Clustering Techniques

Clustering

- Hard vs. Soft
  - Hard: same object can only belong to single cluster
  - Soft: same object can belong to different clusters

Slide credit: Min Zhang

Clustering

- Hard vs. Soft
  - Hard: same object can only belong to single cluster
  - Soft: same object can belong to different clusters
    - E.g. Gaussian mixture model

Slide credit: Min Zhang

Clustering

- Flat vs. Hierarchical
  - Flat: clusters are flat
  - Hierarchical: clusters form a tree
    - Agglomerative
    - Divisive

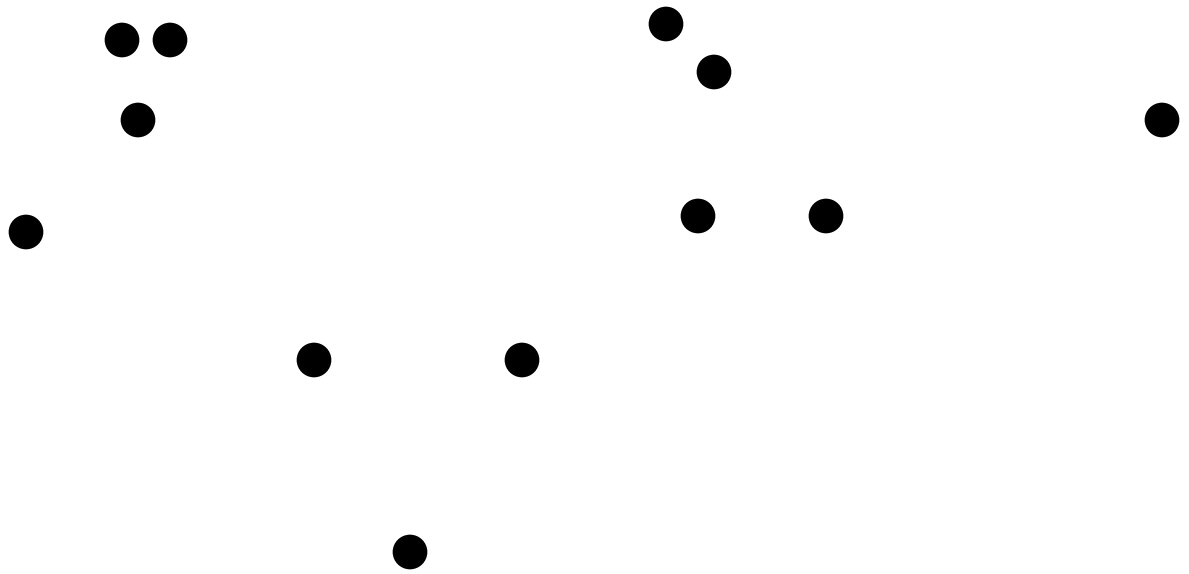
Hierarchical clustering

- Agglomerative (Bottom-up)
  - Compute all pair-wise pattern-pattern similarity coefficients
  - Place each of *n* patterns into a class of its own
  - Merge the two most similar clusters into one
    - Replace the two clusters into the new cluster
    - Re-compute inter-cluster similarity scores w.r.t. the new cluster
  - Repeat the above step until there are *k* clusters left (*k* can be 1)

Slide credit: Min Zhang

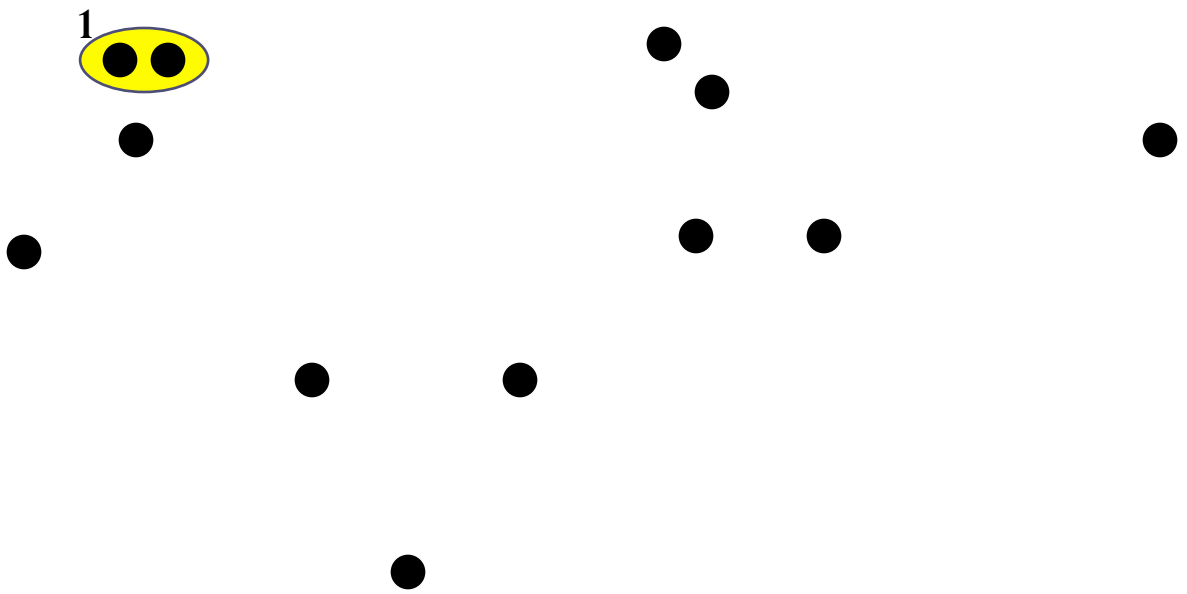
Hierarchical clustering

- Agglomerative (Bottom up)



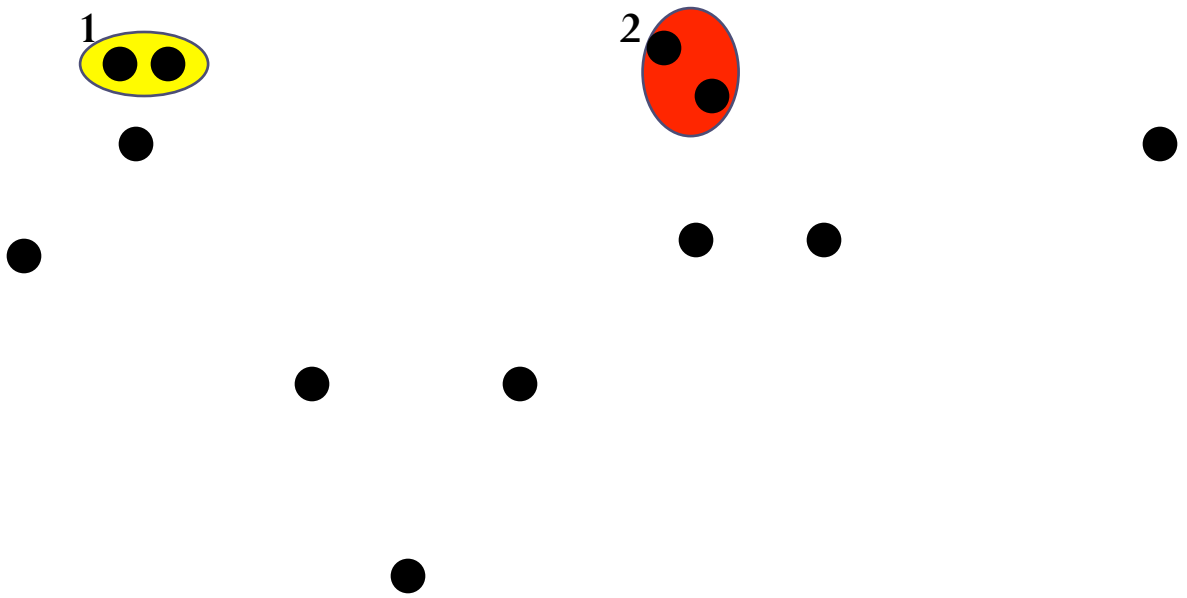
Hierarchical clustering

- Agglomerative (Bottom up)
- 1st iteration



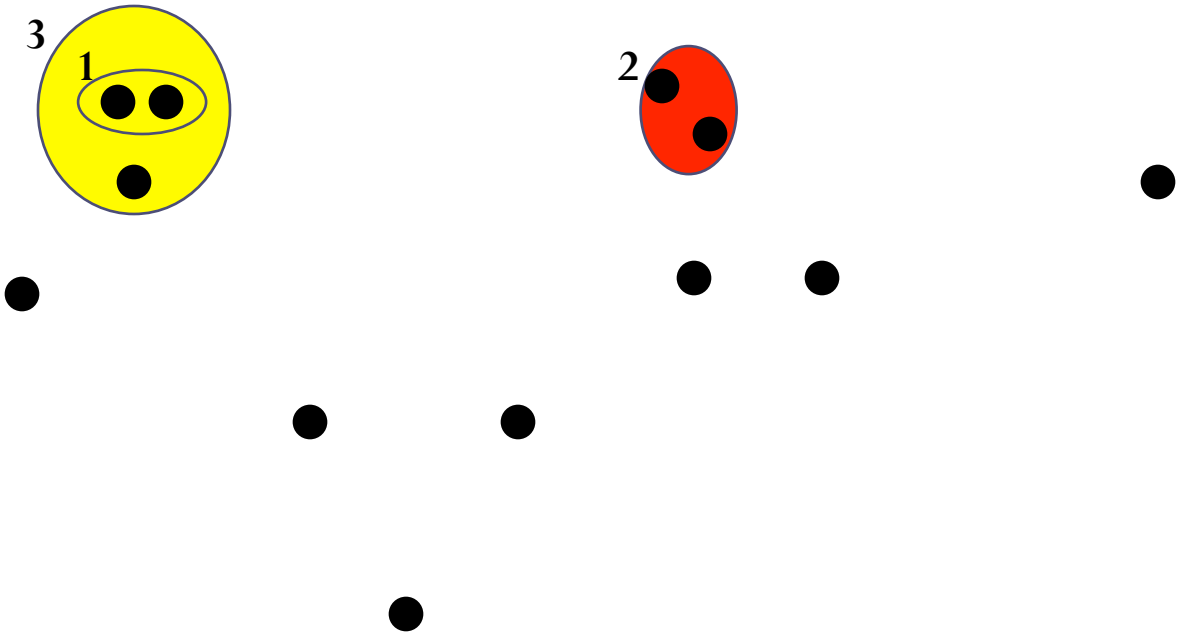
Hierarchical clustering

- Agglomerative (Bottom up)
- 2nd iteration



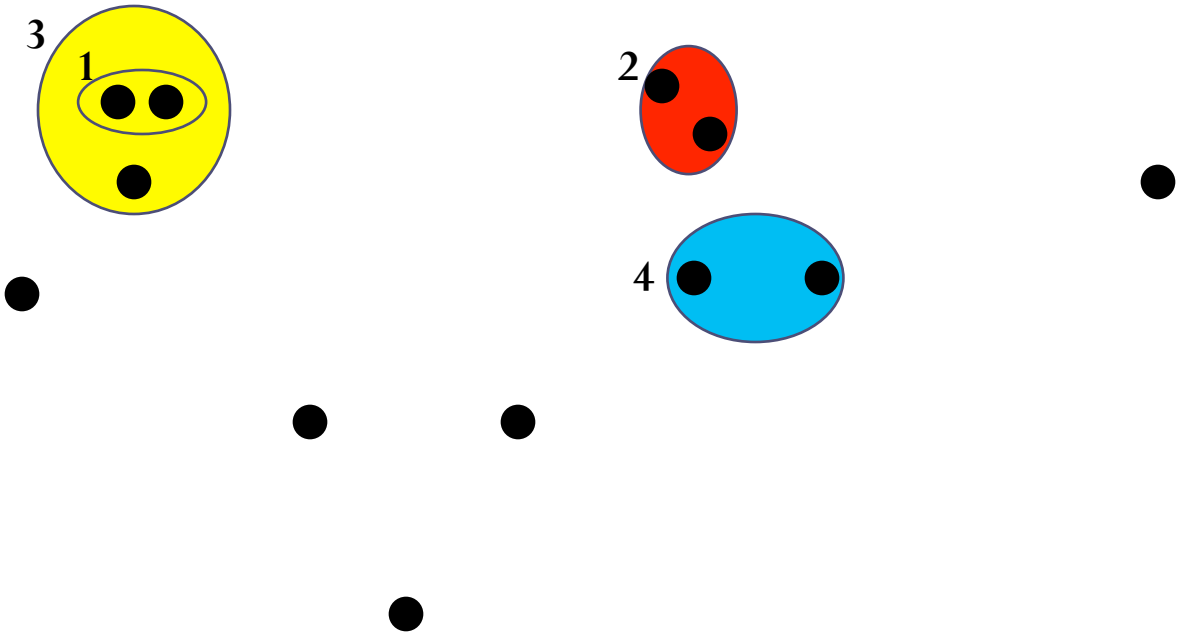
Hierarchical clustering

- Agglomerative (Bottom up)
- 3rd iteration



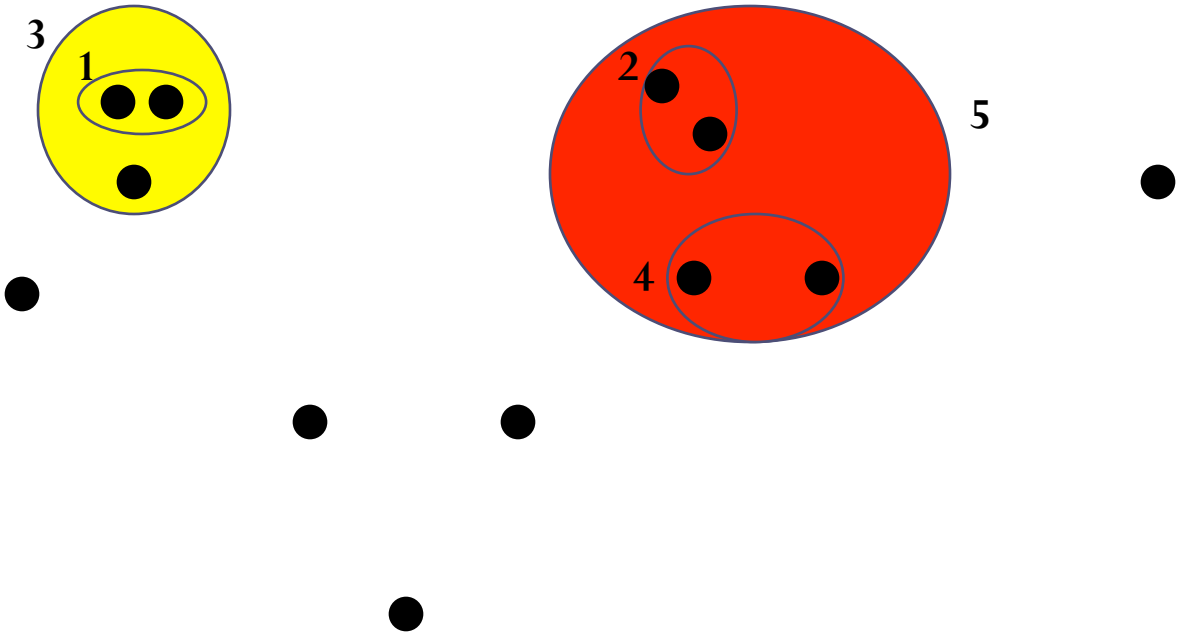
Hierarchical clustering

- Agglomerative (Bottom up)
- 4th iteration



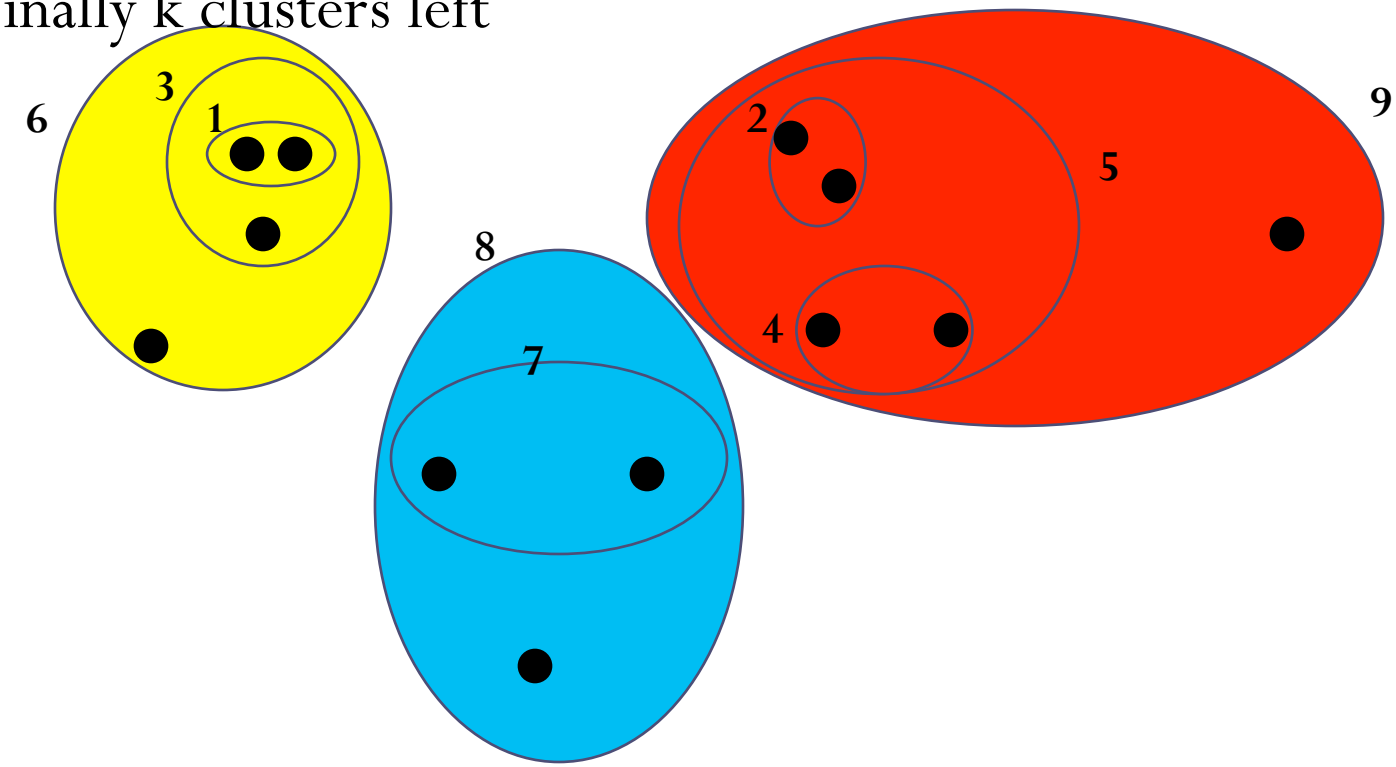
Hierarchical clustering

- Agglomerative (Bottom up)
- 5th iteration



Hierarchical clustering

- Agglomerative (Bottom up)
- Finally k clusters left

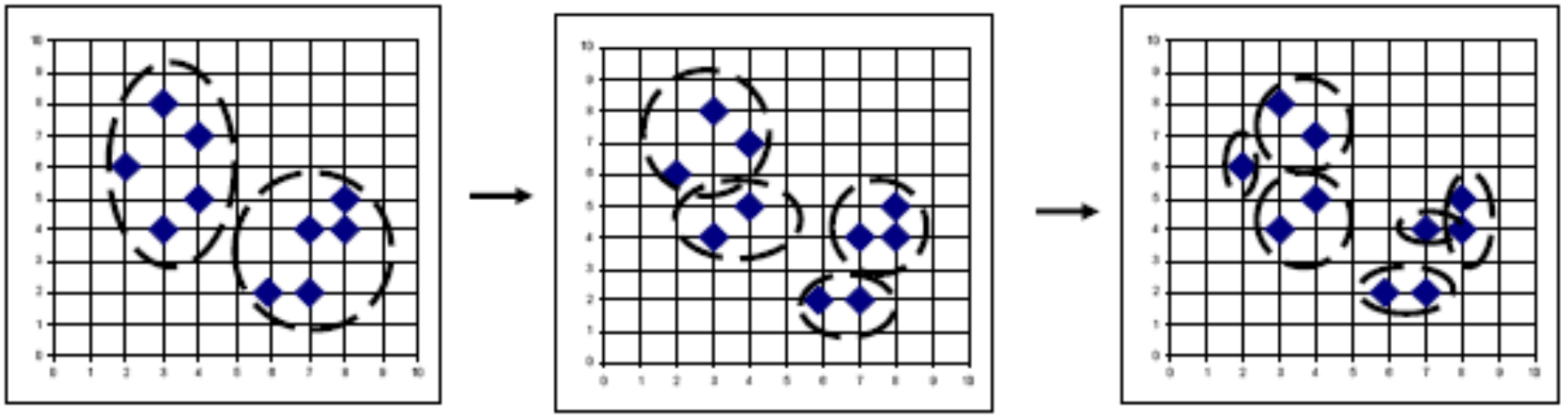


Hierarchical clustering

- Divisive (Top-down)
  - Start at the top with all patterns in one cluster
  - The cluster is split using a flat clustering algorithm
  - This procedure is applied recursively until each pattern is in its own singleton cluster

Hierarchical clustering

- Divisive (Top-down)



Slide credit: Min Zhang

Bottom-up vs. Top-down

- Which one is more complex?
- Which one is more efficient?
- Which one is more accurate?

Bottom-up vs. Top-down

- Which one is more complex?
  - Top-down
  - Because a flat clustering is needed as a "subroutine"
- Which one is more efficient?
- Which one is more accurate?

Bottom-up vs. Top-down

- Which one is more complex?
- Which one is more efficient?
- Which one is more accurate?

Bottom-up vs. Top-down

- Which one is more complex?
- 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 the number of patterns and clusters
  - Agglomerative algorithms are least quadratic
- Which one is more accurate?

Bottom-up vs. Top-down

- Which one is more complex?
- Which one is more efficient?
- Which one is more accurate?

Bottom-up vs. Top-down

- Which one is more complex?
- Which one is more efficient?
- Which one is more accurate?
  - Top-down
  - Bottom-up methods make clustering decisions based on local patterns without initially taking into account the global distribution. These early decisions cannot be undone.
  - Top-down clustering benefits from complete information about the global distribution when making top-level partitioning decisions.

Back

K-means

Data set: $X = \{\bar{x}_1, \bar{x}_2, \mathrm{L}\ , \bar{x}_n\}$
Clusters: $C_1, C_2, \mathrm{L}\ C_k$
Codebook : $V = \{\bar{v}_1, \bar{v}_2, \mathrm{L}\ , \bar{v}_k\}$
Partition matrix: $\Gamma = \{\gamma_{ij}\}$

$$\gamma_{ij} = \begin{cases} 1 & \text{if } \bar{x}_j \in C_i \\ 0 & \text{otherwise} \end{cases}$$

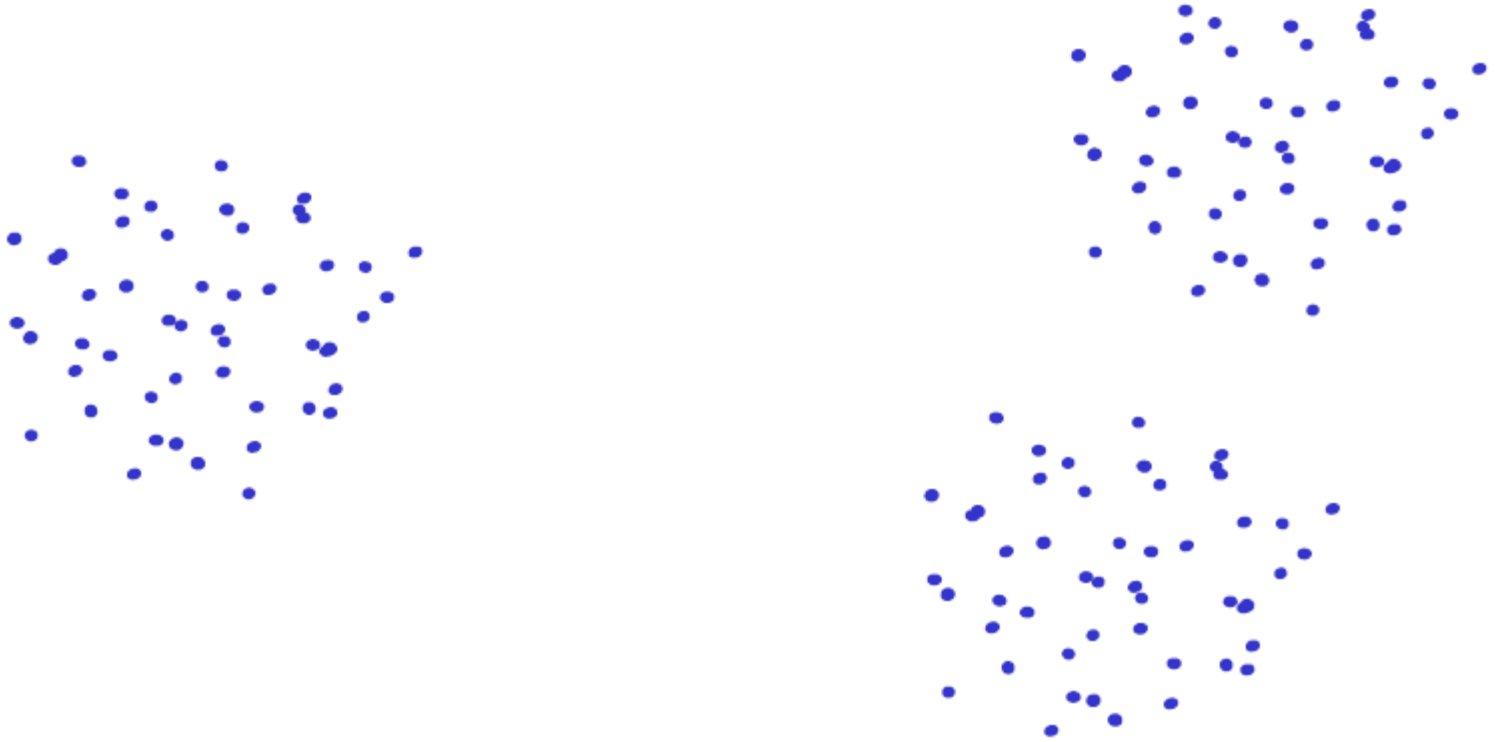
- Minimizes functional:

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

- Iterative algorithm:
  - Initialize the codebook V with vectors randomly picked from X
  - Assign each pattern to the nearest cluster
  - Recalculate partition matrix
  - Repeat the above two steps until convergence

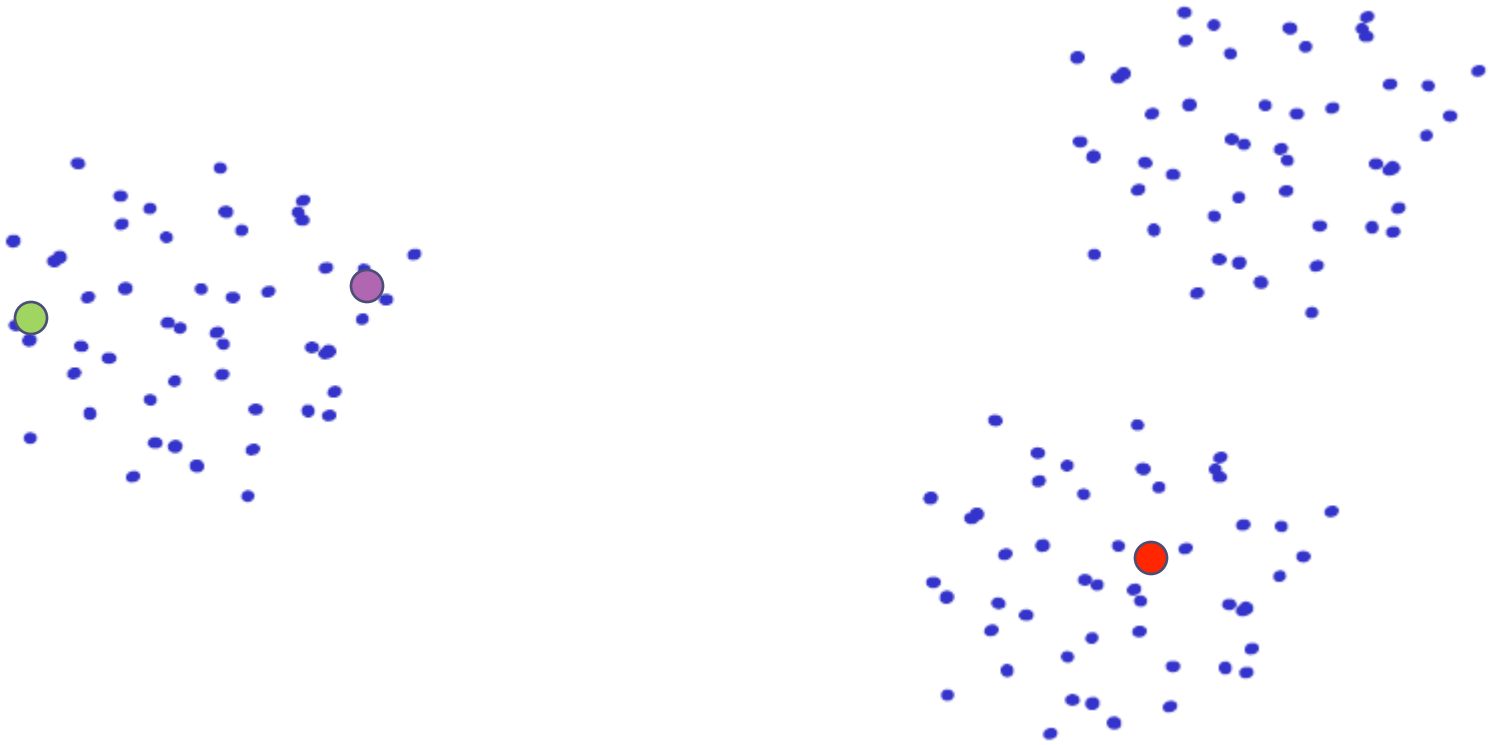
K-means

- Disadvantages
  - Dependent on initialization



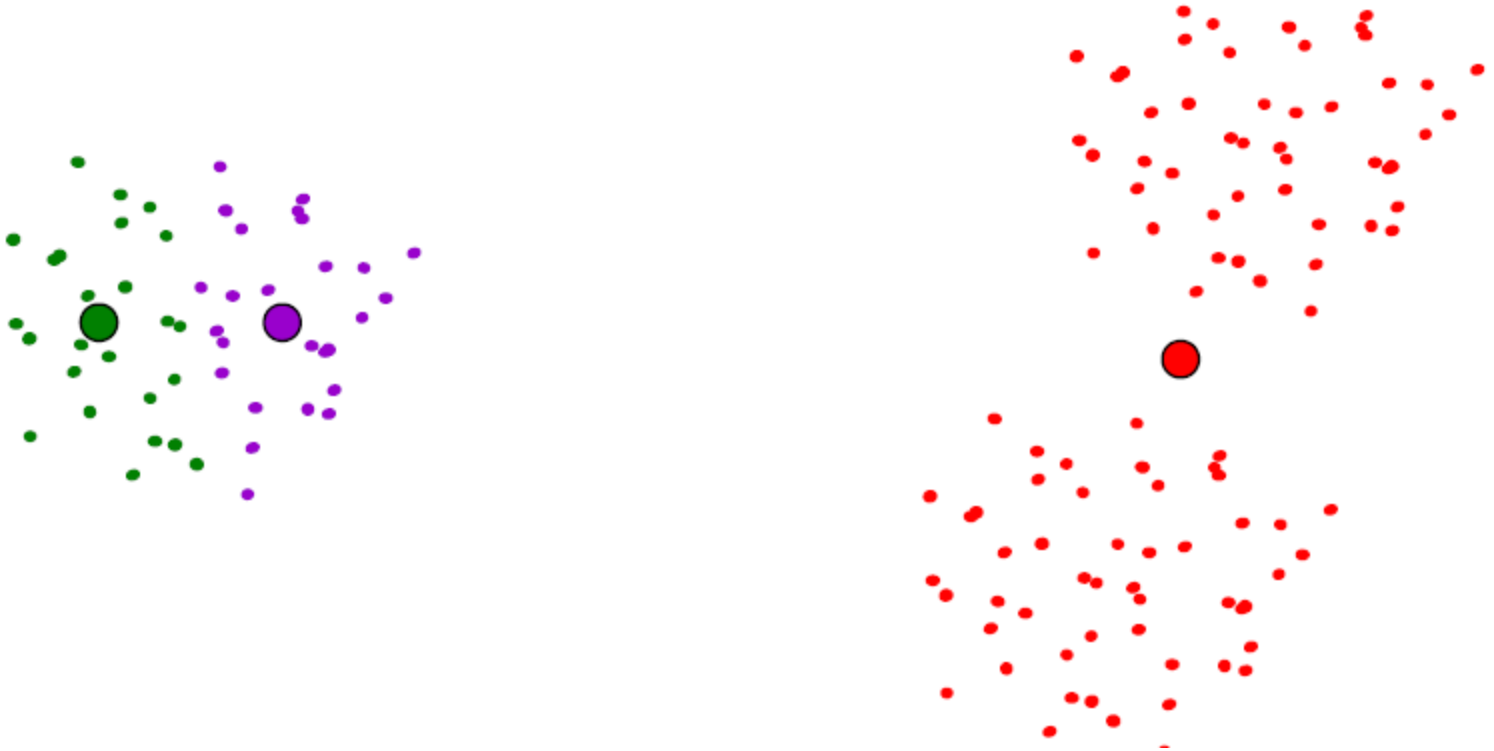
K-means

- Disadvantages
  - Dependent on initialization



K-means

- Disadvantages
  - Dependent on initialization

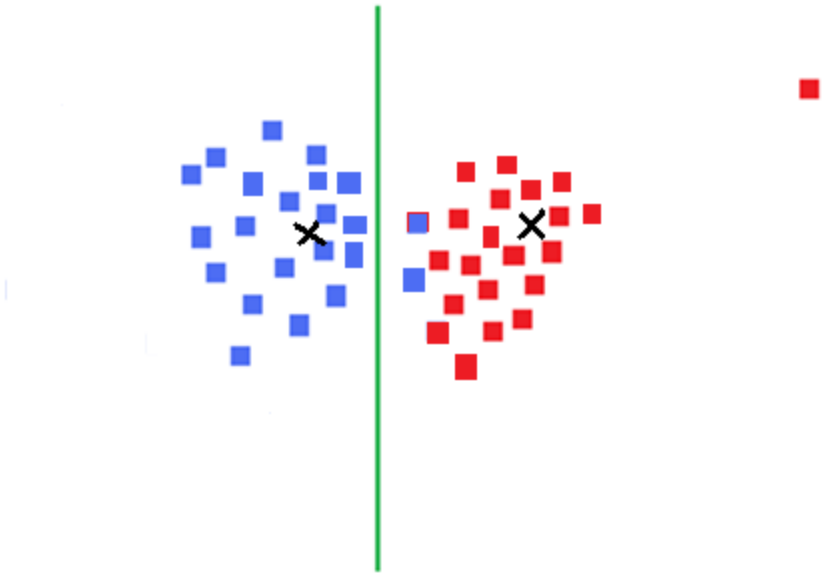


K-means

- Disadvantages
  - Dependent on initialization
    - Select random seeds with at least D_{min}
    - Or, run the algorithm many times

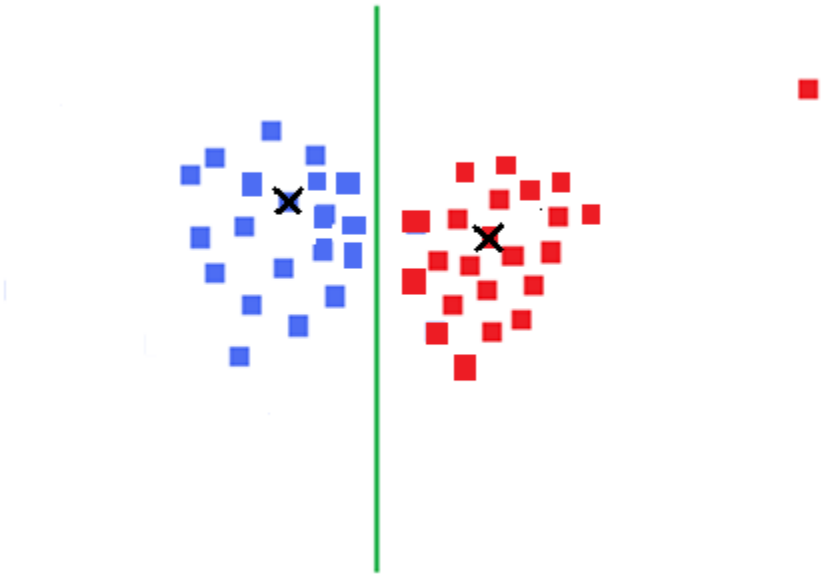
K-means

- Disadvantages
  - Dependent on initialization
  - Sensitive to outliers



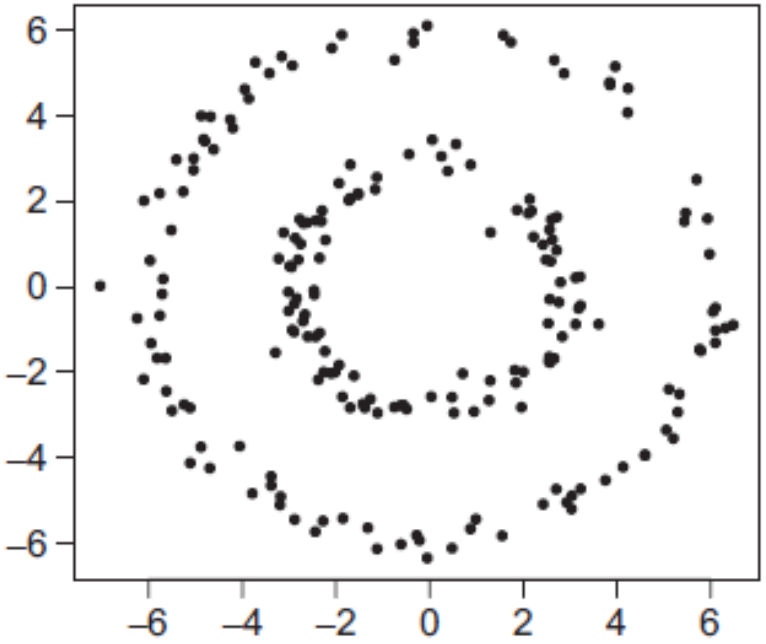
K-means

- Disadvantages
  - Dependent on initialization
  - Sensitive to outliers
    - Use K-medoids



K-means

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



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

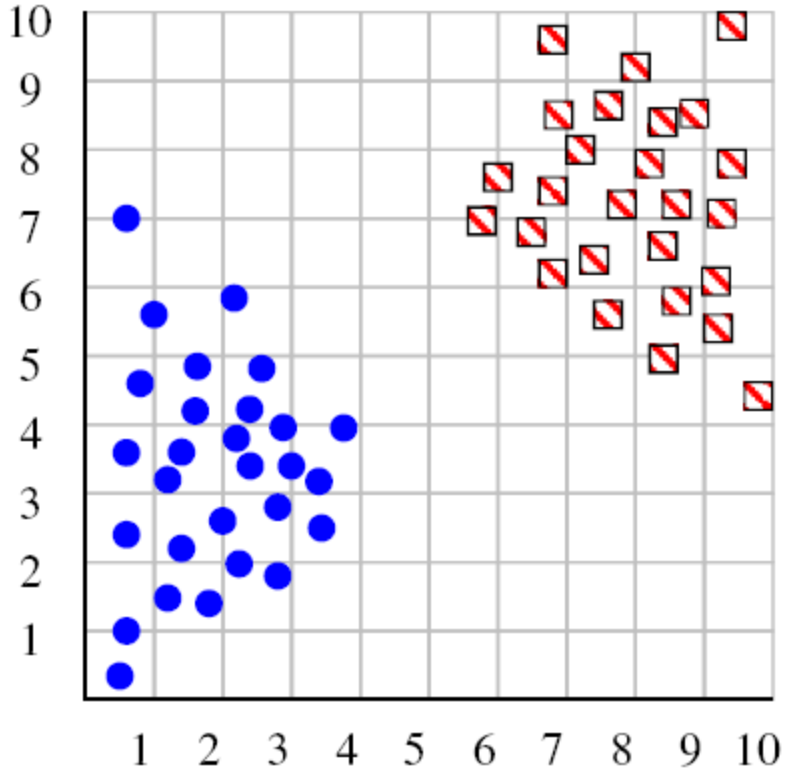


Image: Henry Lin

Deciding K

- When k = 1, the objective function is 873.0

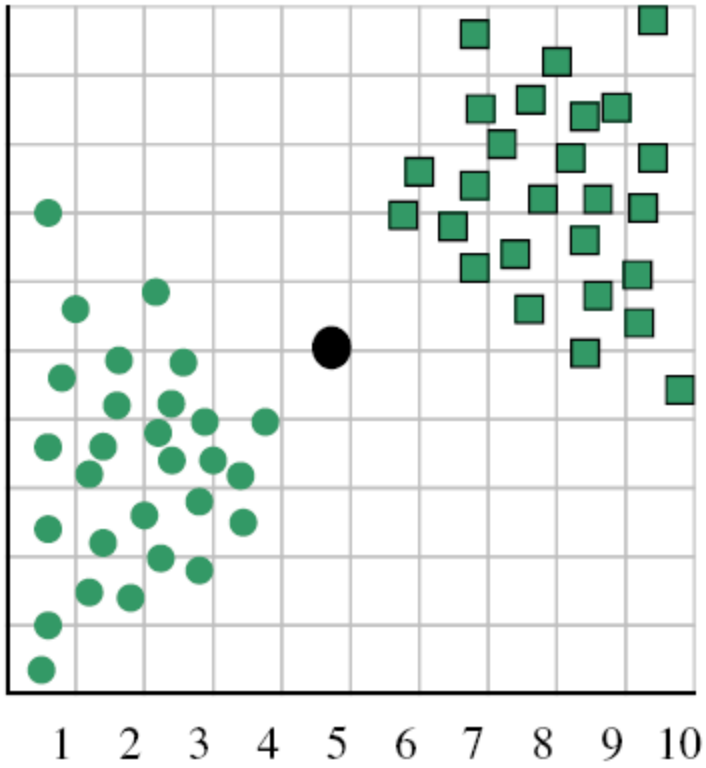


Image: Henry Lin

Deciding K

- When k = 2, the objective function is 173.1

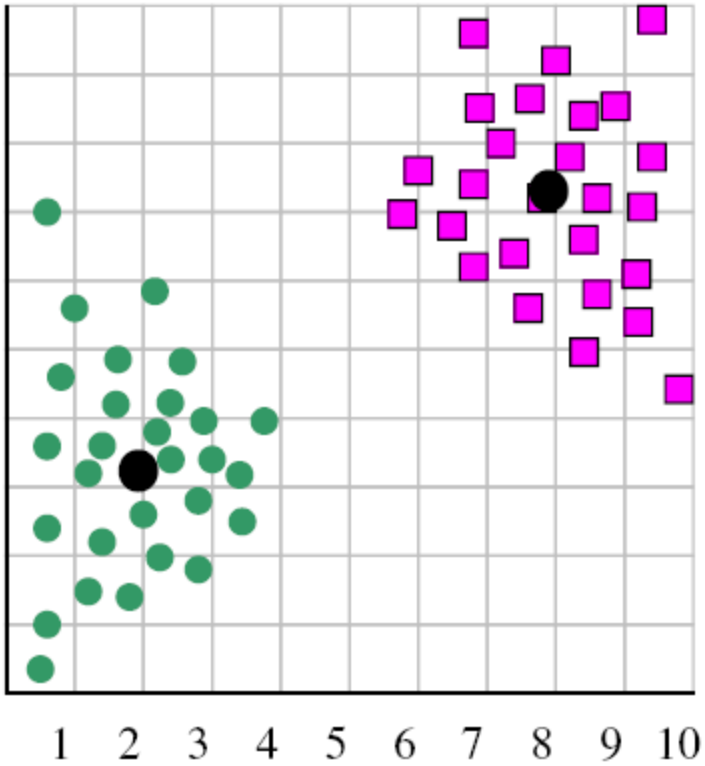


Image: Henry Lin

Deciding K

- When k = 3, the objective function is 133.6

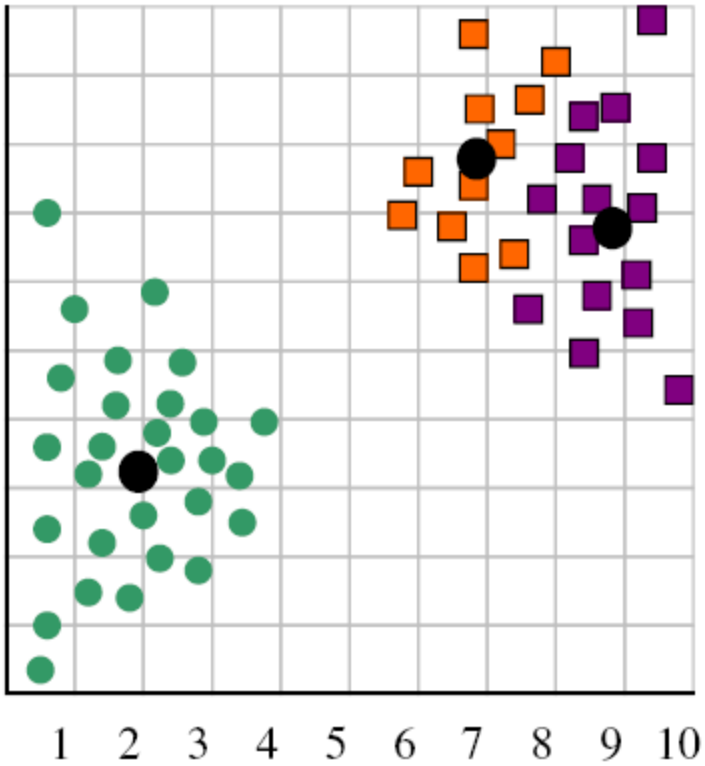


Image: Henry Lin

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

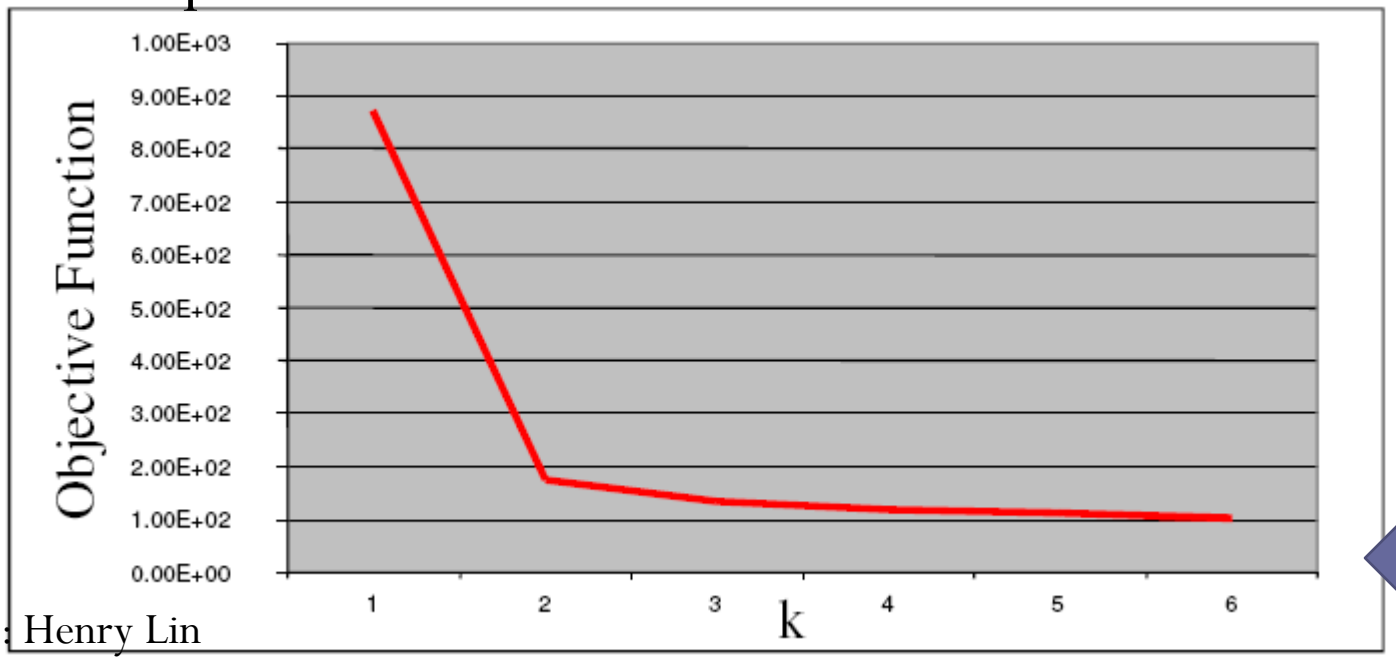


Image: Henry Lin

Back

DBSCAN – Density-Based Spatial Clustering of Applications with Noise

DBSCAN

Density-based Clustering locates regions of high density that are separated from one another by regions of low density.

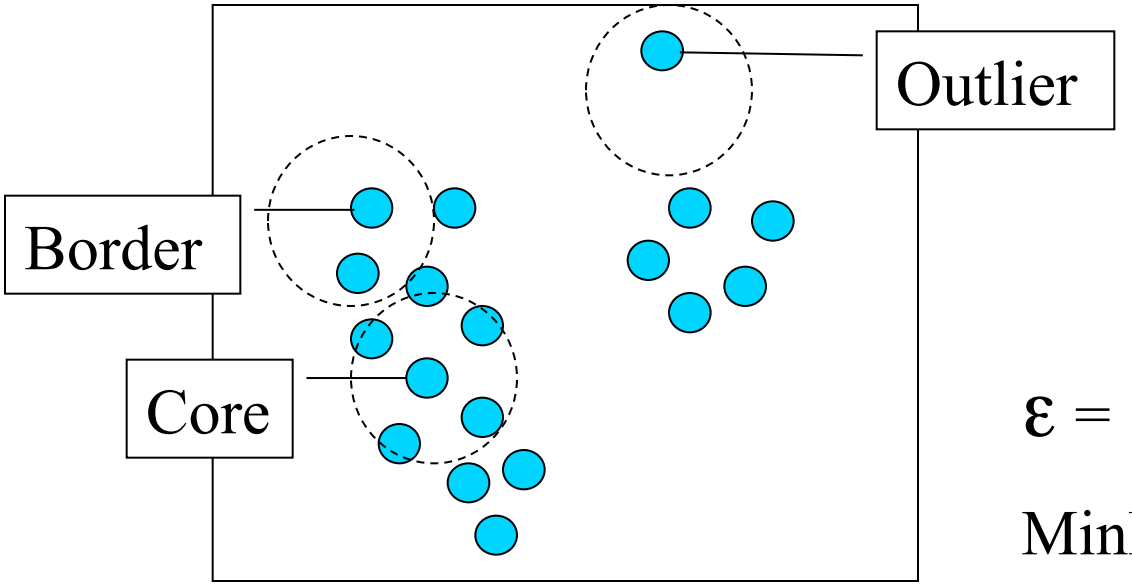
- Density = number of points within a specified radius (Eps)

- DBSCAN is a density-based algorithm.
  - 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

DBSCAN

- A noise point is any point that is not a core point or a border point.
- Any two core points 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

Border & Core

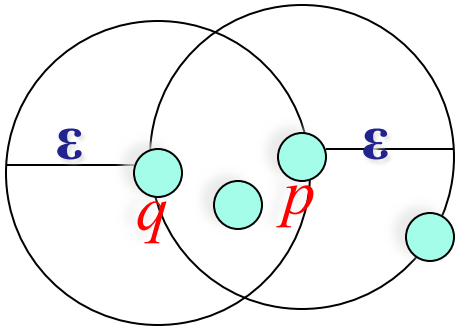


ε = 1unit

MinPts = 5

Concepts: ε-Neighborhood

- **ε -Neighborhood** - Objects within a radius of ε from an object. (epsilon-neighborhood)
- **Core objects** - ε -Neighborhood of an object contains at least **MinPts** of objects



ε-Neighborhood of *p*

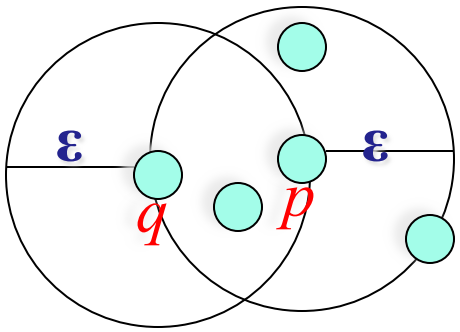
ε-Neighborhood of *q*

p is a core object (MinPts = 4)

q is not a core object

Concepts: Reachability

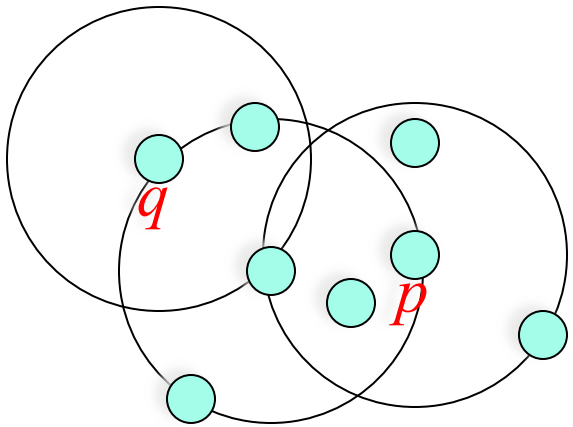
- **Directly density-reachable**
  - An object q is directly density-reachable from object p if q is within the ε -Neighborhood of p and p is a core object.



- q is directly density-reachable from p
- p is not directly density-reachable from q?

Concepts: Reachability

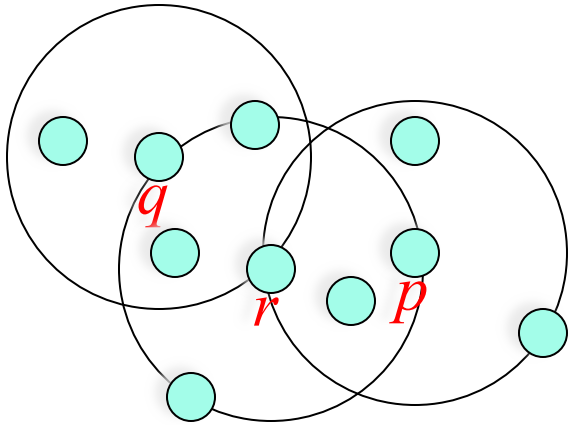
- **Density-reachable:**
  - An object p is density-reachable from q w.r.t ε and *MinPts* if there is a chain of objects $p_1,\ldots,p_n$, with $p_1=q$, $p_n=p$ such that $p_{i+1}$is directly density-reachable from p_i w.r.t ε and *MinPts* for all $1 <= i <= n$



- q is density-reachable from p
- p is not density- reachable from q?
- Transitive closure of direct density-Reachability, asymmetric

Concepts: Connectivity

- **Density-connectivity**
  - Object *p* is density-connected to object *q* w.r.t ε and *MinPts* if there is an object *o* such that both *p* and *q* are density-reachable from *o* w.r.t ε and *MinPts*

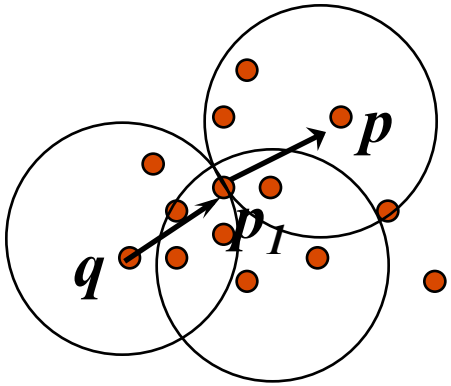


- *P* and *q* are density-connected to each other by *r*
- Density-connectivity is symmetric

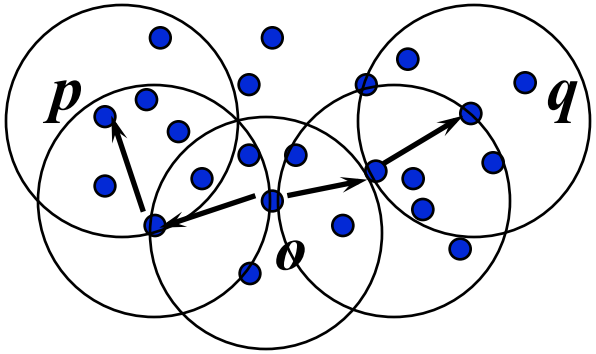
Concepts: cluster & noise

- **Cluster**: a cluster ***C*** in a set of objects ***D*** w.r.t ε and *MinPts* is a non empty subset of ***D*** satisfying
  - Maximality: For all *p*, *q* if $p \in \boldsymbol{C}$ and if *q* is density-reachable from *p* w.r.t ε and *MinPts*, then also $q \in \boldsymbol{C}$.
  - Connectivity: for all $p, q \in \boldsymbol{C}$, *p* is density-connected to *q* w.r.t ε and *MinPts* in ***D***.
  - ***Note:*** cluster contains *core objects* as well as *border objects*
- **Noise:** objects which are not directly density-reachable from at least one core object.

(Indirectly) Density-reachable:



Density-connected

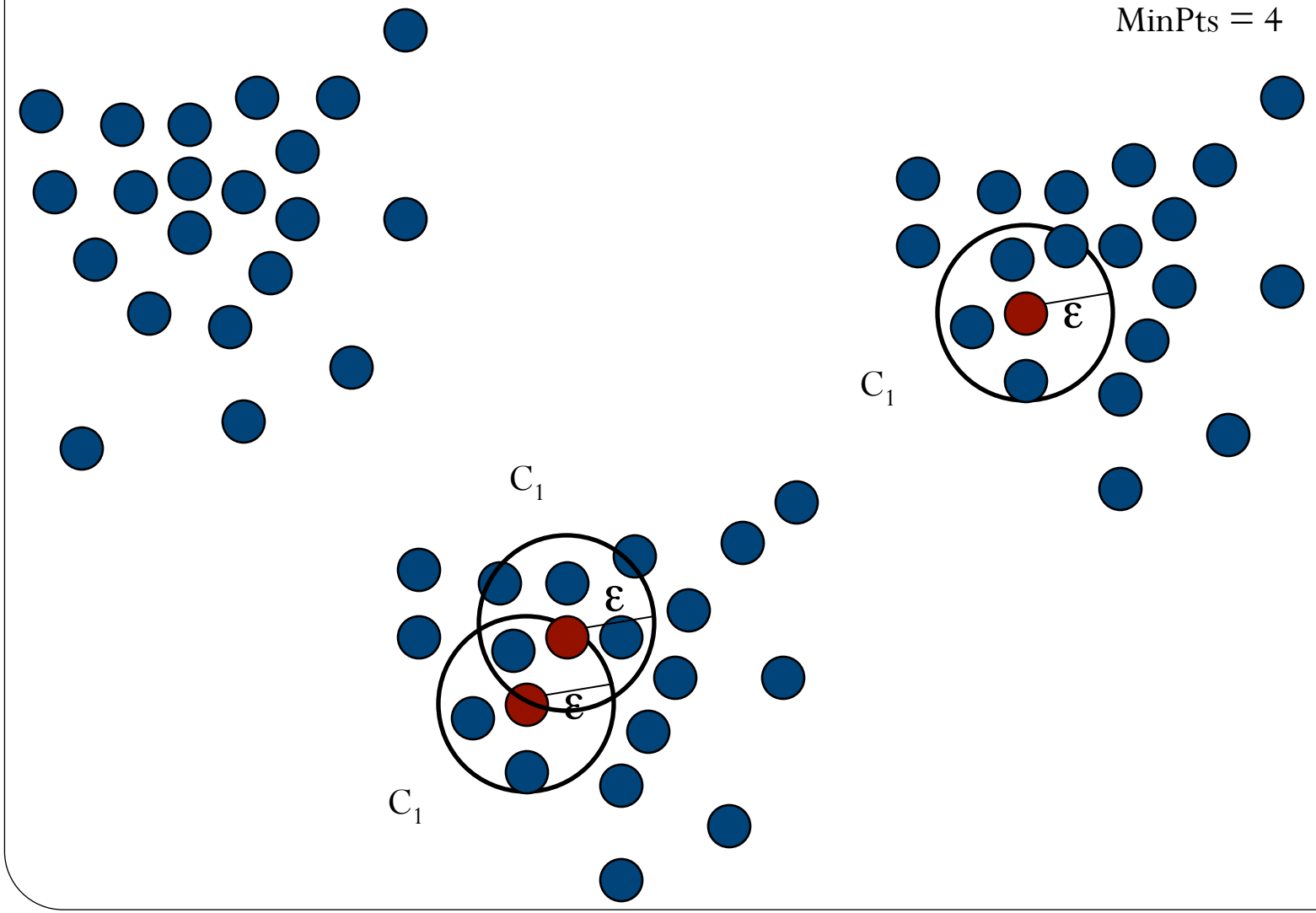


DBSCAN: The Algorithm

- select a point ***p***
- Retrieve all points density-reachable from ***p*** wrt ε and ***MinPts***.
- If ***p*** is a core point, a cluster is formed.
- If ***p*** is a border point, no points are density-reachable from ***p*** and DBSCAN visits the next point of the database.
- Continue the process until all of the points have been processed.

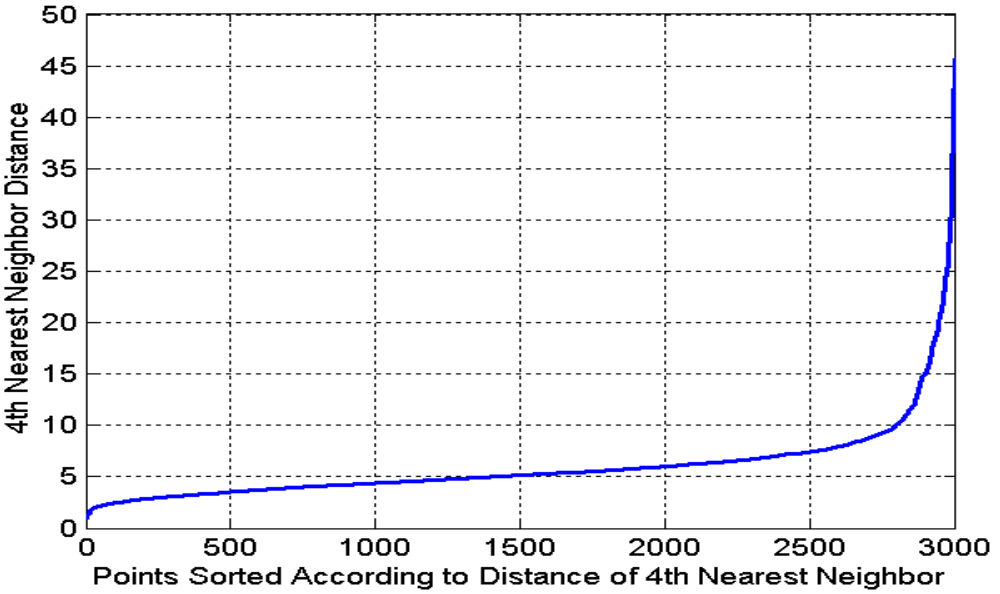
Result is independent of the order of processing the points

An Example



DBSCAN: Determining EPS and MinPts

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



DBSCAN: Determining EPS and MinPts

- Distance from a point to its k^{th} nearest neighbor=>k-dist
- For points that belong to some clusters, the value of k-dist will be small if k is not larger than cluster size
- For points that are not in a cluster such as noise points, the k-dist will be relatively large
- Compute k-dist for all points for some k
- Sort them in increasing order and plot sorted values
- A sharp change at the value of k-dist that corresponds to suitable value of eps and the value of k as MinPts

DBSCAN: Determining EPS and MinPts

- A sharp change at the value of k-dist that corresponds to suitable value of eps and the value of k as MinPts
  - 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.
- If k is too large=> small clusters (of size less than k) are likely to be labeled as noise
- If k is too small=> Even a small number of closely spaced that are noise or outliers will be incorrectly labeled as clusters

What Is A Good Clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:
  - the intra-class (that is, intra-cluster) similarity is high
  - the inter-class similarity is low
  - The measured quality of a clustering depends on both the document representation and the similarity measure used

External criteria for clustering quality

- Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data
- Assesses a clustering with respect to ground truth … requires *labeled data*
- Assume documents with C gold standard classes, while our clustering algorithms produce K clusters, $\omega_1, \omega_2, \ldots, \omega_K$ with n_i members.

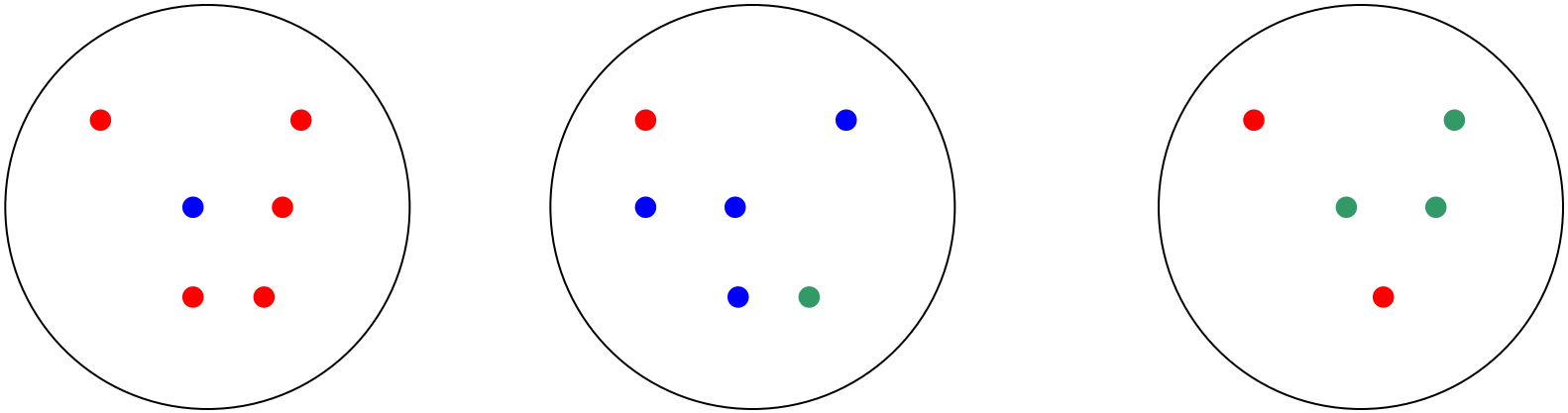
External Evaluation of Cluster Quality

- Simple measure: purity, the ratio between the dominant class in the cluster π_i and the size of cluster ω_i

$$Purity(\omega_i) = \frac{1}{n_i} \max_j (n_{ij}) \quad j \in C$$

- Biased because having *n* clusters maximizes purity
- Others are entropy of classes in clusters (or mutual information between classes and clusters)

Purity example



Cluster I Cluster II Cluster III

Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6

Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6

Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

Rand Index measures between pair decisions. Here RI = 0.68

Number of points	Same Cluster in clustering	Different Clusters in clustering
Same class in ground truth	20	24
Different classes in ground truth	20	72

Rand index and Cluster F-measure

$$RI = \frac{A + D}{A + B + C + D}$$

Compare with standard Precision and Recall:

$$P = \frac{A}{A + B} \qquad R = \frac{A}{A + C}$$

People also define and use a cluster F-measure, which is probably a better measure.

Validation Indices

Find K for K-Means Algorithm