Clustering

Prof. Dr. Renato Pajarola
Visualization and MultiMedia Lab
Department of Informatics
University of Zürich

VISUALIZATIONAND
MULTIMEDIALAB

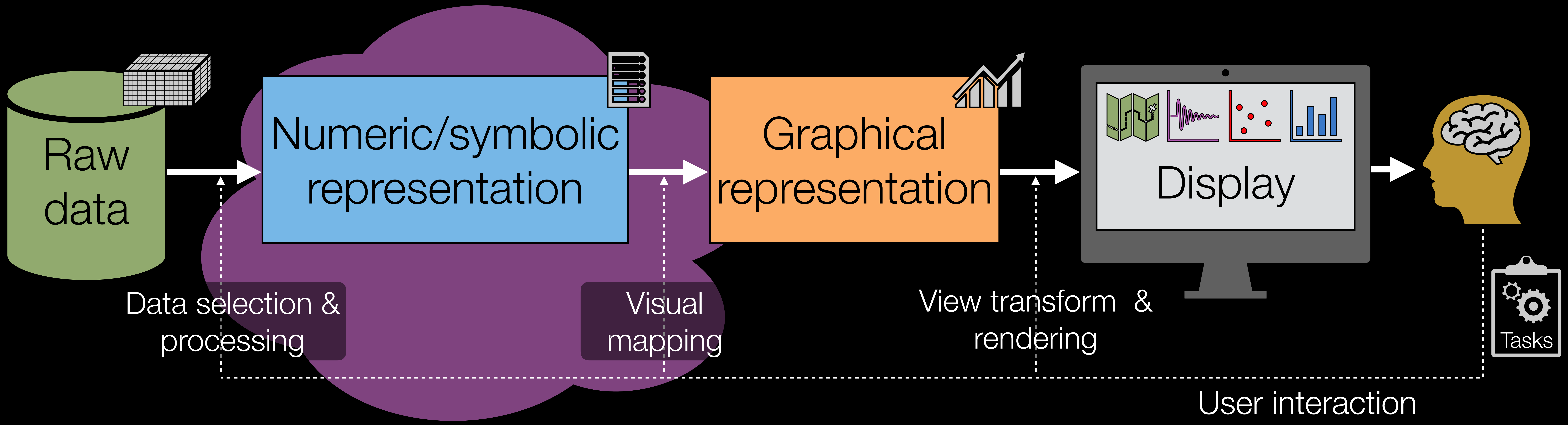
Copyrights

- Most figures of these slides are copyright protected and come from the various sources
- You understand that the slides contain copyright protected material and therefore the following conditions of use apply:
  - The slides may be used for personal teaching purposes only
  - Publishing the slides to any public web site is not allowed
  - Sharing the slides with other persons or institutions is prohibited

Overview

1. What is clustering?
2. Clustering properties and types
3. K-means clustering algorithm
4. Hierarchical clustering

Where are we in the Visualization Pipeline?



- Data preprocessing and transformation
  - Selection of information and mapping to fundamental computer data types
  - Data cleaning, interpolation, sampling, filtering, aggregation, partitioning
- Mapping for visualization
  - Specific visual representation (geometry, color)
  - Embedding in Euclidean 2D/3D space
- Rendering transformations
  - Final image synthesis by 2D imaging and 3D graphics technology
  - Interactive data and view selection

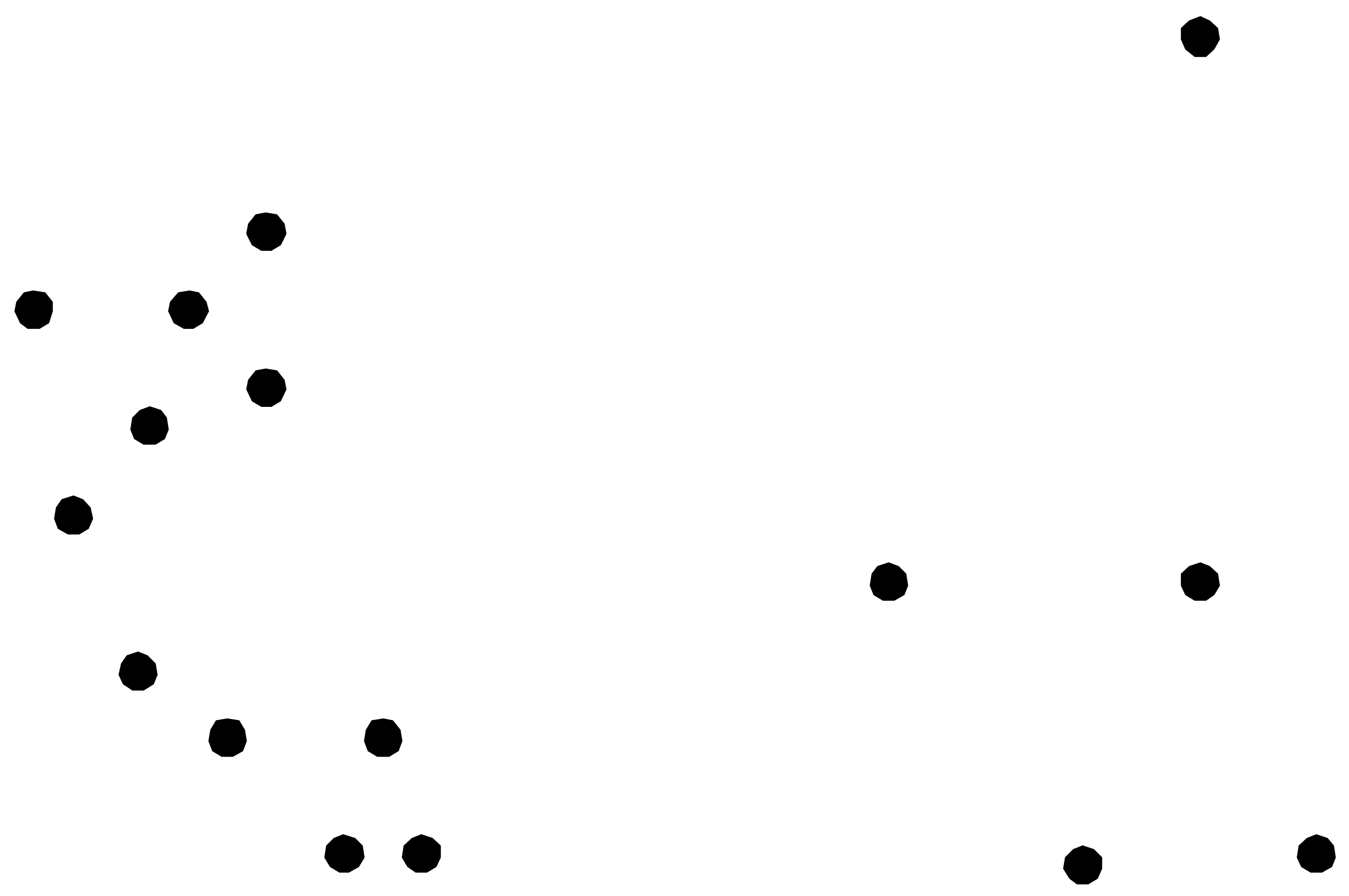
What is Clustering?

- Task of assigning input objects to groups such that objects in the same group are more similar to each other than to objects from other groups
  - Both procedure and set of groups obtained known as "clustering"
  - Impose structure on data for visualization
- Definition of d_{ij} distance between data objects i and j is of fundamental importance
- Hard to provide a rigorous definition of *cluster*
  - Often dependent on specific setting
  - Results in many different clustering algorithms

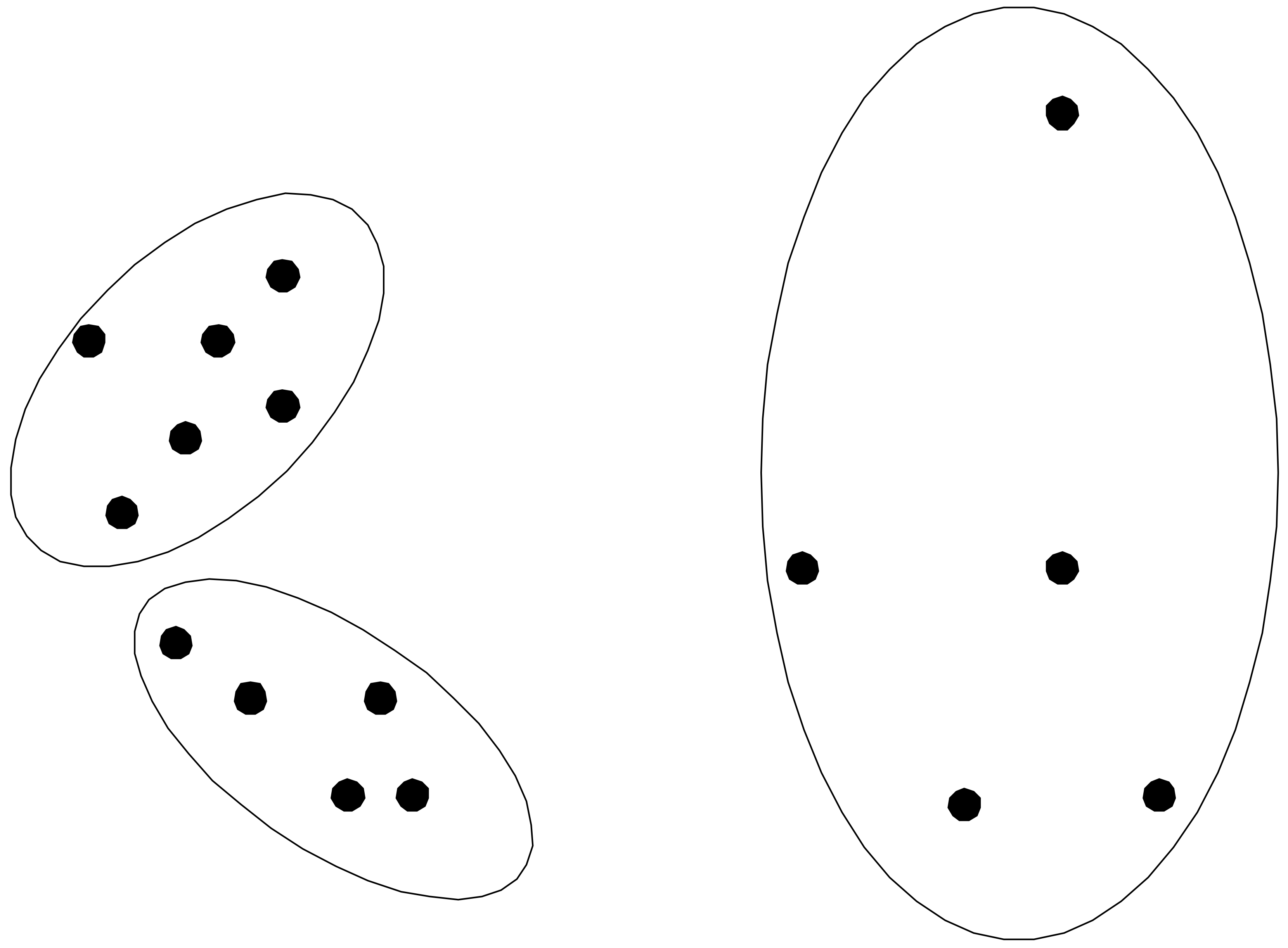
- Common characterization of clustering methods:
  - *Centroid-based*: each cluster represented by a single data vector (k-means, k-medoids)
  - *Connectivity-based*: objects more related to nearby objects than to objects farther away (hierarchical)
  - *Density-based*: clusters as connected dense regions (dbscan, mean-shift)
  - *Distribution-based*: clusters modeled using statistical distributions (Gaussian mixture models)
  - *Hard vs. soft (aka fuzzy) clustering*: object can belong to only one or multiple clusters
  - *Partitional vs. hierarchical clustering*: whether clustering produces a hierarchy of clusters

Partitional Clustering

- Data points divided into non-overlapping subsets
    - Clusters form a partition of the input dataset



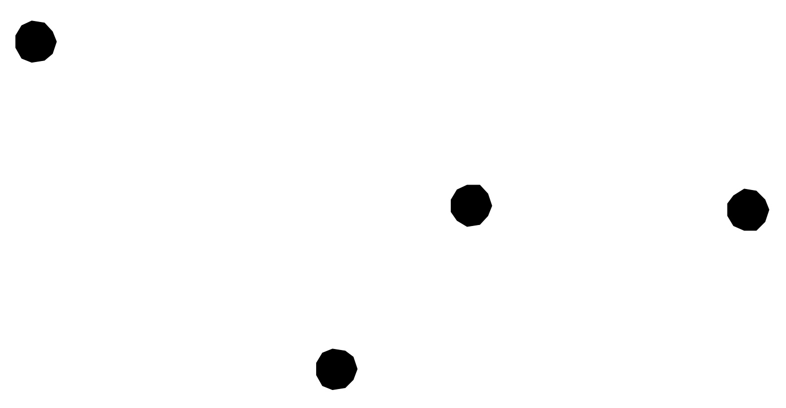
Original Points



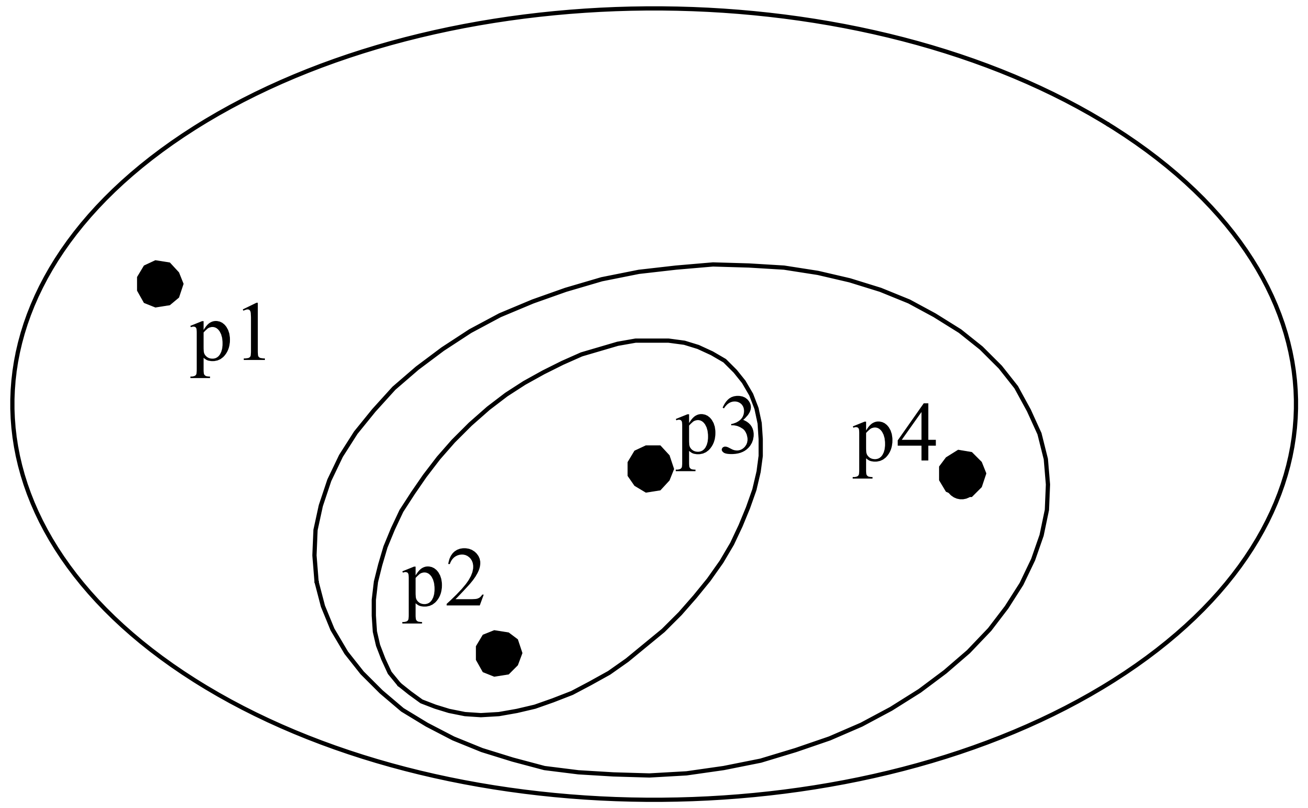
A Partitional Clustering

Hierarchical Clustering

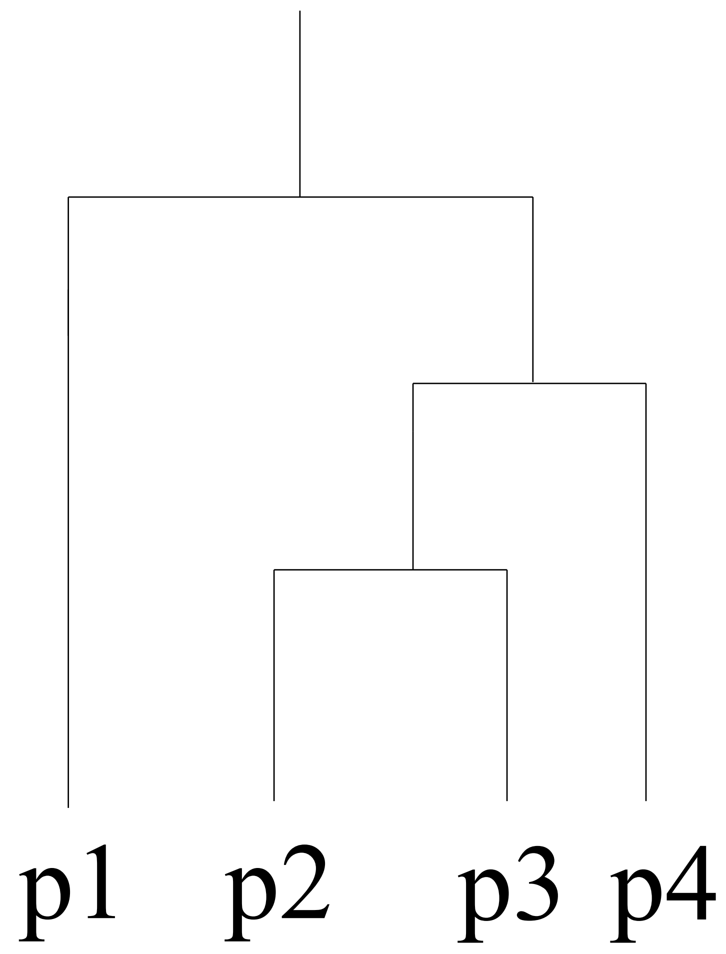
- Data points divided into nested subsets that form a hierarchy
  - Corresponds to a tree structure



Original Points



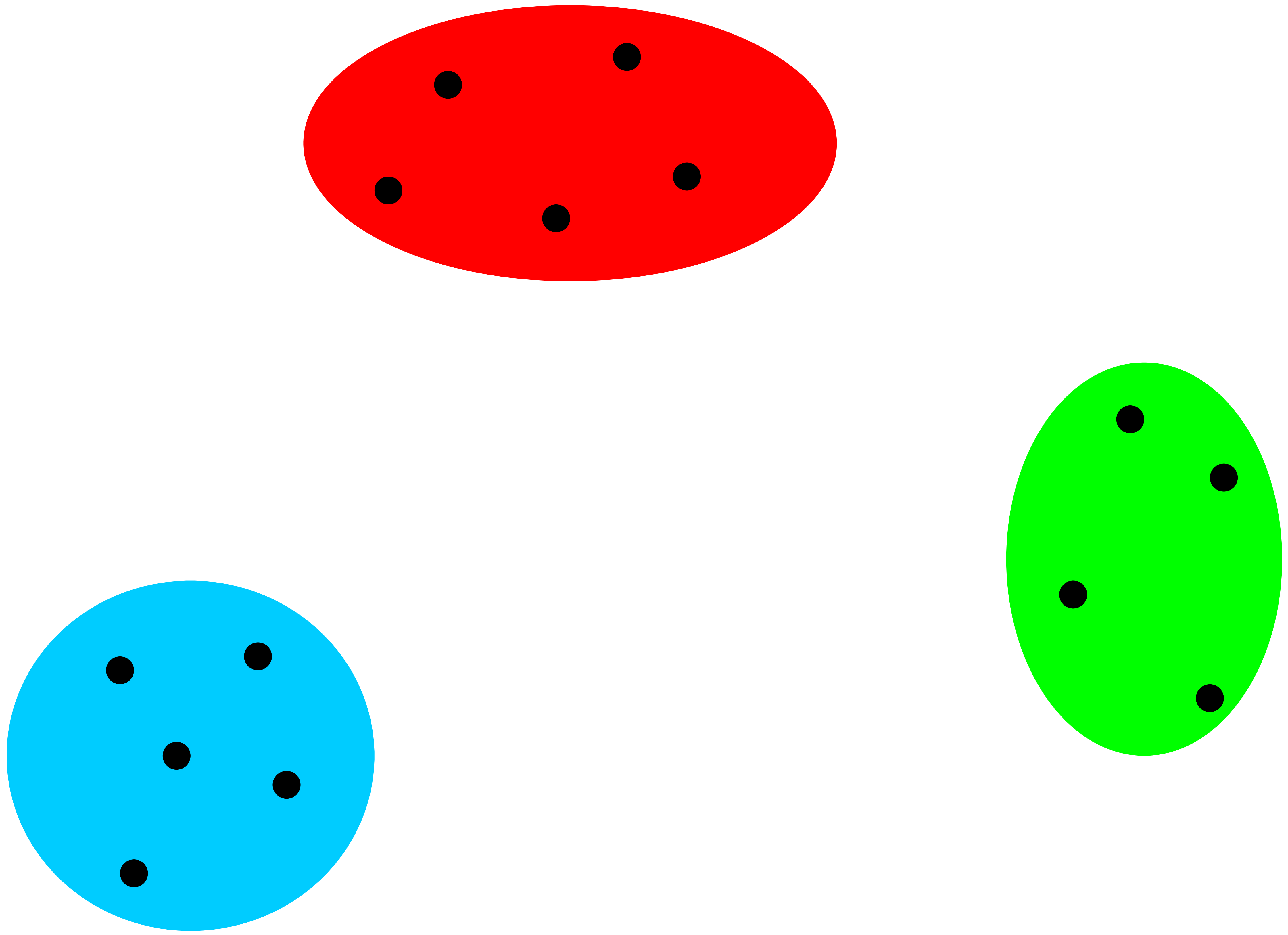
Hierarchical Clustering



Dendrogram

Cluster Separation

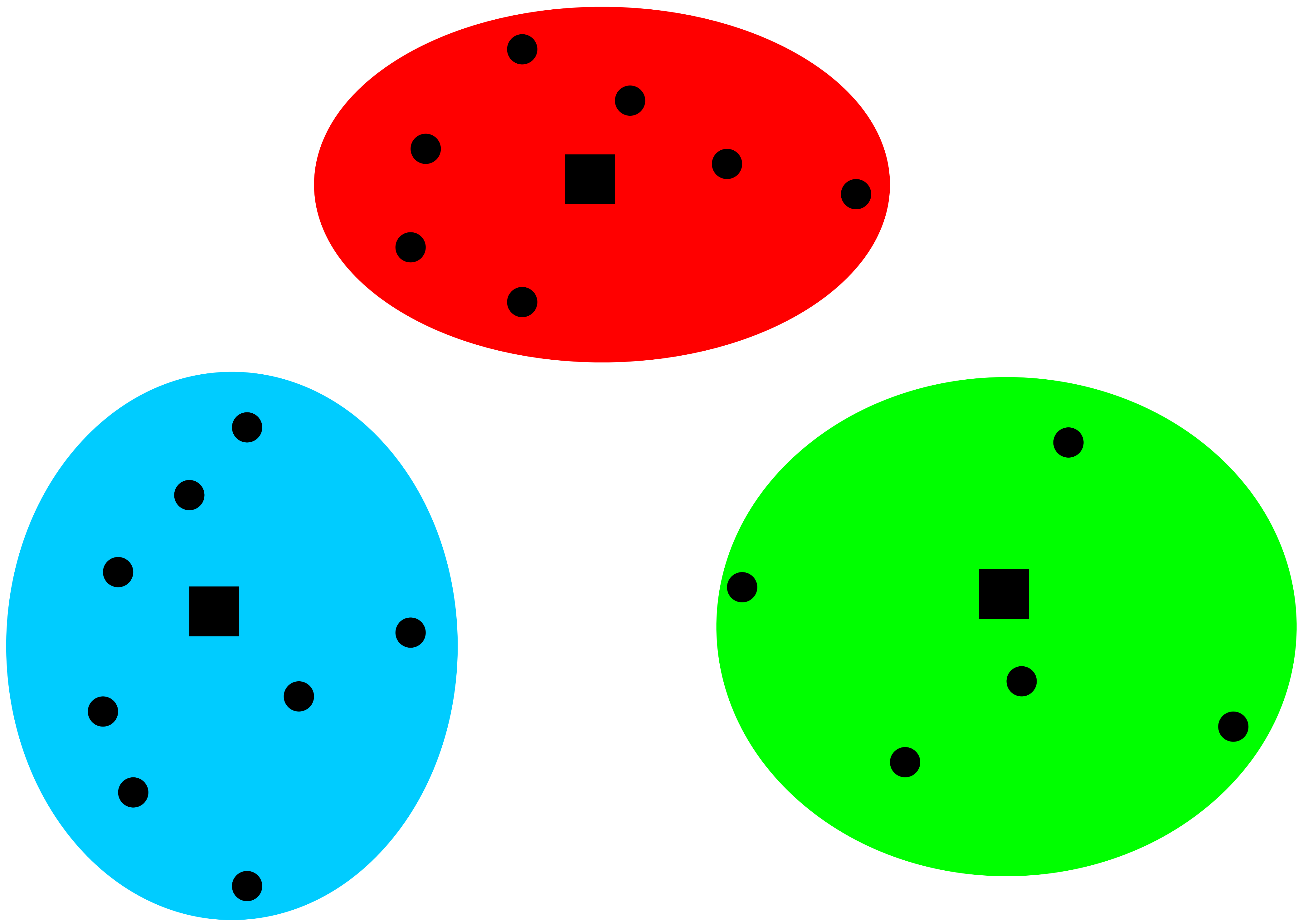
- Well separated clusters
  - Any data point in one cluster is closer or more similar to **every point** in that cluster than to any other point from other clusters



3 well-separated clusters

Cluster Separation

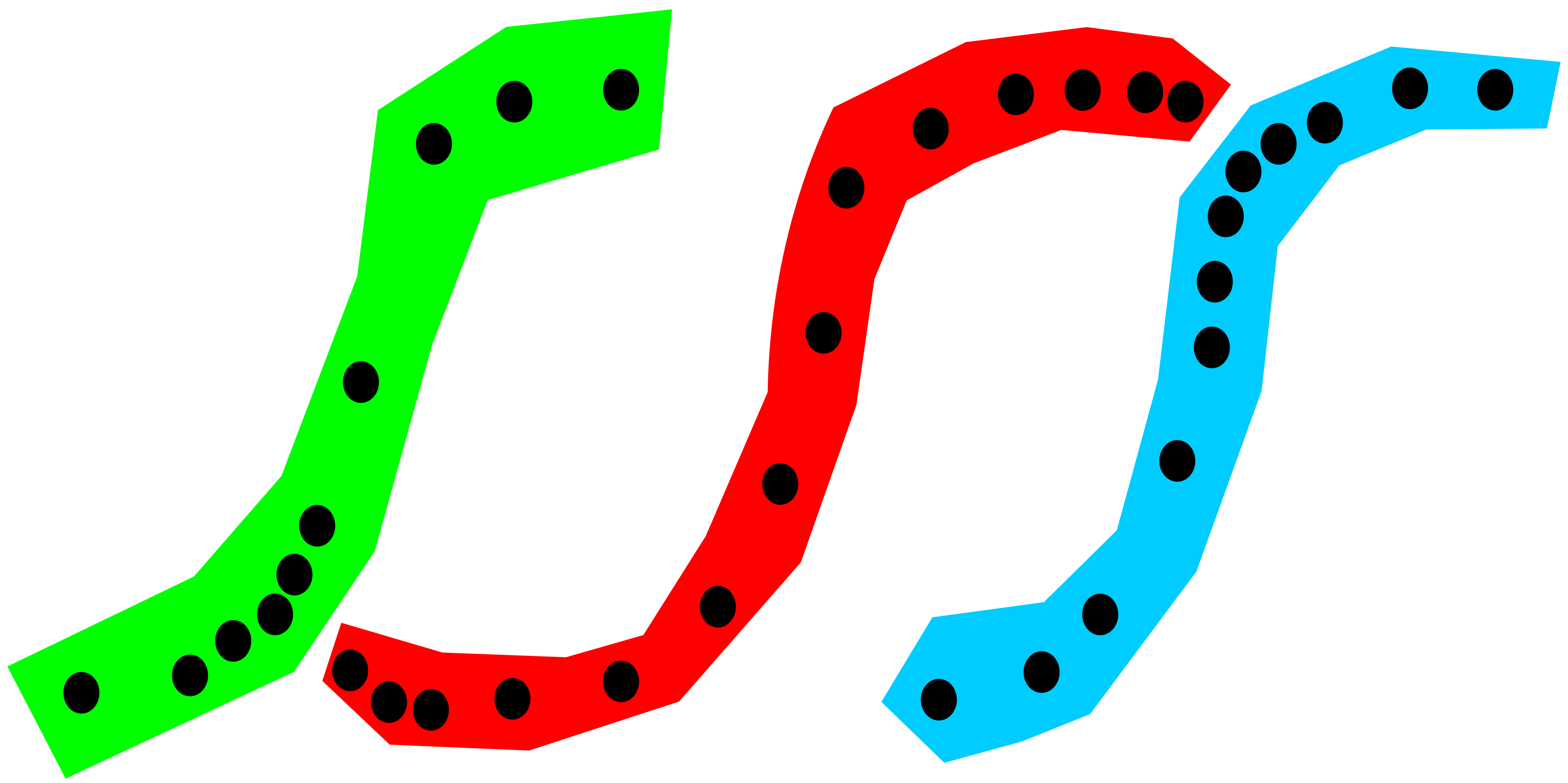
- Well separated clusters
  - Any data point in one cluster is closer or more similar to **every point** in that cluster than to any other point from other clusters
- Center based clusters
  - Any data point in one cluster is closer to the **center** of that cluster than to the center of any other cluster
    - often the centroid, average of all points, or the medoid, the most representative point



3 center-based clusters

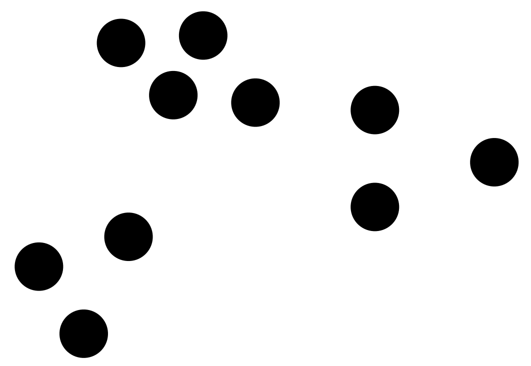
Cluster Separation

- Well separated clusters
  - Any data point in one cluster is closer or more similar to **every point** in that cluster than to any other point from other clusters
- Center based clusters
  - Any data point in one cluster is closer to the **center** of that cluster than to the center of any other cluster
    - often the centroid, average of all points, or the medoid, the most representative point
- Contiguous clusters
  - Any data point in one cluster is closer or more similar to one or more other points in that cluster than to any other point from other clusters



3 contiguous clusters

Clustering can be Ambiguous



How many clusters?

Six Clusters

Two Clusters

Four Clusters

Clustering Algorithms

- Types of clustering algorithms:
  - Single-pass, iterative, incremental, hierarchical …
- Iterative k-means clustering
  - Repeated assignment of data points to k clusters followed by update of cluster center
- Hierarchical clustering
  - Recursively group 2 or more closest clusters into a new parent cluster
  - Recursively split one cluster into 2 or more smaller child clusters
- Other clustering methods:
  - k-medoids, mean-shift, dbscan, spectral, Gaussian mixture …

K-Means Clustering

- Simple and efficient method to partition the dataset into *k* clusters
  - Each cluster represented by a *centroid*
  - *k* is a user-defined parameter
- Partitional; centroid-based; produces hard clusters
- Compute centroids $\boldsymbol{v}_j$ that minimize: $E(\Gamma, V) = \sum_{j=1}^{k} \sum_{i=1}^{n} \gamma_{ij} \|\boldsymbol{x}_i - \boldsymbol{v}_j\|^2$

Data set: $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots \boldsymbol{x}_n\}$

Clusters: $C_1, C_2, \cdots C_k$

Centroids: $\boldsymbol{V} = \{\boldsymbol{v}_1, \cdots \boldsymbol{v}_k\}$

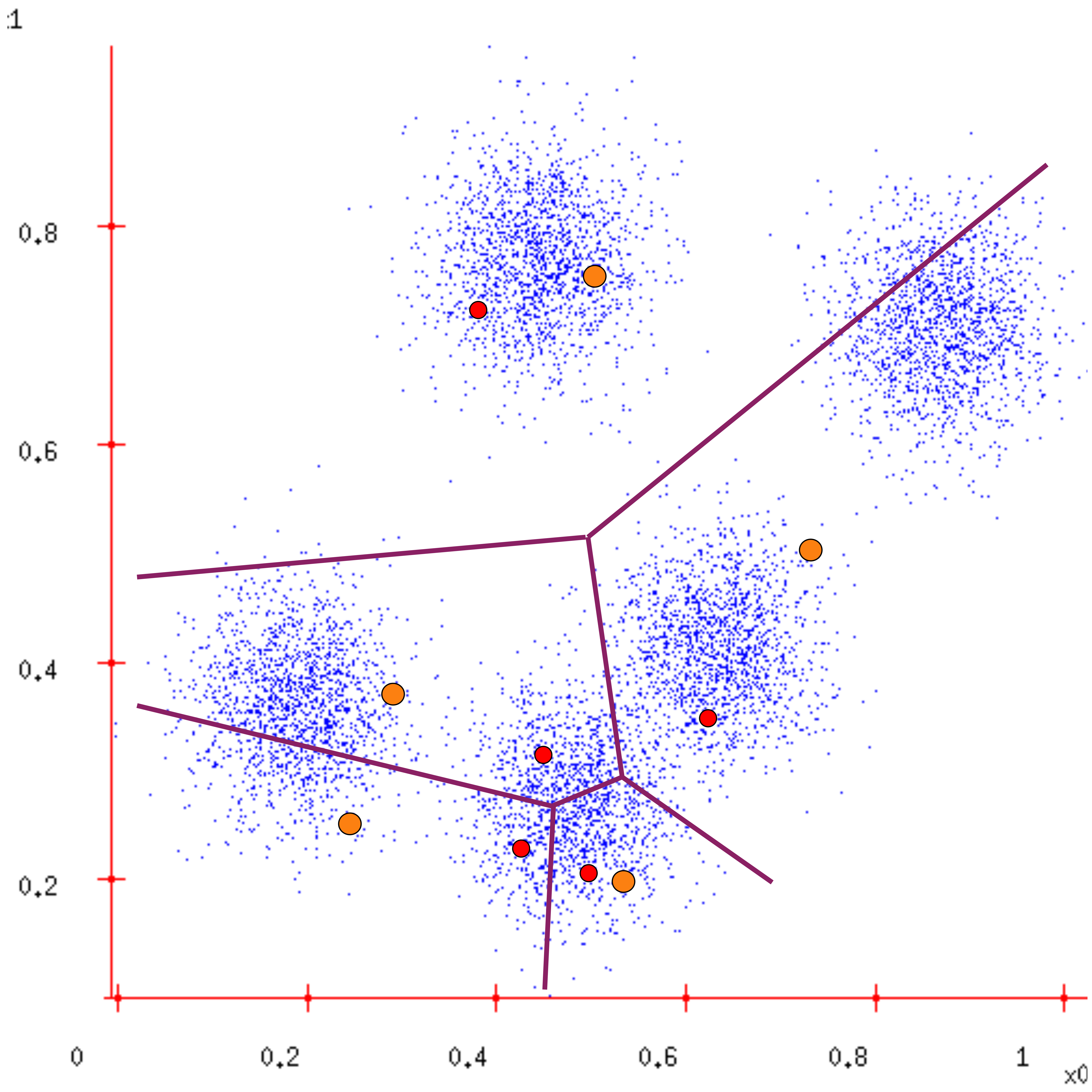
Partition matrix: $\Gamma = \{\gamma_{ij}\}$

$\gamma_{ij} = 1$ iff $\boldsymbol{x}_i \in C_j$

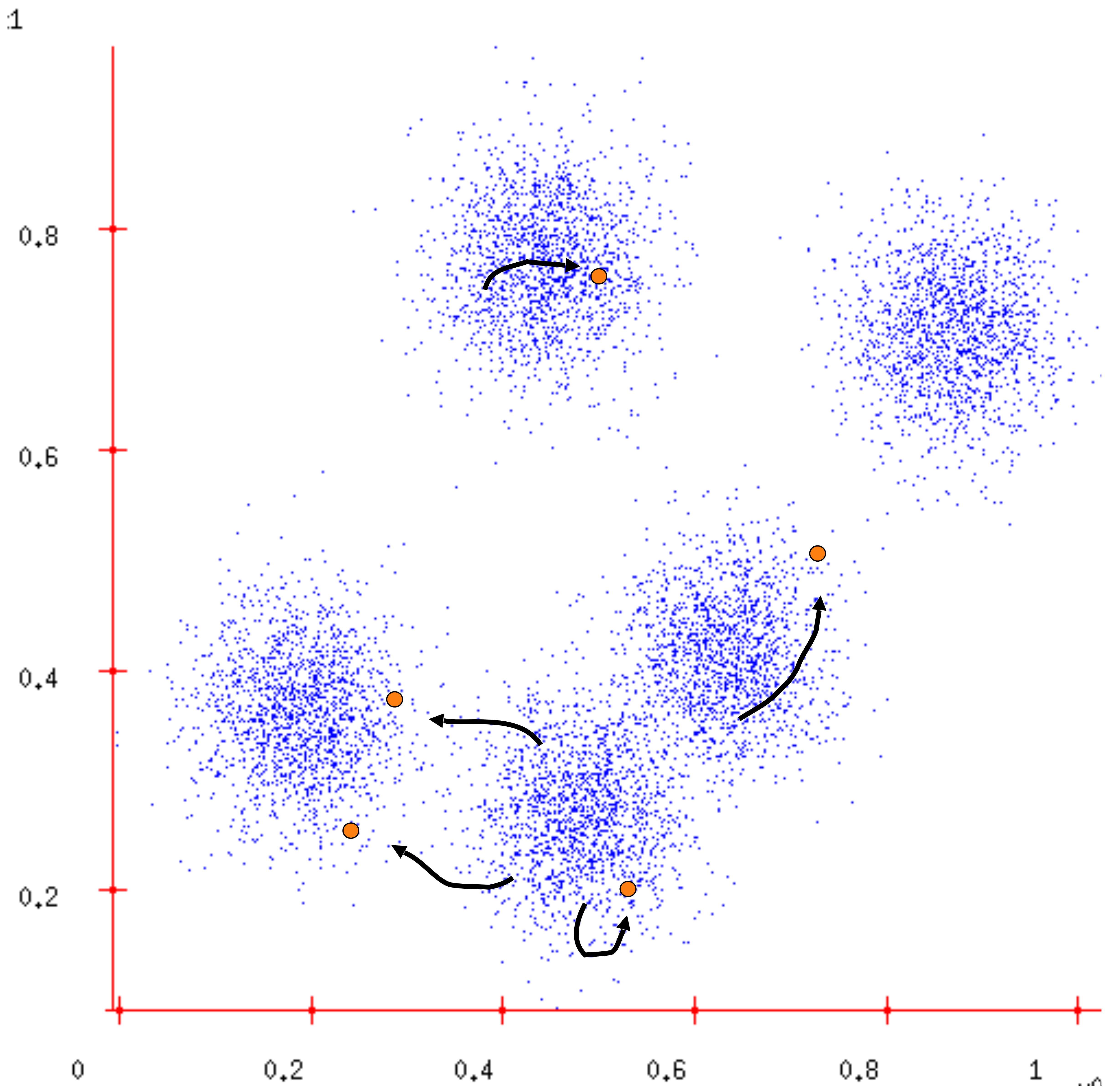
0 otherwise

- Iterative algorithm:
  - select *k* points as initial centroids (e.g. pick *k* random points from *X*)
  - **repeat**
    - form *k* clusters by assigning each point to its closest centroid
    - recompute the centroid of each cluster (e.g. as mean of its assigned points)
  - **until** convergence (e.g. centroids do not change)

1. Ask user how many clusters are needed.
   (e.g. k=5)
2. Randomly guess k cluster center locations.
3. Each datapoint finds out which center it is closest to.
   - Voronoi diagram in 2D
4. Each center finds the centroid of its closest points …

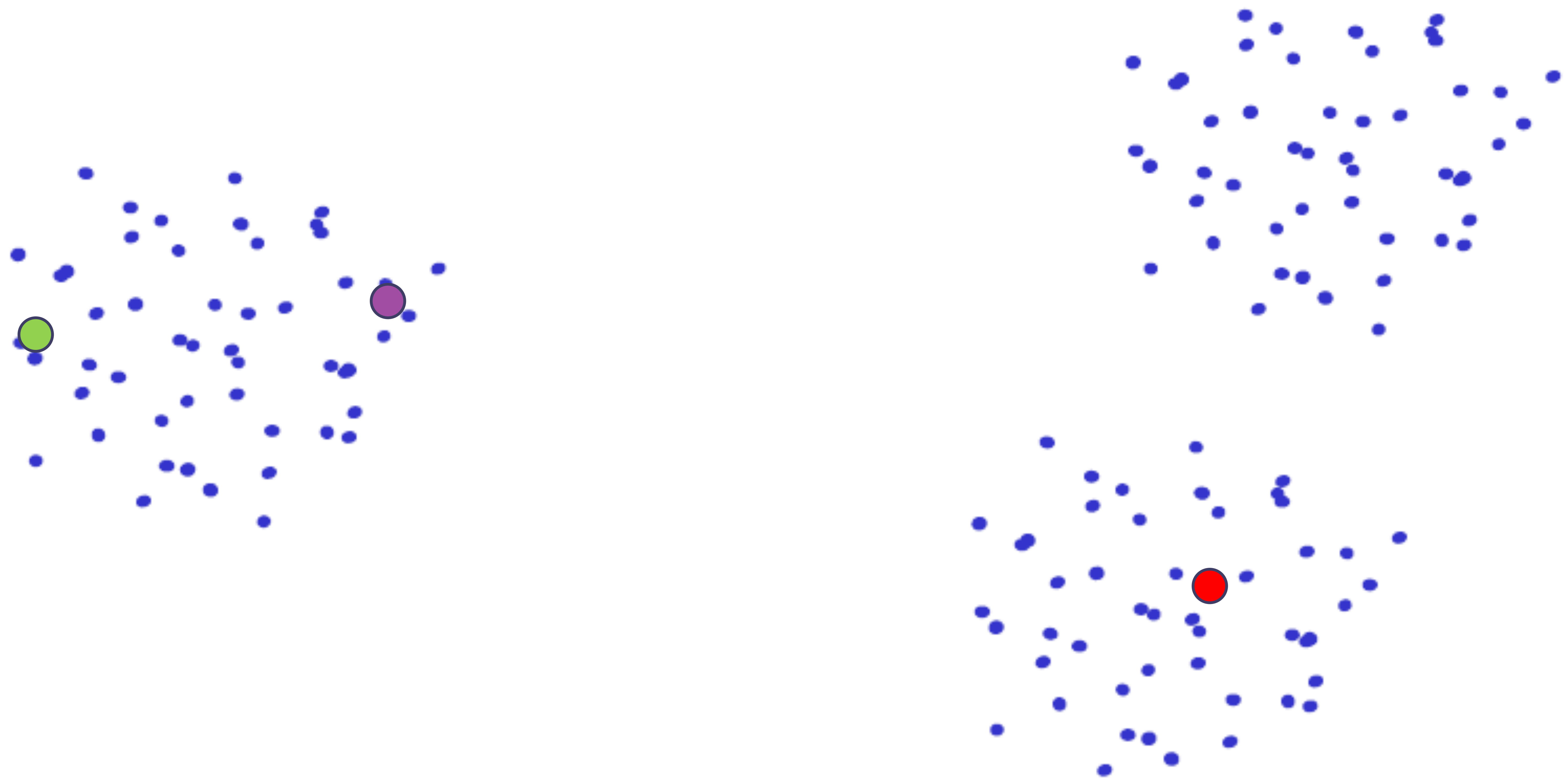


1. Ask user how many clusters are needed.
   (e.g. k=5)
2. Randomly guess k cluster center locations.
3. Each datapoint finds out which center it is closest to.
   - Voronoi diagram in 2D
4. Each center finds the centroid of its closest points …
5. …and jumps there
6. …Repeat with step 3. until terminated!
   - centroids don't move



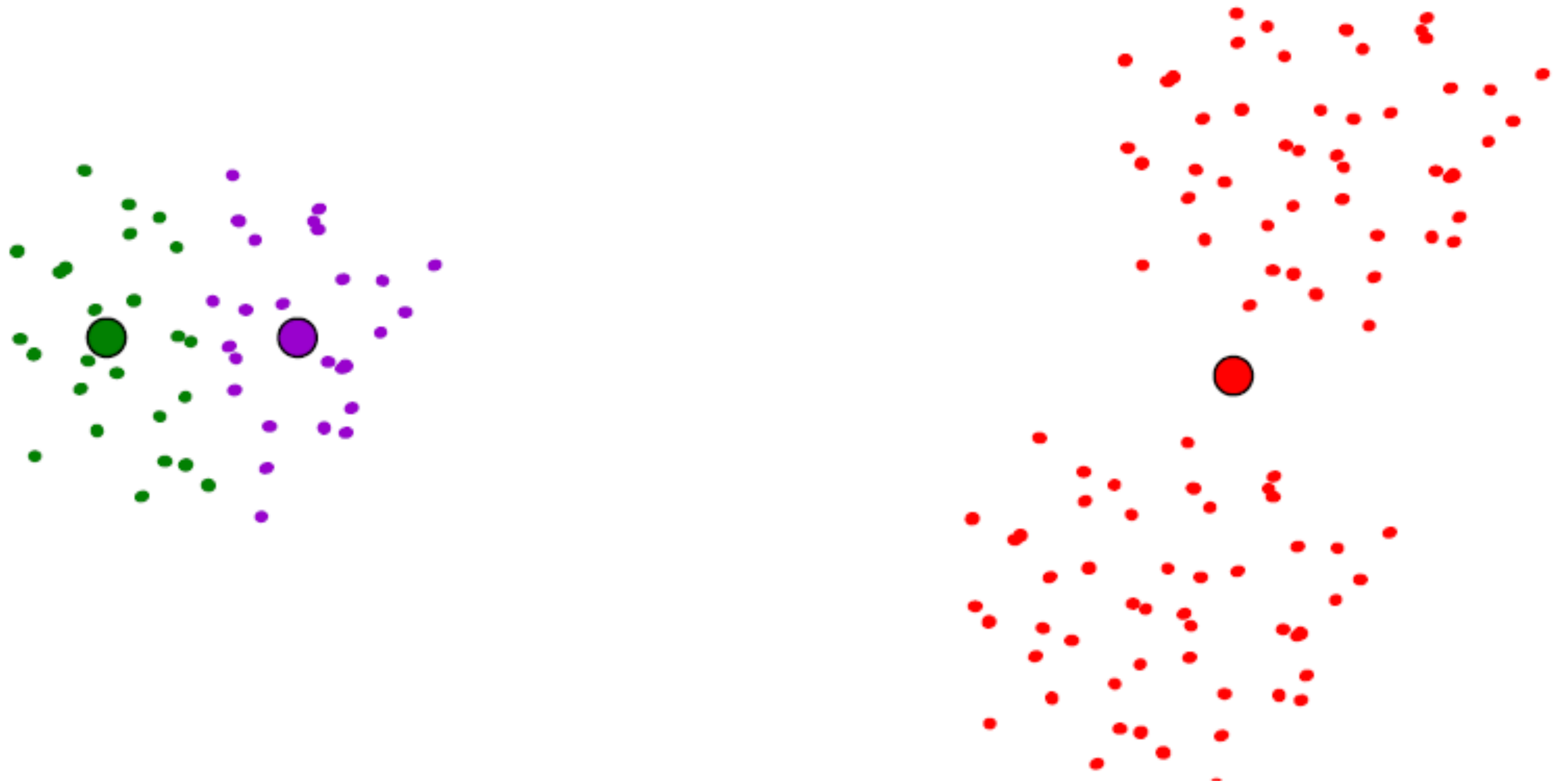
K-means

- Disadvantages
  - Dependent on initialization



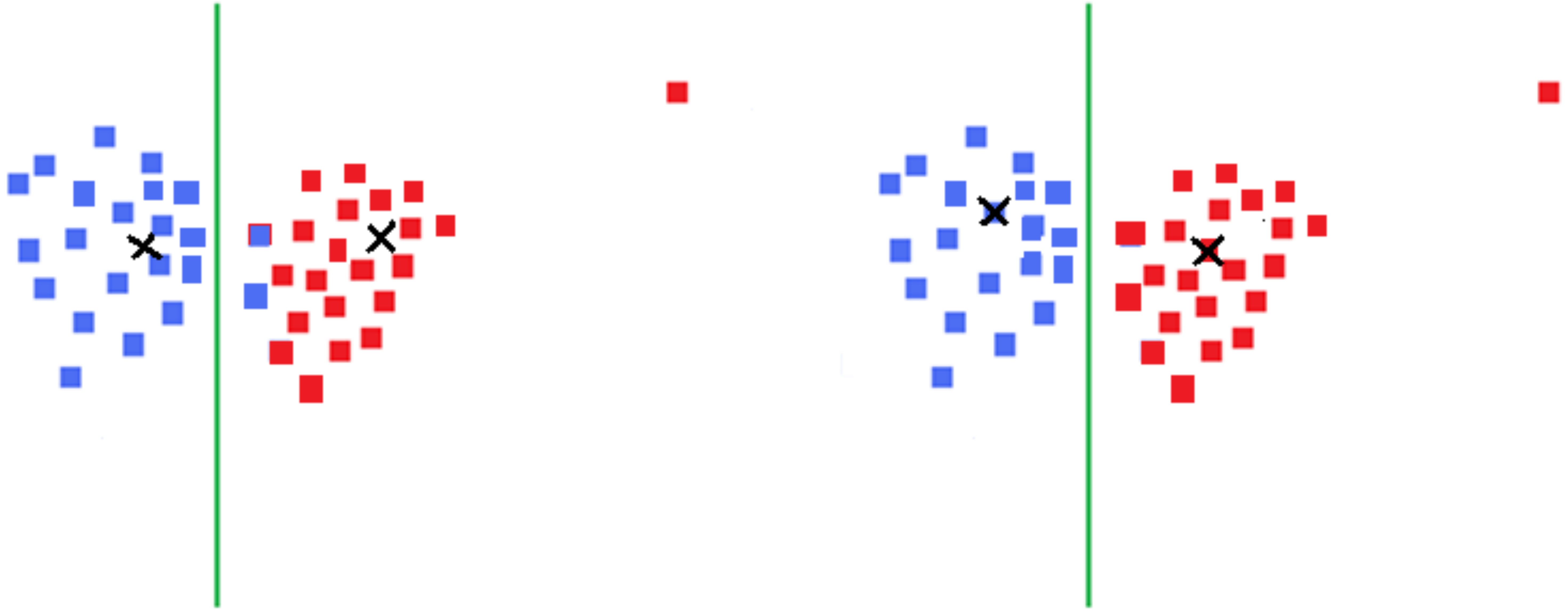
K-means

- Disadvantages
  - Dependent on initialization
    - select random seeds with at least $D_{\min}$ distance
    - or, run the algorithm many times



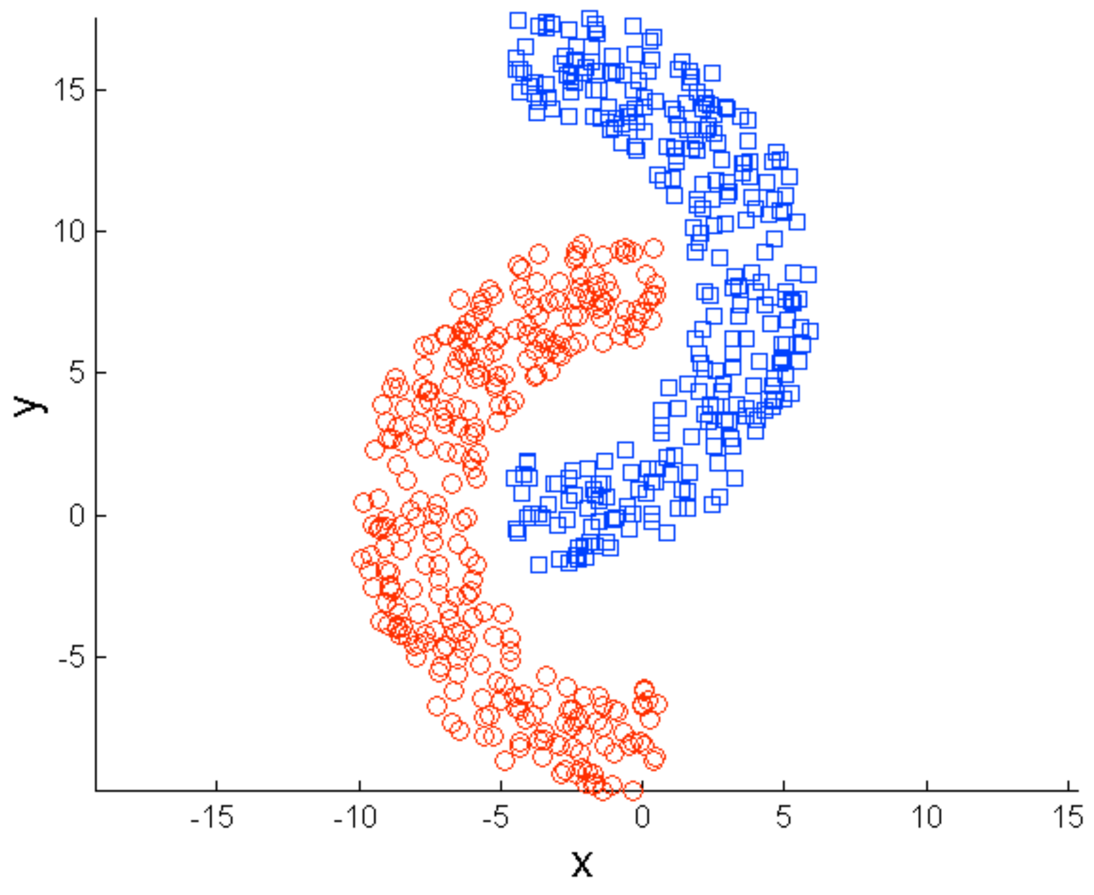
K-means

- Disadvantages
  - Dependent on initialization
  - Sensitive to outliers
    - use K-medoids: update centroids by picking best data point from those currently assigned to the cluster

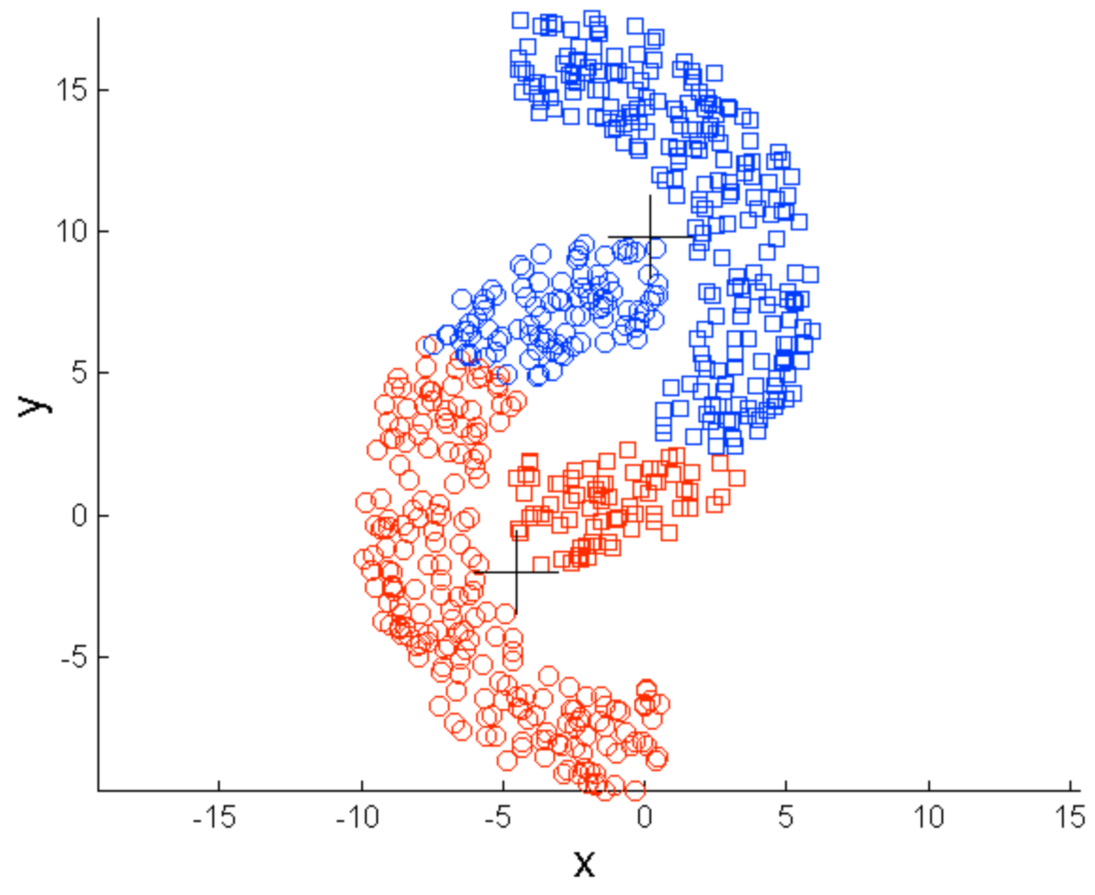


K-means

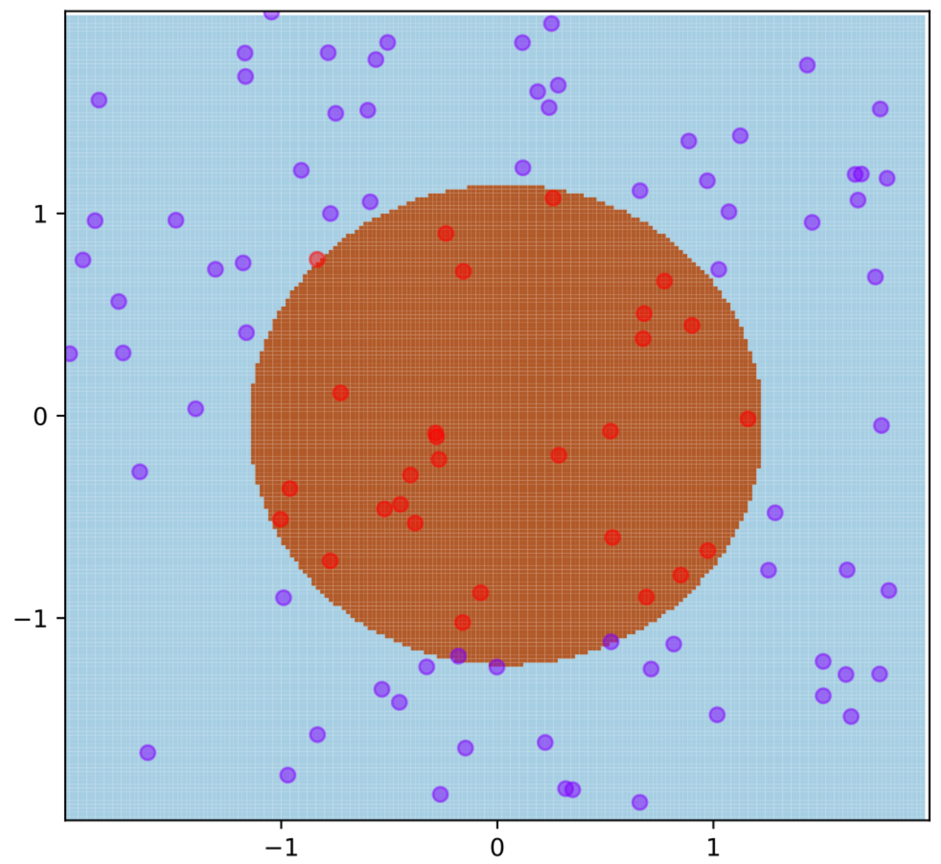
- Disadvantages
  - Dependent on initialization
  - Sensitive to outliers (K-medoids)
  - Can deal only with clusters with spherical symmetrical point distribution
    - *kernel trick*: corresponds to transforming data points into a higher-dimensional space in which they can be more easily separated

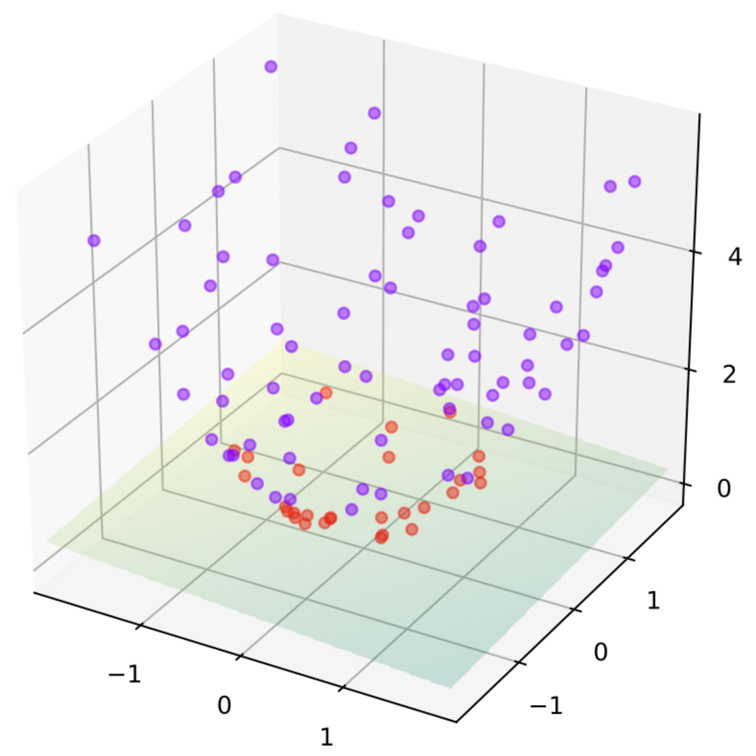


Original Points



K-means (2 Clusters)





K-means

- Disadvantages
  - Dependent on initialization
  - Sensitive to outliers (K-medoids)
  - Can deal only with clusters with spherical symmetrical point distribution
  - Deciding K

Deciding K

- Try a couple of K and check objective function

$$E(\Gamma, V) = \sum_{j=1}^{k} \sum_{i=1}^{n} \gamma_{ij} \|\boldsymbol{x}_i - \boldsymbol{v}_j\|^2$$

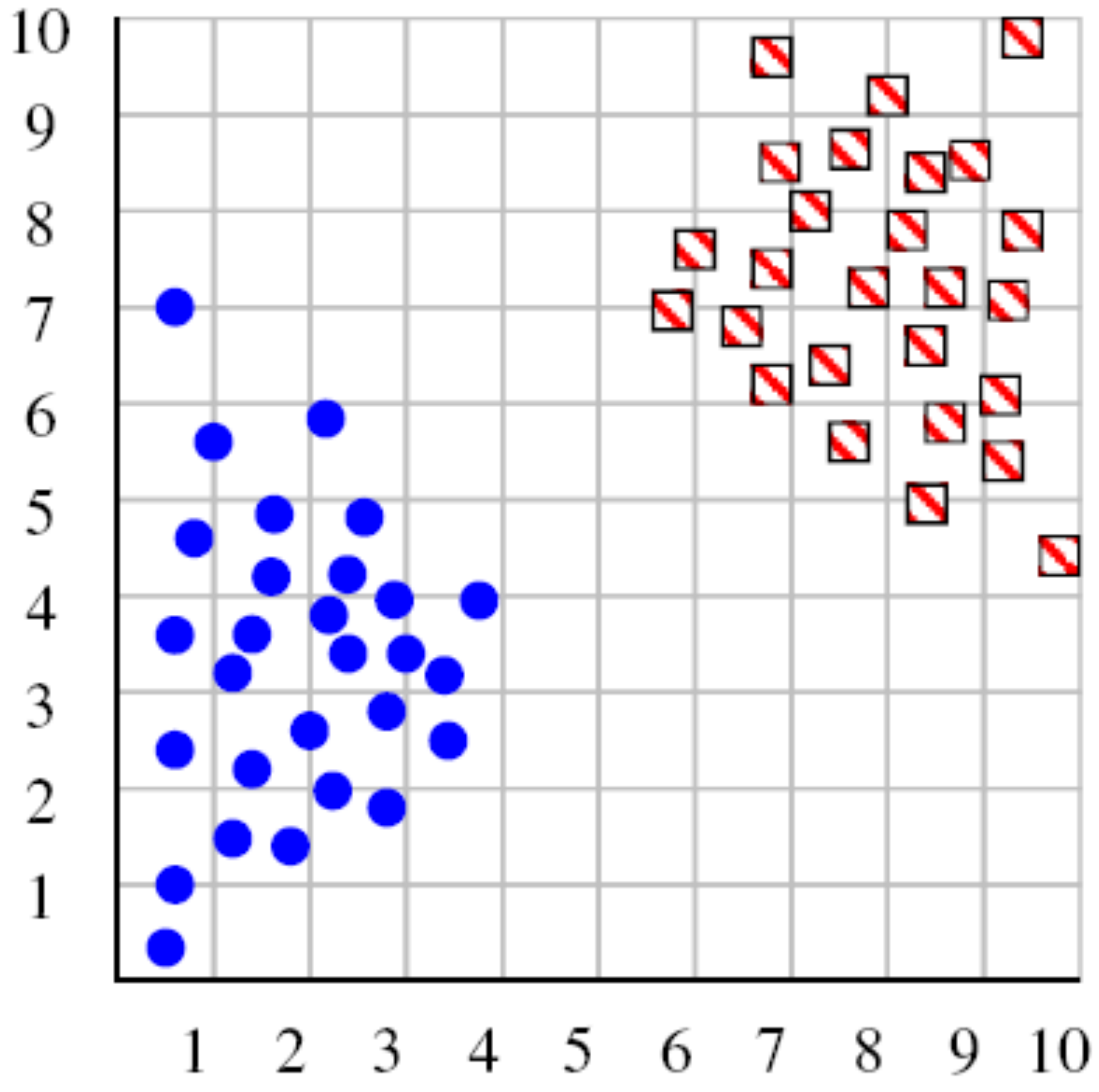


Image: Henry Lin

Deciding K

- When k = 1, the objective function is 873.0

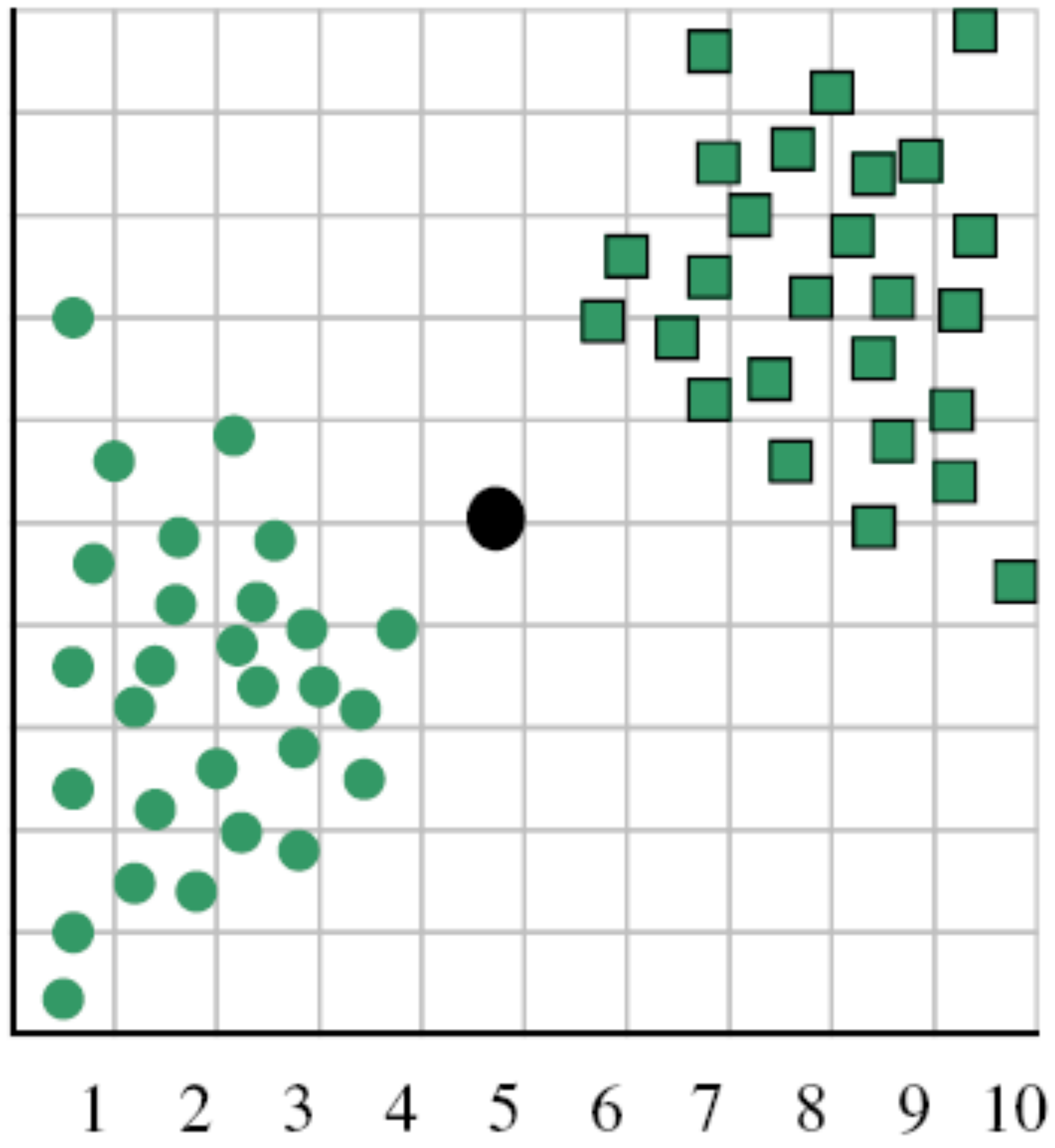


Image: Henry Lin

Deciding K

- When k = 2, the objective function is 173.1

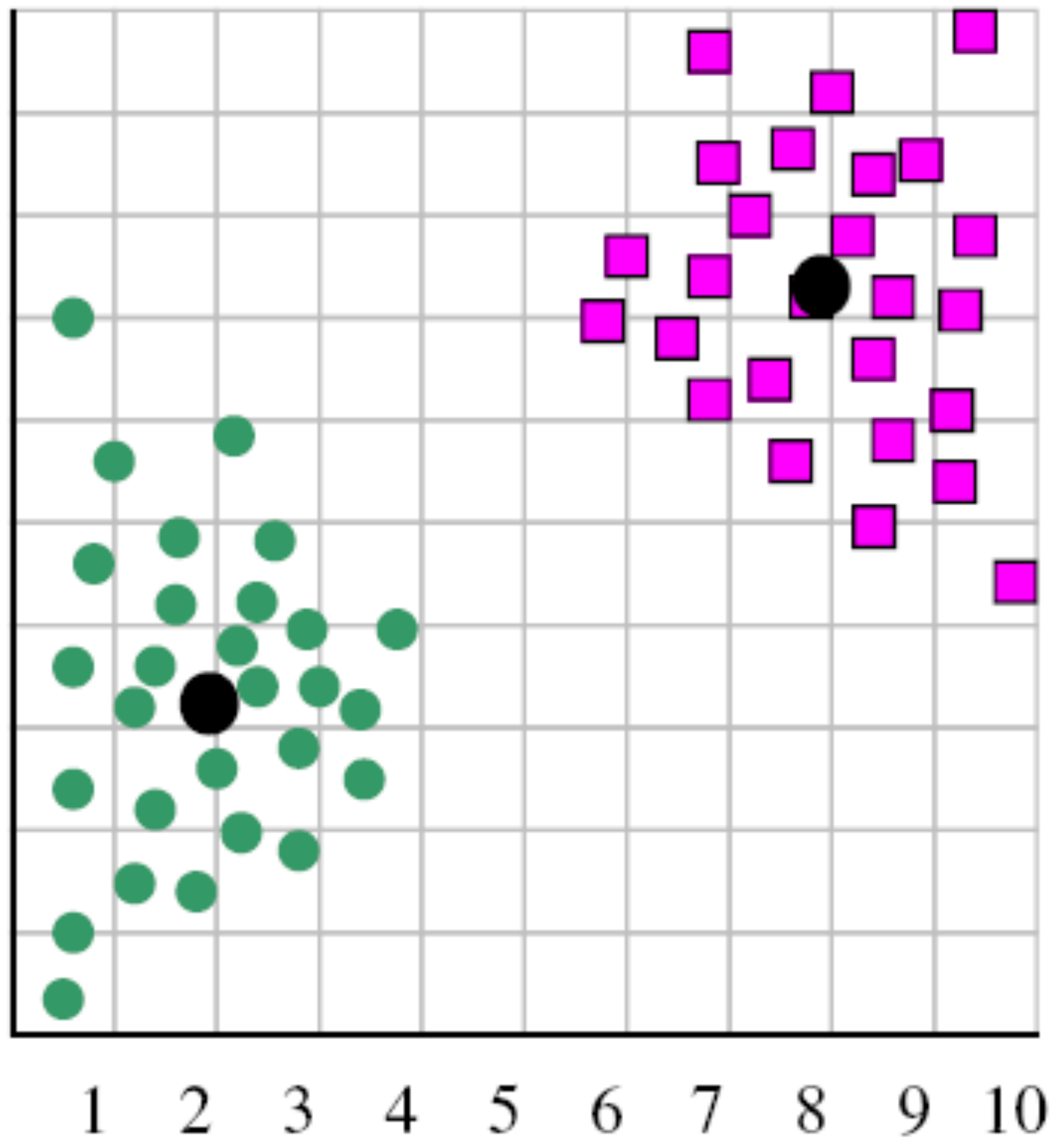


Image: Henry Lin

Deciding K

- When k = 3, the objective function is 133.6

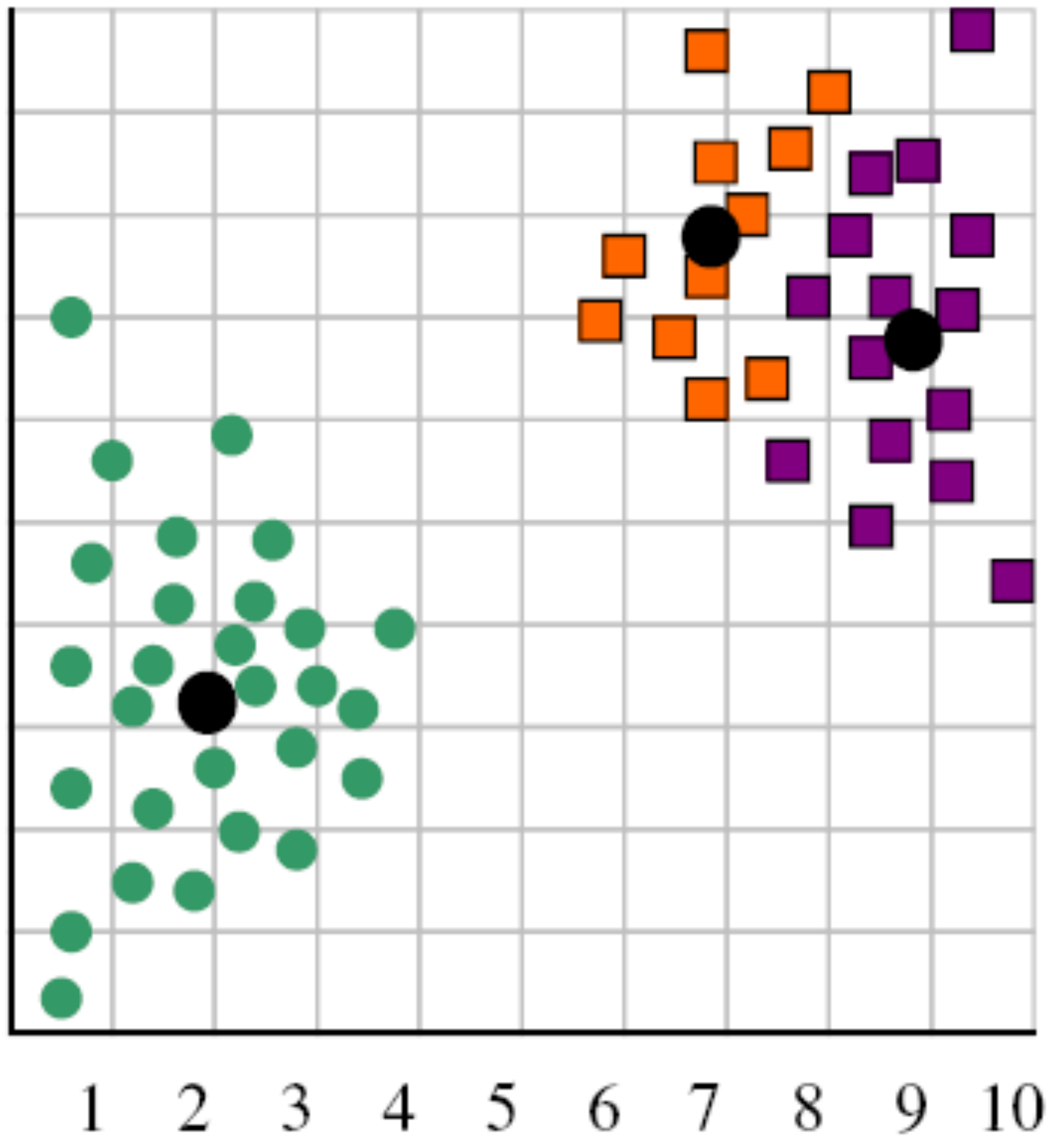


Image: Henry Lin

Deciding K

- We can plot objective function values for k=1 to 6
- The abrupt change at k=2 is highly suggestive of two clusters
  - "knee finding" or "elbow finding"
- Note that the results are not always as clear cut as in this toy example

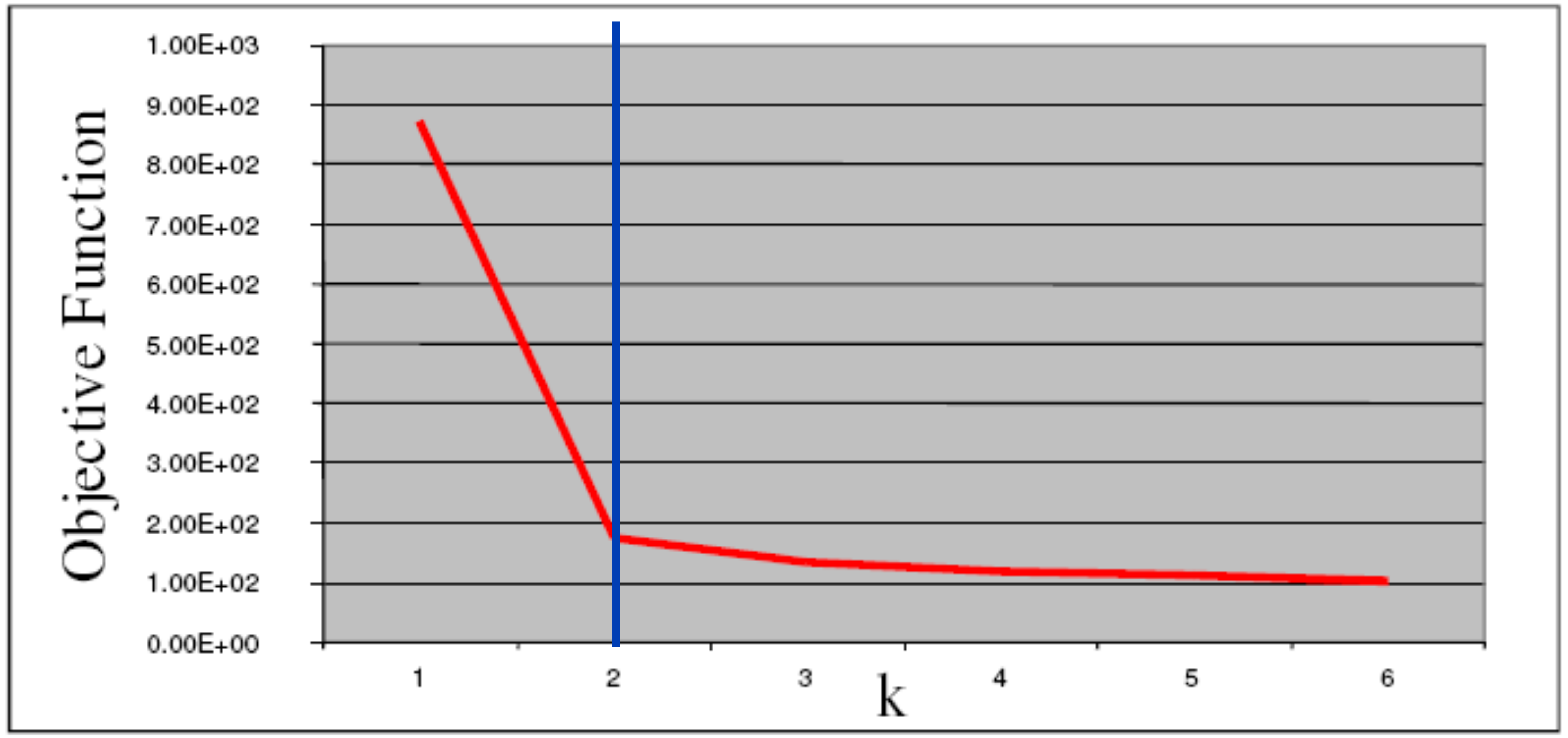
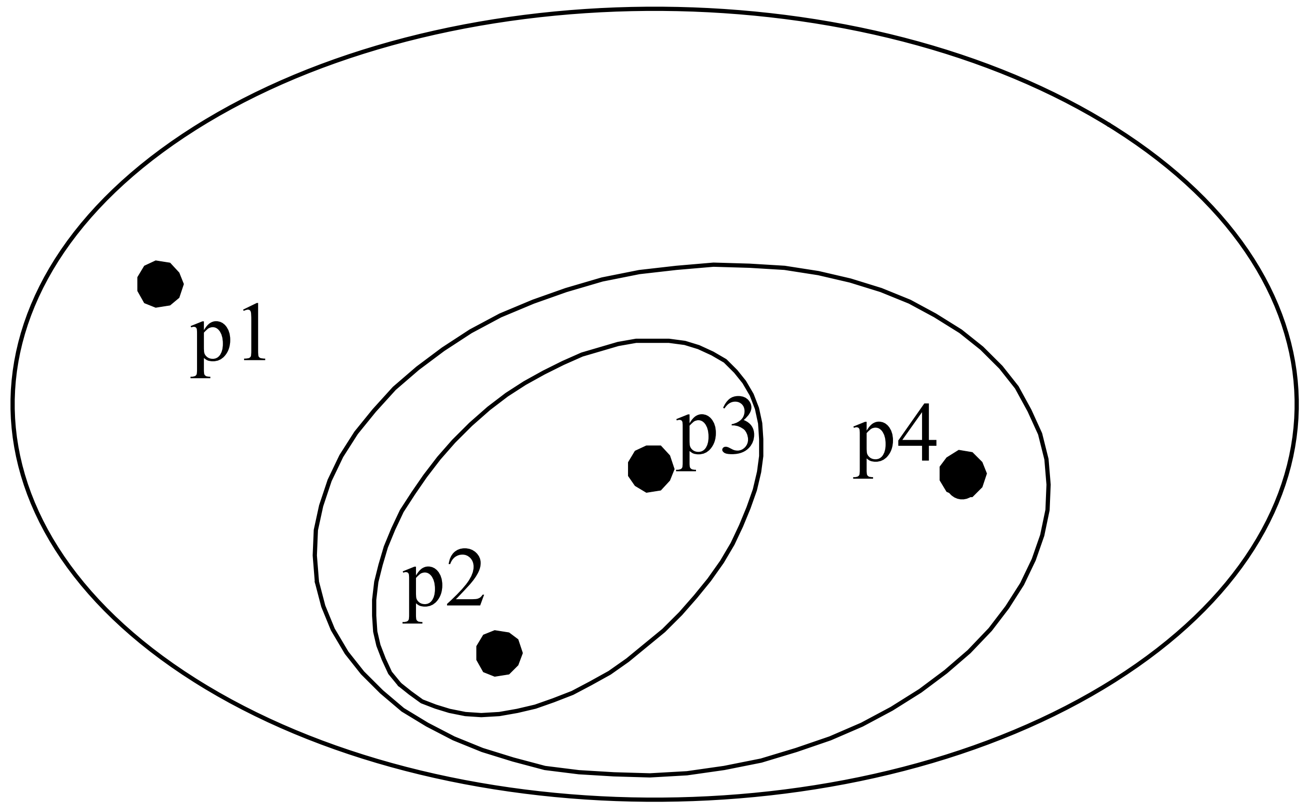
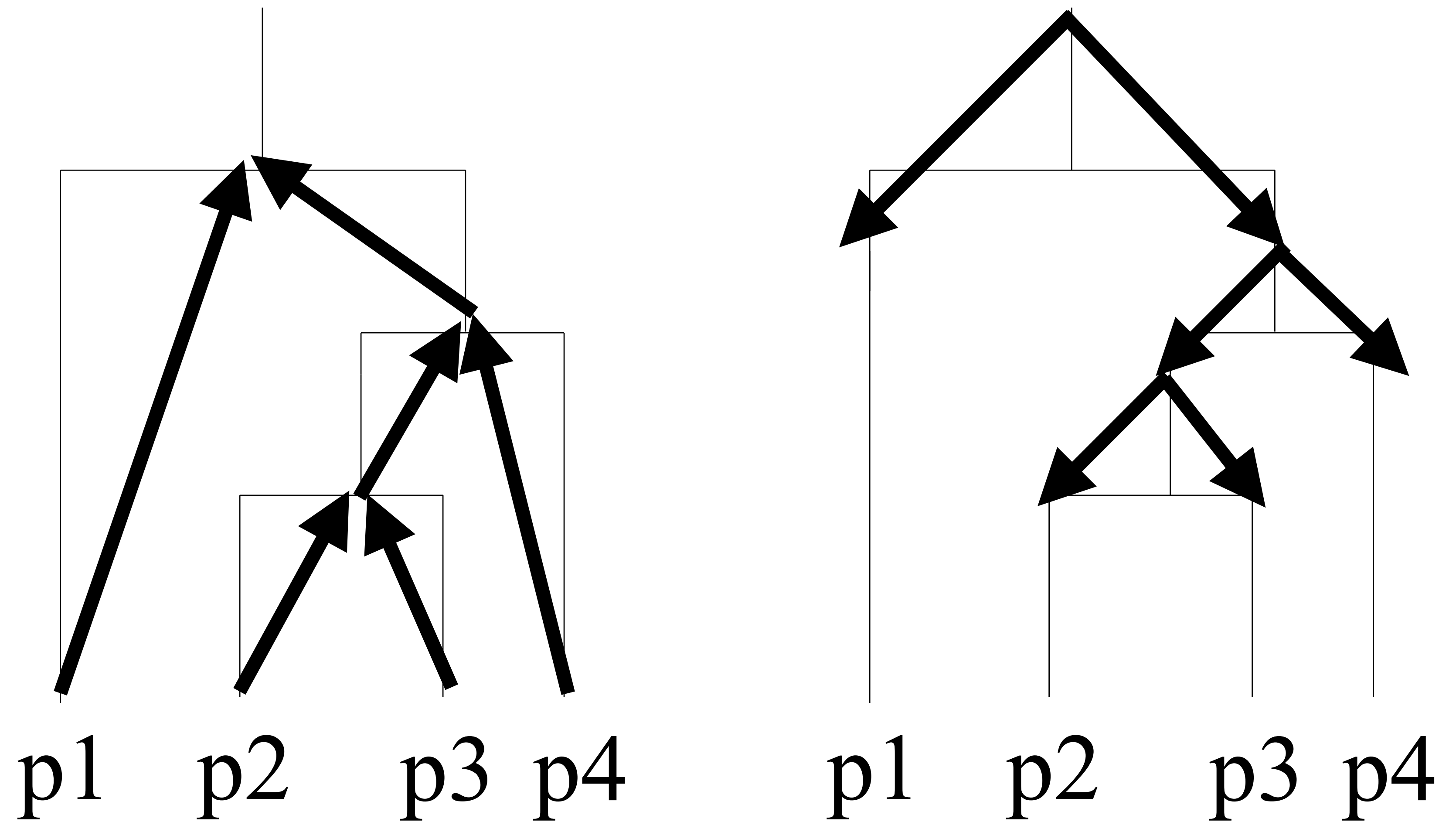


Image: Henry Lin

Hierarchical clustering

- Produces overlapping (nested) clusters that form a hierarchy
- Hierarchy displayed graphically in a tree structure called *dendrogram*
  - Cluster-subcluster relationships
  - Order in which clusters were produced
- Two possible strategies:
  - Agglomerative (bottom-up) clustering
    - start with one cluster per data point, merge pairs of clusters
  - Divisive (top-down) clustering
    - start with one combined cluster for all data points, perform splits recursively



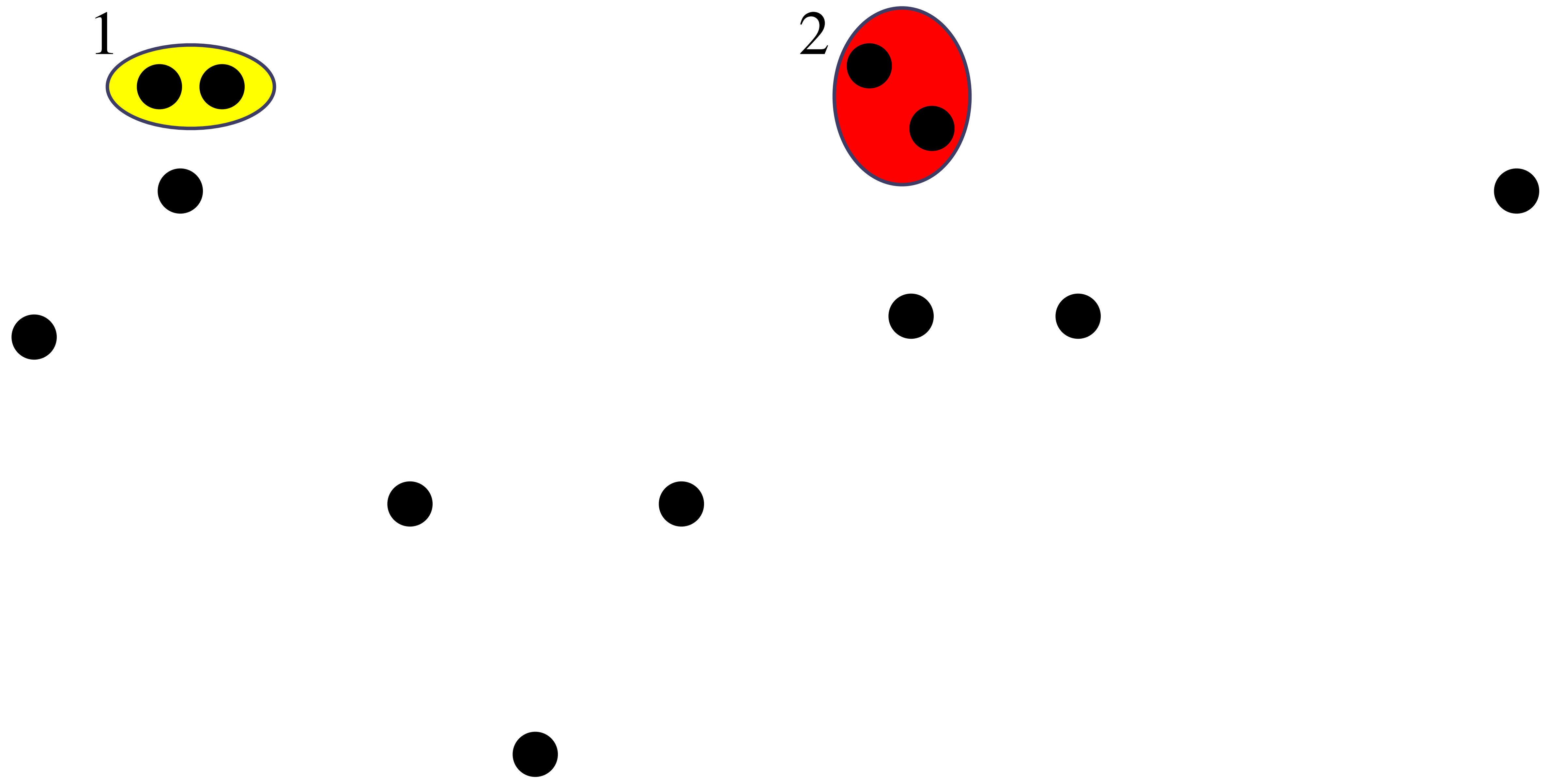


Agglomerative (bottom-up) Clustering

- General iterative algorithm:
  - Place each object into a cluster of its own
  - Compute the *proximity matrix*
    - all pairwise object-to-object distances/similarity scores
    - inter-cluster distances/similarity scores initialized from pairwise object-to-object values
  - **repeat**
    - merge the closest two clusters
      - replace the two clusters with the new cluster
    - update proximity matrix
      - re-compute inter-cluster distances/similarity scores w.r.t. the new cluster
  - **until** only *k* cluster remain (*k* can be 1)

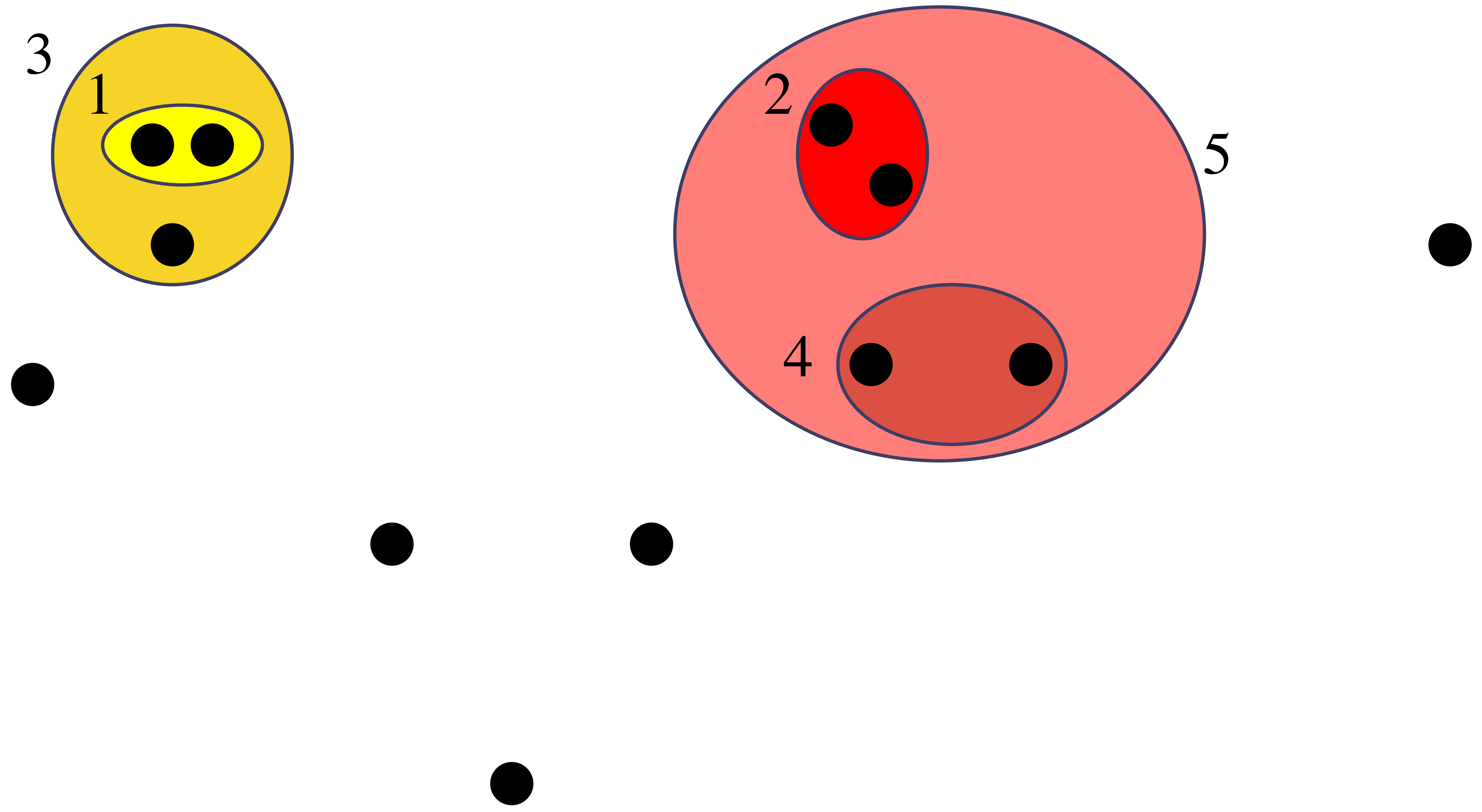
Agglomerative (bottom-up) Clustering

- Agglomerative hierarchical clustering, with $k = 3$
- 2nd iteration



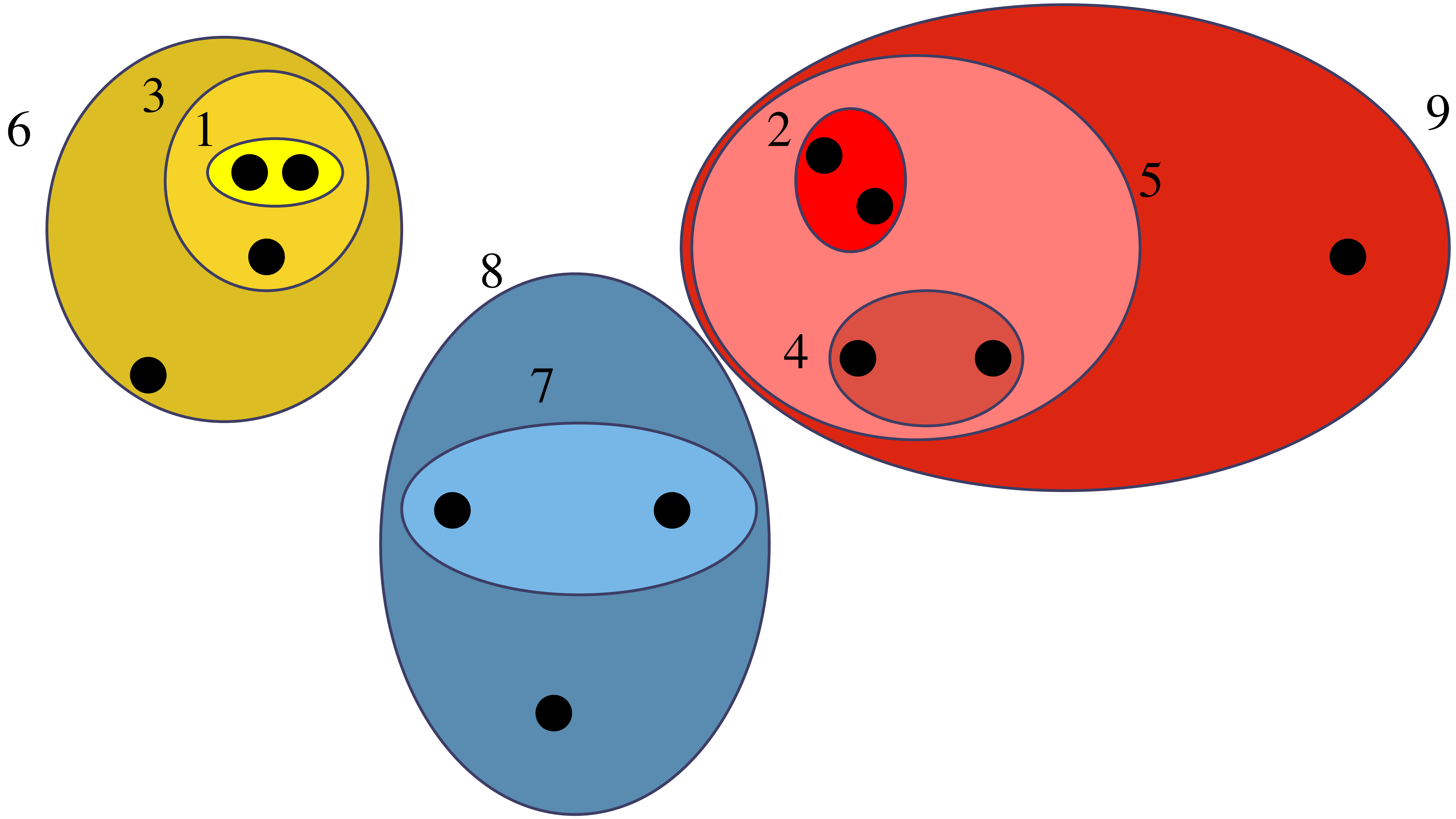
Agglomerative (bottom-up) Clustering

- Agglomerative hierarchical clustering, with *k = 3*
- 5th iteration



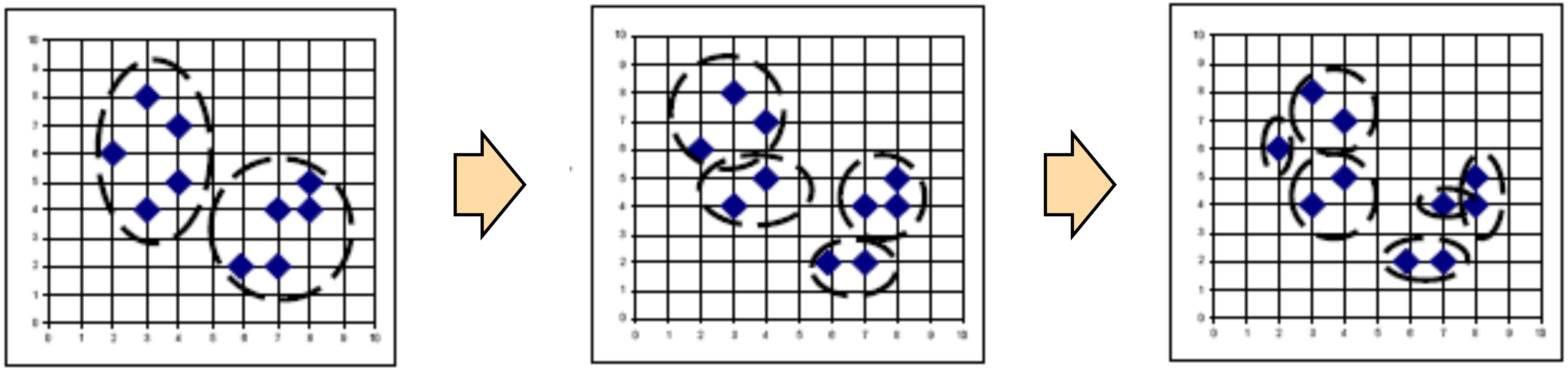
Agglomerative (bottom-up) Clustering

- Agglomerative hierarchical clustering, with *k = 3*
- 9th iteration
- Reached target of *k* clusters
  - Terminate



Divisive (top-down) clustering

- General recursive algorithm:
  - Place all input objects into *one* single cluster
  - **if** cluster has less than *n* objects (e.g. $n = 3$), **return** this cluster
  - **else**
    - split cluster into two clusters C_1, C_2 using flat clustering algorithm (k-means, k=2)
    - perform top-down recursive clustering of C_1
    - perform top-down recursive clustering of C_2



Bottom-up vs. Top-down

- Which one is more complex?
  - Top-down requires a flat clustering “subroutine”
  - Bottom-up requires updating inter-cluster distance/similarity scores
- Which one is more efficient?
  - Top-down
  - For a *fixed number* of top levels, using an efficient flat algorithm like K-means, divisive algorithms are linear in practice in the number of objects and clusters
  - Agglomerative algorithms are at least quadratic
- Which one is more accurate?
  - Top-down
  - Bottom-up methods make clustering decisions based on local object information without initially taking into account the global distribution
    - early decisions cannot be undone
  - Top-down clustering benefits from complete information about the global distribution when making top-level partitioning decisions

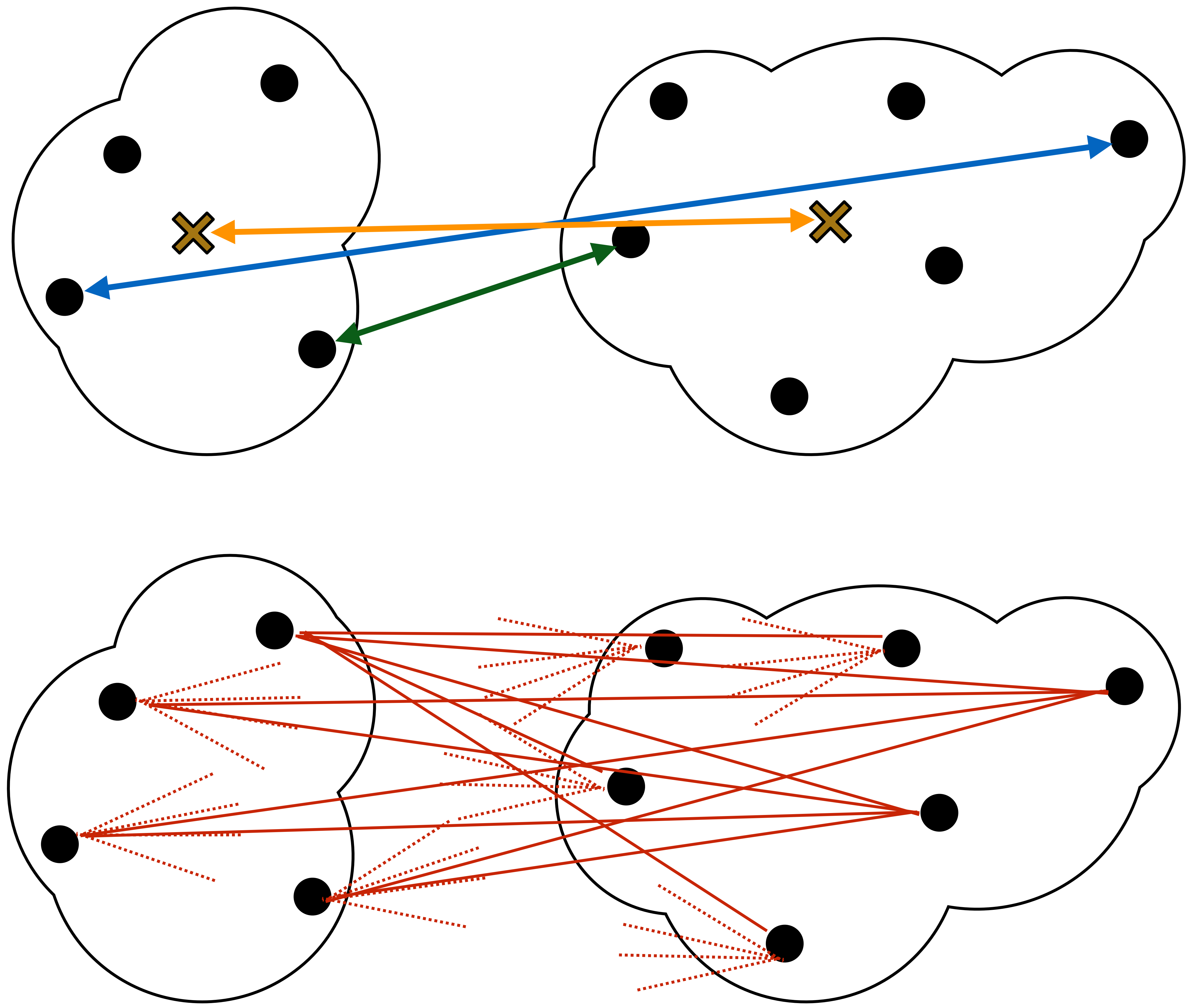
Strength and Weakness of Hierarchical Clustering

- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by ‘cutting’ the tree hierarchy, dendogram at the desired level
- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)
- Standard hierarchical agglomerative clustering has a time complexity of $O(n^3)$ and requires $\Omega(n^2)$ memory
  - Too slow for even medium sized datasets
  - Runtime of the general (agglomerative) case can be reduced to $O(n^2 \log n)$ using a heap structure
    - but further increasing the memory requirements
- Divisive (top-down) clustering can run in of $O(n^2)$

Inter-Cluster Similarity

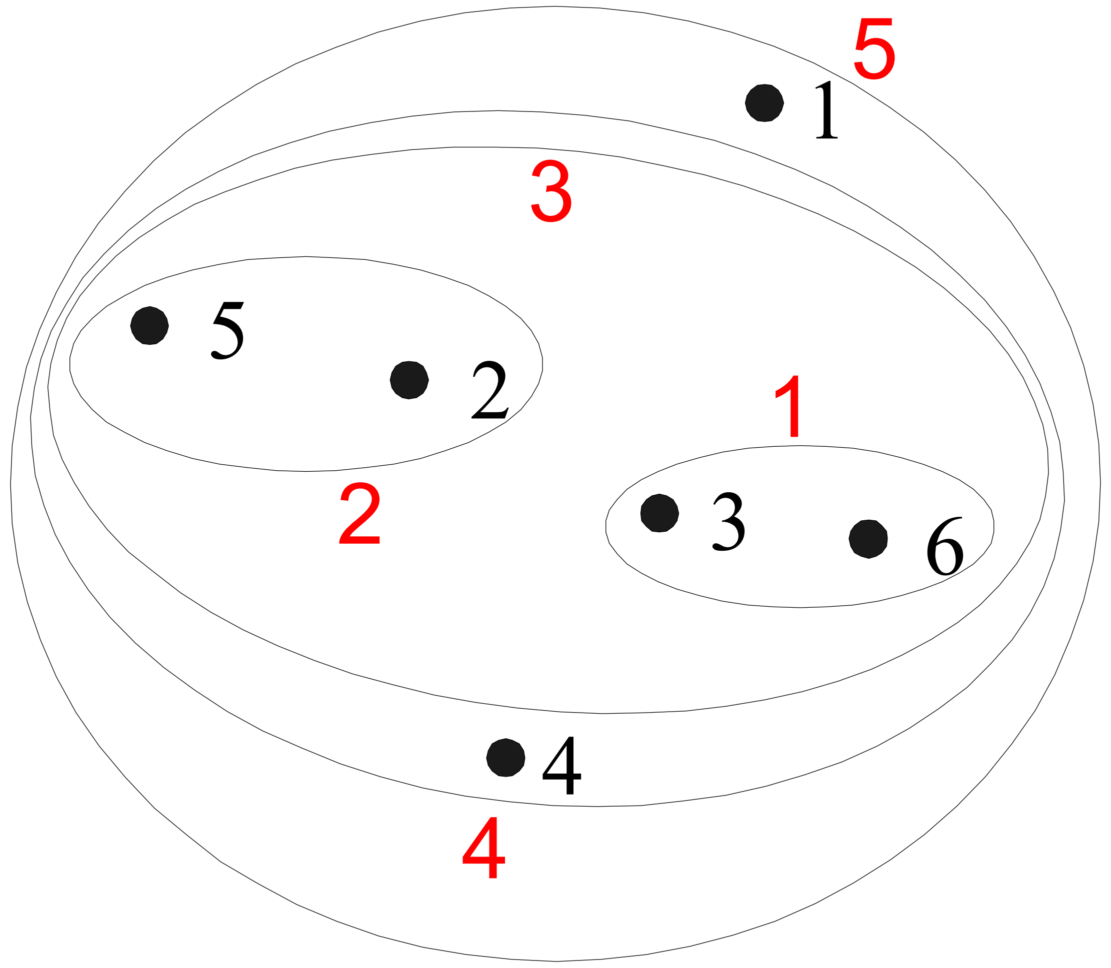
- Given two clusters $\boldsymbol{A}$ and $\boldsymbol{B}$, how do we define their distance?
  - Determines which clusters should be combined (bottom-up) or where a cluster should be split (top-down)
- Need a measure of similarity between *sets of elements*, which involves:
  - A distance metric $d(a, b)$ (e.g. Euclidean), applied to extract pairwise object-to-object distances
  - A *linkage* criterion
- Linkage determines the distance between sets as a function of pairwise distances
  - *Single-linkage*: minimum distance between any object in the first cluster and any in the second: $\min\{d(a,b) : a \in A, b \in B\}$
    - distance defined by the two most similar objects
  - *Complete-linkage*: maximum distance between any object in the first cluster and any in the second: $\max\{d(a,b) : a \in A, b \in B\}$
    - distance defined by the two most dissimilar objects
  - *Centroid-linkage*: distance between centroids c_s and c_t of two clusters s and t: $d(c_s, c_t)$
  - *Average-linkage*: average distance between any object in the first cluster and any in the second: $\frac{1}{|A||B|}\sum_{a \in A}\sum_{b \in B} d(a,b)$

Inter-Cluster Similarity



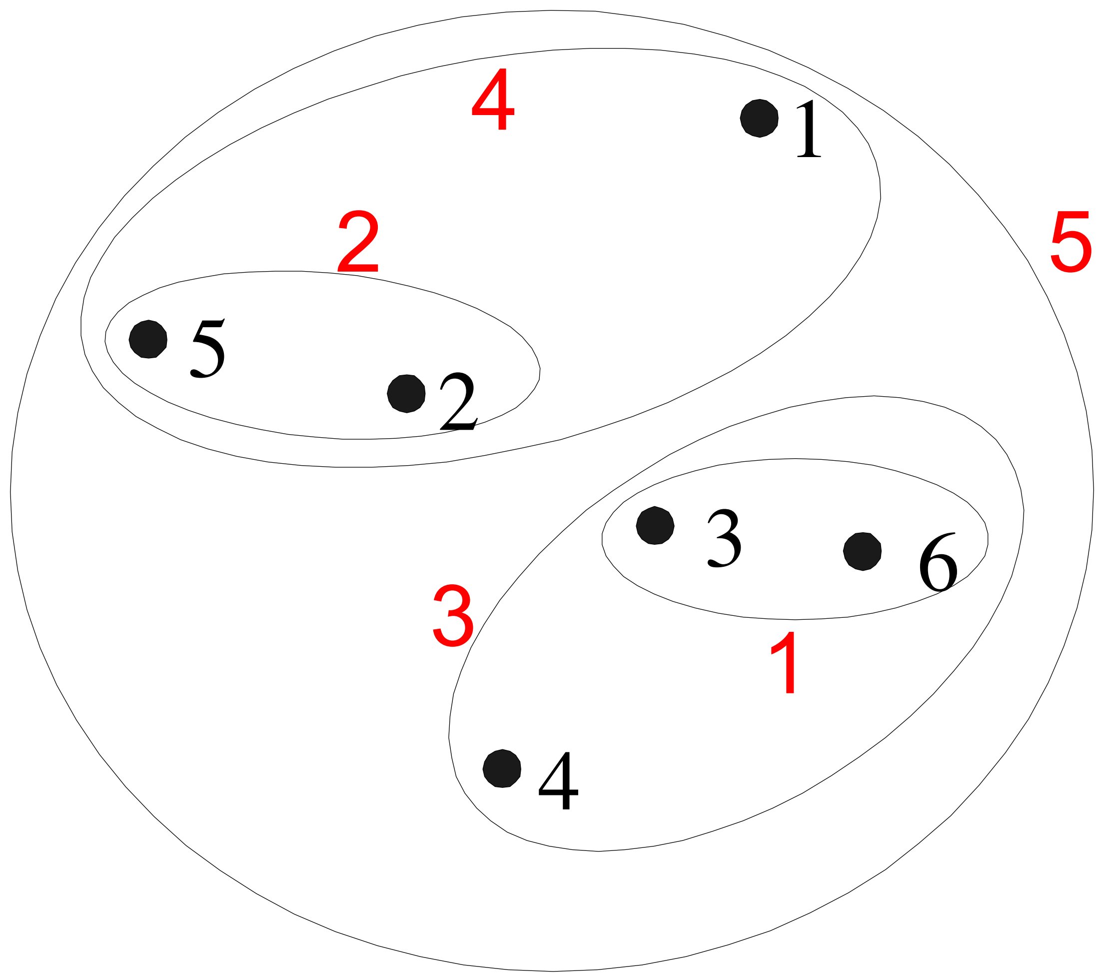
- Single-linkage: min
- Complete-linkage: max
- Centroid-linkage: distance between centroids
- Average-linkage: group average

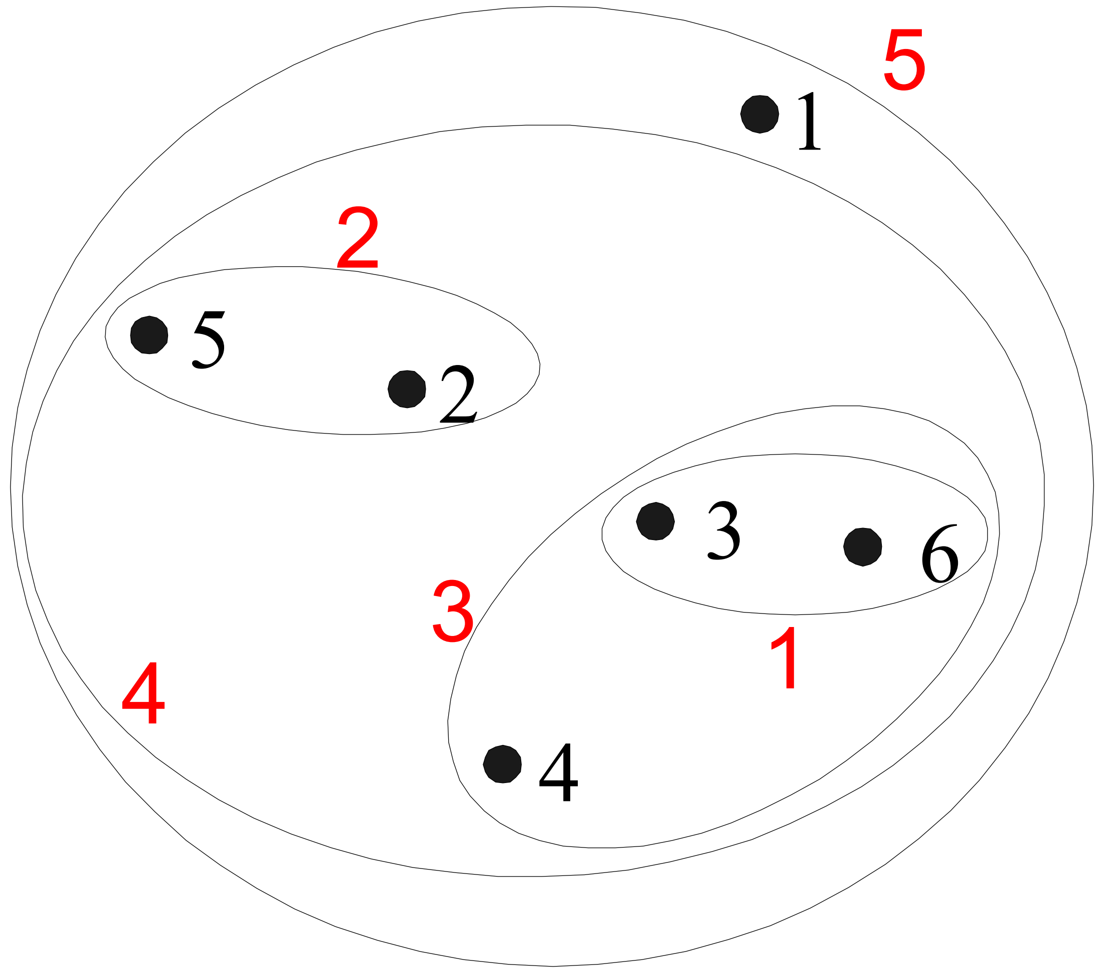
Comparison



MIN

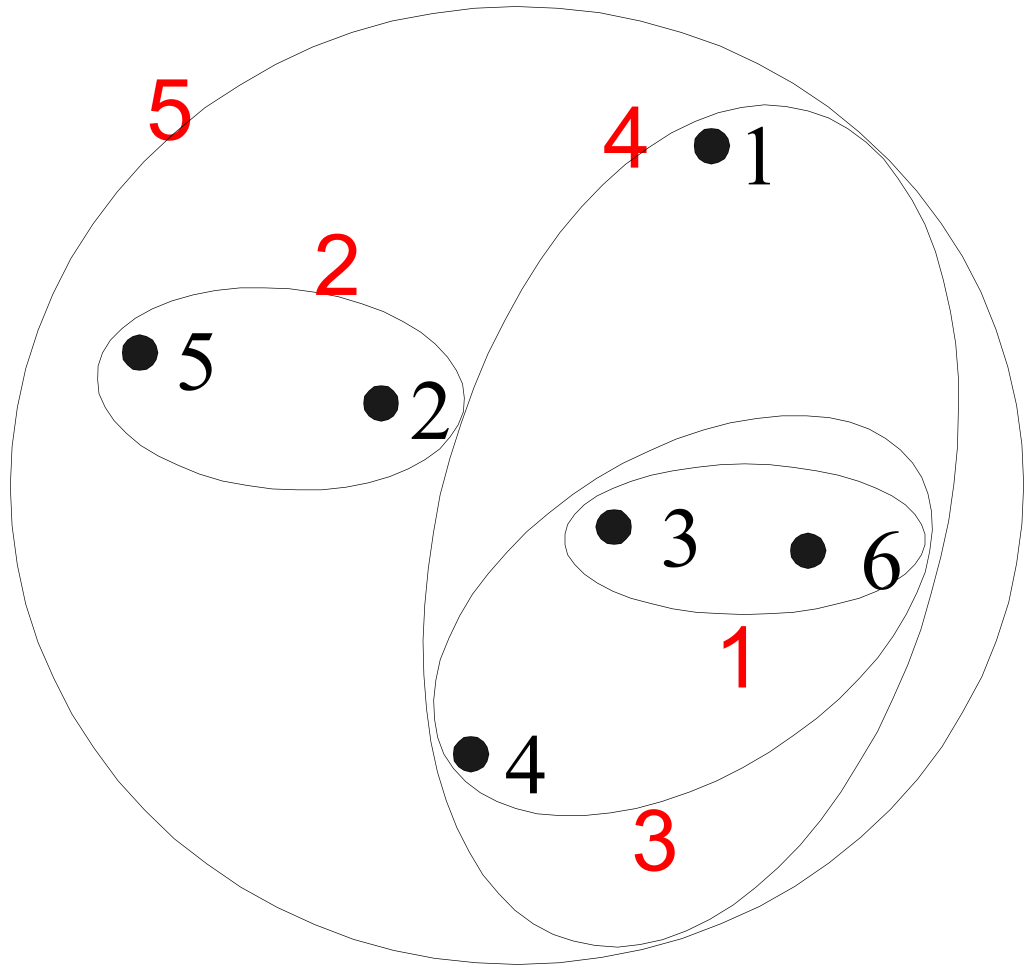
MAX





Group Average

Ward’s Method



Ward's method uses initial squared errors between data points and recursively updates inter-cluster distances as:

$$d(C_i \cup C_j, C_k) = \frac{n_i + n_k}{n_i + n_j + n_k} d(C_i, C_k) + \frac{n_j + n_k}{n_i + n_j + n_k} d(C_j, C_k) + \frac{n_k}{n_i + n_j + n_k} d(C_i, C_j)$$

Recap

- **Clustering:** grouping of data points by similarity/distance; connectivity, centroid, distribution or density based clustering; hard/soft, partitional/hierarchical clustering properties
- **Cluster separation:** well-separated, center-based, contiguous
- **K-means:** clustering algorithm, alternates (a) move centroids and (b) reassign points to closest centroid
- **Hierarchical clustering:** agglomerative (bottom-up) or divisive (top-down), strengths and weaknesses
- **Cluster dissimilarity:** single, complete, average, centroid linkage as distance measures between sets of data points