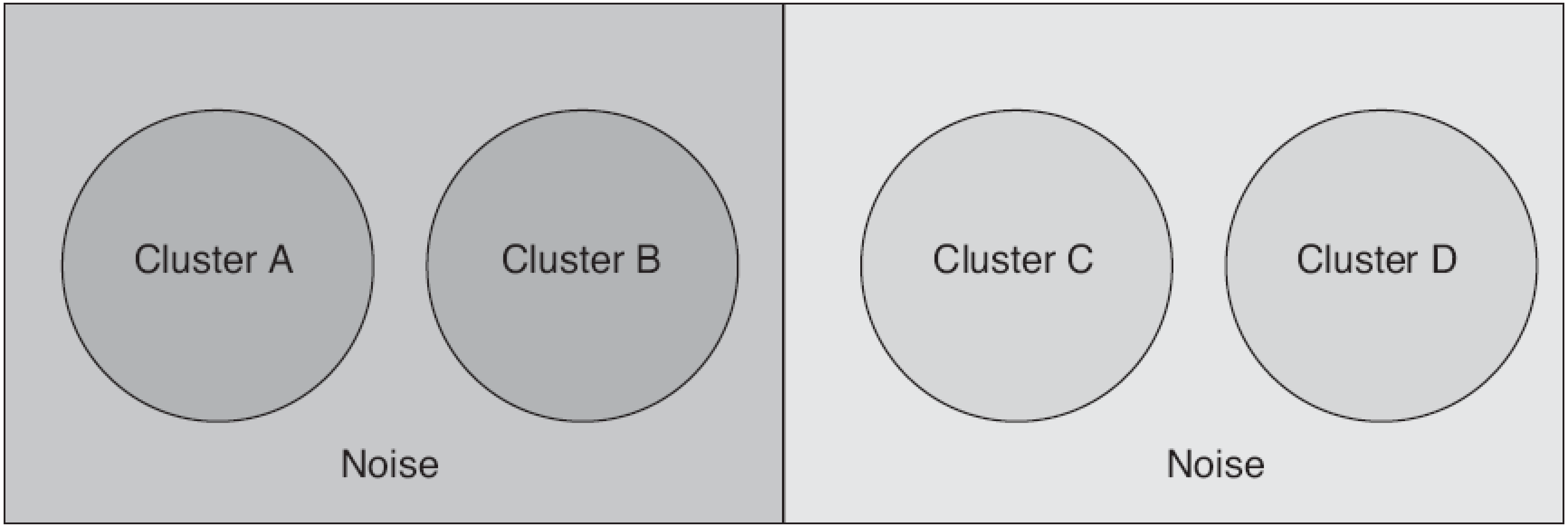
Original Points



Clusters

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

When DBSCAN Does NOT Work Well

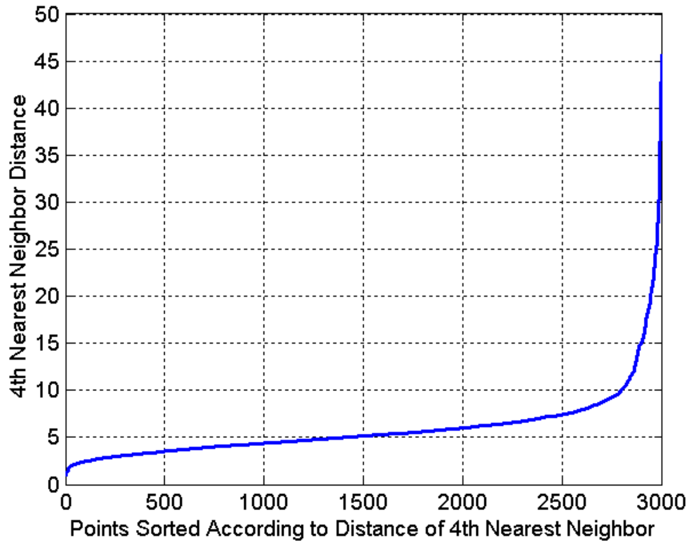


Why DBSCAN doesn’t work well here?

DBSCAN: Determining EPS and MinPts

- Look at the behavior of the distance from a point to its *k*-th nearest neighbor, called the *k*dist.
- For points that belong to some cluster, the value of *k*dist will be small [if *k* is not larger than the cluster size].
- However, for points that are not in a cluster, such as noise points, the *k*dist will be relatively large.
- So, if we compute the *k*dist for all the data points for some *k*, sort them in increasing order, and then plot the sorted values, we expect to see a **sharp change** at the value of *k*dist that corresponds to a suitable value of Eps.
- If we select this distance as the Eps parameter and take the value of *k* as the MinPts parameter, then points for which *k*dist is less than Eps will be labeled as core points, while other points will be labeled as noise or border points.

DBSCAN: Determining EPS and MinPts



- Eps determined in this way depends on *k*, but does not change dramatically as *k* changes.
- If *k* is too small ?

  then even a small number of closely spaced points that are noise or outliers will be incorrectly labeled as clusters.
- If *k* is too large ?

  then small clusters (of size less than *k*) are likely to be labeled as noise.
- Original DBSCAN used *k* = 4, which appears to be a reasonable value for most data sets.