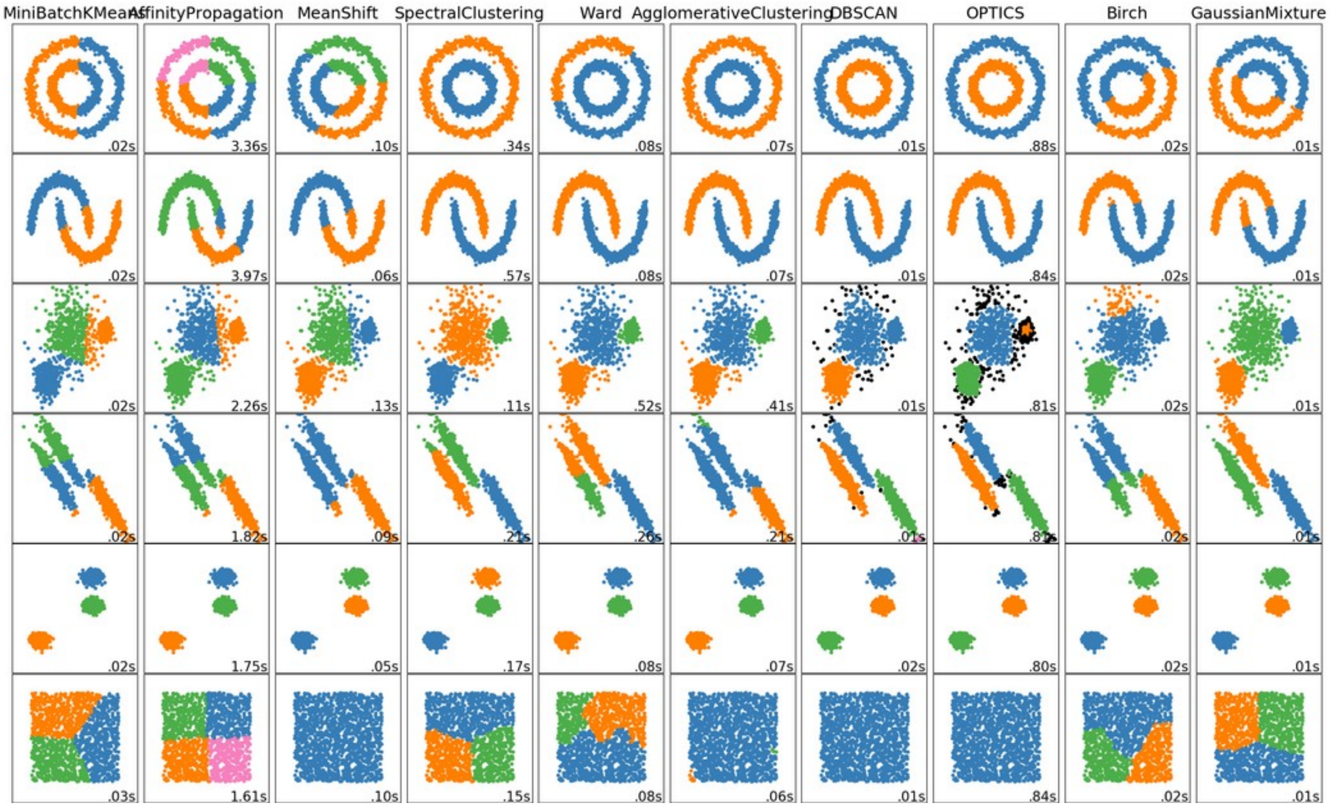
Machine learning
Lecture 9



- Cluster analysis

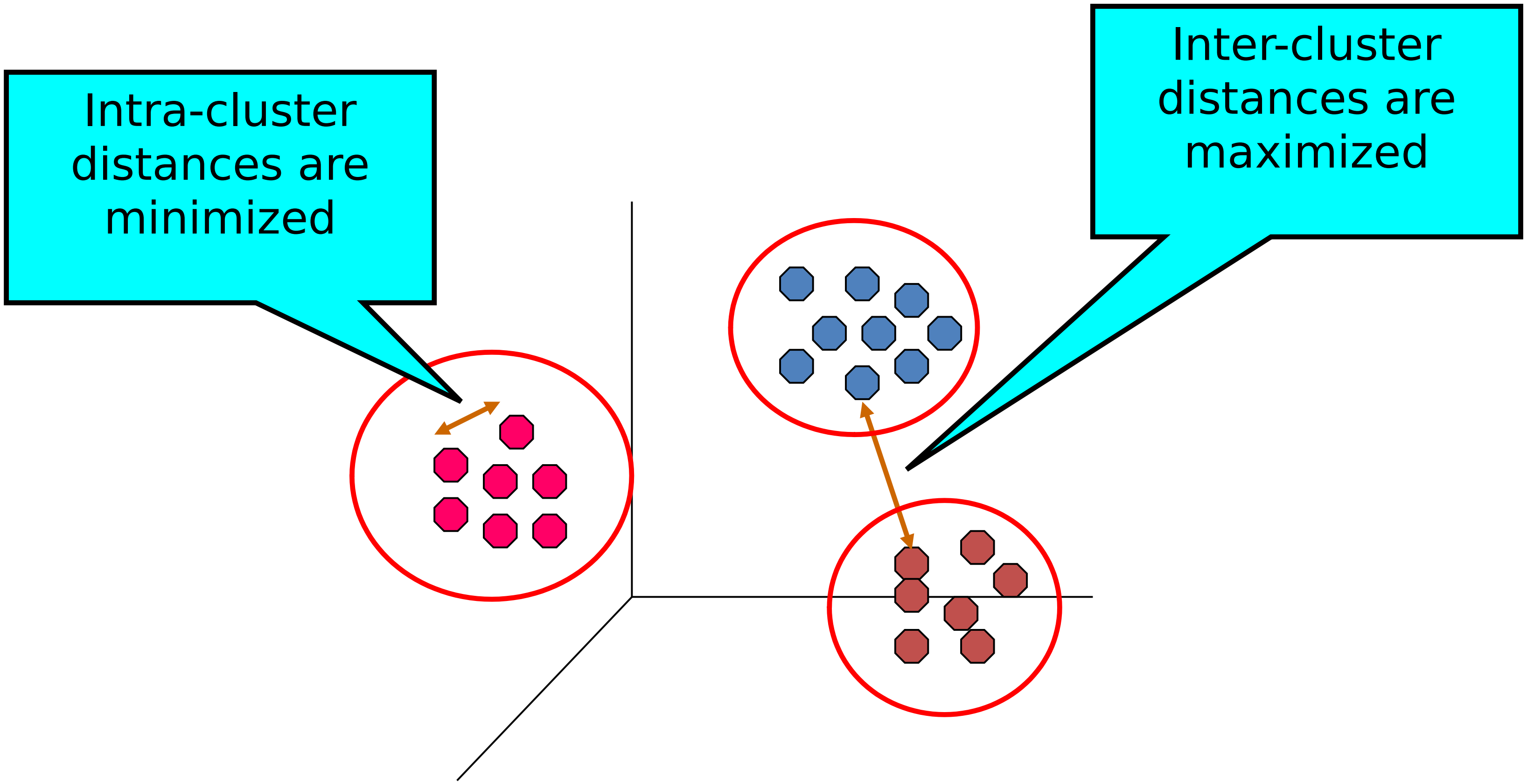
Marcin Wolter

IFJ PAN

16 December 2020

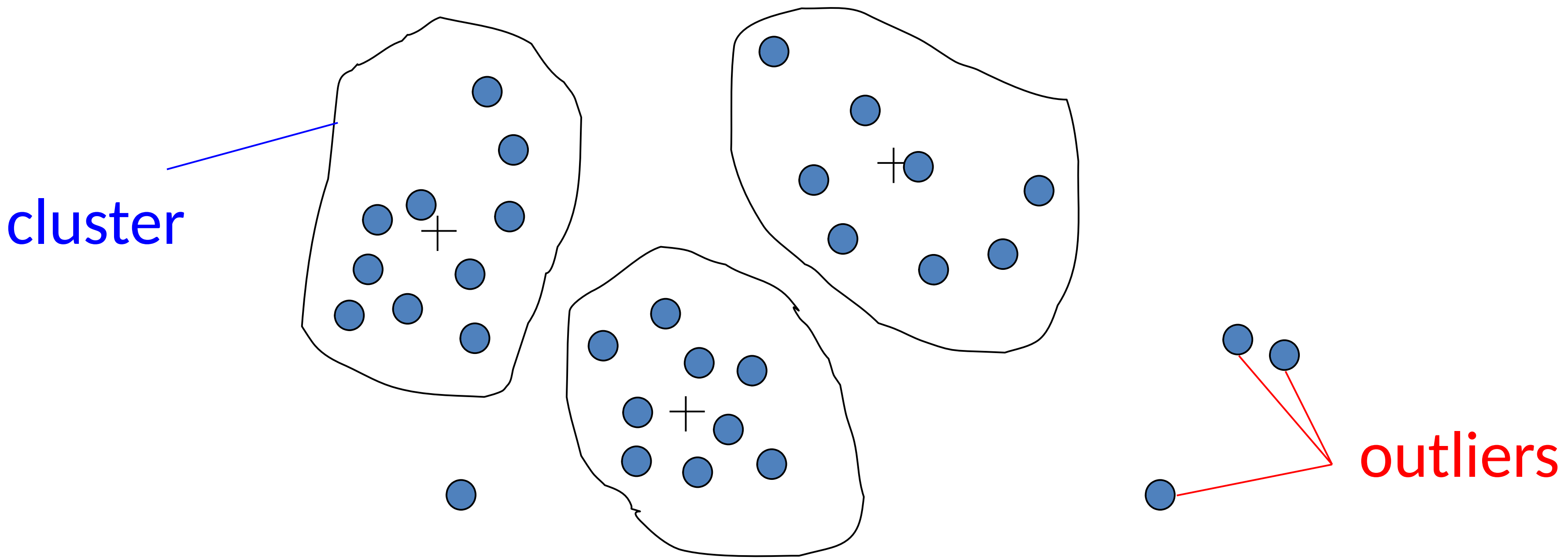
What is clustering?

- A **grouping** of data objects such that the objects **within a group are similar** (or related) to one another **and different from (or unrelated to) the objects in other groups**



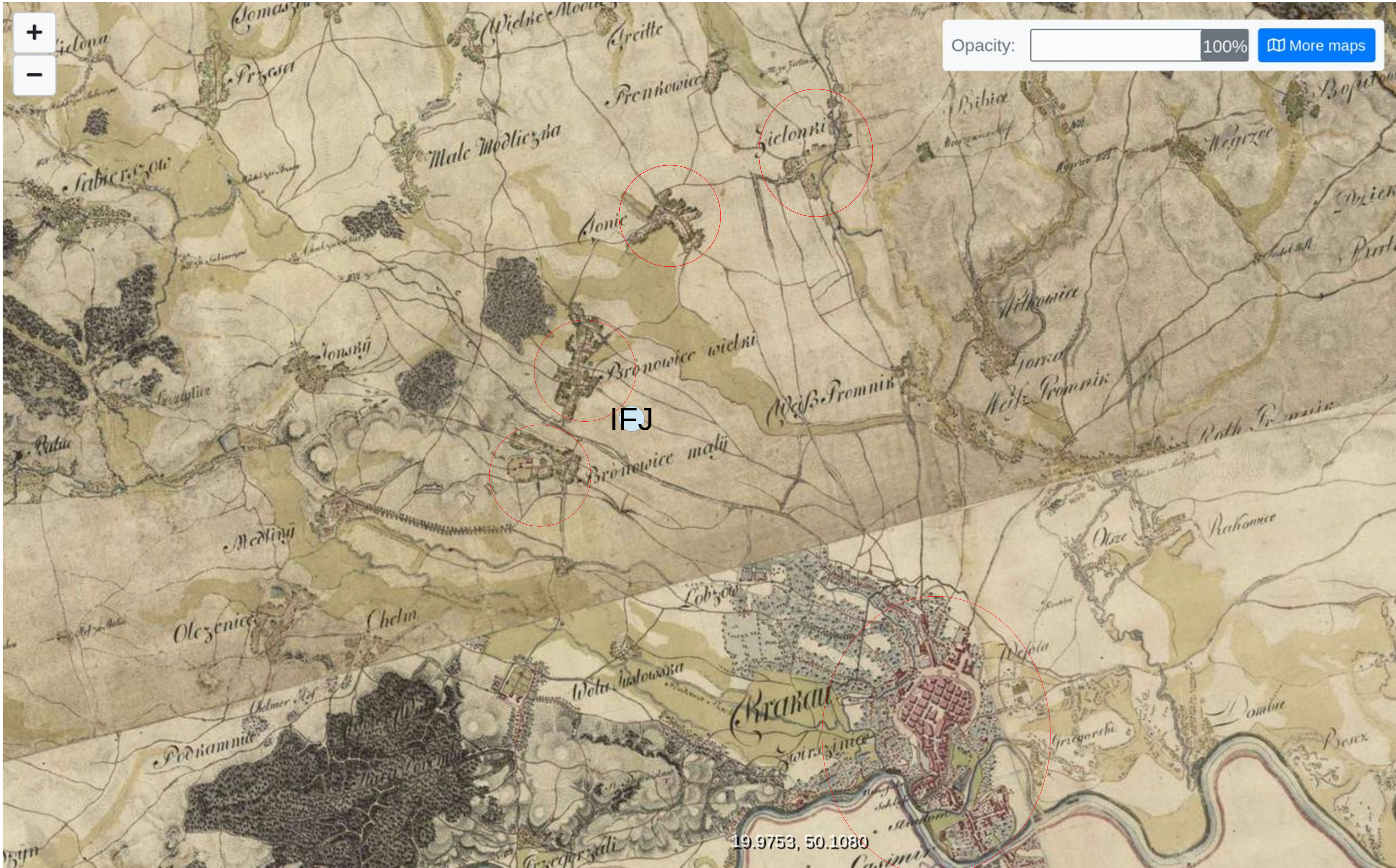
Outliers

- **Outliers** are **objects that do not belong to any cluster** or form clusters of very small cardinality (number of cluster members).



- In some applications we are interested in discovering outliers, not clusters (outlier analysis)

Kraków 1804



Opacity:
100%
More maps
IFJ
Bronowice wielki
Bronowice mały
Krakau
Lobzow
Weiß Promnik
19.9753, 50.1080

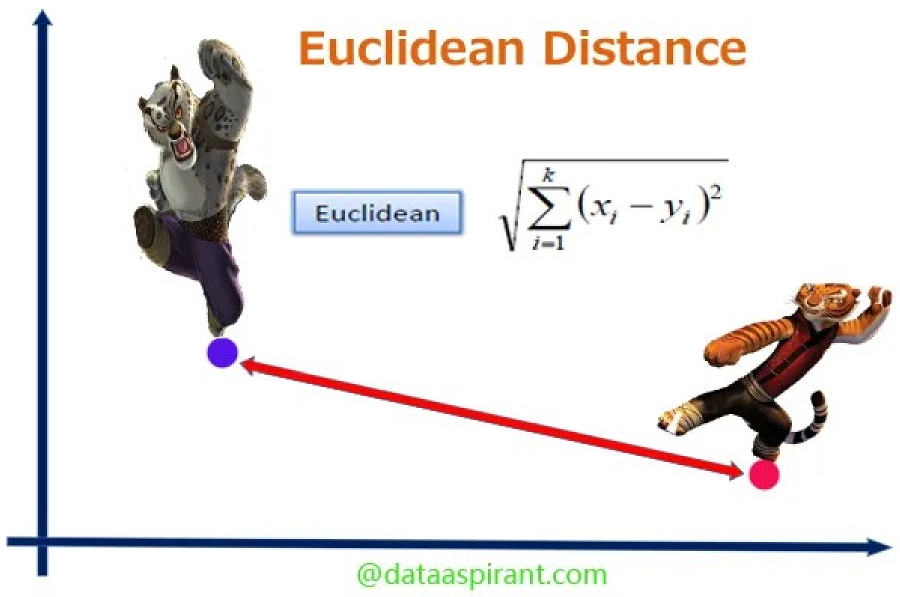
The clustering task

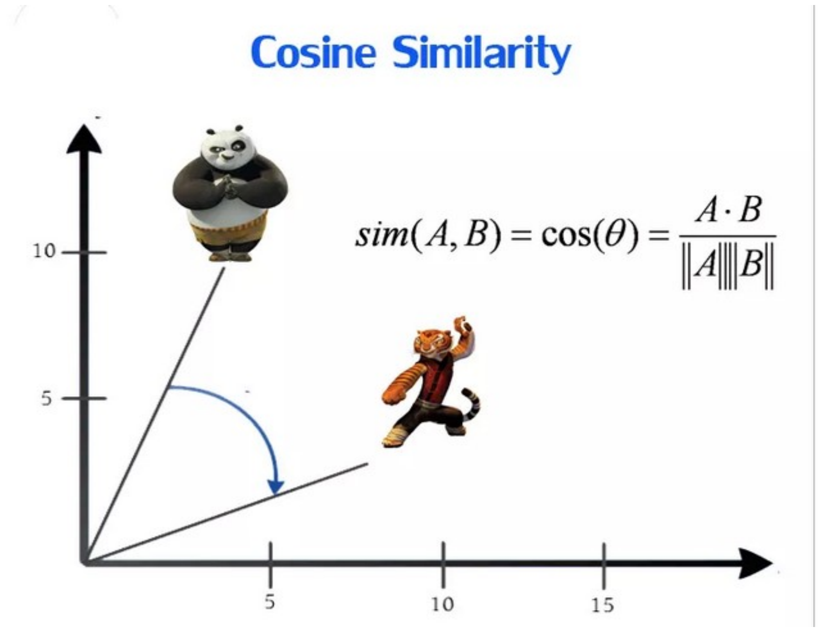
- Assign observations into groups so that the observations belonging in the same group are similar, whereas observations in different groups are different =>
- **We need a distance between points:**

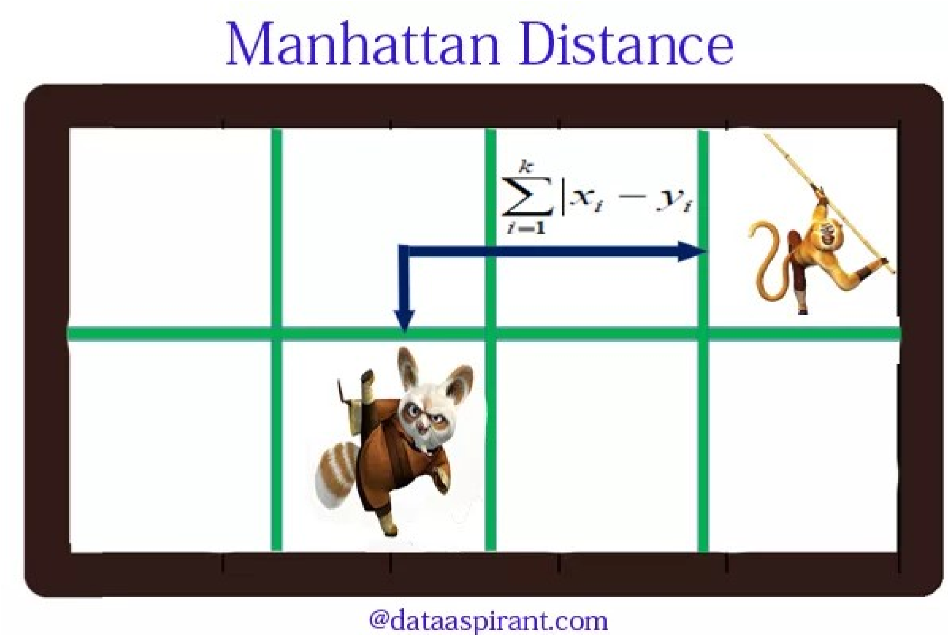
The distance **d(x, y)** between two objects **x** and **y** is a metric if:

- **d(i, j)≥0** (non-negativity)
- **d(i, i)=0** (isolation)
- **d(i, j)= d(j, i)** (symmetry)
- **d(i, j) ≤ d(i, h)+d(h, j)** (triangular inequality)

Distance







- Euclidian
- Manhattan
- Cosine similarity
- …. many other

Data Structures

- *data* matrix

$$\text{tuples/objects}\left\{\overbrace{\begin{bmatrix} x_{11} & \cdots & x_{1\ell} & \cdots & x_{1d} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{i\ell} & \cdots & x_{id} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{n\ell} & \cdots & x_{nd} \end{bmatrix}}^{\text{attributes/dimensions}}\right.$$

- *Distance* matrix

$$\text{objects}\left\{\overbrace{\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}}^{\text{objects}}\right.$$

Non-hierarchical methods
the k-means algorithm

- Given a set **X** of **n** points in a **d**-dimensional space and an integer **k**

- **Task:** choose a set of **k** points (cluster centers) $\{c_1, c_2, \ldots, c_k\}$ in the **d**-dimensional space to form clusters $\{C_1, C_2, \ldots, C_k\}$ such that

$$Cost(C) = \sum_{i=1}^{k} \sum_{x \in C_i} L_2^{\,2}(x - c_i)$$

  is minimized

- Some special cases: k = 1, k = n

The k-means algorithm

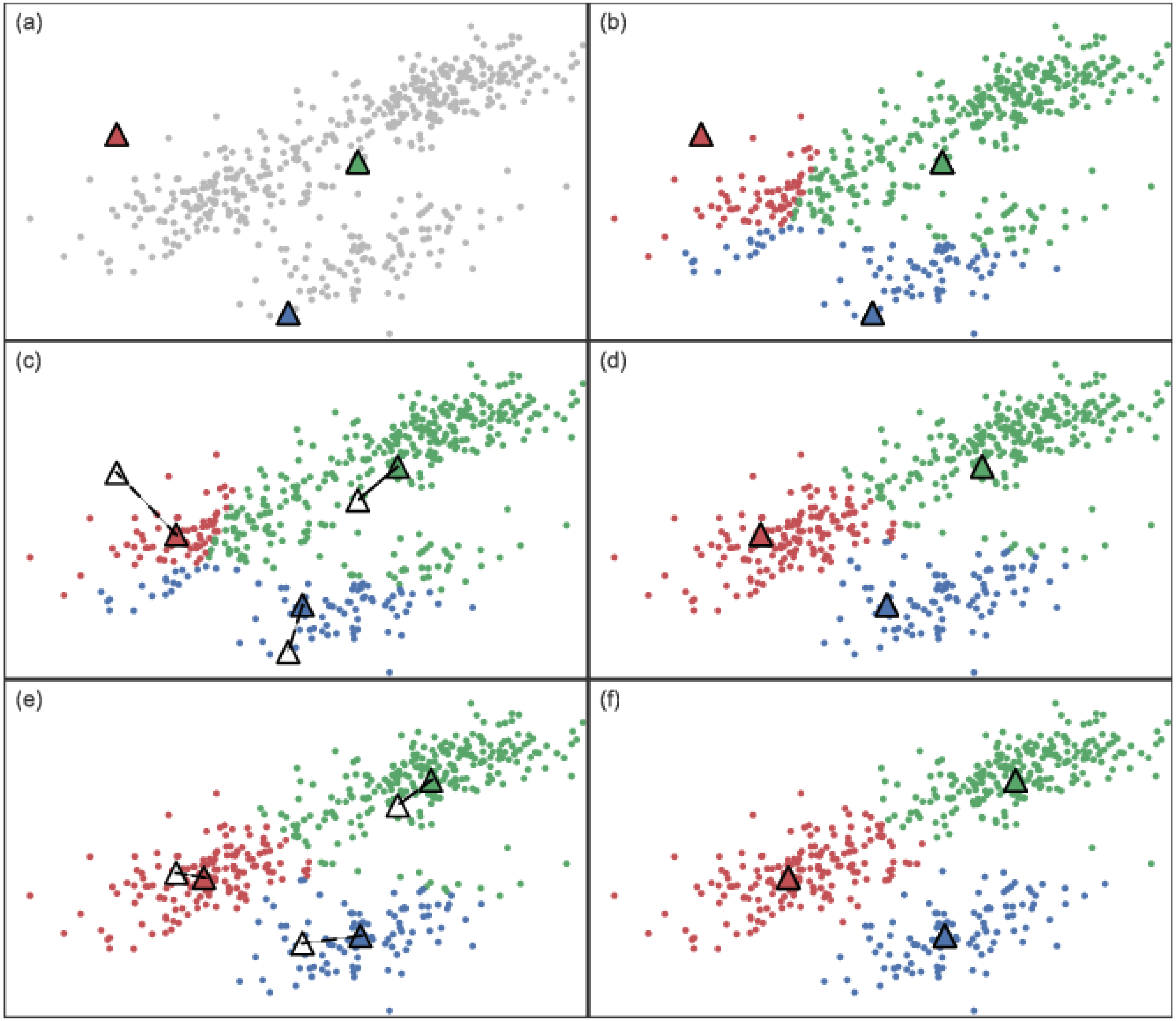
- Randomly pick **k** cluster centers $\{c_1,\ldots,c_k\}$

- For each **i**, set the cluster C_i to be the set of points in **X** that are closer to c_i than they are to c_j for all $i \neq j$

- For each **i** let c_i be the center of cluster C_i (mean of the vectors in C_i)

- Repeat until convergence

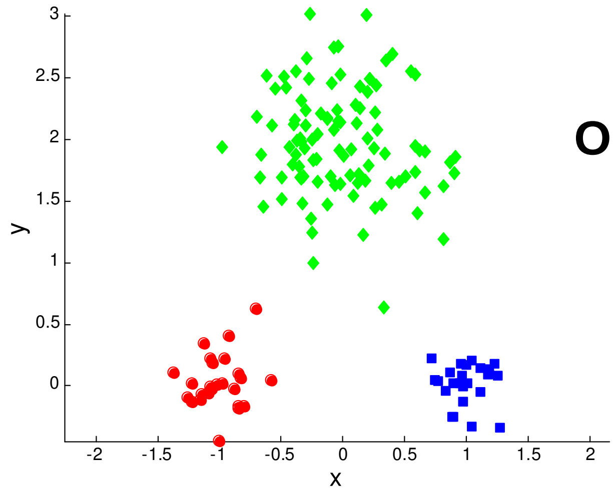
K-means algorithm



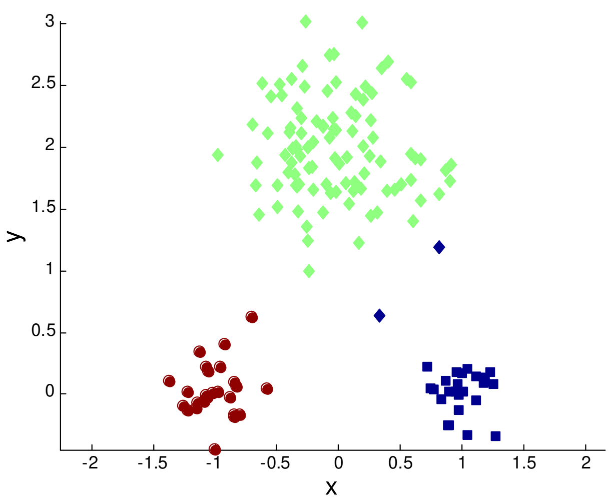
Properties of the k-means algorithm

- Finds a local optimum
- Converges often quickly (but not always)
- The choice of initial points can have large influence in the result

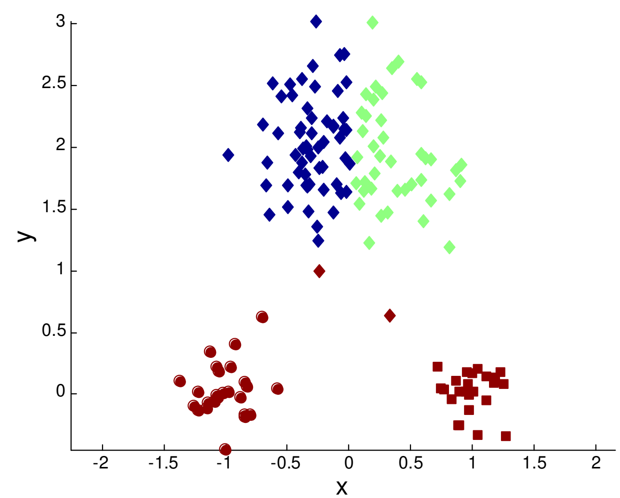
Two different K-means clusterings



Original Points



Optimal Clustering



Sub-optimal Clustering

Some alternatives to random initialization of the central points

- Multiple runs
  - Helps, but probability is not on your side

- Select original set of points by methods other than random. E.g., pick the most distant (from each other) points as cluster centers (kmeans++ algorithm in Scikit Learn)

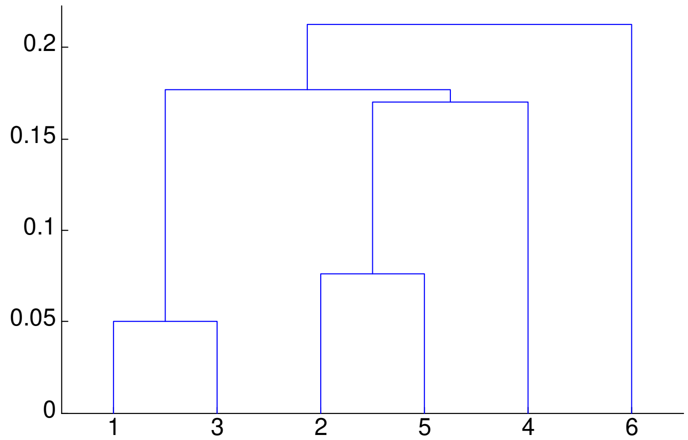
Example of k-means algorithm

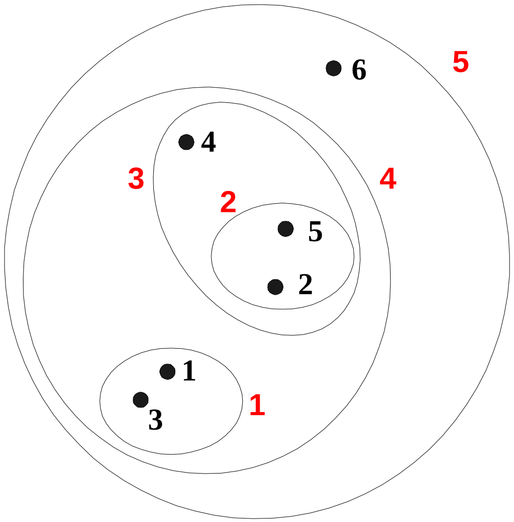
- https://github.com/marcinwolter/MachineLearning2020/blob/main/plot_kmeans_assumptions.ipynb

- The KMeans algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares (see below). This algorithm requires the number of clusters to be specified. It scales well to large number of samples and has been used across a large range of application areas in many different fields.

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

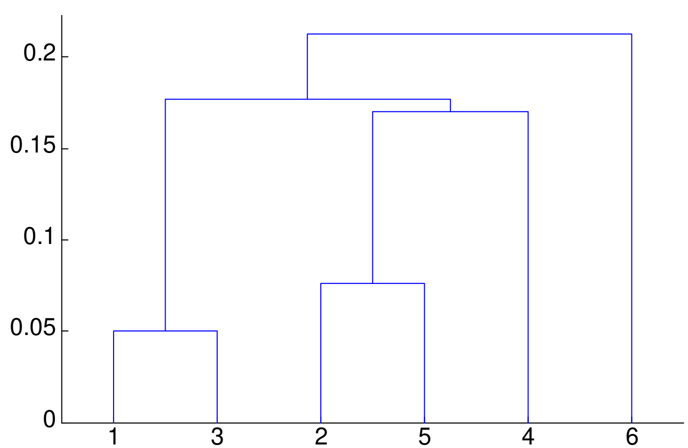


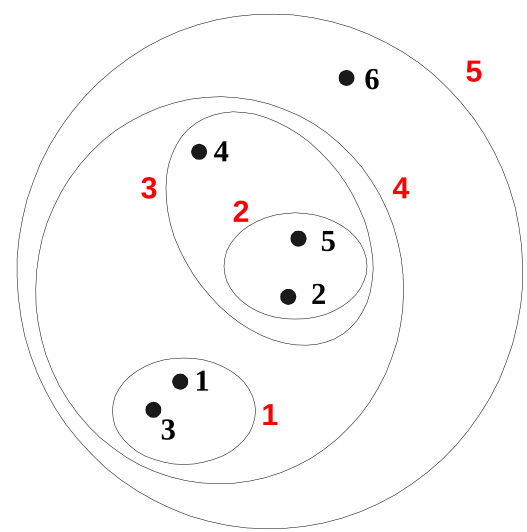


Strengths of Hierarchical Clustering

- No assumptions on the number of clusters
  - Any desired number of clusters can be obtained by ‘cutting’ the dendrogram at the proper level

- Hierarchical clustering may correspond to some meaningful features





Hierarchical Clustering

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

Complexity of hierarchical clustering

- Distance matrix is used for deciding which clusters to merge/split
- At least quadratic in the number of data points
- Not usable for large datasets

Agglomerative clustering algorithm

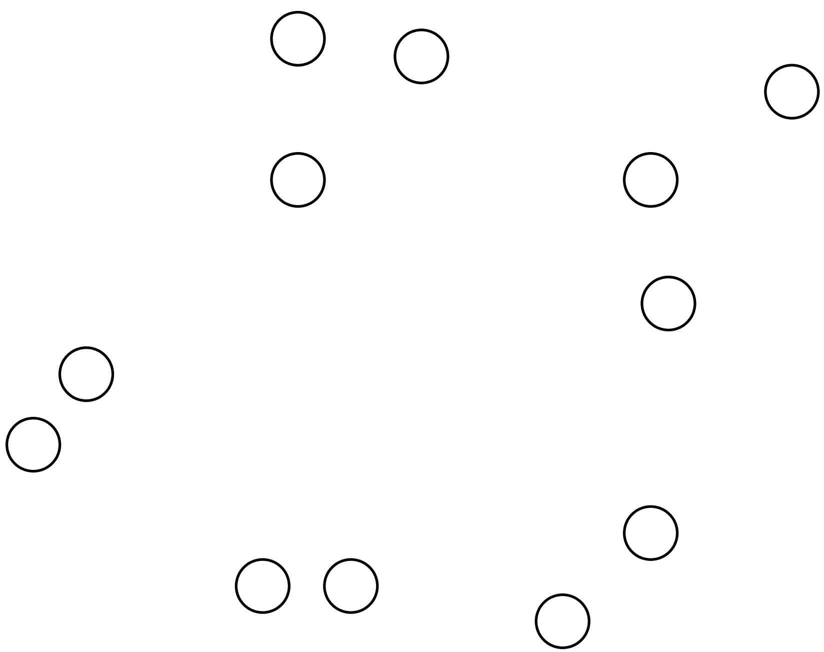
- **Most popular hierarchical clustering technique**

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

- Key operation is the computation of the distance between two clusters
  – Different definitions of the distance between clusters lead to different algorithms

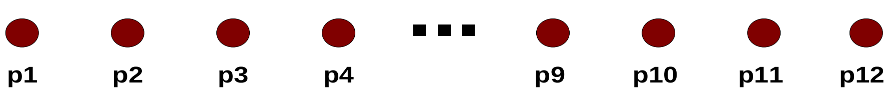
Input / Initial setting

- Start with clusters of individual points and a distance/proximity matrix



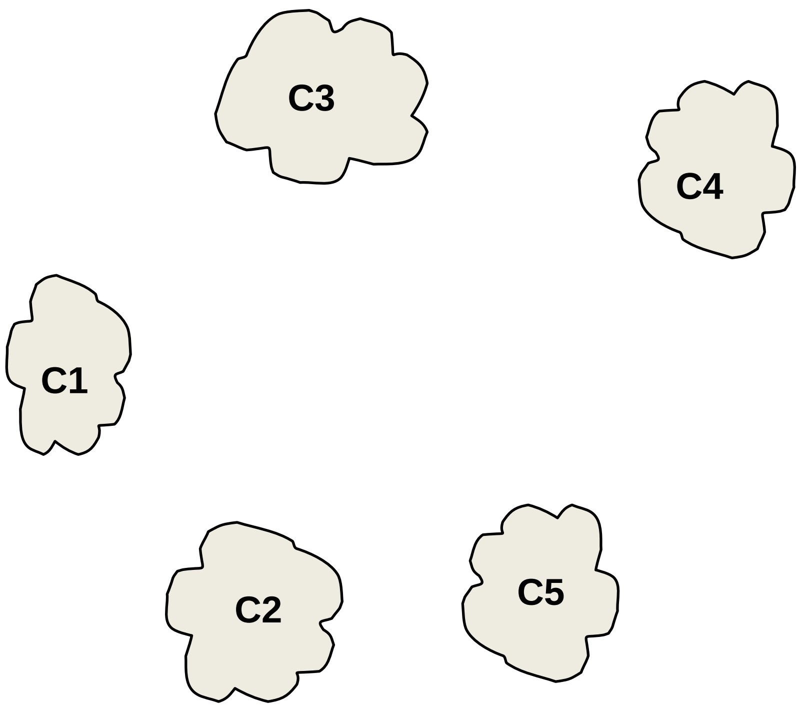
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
. . .						

Distance/Proximity Matrix



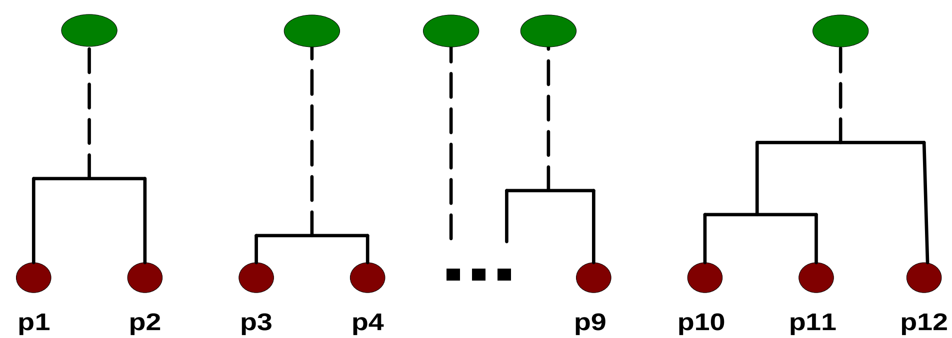
Intermediate State

- After some merging steps, we have some clusters



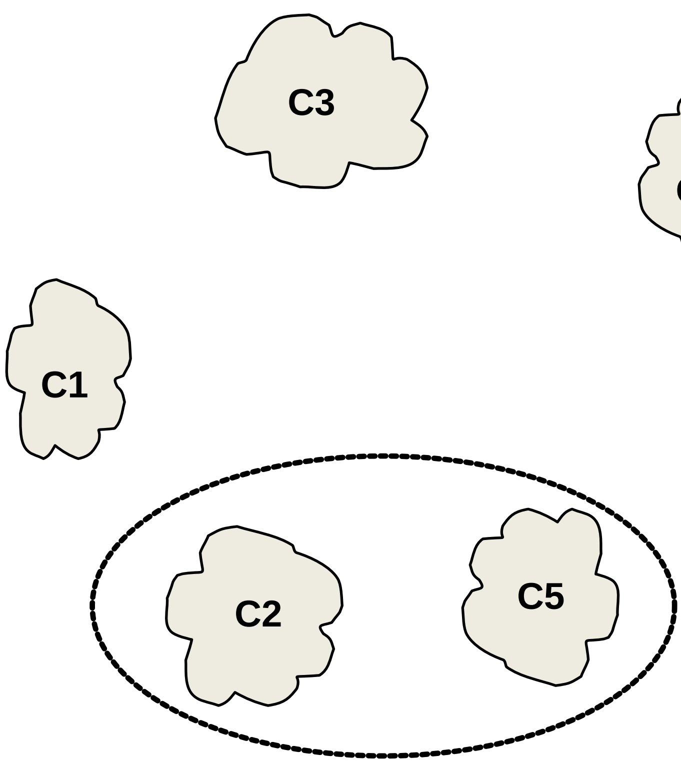
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

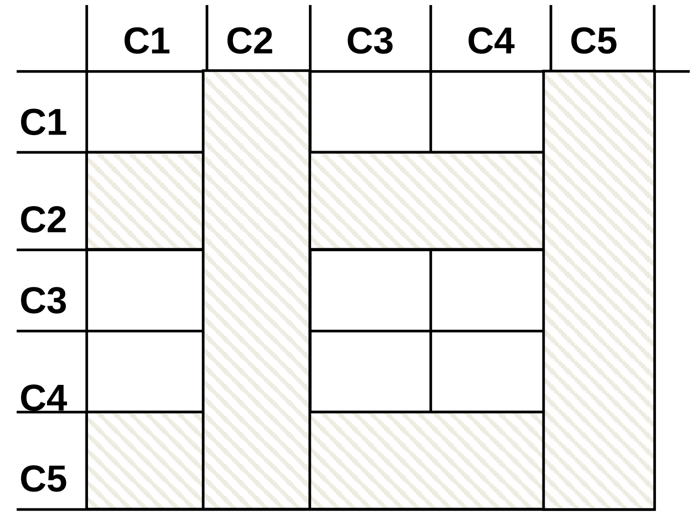
Distance/Proximity Matrix



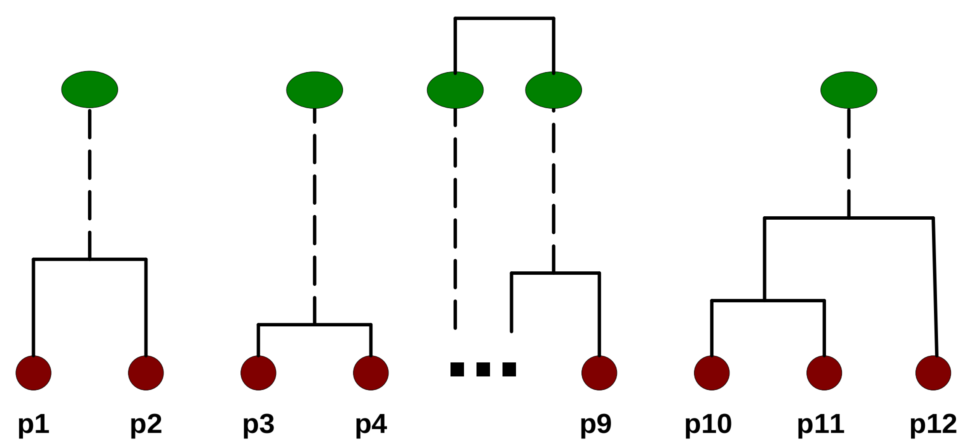
Intermediate State

- Merge the two closest clusters (C2 and C5) and update the distance matrix.



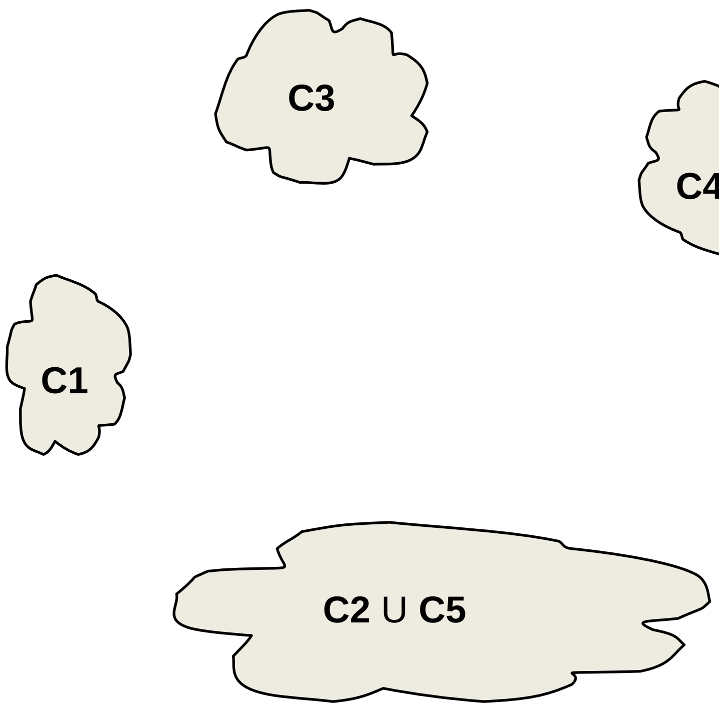


Distance/Proximity Matrix

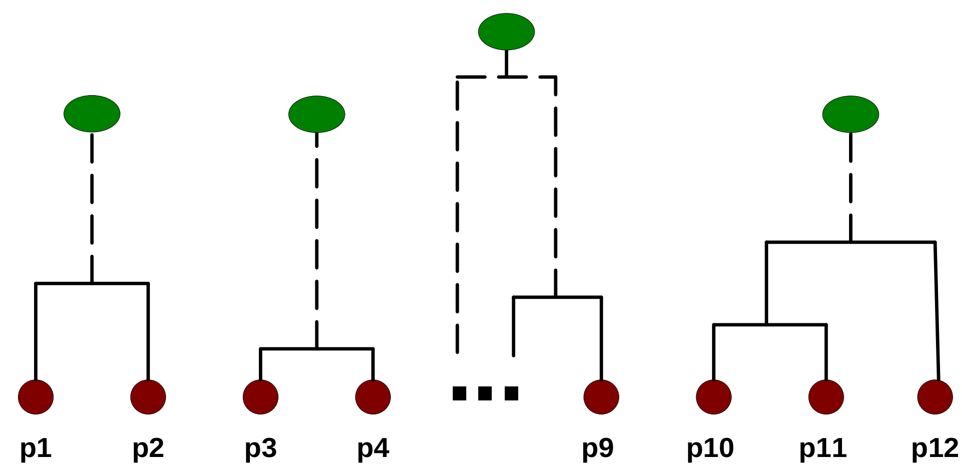


After Merging

- “How do we update the distance matrix?”



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



Distance between two clusters

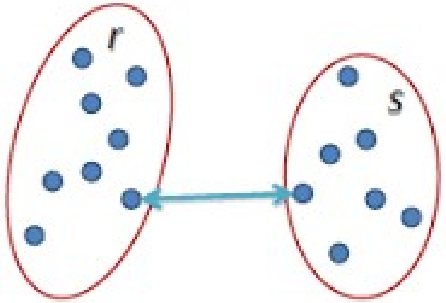
- Each cluster is a set of points
- How do we define distance between two sets of points?
  - Lots of alternatives
  - Not an easy task

Distance between two clusters

- **Single-link distance** between clusters C_i and C_j is the ***minimum distance*** between any object in C_i and any object in C_j

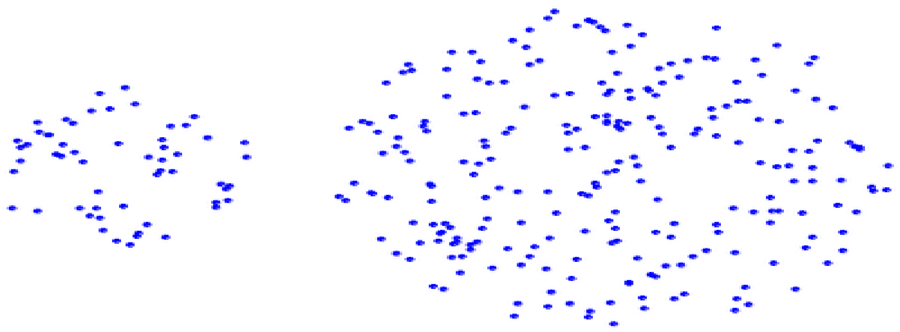
- The distance is **defined by the two most similar objects**

$$D_{sl}(C_i, C_j) = \min_{x,y} \{ d(x,y) \mid x \in C_i, y \in C_j \}$$

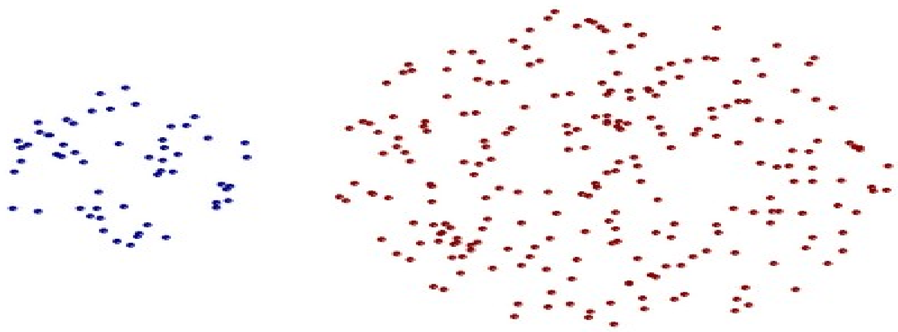


$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

Strengths of single-link clustering



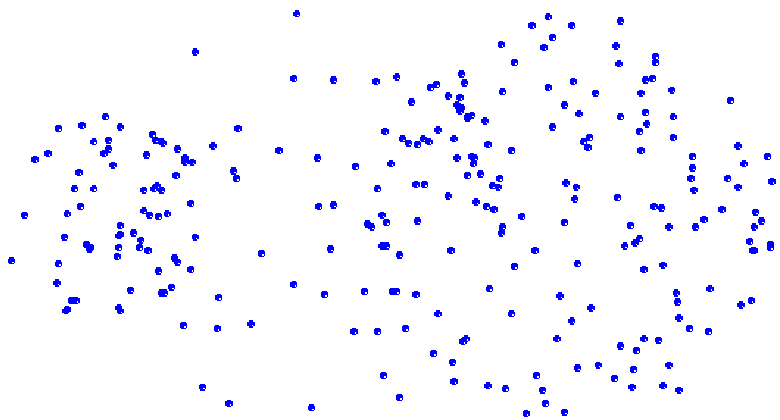
Original Points



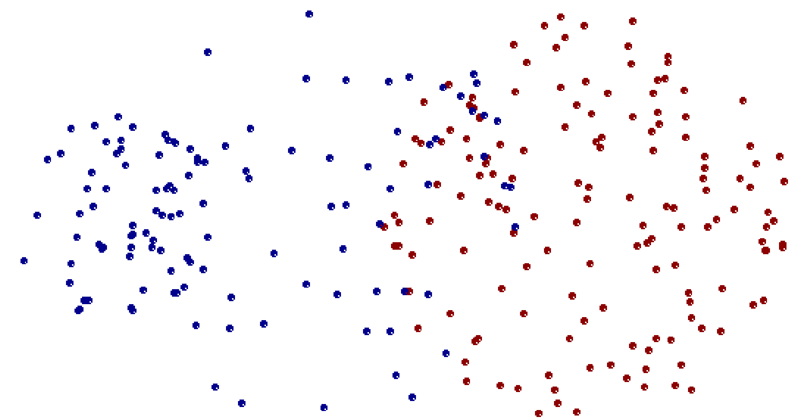
Two Clusters

- Can handle non-elliptical shapes

Limitations of single-link clustering



Original Points



Two Clusters

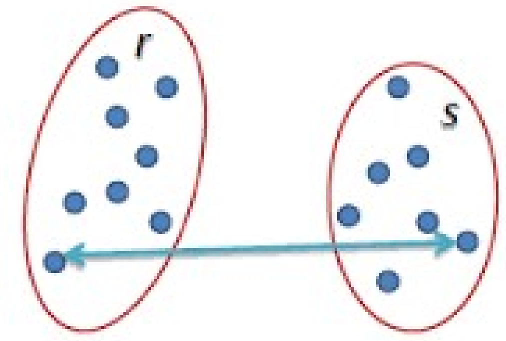
- Sensitive to noise and outliers
- Produces long, elongated clusters

Distance between two clusters

- **Complete-link distance** between clusters C_i and C_j is the ***maximum distance*** between any object in C_i and any object in C_j

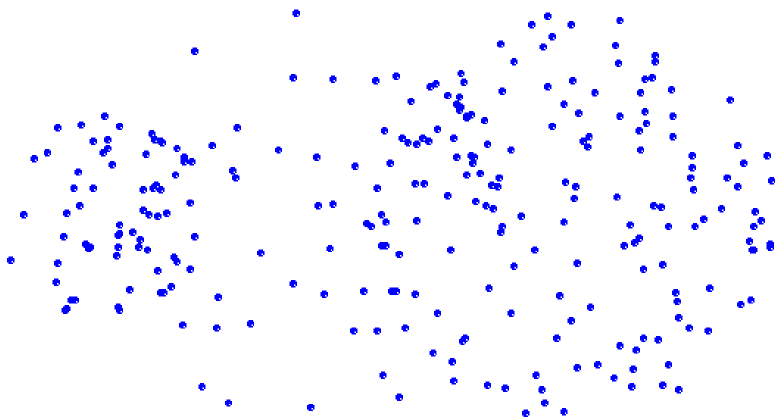
- The distance is **defined by the two most dissimilar objects**

$$D_{cl}(C_i, C_j) = \max_{x,y}\{d(x,y) \mid x \in C_i, y \in C_j\}$$

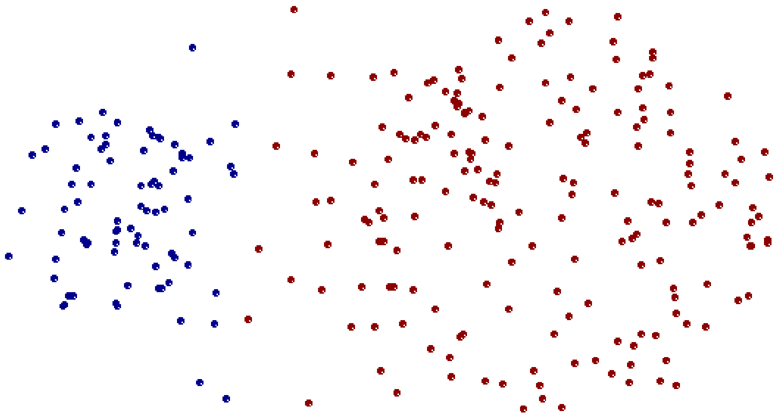


$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

Strengths of complete-link clustering



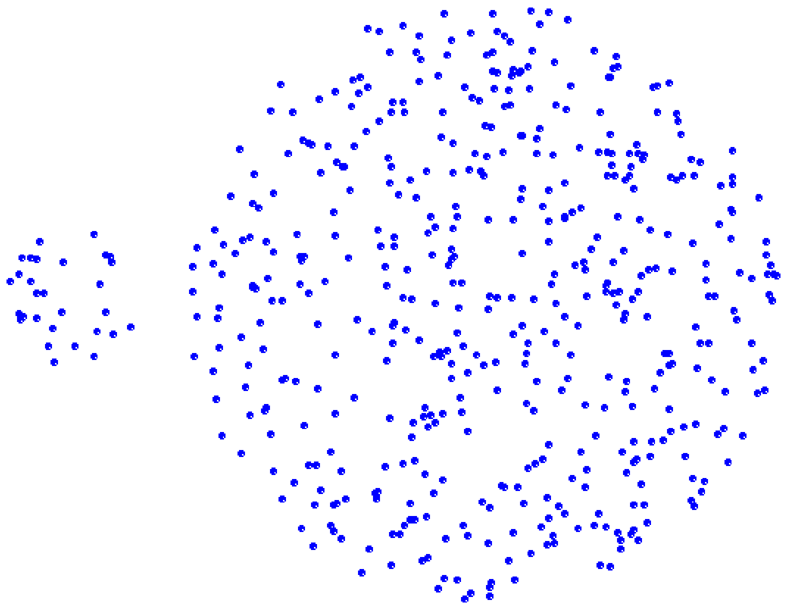
Original Points



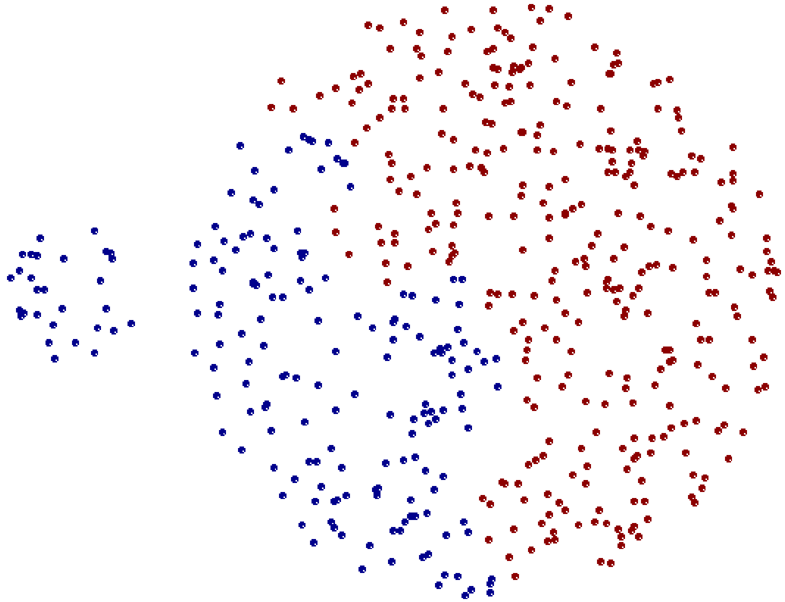
Two Clusters

- More balanced clusters (with equal diameter)
- Less susceptible to noise

Limitations of complete-link clustering



Original Points



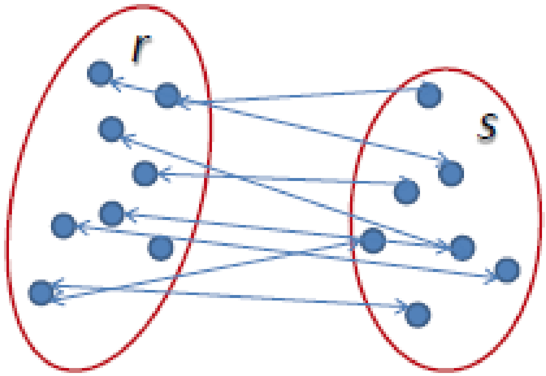
Two Clusters

- Tends to break large clusters
- All clusters tend to have the same diameter – small clusters are merged with larger ones

Distance between two clusters

- **Group average distance** between clusters C_i and C_j is the ***average distance*** between any object in C_i and any object in C_j

$$D_{avg}(C_i, C_j) = \frac{1}{|C_i| \times |C_j|} \sum_{x \in C_i, y \in C_j} d(x, y)$$



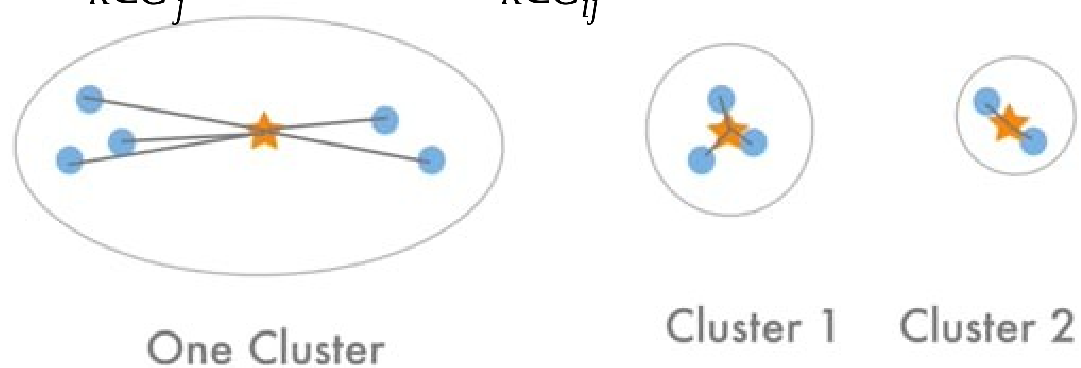
$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

Distance between two clusters

- **Ward's distance** between clusters C_i and C_j is the ***difference*** between the ***total within cluster sum of squares for the two clusters separately***, and the ***within cluster sum of squares resulting from merging the two clusters*** in cluster C_{ij}

$$D_w\left(C_i, C_j\right) = \sum_{x \in C_i} \left(x - r_i\right)^2 + \sum_{x \in C_j} \left(x - r_j\right)^2 - \sum_{x \in C_{ij}} \left(x - r_{ij}\right)^2$$

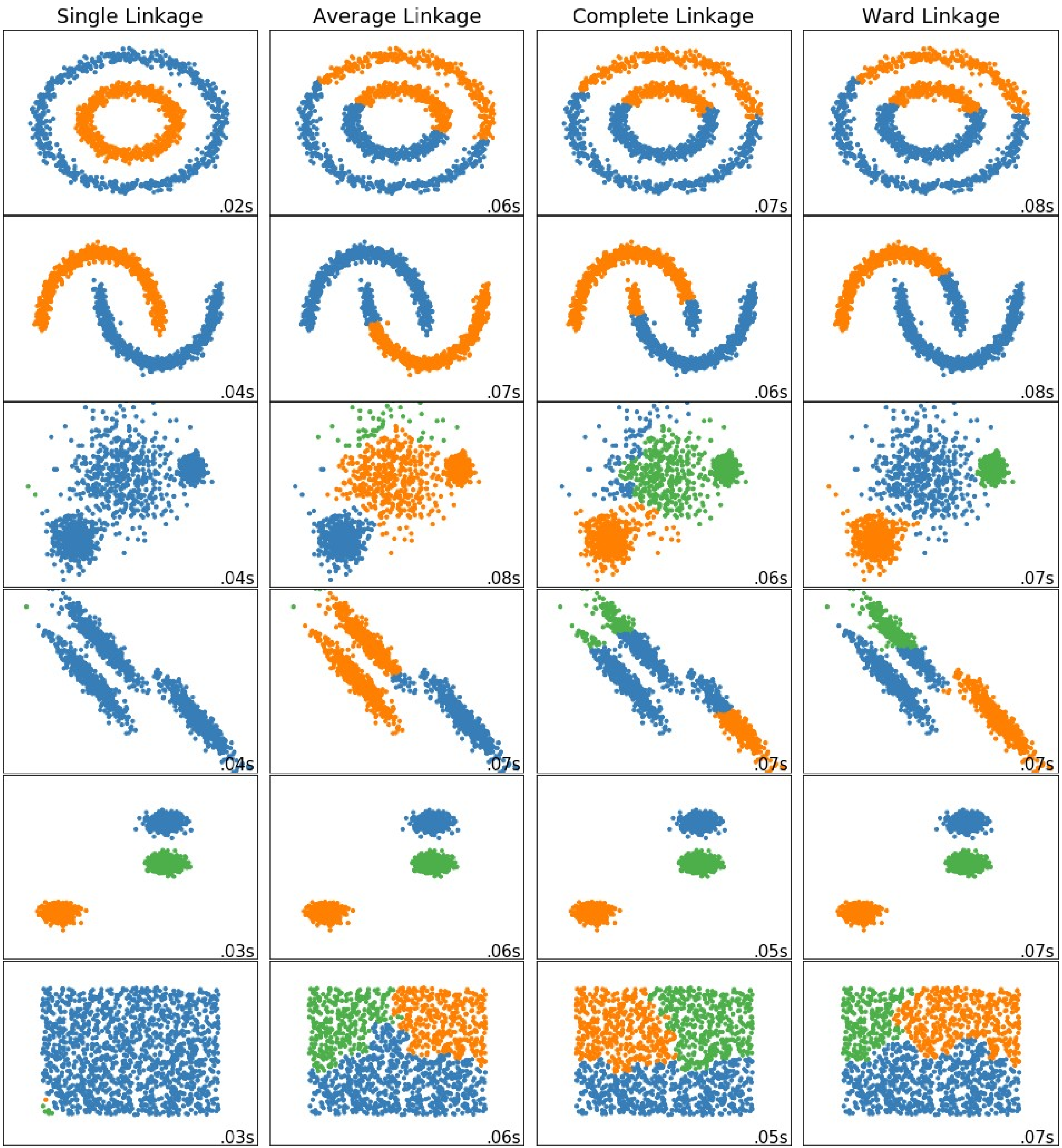
- r_i: centroid of C_i
- r_j: centroid of C_j
- r_{ij}: centroid of C_{ij}



Ratio of sum of squared distance from means, between one cluster, and the two clusters defines the intercluster distance

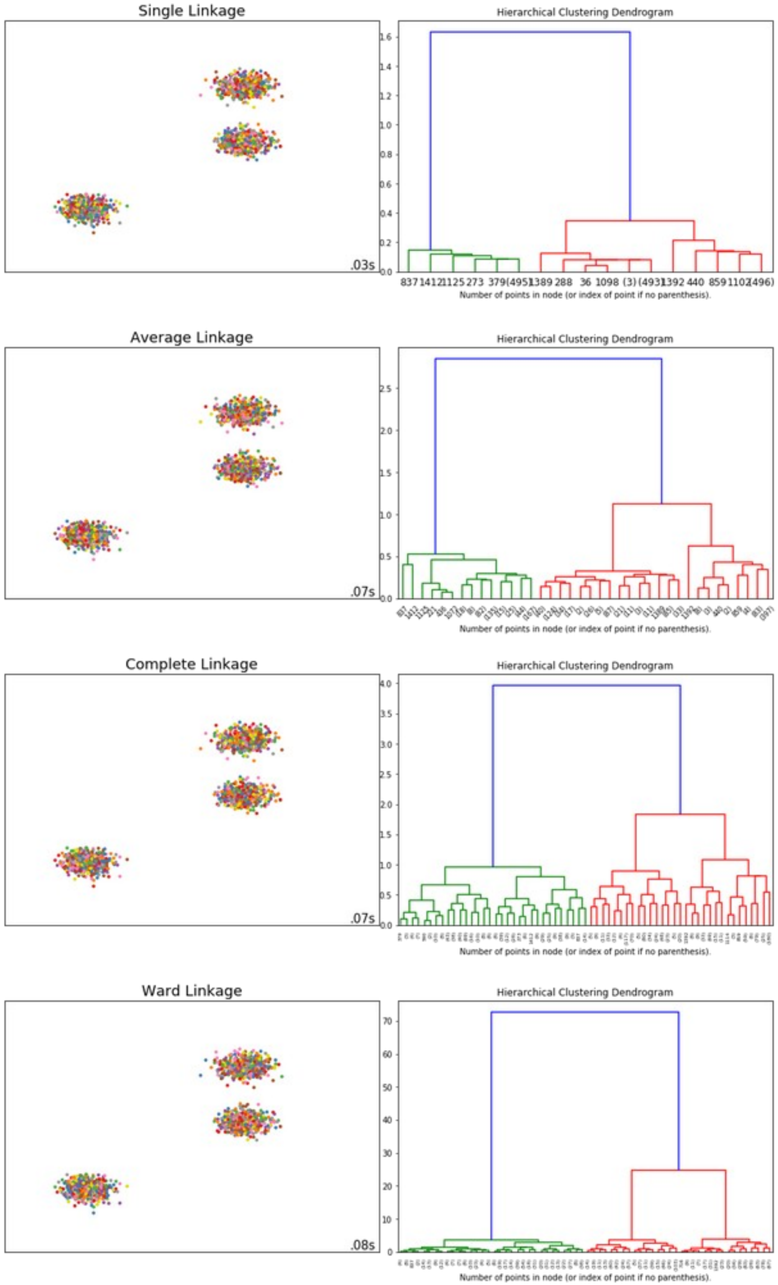
Ward’s distance for clusters

- Similar to group average and centroid distance
- Less susceptible to noise and outliers
- Hierarchical analogue of k-means
  - Can be used to initialize k-means


Single Linkage
Average Linkage
Complete Linkage
Ward Linkage
.02s
.06s
.07s
.08s
.04s
.07s
.06s
.08s
.04s
.08s
.06s
.07s
.04s
.07s
.07s
.07s
.03s
.06s
.05s
.07s
.03s
.06s
.05s
.07s

Comparison of distance measurements

https://github.com/marcinwolter/MachineLearning2020/blob/main/plot_linkage_comparison.ipynb



Cluster analysis

- Clustering analysis is broadly used in many applications such as market research, pattern recognition, data analysis, and image processing.
- As a data mining function, cluster analysis serves as a tool to gain insight into the distribution of data to observe characteristics of each cluster.