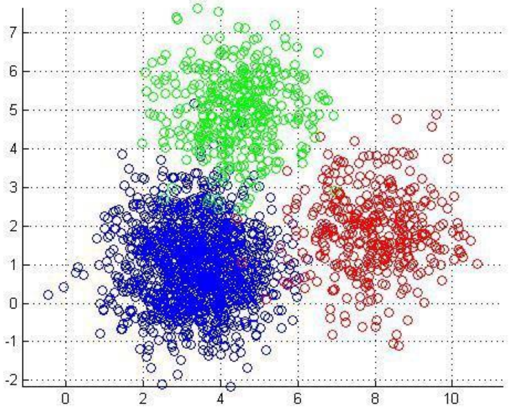
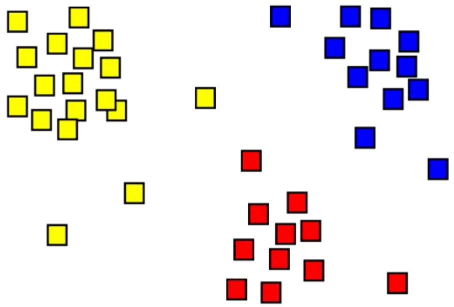
CS4104 Applied Machine Learning

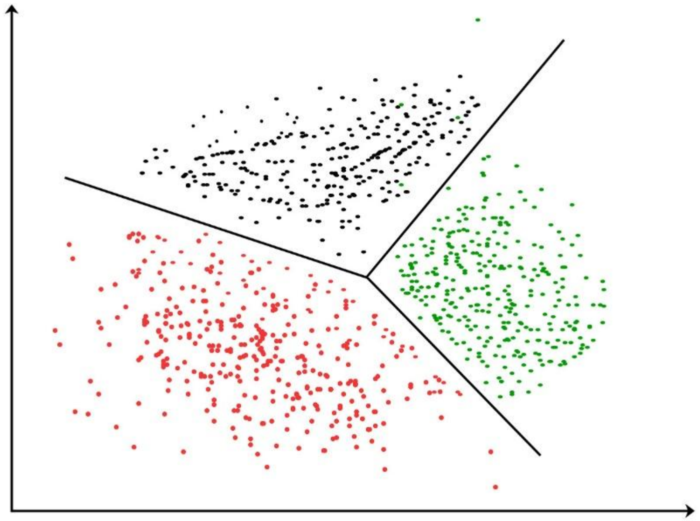
Clustering: Un Supervised Learning

Clustering

- The organization of unlabeled data into similarity groups called clusters.
- A cluster is a collection of data items which are “similar” between them, and “dissimilar” to data items in other clusters.







zeshan.khan@nu.edu.pk

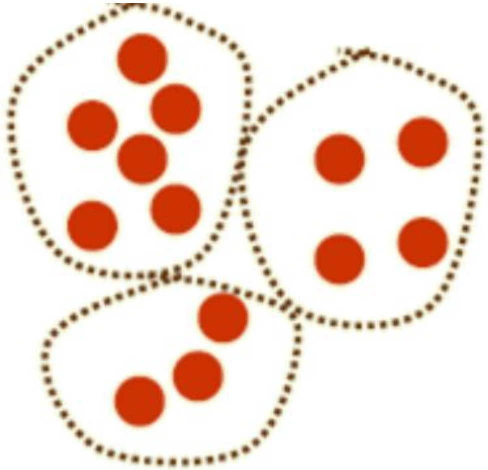
History

- John Snow (London Physician) plotted the location of cholera deaths on a map during an outbreak in 1850s.
- The location indicated that the cases were clustered around certain intersections where there were polluted wells that leaded to the solution.

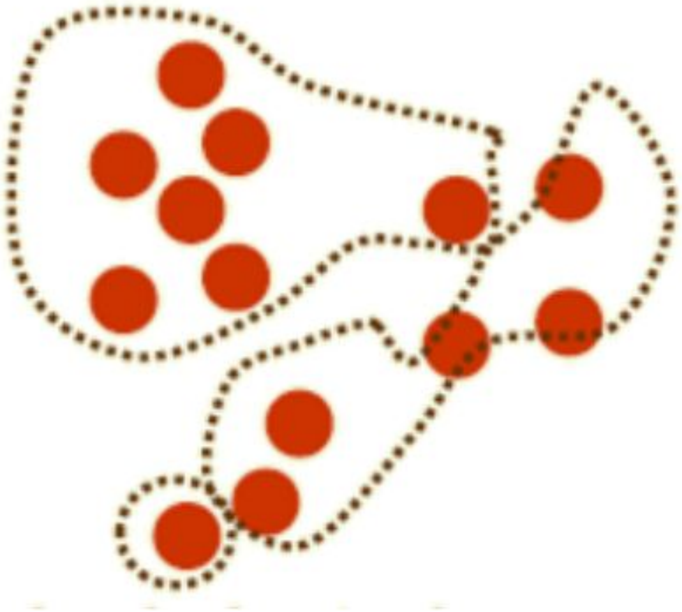
zeshan.khan@nu.edu.pk

Clustering measure

- Proximity measure
- $similarity(p_1, p_2)$ is large if p_1 and p_2 are similar
- $distance(p_1, p_2)$ is larger if the points are different



Good cluster



Bad cluster

zeshan.khan@nu.edu.pk

Distance Measure

-
- Euclidean Distance $d(p,q) = \sqrt{\sum_{i=1}^{d}(p_i - q_i)^2}$
  - $d(x,y) = \sqrt{(x_i - y_i)^2 + (x_j - y_j)^2}$ if 2D
- Manhattan (City Block) Distance $d(p,q) = \sum_{i=1}^{d} |(p_i - q_i)|$
- Minkowski Distance $dist(p,q) = \left(\sum_{i=1}^{d}|(p_i - q_i)|^d\right)^{\frac{1}{d}}$

zeshan.khan@nu.edu.pk

Cluster evaluation

- **Intra-cluster cohesion** (compactness)
  - Cohesion measures how near the data points in a cluster are to the cluster centroid.
  - Sum of squared error (SSE) is a commonly used measure.
- **Inter-cluster separation** (isolation)
  - Separation means that different cluster centroids should be far away from one another.
- In most applications, expert judgments are still the key

Number of clusters?

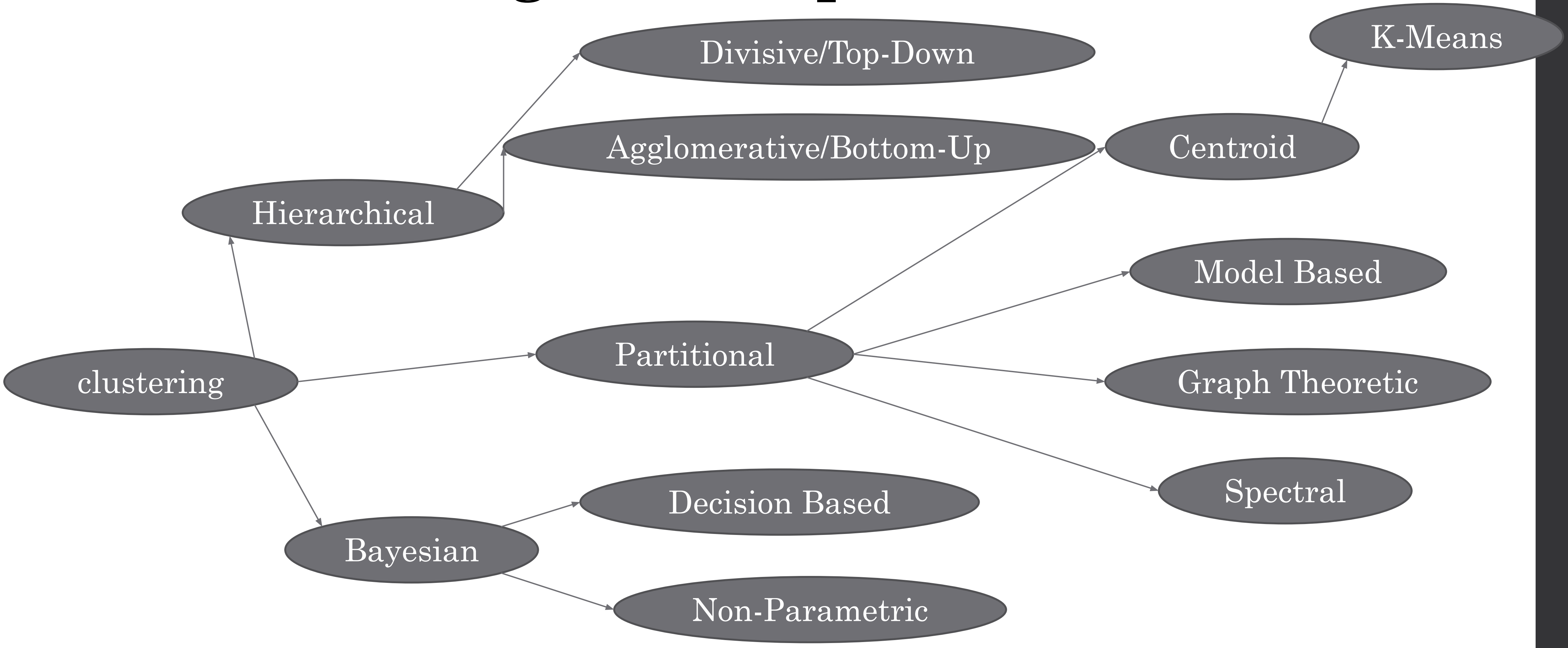
zeshan.khan@nu.edu.pk

Number of Clusters?

- Fix number of clusters
- Vitiate number of clusters
- Best number of clusters

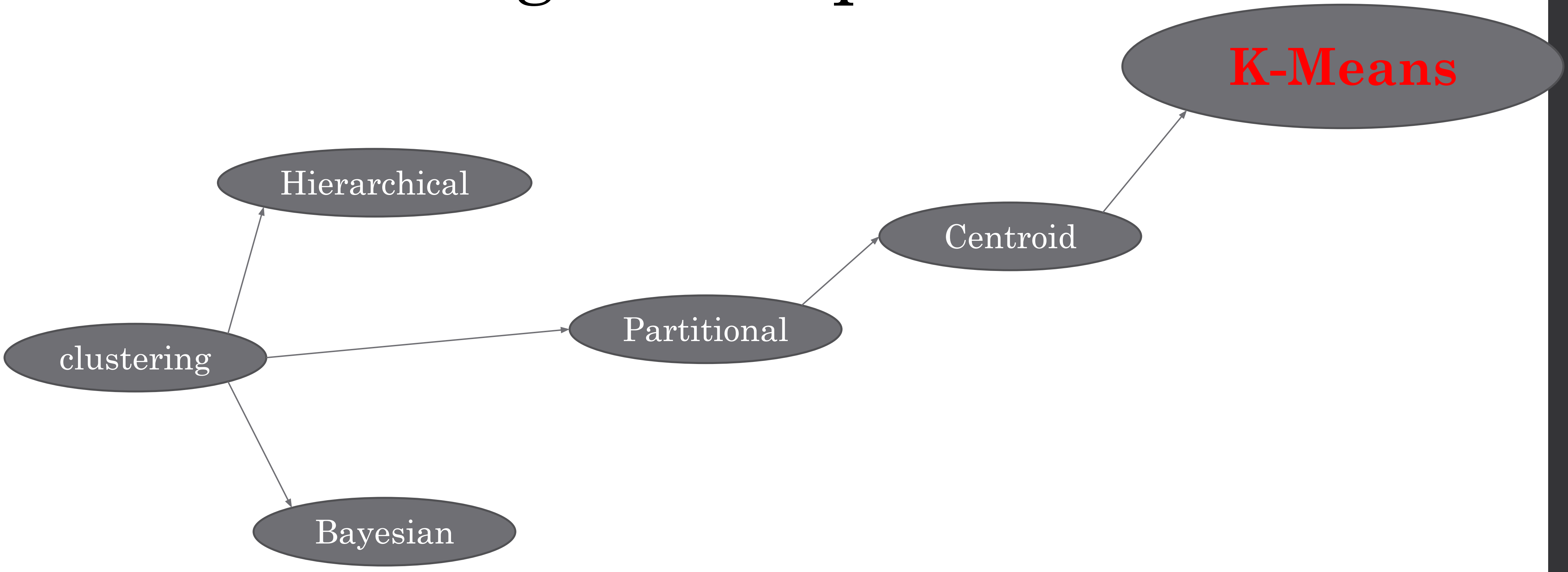
zeshan.khan@nu.edu.pk

Clustering techniques



zeshan.khan@nu.edu.pk

Clustering techniques



zeshan.khan@nu.edu.pk

K-Means clustering

- K-means (MacQueen, 1967) is a partitional clustering algorithm
- Let the set of data points $D = \{p_1, p_2, \dots, p_n\}$, where $x_i = (x_{i1}, x_{i2}, \dots, x_{ir})$ is a vector in r dimensional space. *r* is the number of dimensions.
- The *k*-means algorithm partitions the given data into *k* clusters:
  - Each cluster has a cluster **center**, called **centroid**.
  - $centeroid = Average(p_{1i}, p_{2i}, \dots, p_{mi})$
  - *k* is specified by the user

zeshan.khan@nu.edu.pk

K-means Clustering

- Given k, the k-means algorithm works as follows:
  1. Choose k (random) data points (seeds) to be the initial centroids, cluster centers
  2. Assign each data point to the closest centroid
  3. Re-compute the centroids using the current cluster memberships
  4. If a convergence criterion is not met, repeat steps 2 and 3

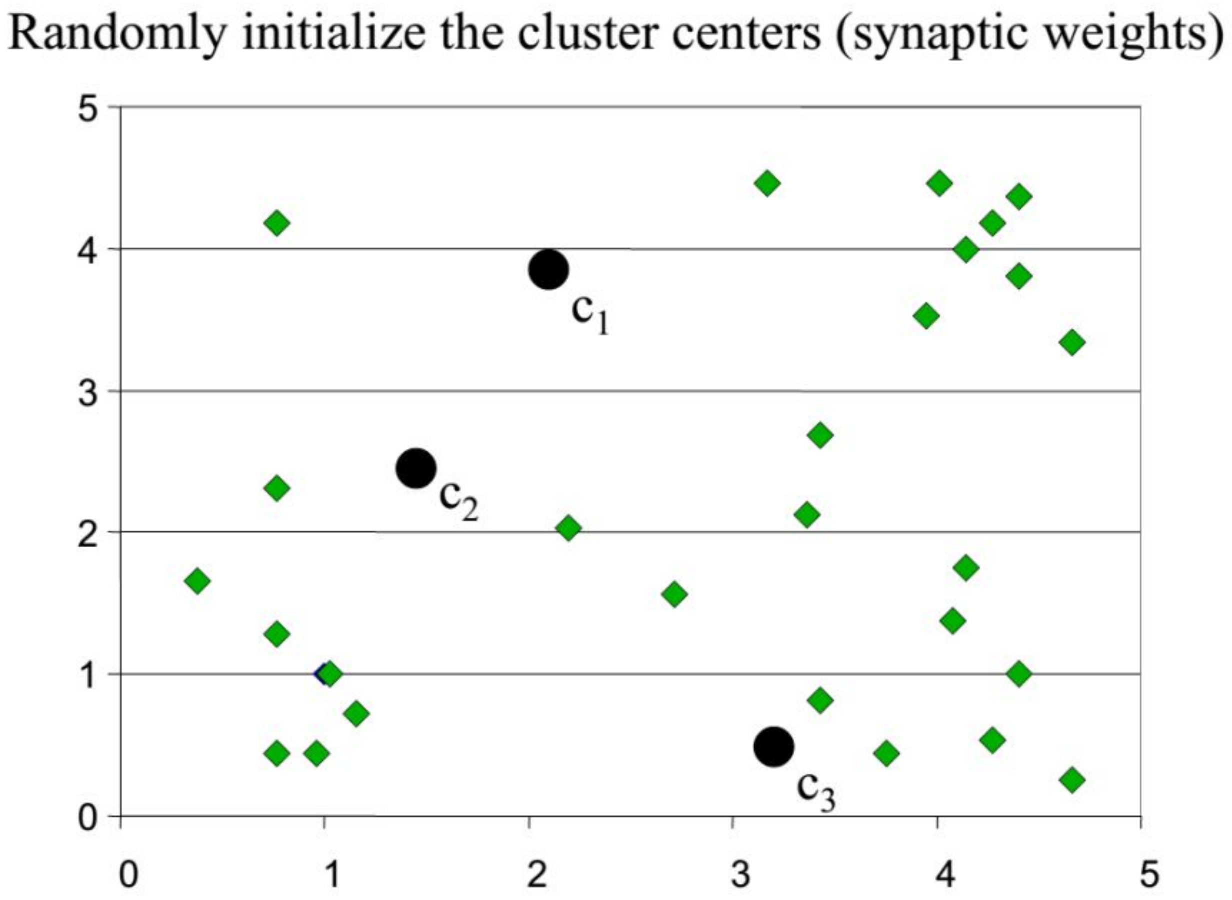
zeshan.khan@nu.edu.pk

K-means convergence (stopping) criterion

- no (or minimum) re-assignments of data points to different clusters, *or*
- no (or minimum) change of centroids, *or*
- minimum decrease in the **sum of squared error** (SSE),
- $SSE = \sum_{j=1}^{k} \left(\sum_{x \in C_j} d(X, m_j)^2 \right)$
- C_j is the jth cluster,
- m_j is the centroid of cluster C_j (the mean vector of all the data points in C_j),
- $d(x, m_j)$ is the (Euclidian) distance between data point **x** and centroid m_j

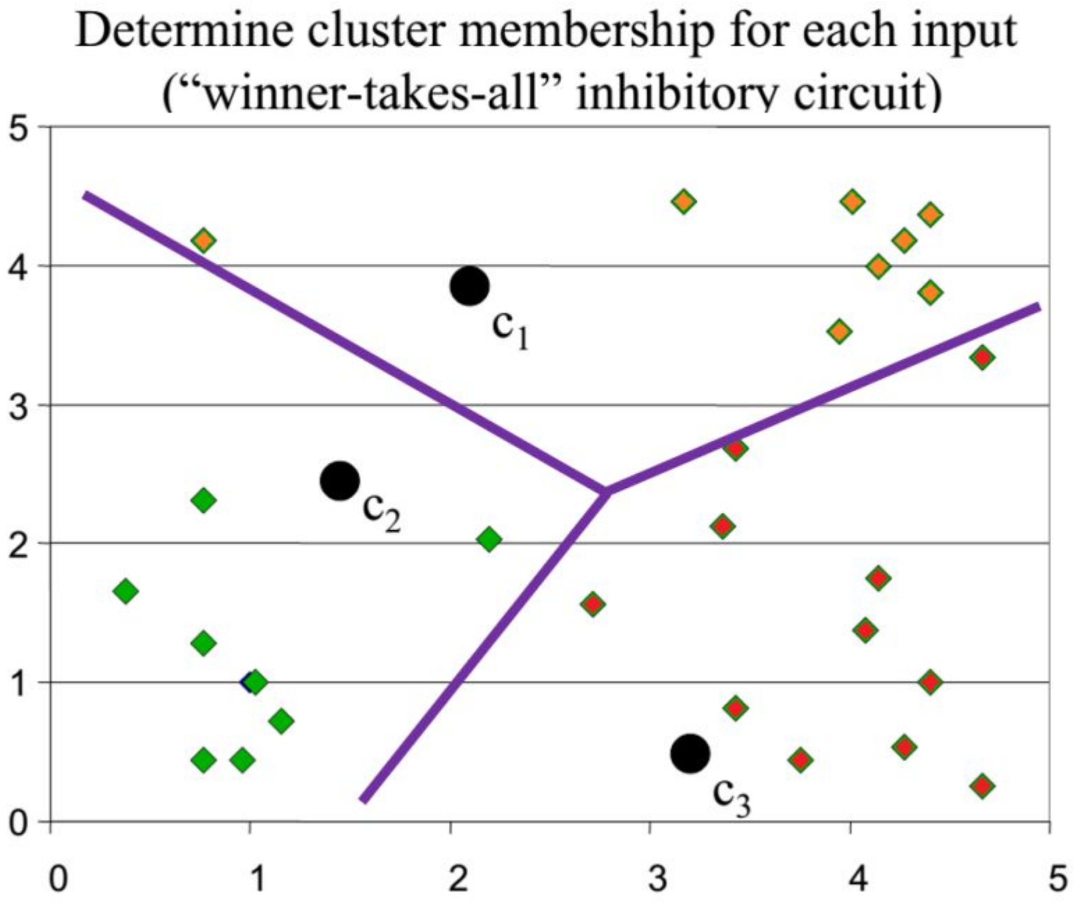
zeshan.khan@nu.edu.pk

K-means clustering example



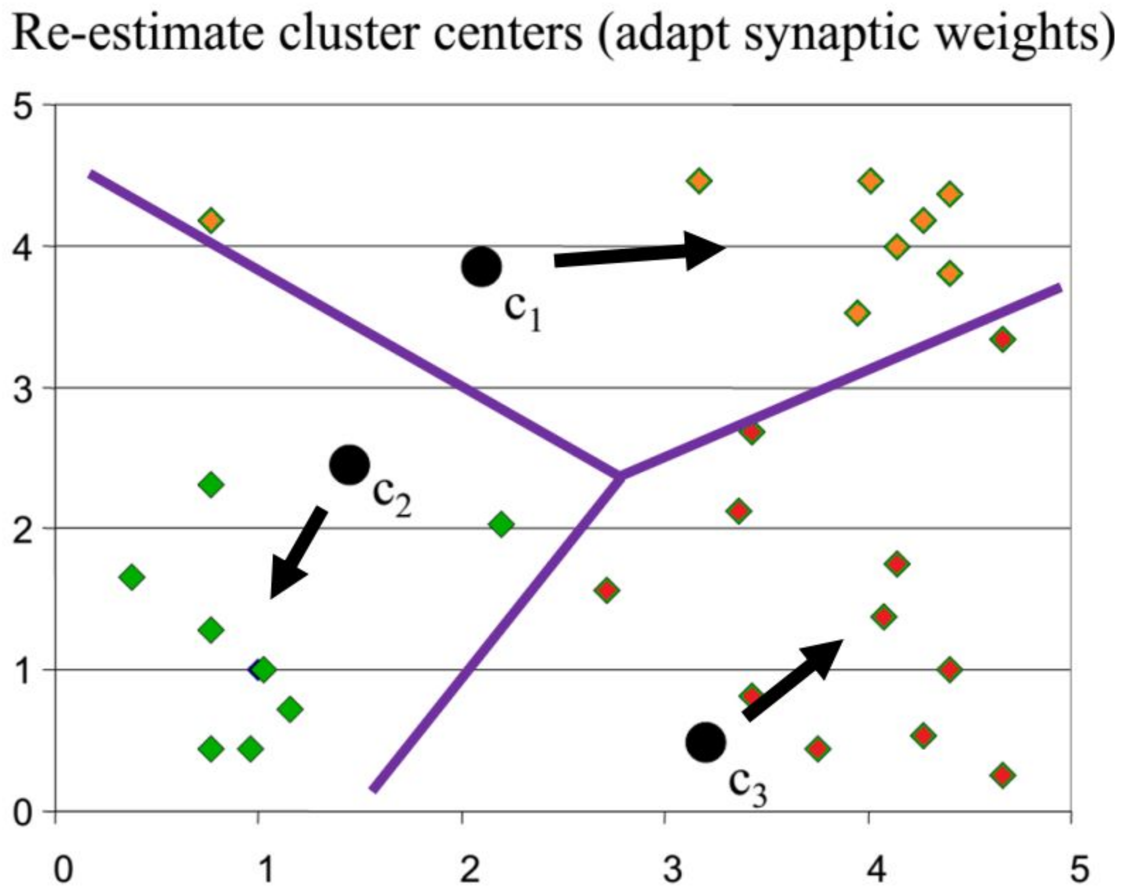
zeshan.khan@nu.edu.pk

K-means clustering example



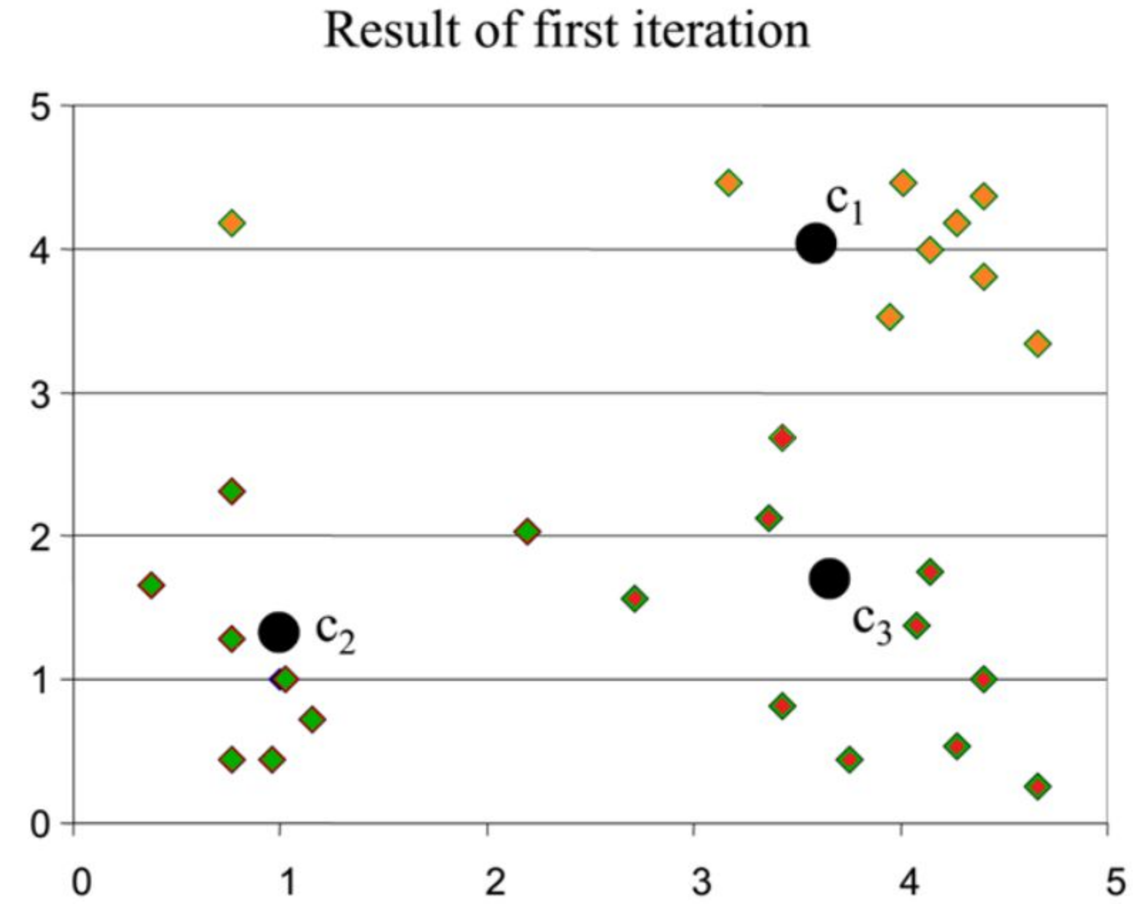
zeshan.khan@nu.edu.pk

K-means clustering example



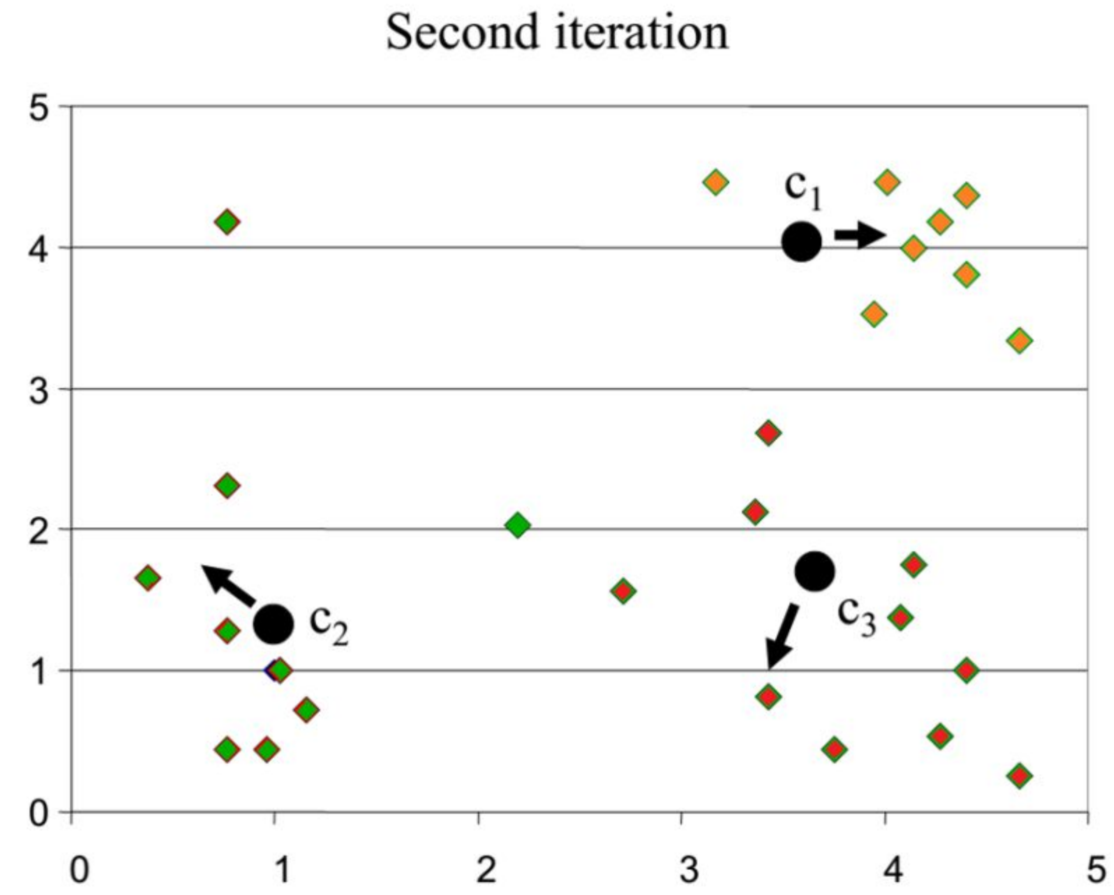
zeshan.khan@nu.edu.pk

K-means clustering example

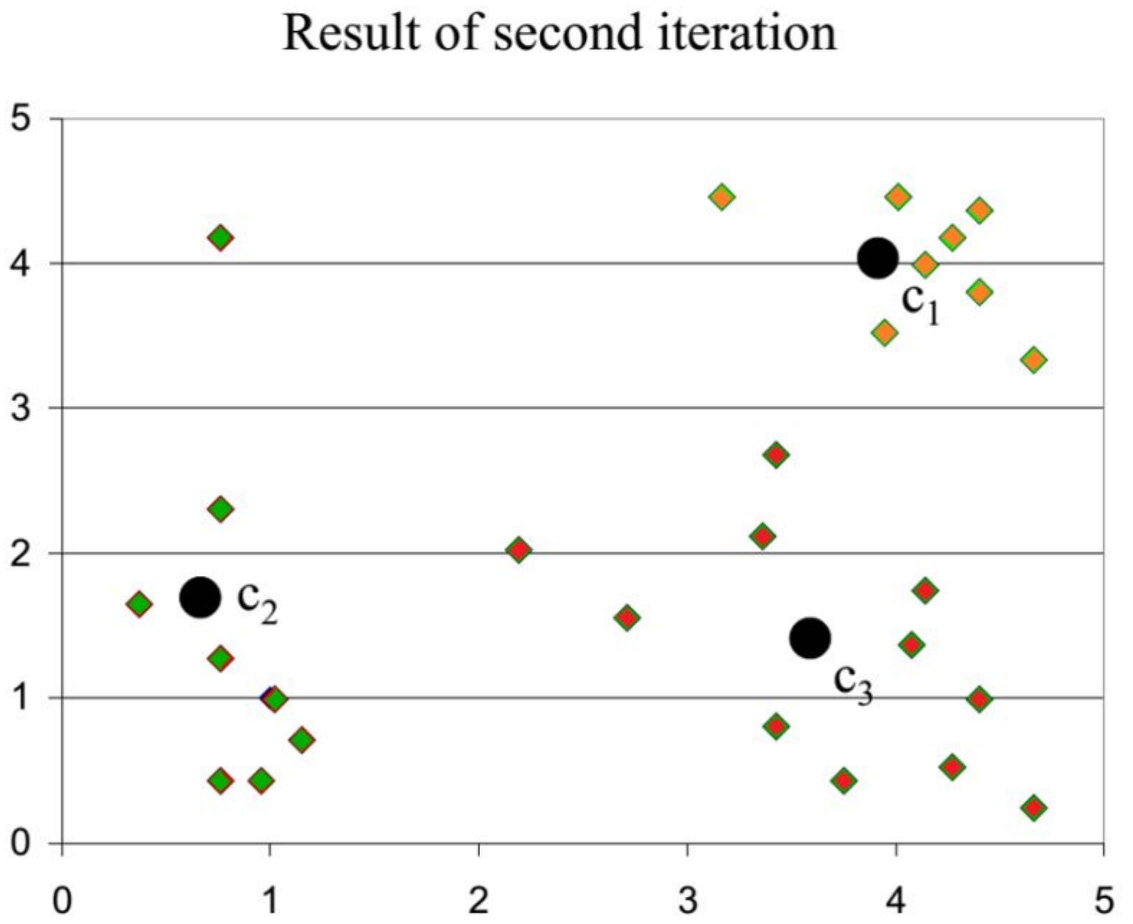


zeshan.khan@nu.edu.pk

K-means clustering example



K-means clustering example



K-Means Analysis

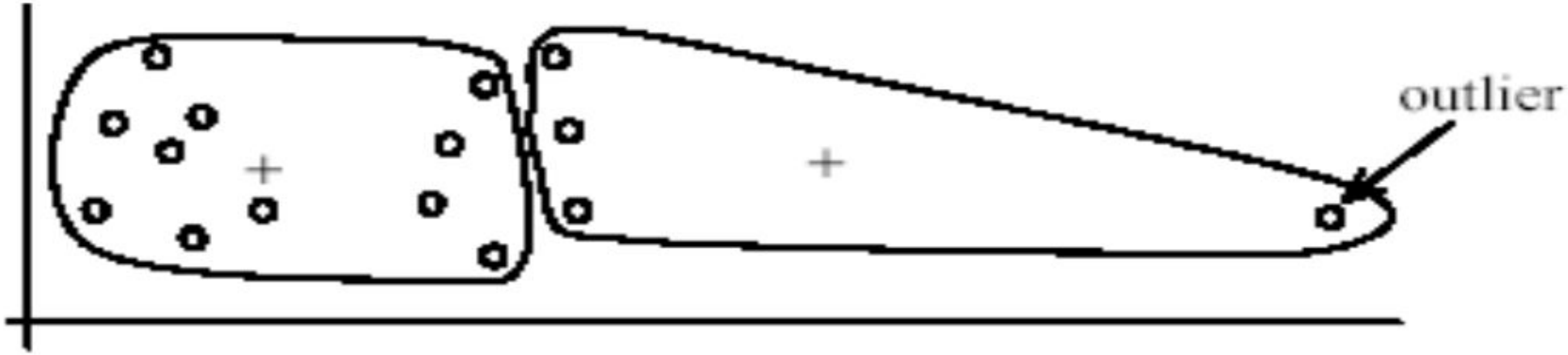
Strengths

- Simple: easy to understand and to implement
- Efficient: Time complexity: $O(tkn)$,
  - where n is the number of data points,
  - k is the number of clusters, and
  - t is the number of iterations.
- Since both k and t are small. k-means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.
- Note that: it terminates at a local optimum if SSE is used.
- The global optimum is hard to find due to complexity

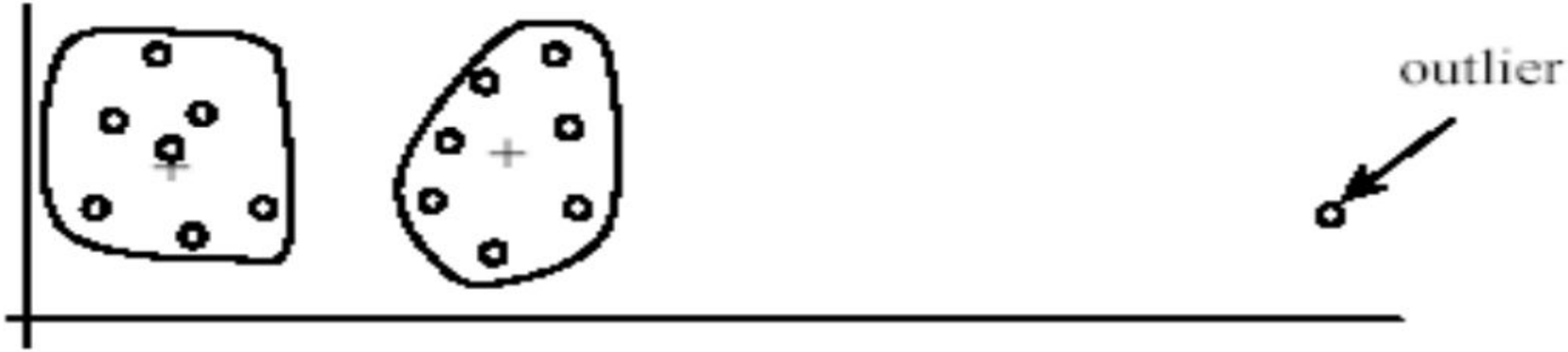
Weaknesses

- The algorithm is only applicable if the mean is defined.
  - For categorical data, k-mode - the centroid is represented by most frequent values.
- The user needs to specify k.
- The algorithm is sensitive to **outliers**
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values.
- No clear evidence that any other clustering algorithm performs better in general

zeshan.khan@nu.edu.pk



(A): Undesirable clusters



(B): Ideal clusters

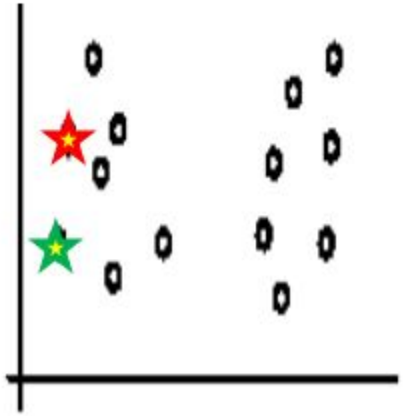
zeshan.khan@nu.edu.pk

Outliers: Handling

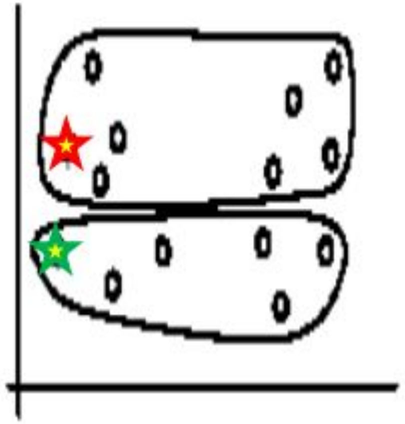
- Remove some data points that are much further away from the centroids than other data points
  - To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.
- Perform random sampling: by choosing a small subset of the data points, the chance of selecting an outlier is much smaller
- Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

zeshan.khan@nu.edu.pk

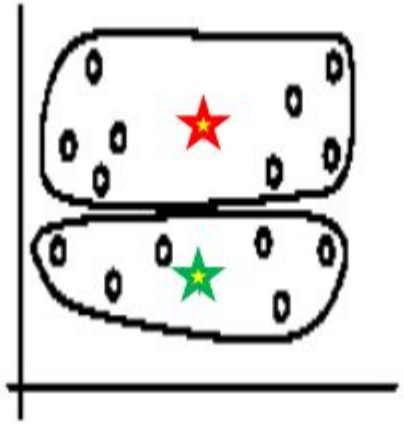
Sensitivity to initial seeds



Initial seeds



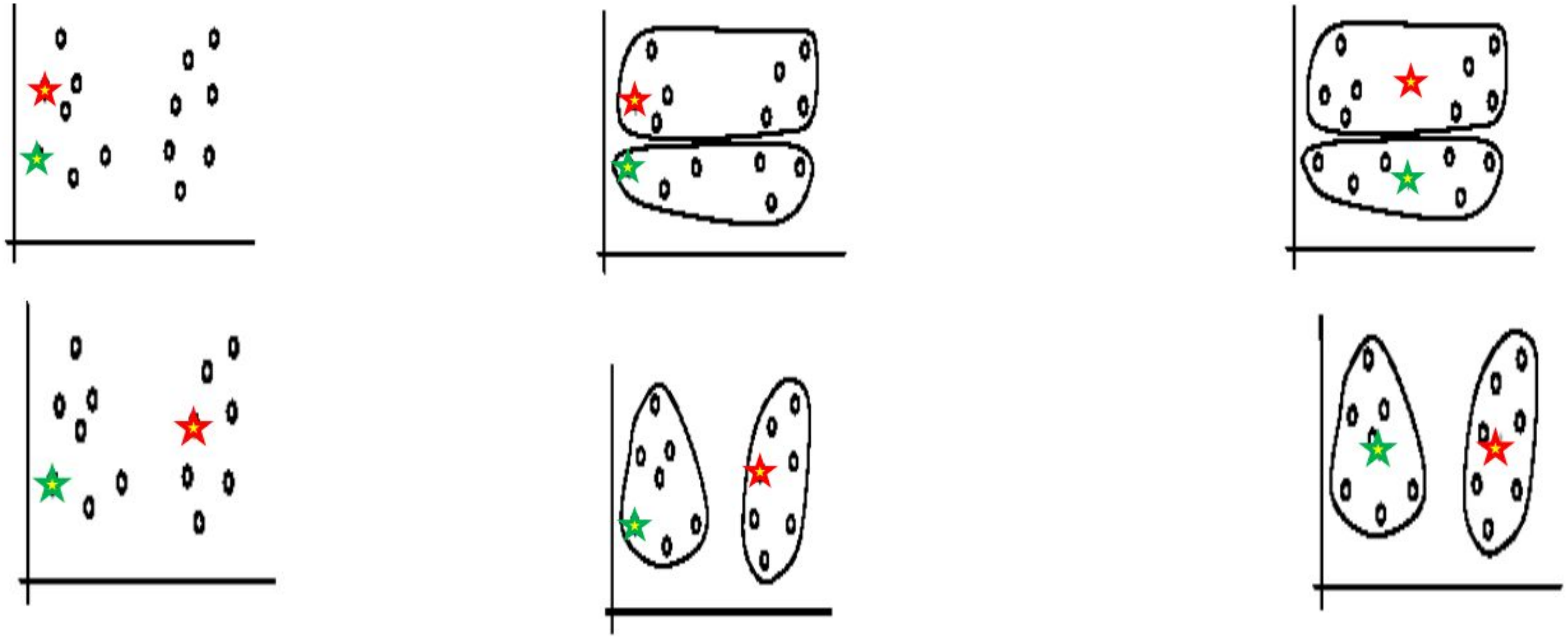
Iteration 1



Iteration 2

zeshan.khan@nu.edu.pk

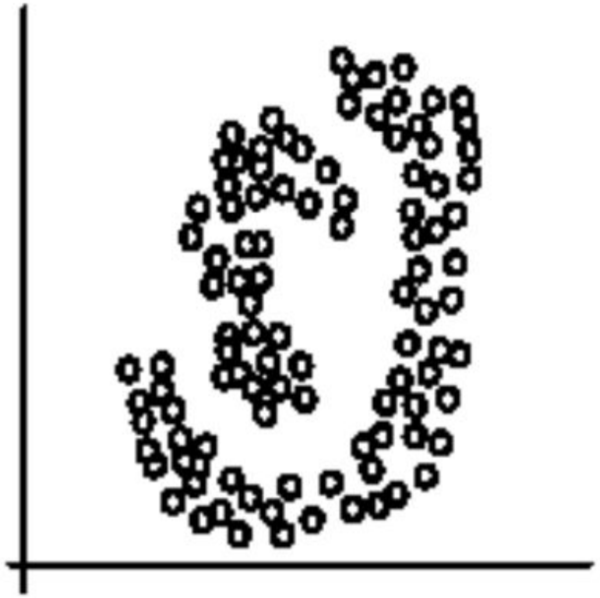
Sensitivity to initial seeds



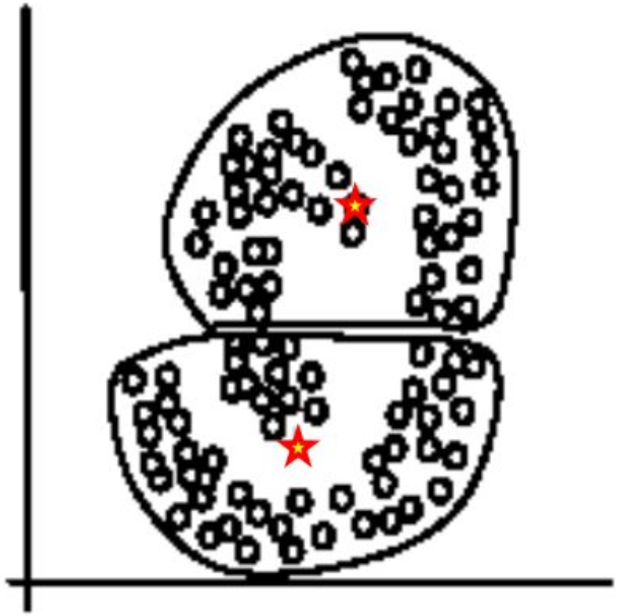
Initial seeds Iteration 1 Iteration 2

zeshan.khan@nu.edu.pk

Sensitivity to initial seeds



Two Clusters



K-Mean Clusters

zeshan.khan@nu.edu.pk

CS4104 Applied Machine Learning

Density Based Clustering

zeshan.khan@nu.edu.pk

Density-based Approaches

- Why Density-Based Clustering methods?
    - Discover clusters of arbitrary shape.
    - Clusters – Dense regions of objects separated by regions of low density
- DBSCAN – the first density based clustering
- OPTICS – density based cluster-ordering
- DENCLUE – a general density-based description of cluster and clustering

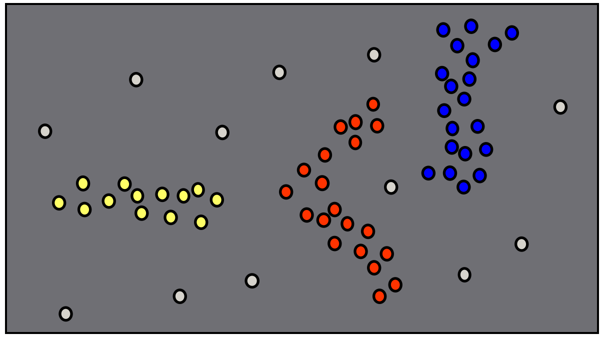
DBSCAN: Density Based Spatial Clustering of Applications with Noise

- Proposed by Ester, Kriegel, Sander, and Xu (KDD96)
- Relies on a density-based notion of cluster: A cluster is defined as a maximal set of density-connected points.
- Discovers clusters of arbitrary shape in spatial databases with noise

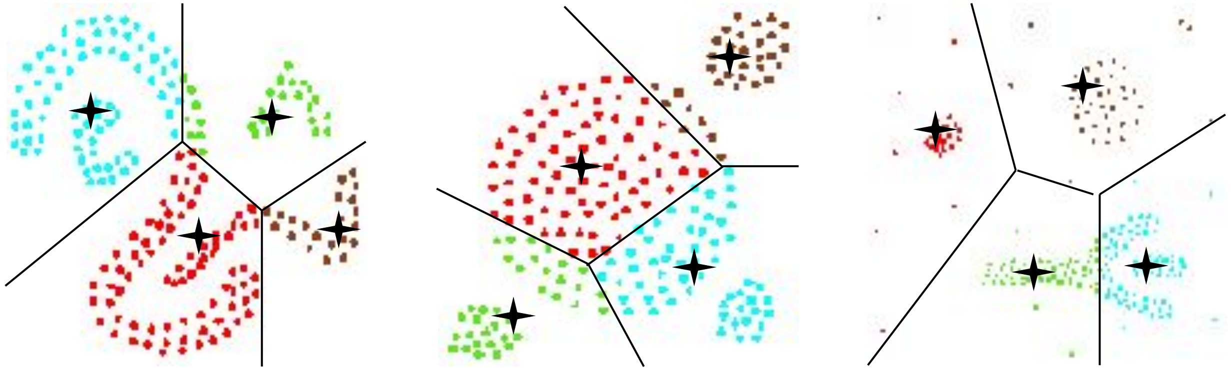
Density-Based Clustering

- *Basic Idea*:

  Clusters are dense regions in the data space, separated by regions of lower object density



- Why Density-Based Clustering?



Results of a k-medoid algorithm for k=4

Different density-based approaches exist (see Textbook & Papers)
Here we discuss the ideas underlying the DBSCAN algorithm

Density Based Clustering: Basic Concept

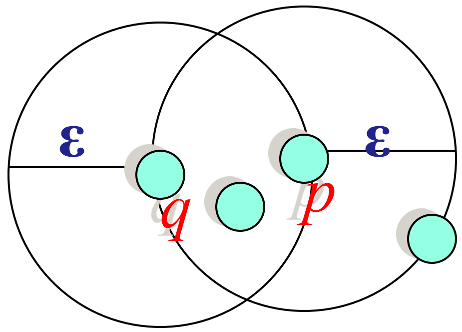
- Intuition for the formalization of the basic idea
  - For any point in a cluster, the local point density around that point has to exceed some threshold
  - The set of points from one cluster is spatially connected
- Local point density at a point *p* defined by two parameters
  - ε – radius for the neighborhood of point p: $N_\varepsilon(p) := \{q \text{ in data set } D \mid dist(p, q) \leq \varepsilon\}$
  - *MinPts* – minimum number of points in the given neighbourhood *N*(*p*)

ε-Neighborhood

- ε-Neighborhood – Objects within a radius of ε from an object.

$$N_{\varepsilon}(p):\{q \mid d(p,q) \leq \varepsilon\}$$

- "High density" - ε-Neighborhood of an object contains at least *MinPts* of objects.

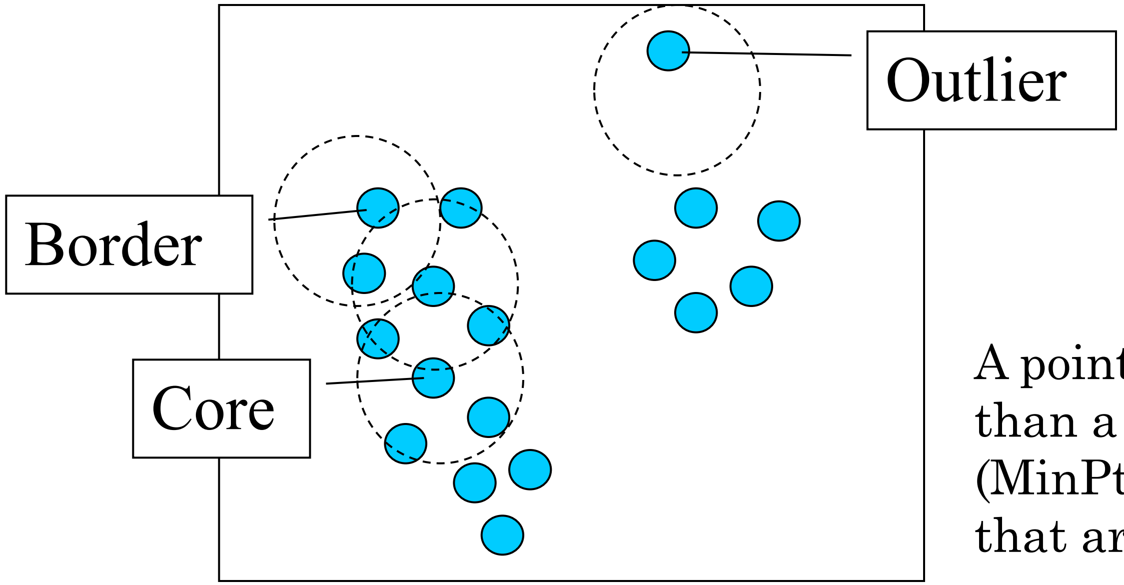


ε-Neighborhood of *p*
ε-Neighborhood of *q*

Density of p is "high" (MinPts = 4)

Density of q is "low" (MinPts = 4)

Core, Border & Outlier



ε = 1unit, MinPts = 5

Given ε and *MinPts*, categorize the objects into three exclusive groups.

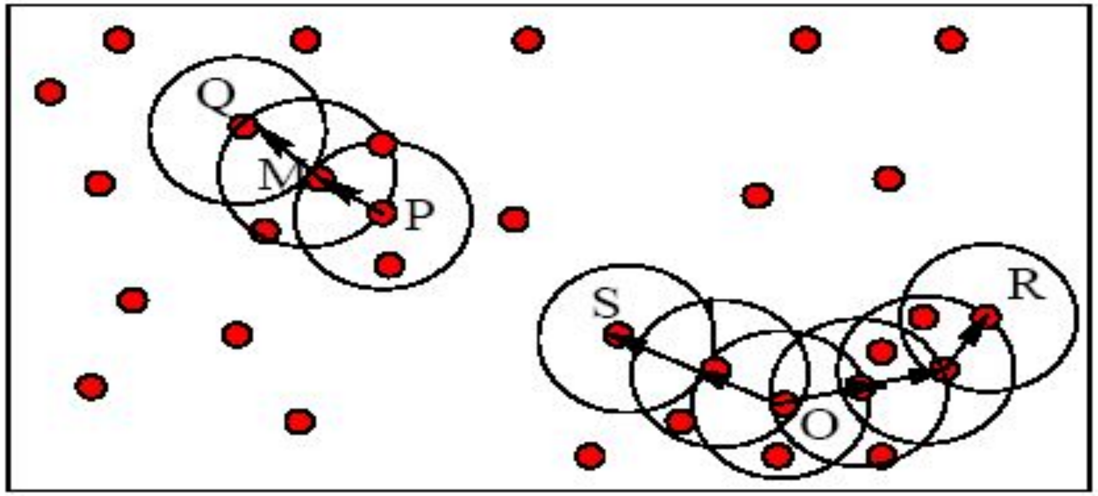
A point is a core point if it has more than a specified number of points (MinPts) within Eps These are points that are at the interior of a cluster.

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 not a core point nor a border point.

Example

- M, P, O, and R are core objects since each is in an Eps neighborhood containing at least 3 points

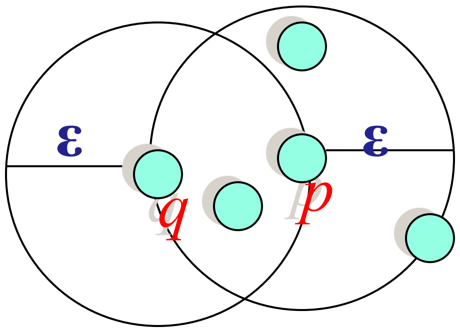


Minpts = 3

Eps=radius of the circles

Density-Reachability

- Directly density-reachable
  - An object q is directly density-reachable from object p if p is a core object and q is in p's ε-neighborhood.

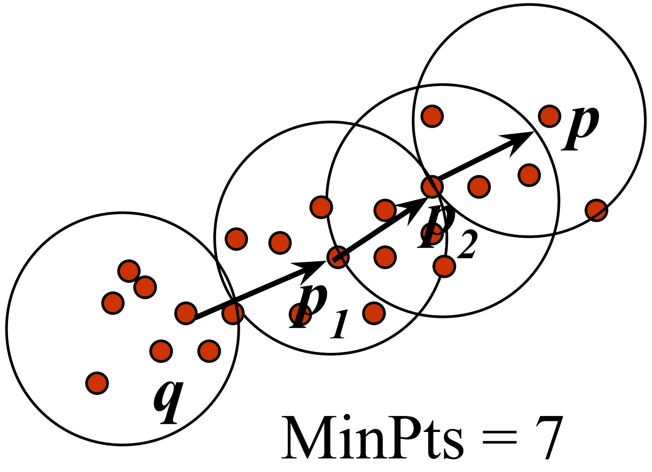


MinPts = 4

- q is directly density-reachable from p
- p is not directly density- reachable from q?
- Density-reachability is asymmetric.

Density-reachability

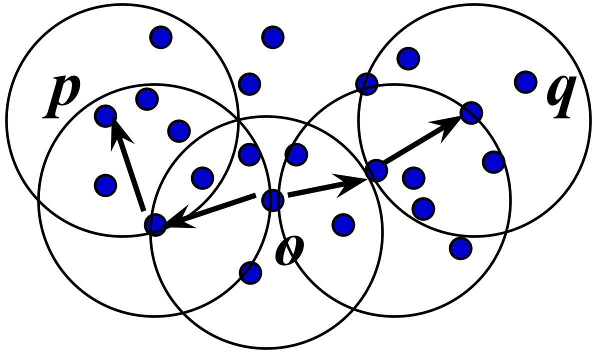
- Density-Reachable (directly and indirectly):
  - A point p is directly density-reachable from p2;
  - p2 is directly density-reachable from p1;
  - p1 is directly density-reachable from q;
  - p□p2□p1□q form a chain.



- **p is (indirectly) density-reachable from q**
- **q is not density- reachable from p?**

Density-Connectivity

- Density-reachable is not symmetric
  - not good enough to describe clusters
- Density-Connected
  - A pair of points p and q are density-connected if they are commonly density-reachable from a point o.



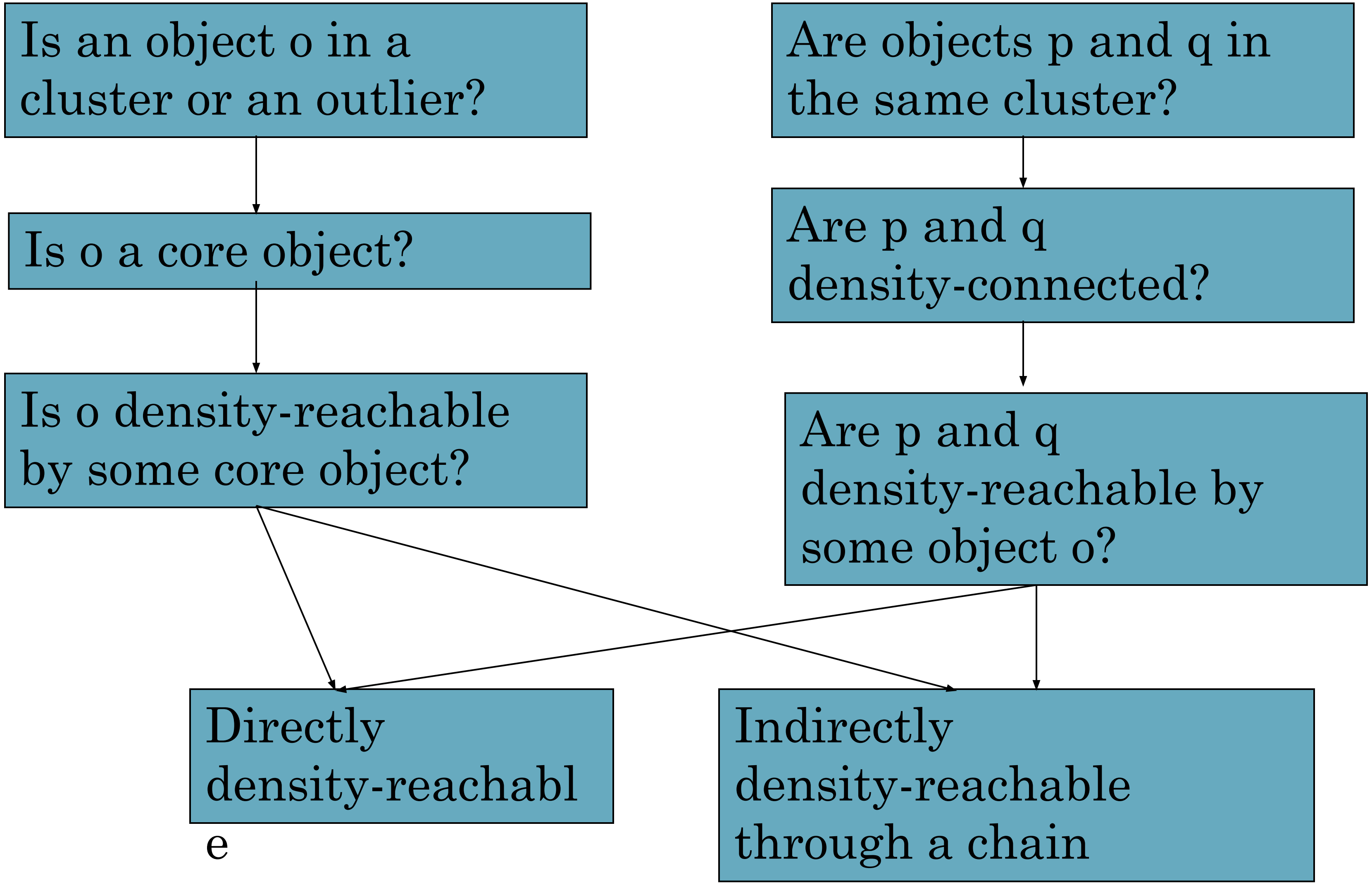
- Density-connectivity is symmetric

Formal Description of Cluster

- Given a data set D, parameter ε and threshold MinPts.
- A cluster C is a subset of objects satisfying two criteria:
  - *Connected:* □ p,q □C: p and q are density-connected.
  - *Maximal:* □ p,q: if p □C and q is density-reachable from p, then q □C. (avoid redundancy)

P is a core object.

Review of Concepts



DBSCAN Algorithm

```
Input: The data set D

Parameter: ε, MinPts

For each object p in D
    if p is a core object and not processed then
        C = retrieve all objects density-reachable from p
        mark all objects in C as processed
          report C as a cluster
    else mark p as outlier
    end if

End For
```

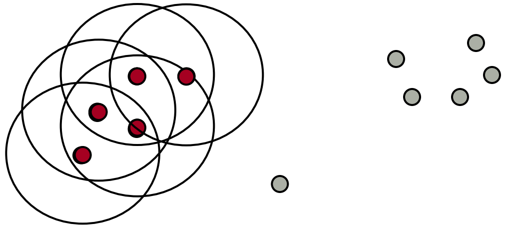
DBScan Algorithm

DBSCAN: The Algorithm

- Arbitrary select a point *p*
- Retrieve all points density-reachable from *p* wrt *Eps* 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.

DBSCAN Algorithm: Example

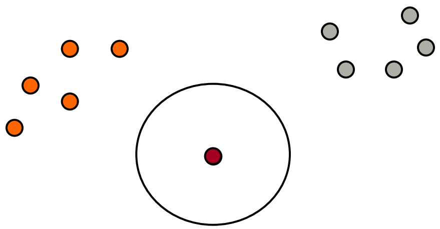
- Parameter
  - ε = 2 cm
  - $MinPts$ = 3



for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

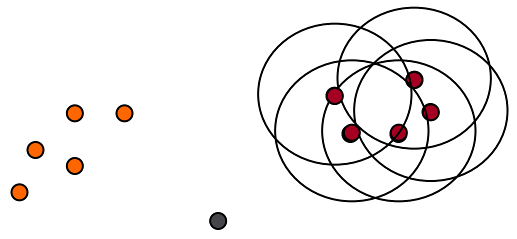
- Parameter
  - $\varepsilon = 2$ cm
  - $MinPts = 3$



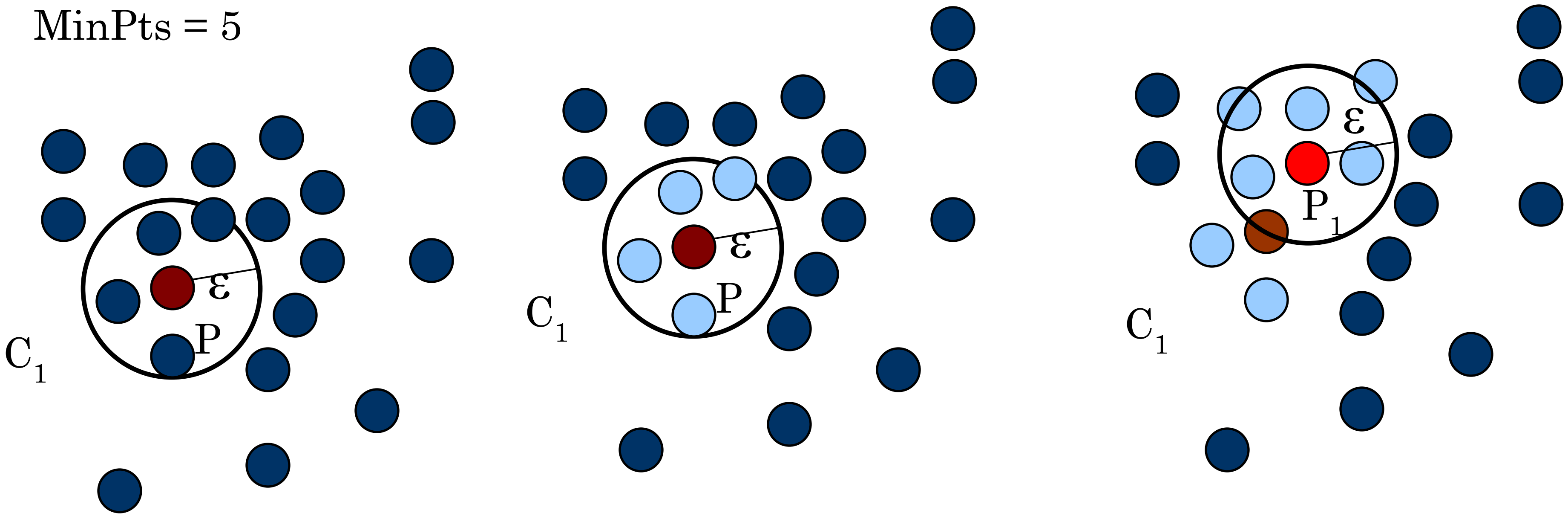
for each $o \in D$ **do**
    if o is not yet classified **then**
        if o is a core-object **then**
            collect all objects density-reachable from o
            and assign them to a new cluster.
        else
            assign o to NOISE

DBSCAN Algorithm: Example

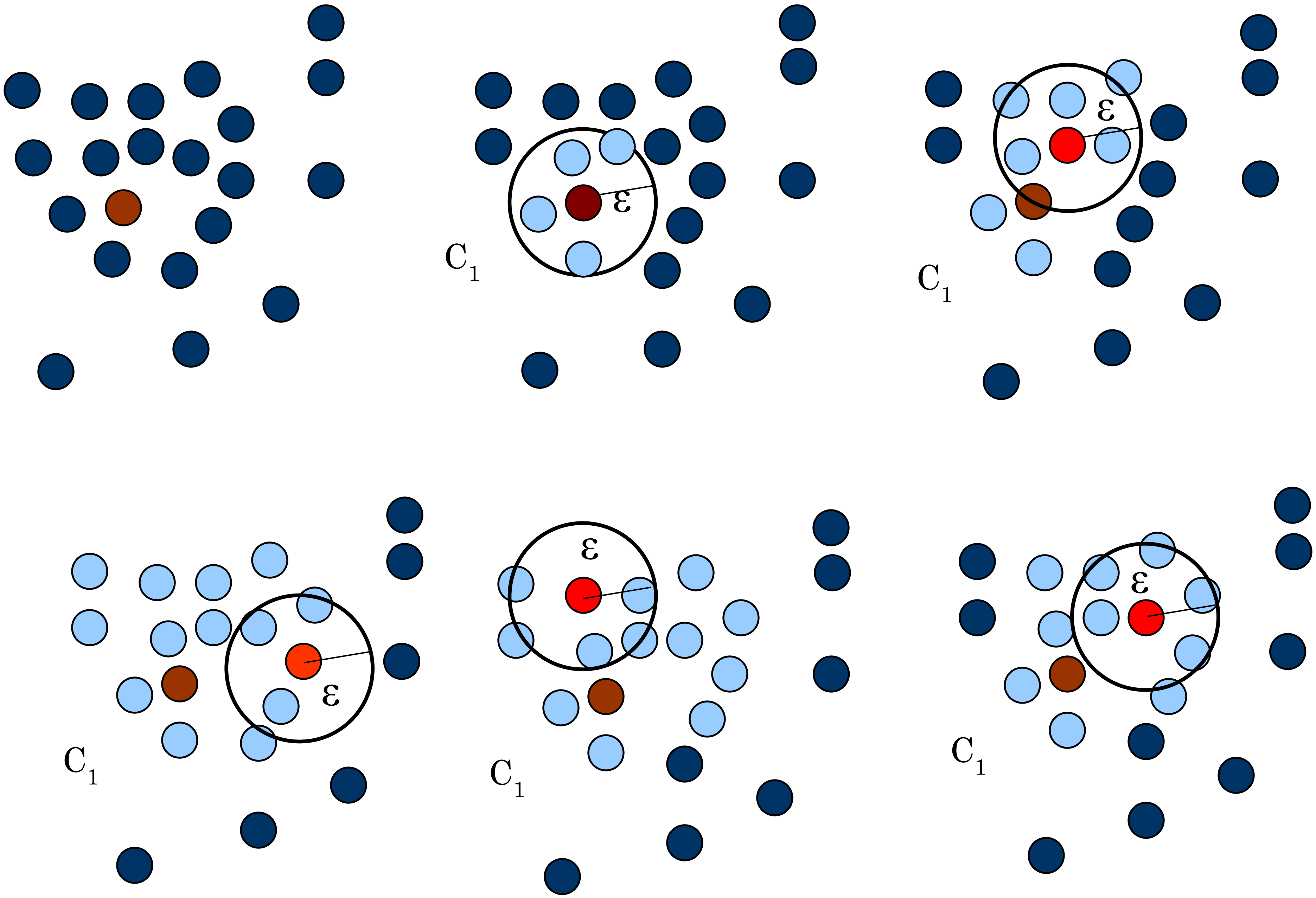
- Parameter
  - $\varepsilon = 2$ cm
  - $MinPts = 3$



for each $o \in D$ **do**
  if o is not yet classified **then**
    if o is a core-object **then**
      collect all objects density-reachable from o
      and assign them to a new cluster.
    else
      assign o to NOISE

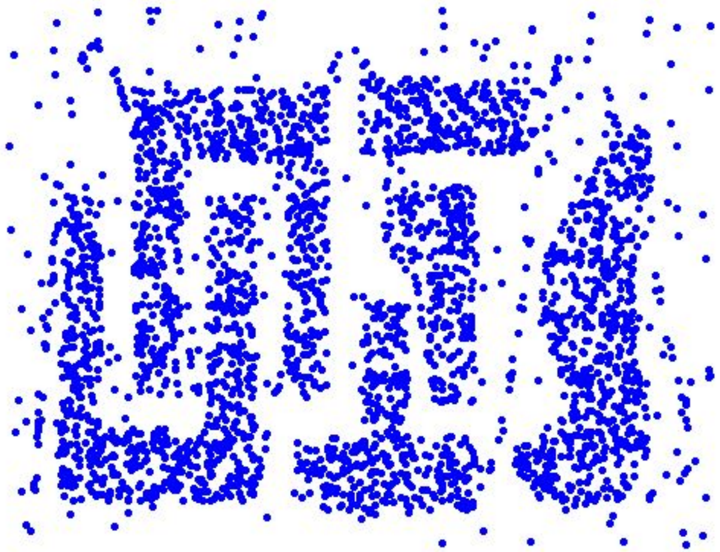


1. Check the ε-neighborhood of p;	1. Check the unprocessed objects in C
2. If p has less than MinPts neighbors then mark p as outlier and continue with the next object	2. If no core object, return C
3. Otherwise mark p as processed and put all the neighbors in cluster C	3. Otherwise, randomly pick up one core object p_1, mark p_1 as processed, and put all unprocessed neighbors of p_1 in cluster C

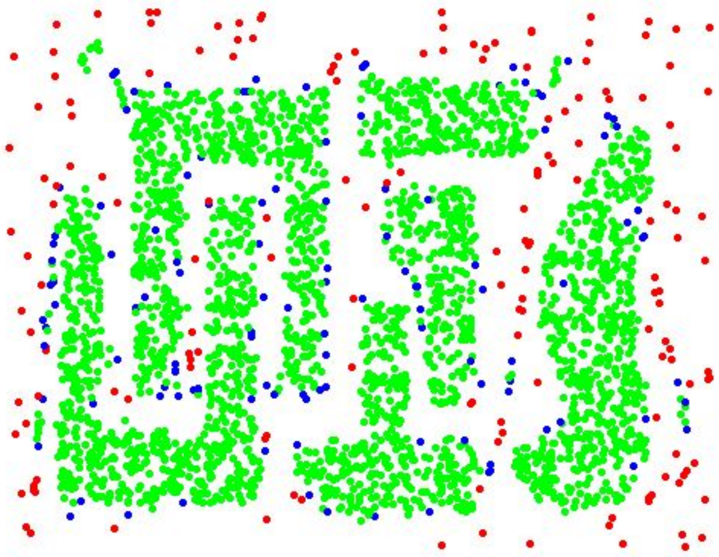


ε
C_1
ε
C_1
C_1
ε
ε
C_1
ε
C_1

Example



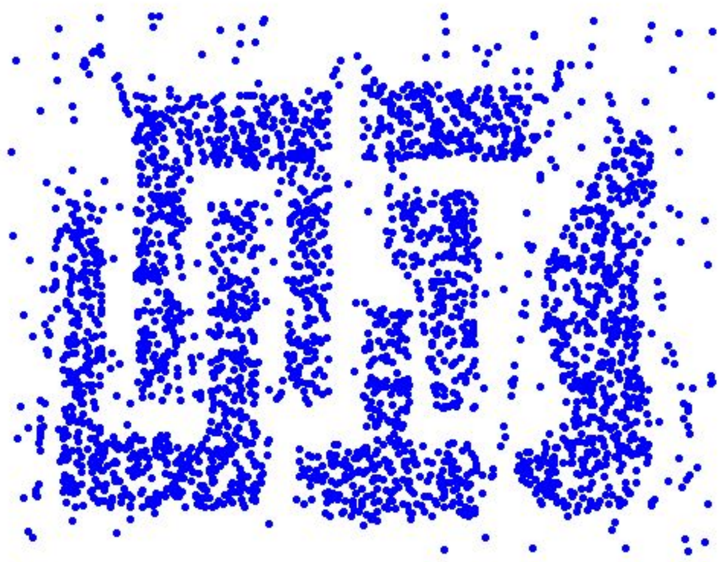
Original Points



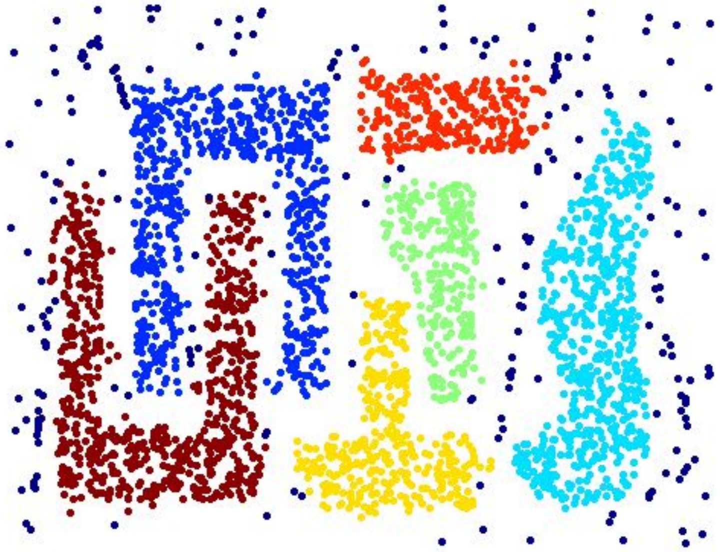
Point types: core,
border and outliers

ε = 10, MinPts = 4

When DBSCAN Works Well



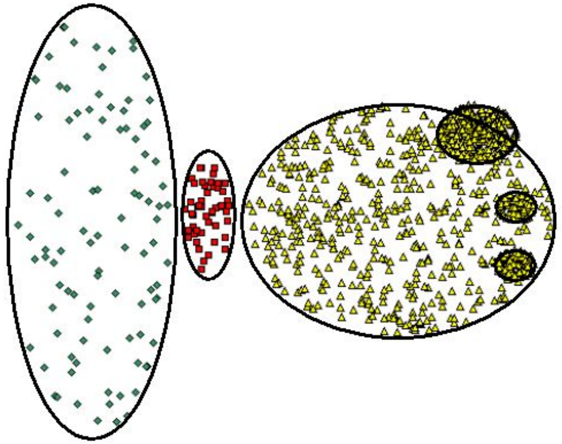
Original Points



Clusters

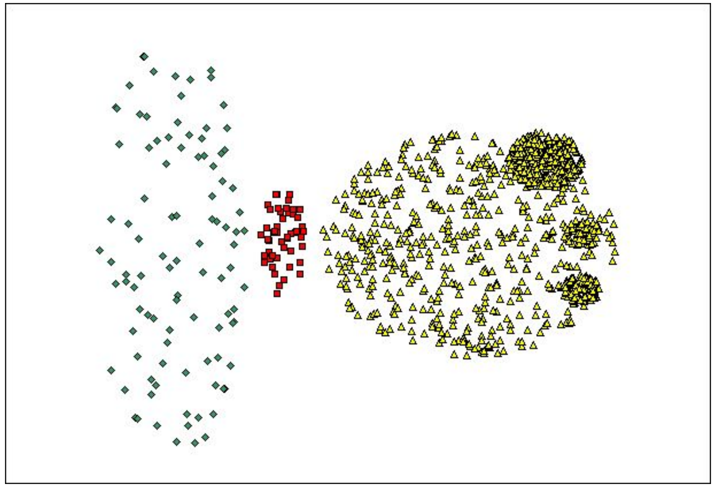
- **Resistant to Noise**
- **Can handle clusters of different shapes and sizes**

When DBSCAN Does NOT Work Well

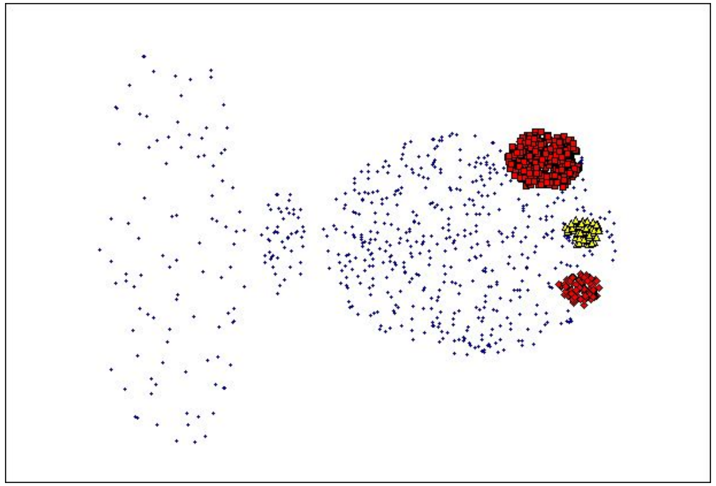


Original Points

- **Cannot handle Varying densities**
- **sensitive to parameters**



(MinPts=4, Eps=9.92).



(MinPts=4, Eps=9.75)

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

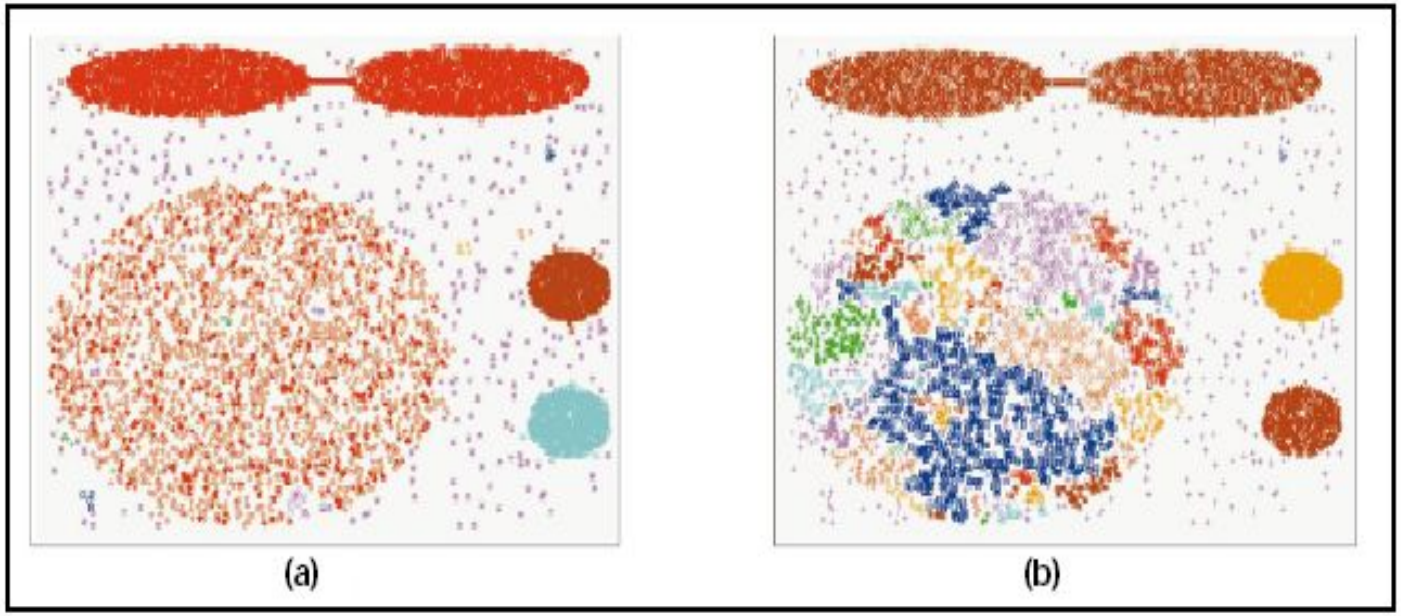
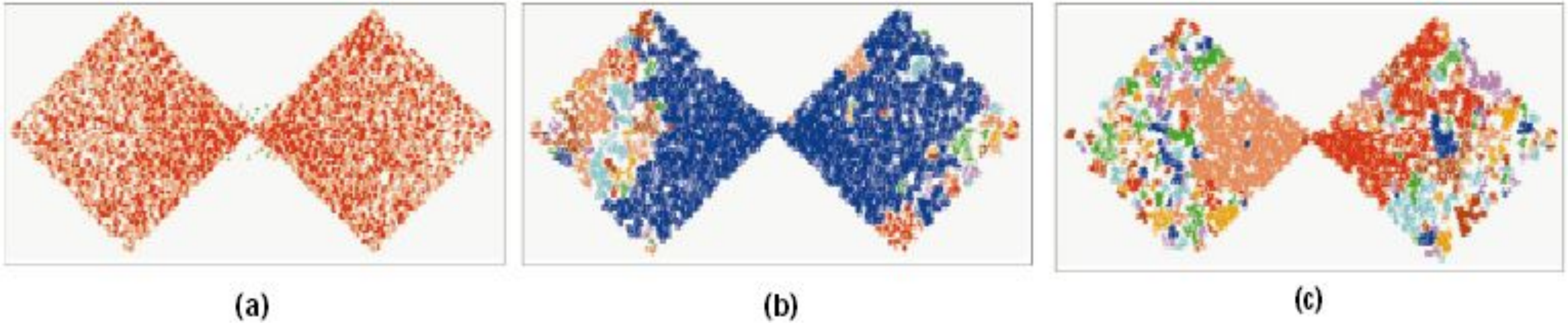
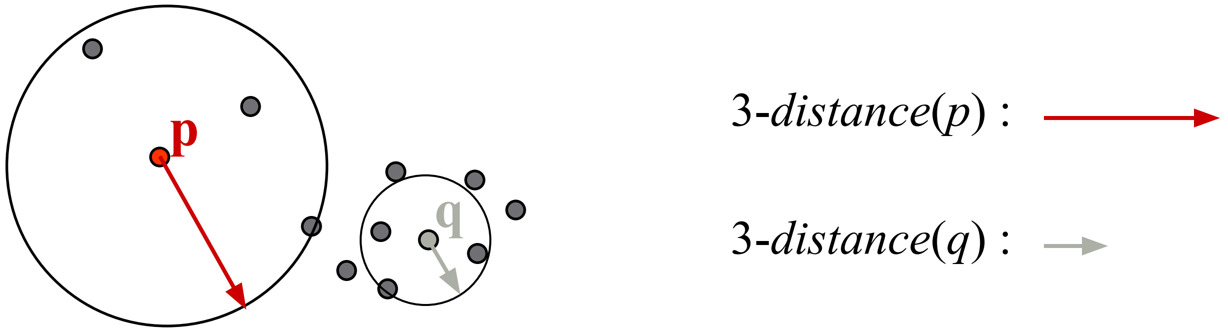


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



Determining the Parameters ε and *MinPts*

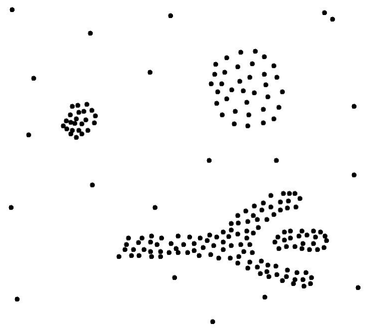
- Cluster: Point density higher than specified by ε and *MinPts*
- Idea: use the point density of the least dense cluster in the data set as parameters – but how to determine this?
- Heuristic: look at the distances to the k-nearest neighbors

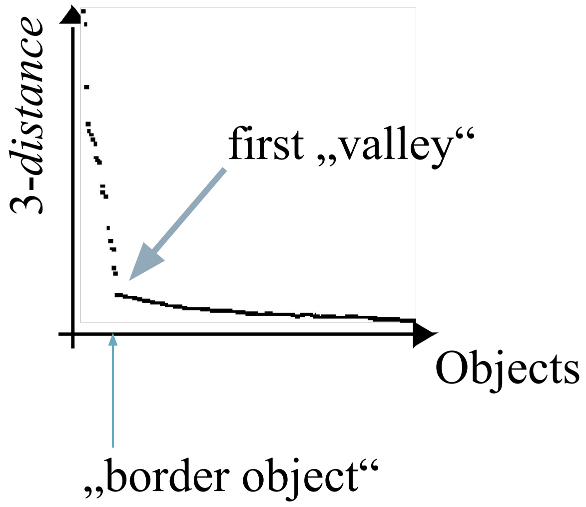


- Function k-*distance*(p): distance from p to the its k-nearest neighbor
- k-*distance plot*: k-distances of all objects, sorted in decreasing order

Determining the Parameters ε and *MinPts*

- Example k-distance plot

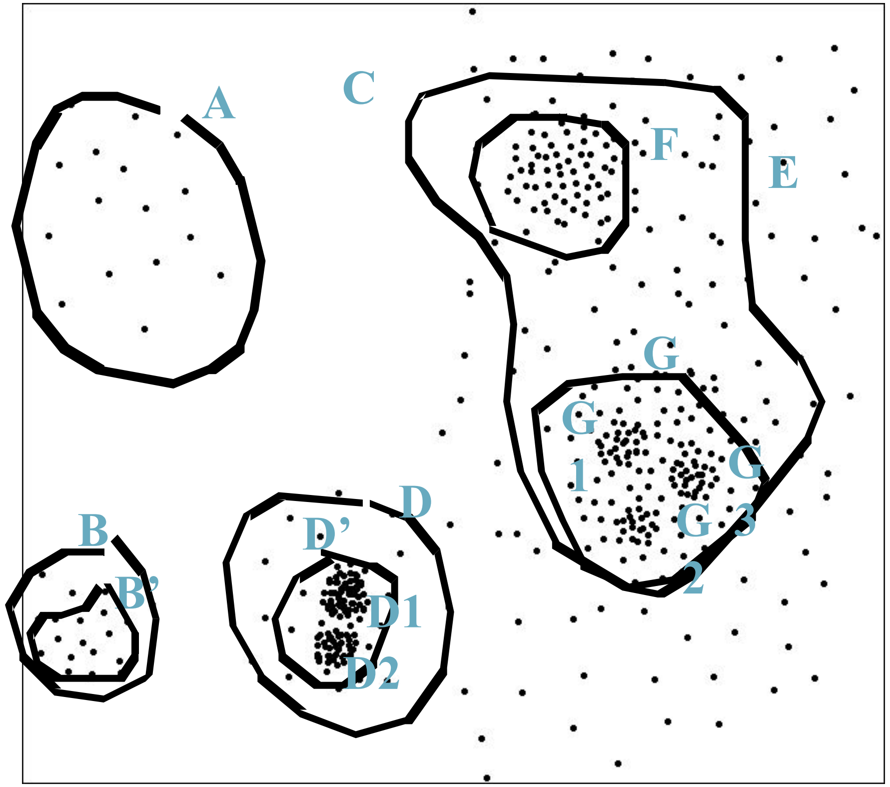


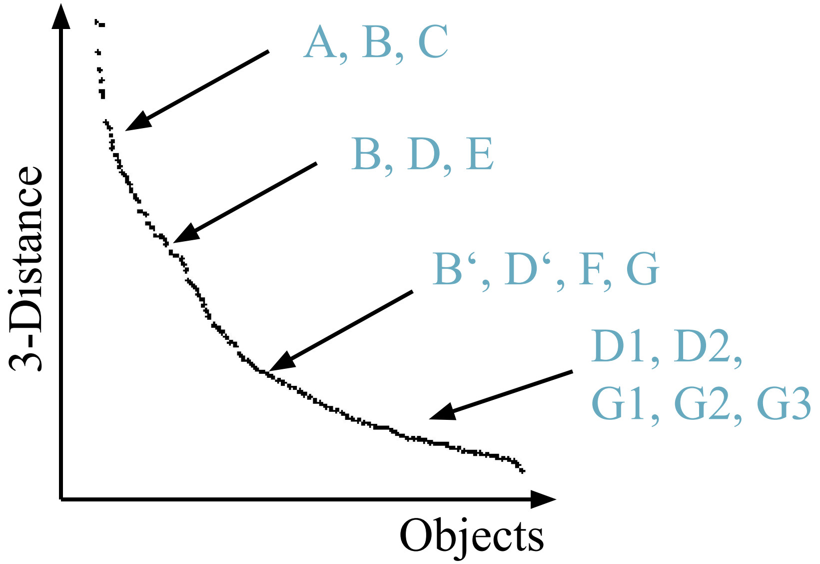


- Heuristic method:
  - Fix a value for *MinPts* (default: $2 \times d - 1$)
  - User selects “border object” o from the *MinPts-distance* plot;
    ε is set to *MinPts-distance*(o)

Determining the Parameters ε and *MinPts*

- Problematic example





Density Based Clustering: Discussion

- Advantages
  - Clusters can have arbitrary shape and size
  - Number of clusters is determined automatically
  - Can separate clusters from surrounding noise
  - Can be supported by spatial index structures
- Disadvantages
  - Input parameters may be difficult to determine
  - In some situations very sensitive to input parameter setting

OPTICS: Ordering Points To Identify the Clustering Structure

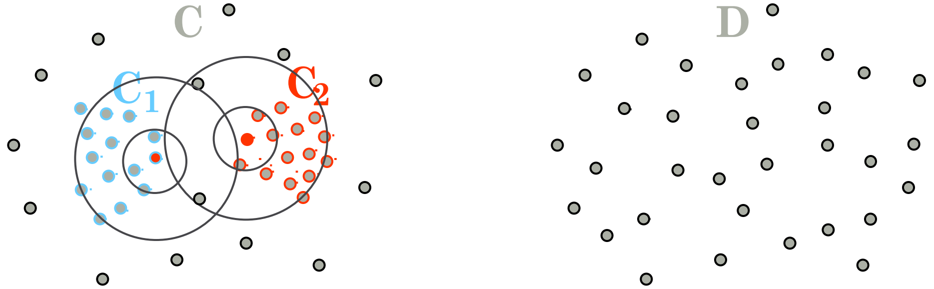
- DBSCAN
  - Input parameter – hard to determine.
  - Algorithm very sensitive to input parameters.
- OPTICS – Ankerst, Breunig, Kriegel, and Sander (SIGMOD'99)
  - Based on DBSCAN.
  - Does not produce clusters explicitly.
  - Rather generate an ordering of data objects representing density-based clustering structure.

OPTICS con’t

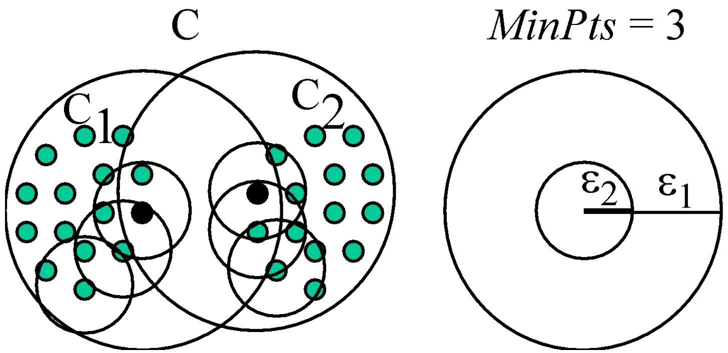
- Produces a special order of the database wrt its density-based clustering structure
- This cluster-ordering contains info equiv to the density-based clusterings corresponding to a broad range of parameter settings
- Good for both automatic and interactive cluster analysis, including finding intrinsic clustering structure
- Can be represented graphically or using visualization techniques

Density-Based Hierarchical Clustering

- *Observation*: Dense clusters are completely contained by less dense clusters



- *Idea*: Process objects in the "right" order and keep track of point density in their neighborhood

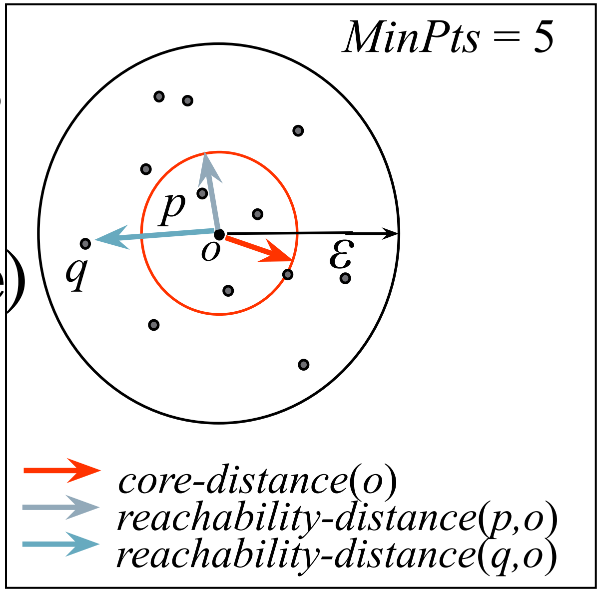


Core- and Reachability Distance

- Parameters: "generating" distance ε, fixed value *MinPts*
- *core-distance*$_{\varepsilon,MinPts}(o)$

  "smallest distance such that o is a core object"
  (if that distance is $\leq \varepsilon$; "?" otherwise)
- *reachability-distance*$_{\varepsilon,MinPts}(p, o)$

  "smallest distance such that p is *directly* density-reachable from o"
  (if that distance is $\leq \varepsilon$; "?" otherwise)



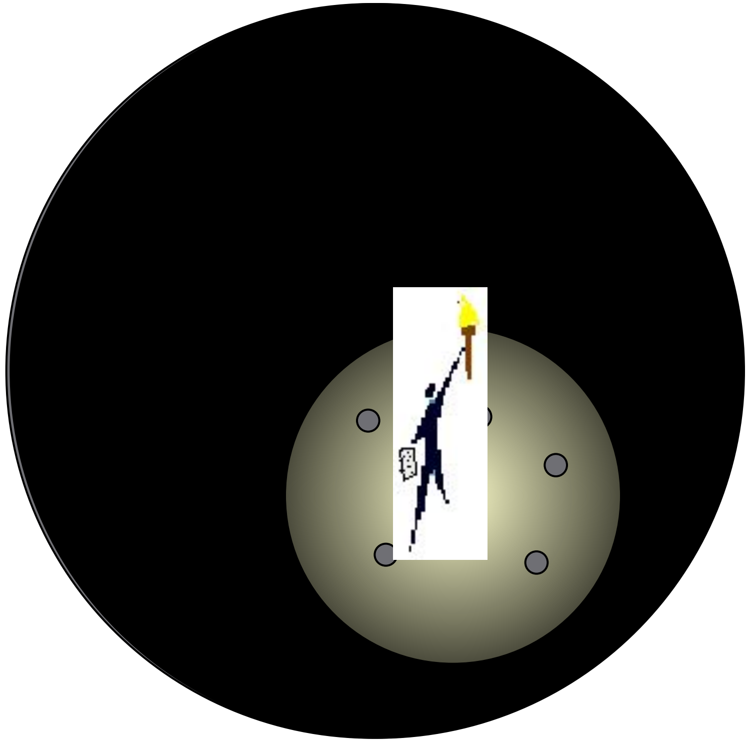
OPTICS: Extension of DBSCAN

- Order points by shortest *reachability distance* to guarantee that clusters w.r.t. higher density are finished first. (for a constant MinPts, higher

  density requires lower ε)

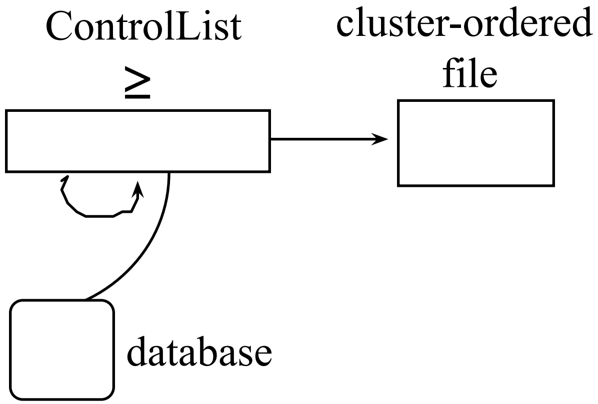
The Algorithm OPTICS

- Basic data structure: controlList
  - Memorize shortest reachability distances seen so far ("distance of a jump to that point")
- Visit each point
  - Make always a shortest jump
- Output:
  - order of points
  - core-distance of points
  - reachability-distance of points



The Algorithm OPTICS

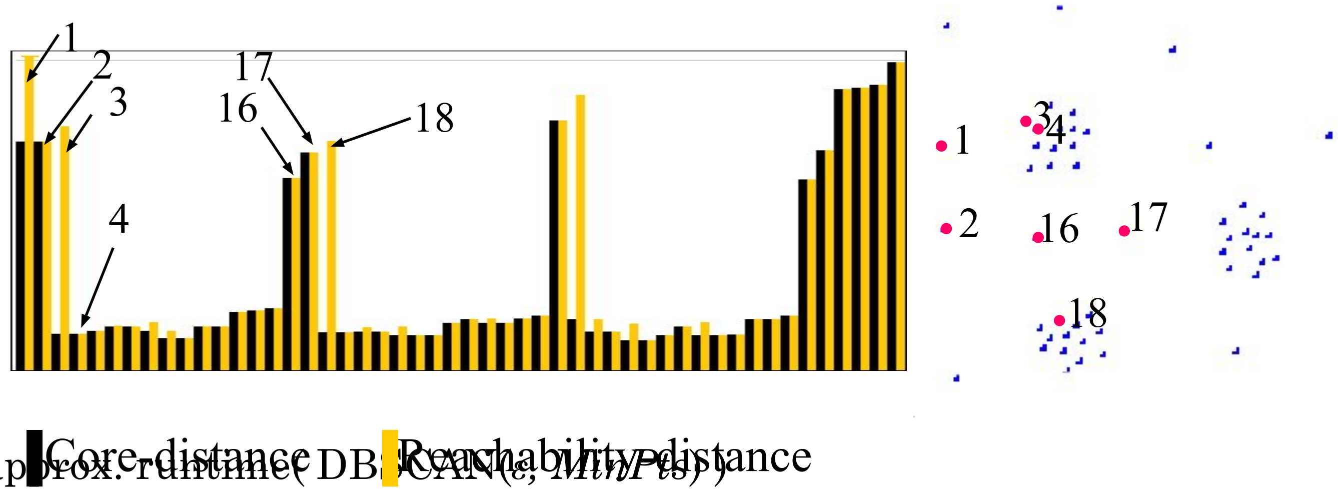
- *ControlList* ordered by reachability-distance.



foreach $o \in$ Database
// initially, o.processed = false for all objects o
if o.processed = false;
insert (o, *"?"*) into *ControlList;*
while *ControlList* is not empty
select first element (o, *r-dist*) from *ControlList*;
retrieve $N_\varepsilon(o)$ and determine *c_dist*= *core-distance*(o);
set o.processed = true;
write (o, *r_dist*, *c_dist*) to file;
if o is a core object at any distance $\leq \varepsilon$
foreach $p \in N_\varepsilon(o)$ not yet processed;
determine r_dist_p = *reachability-distance*(p, o);
if (p, _) $\notin$ *ControlList*
insert (p, r_dist_p) in *ControlList;*
else if (p, *old_r_dist*) $\in$ *ControlList* **and** r_dist_p < *old_r_dist*
update (p, r_dist_p) in *ControlList*;

OPTICS: Properties

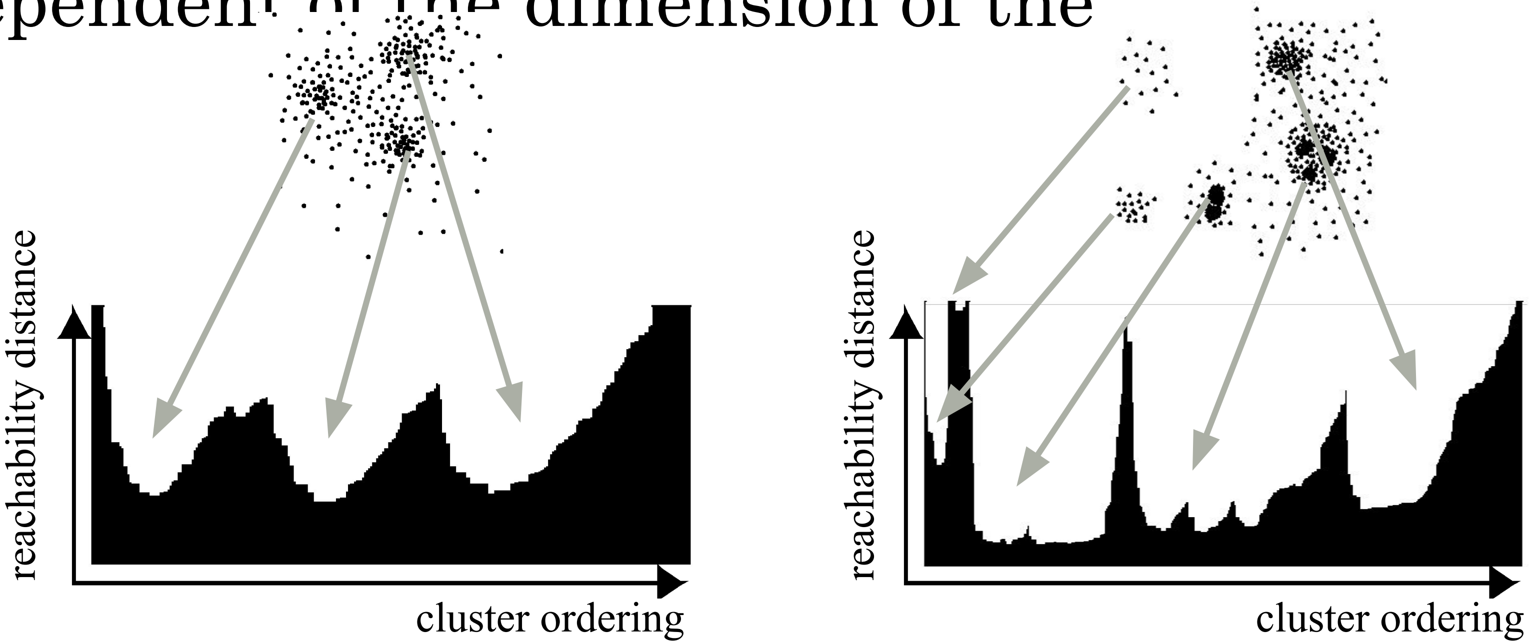
- “Flat” density-based clusters wrt. ε* ≤ ε and *MinPts* afterwards:
  - Starts with an object *o* where *c-dist*(*o*) ≤ ε* and *r-dist*(*o*) > ε*
  - Continues while *r-dist* ≤ ε*



- Performance: approx. runtime(DBSCAN(ε, *MinPts*))
  - O(*n* * runtime(ε-neighborhood-query))
    - without spatial index support (worst case): O(n^2)
    - e.g. tree-based spatial index support: O(*n* * log(*n*))

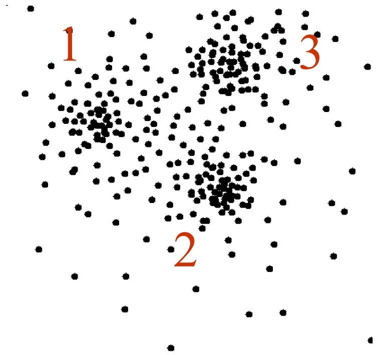
OPTICS: The Reachability Plot

- represents the density-based clustering structure
- easy to analyze
- independent of the dimension of the

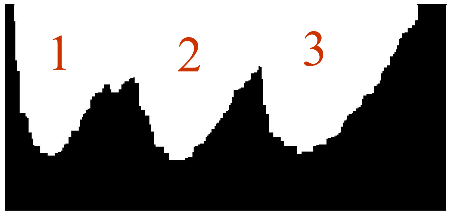


OPTICS: Parameter Sensitivity

- Relatively insensitive to parameter settings
- Good result if parameters are just "large enough"

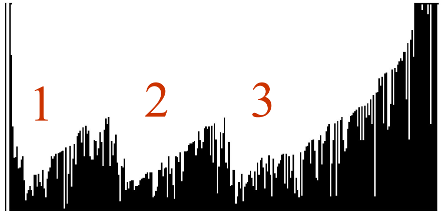


MinPts = 10, ε = 10



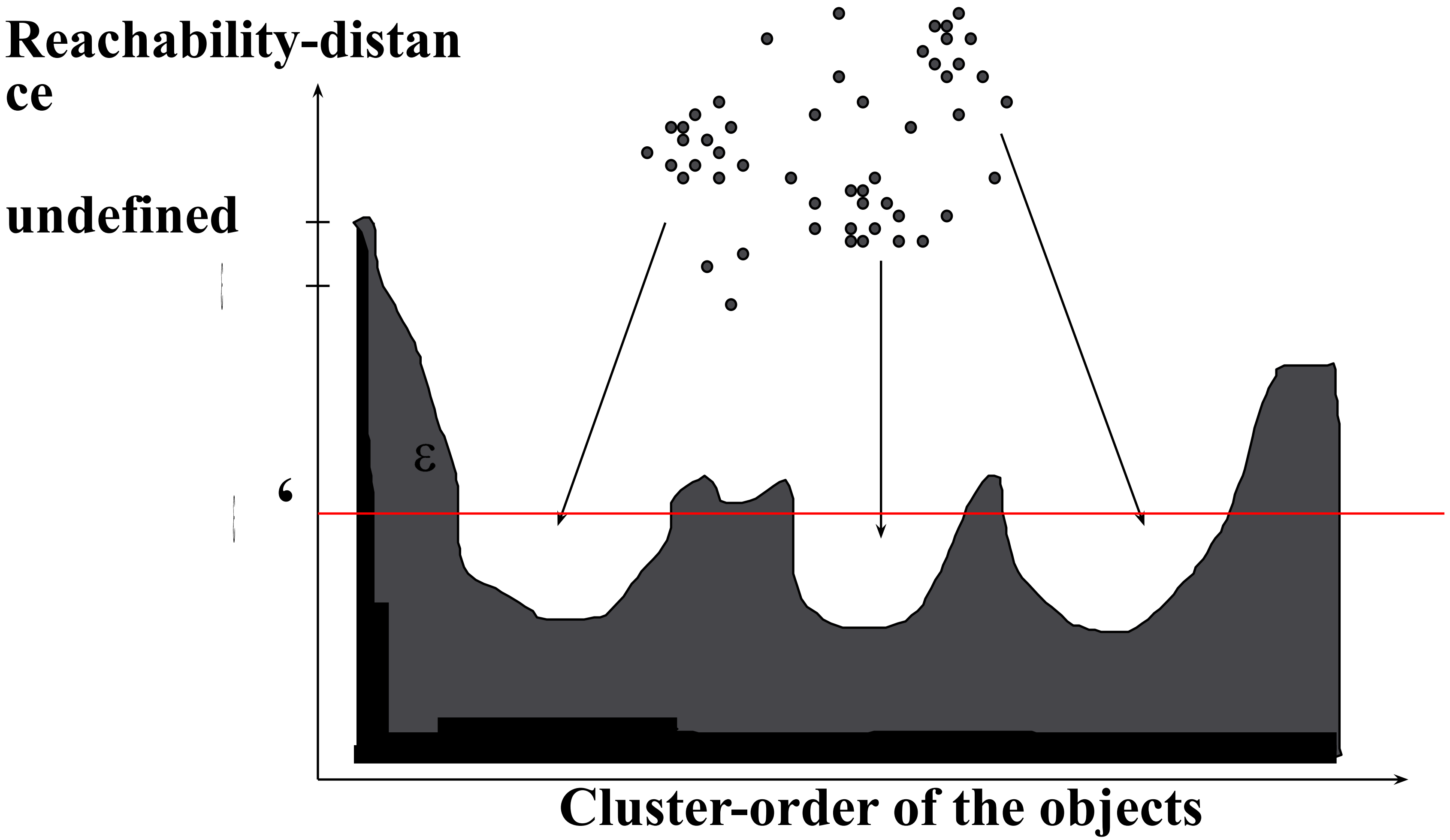
MinPts = 10, **ε = 5**

***MinPts* = 2**, ε = 10



An Example of OPTICS

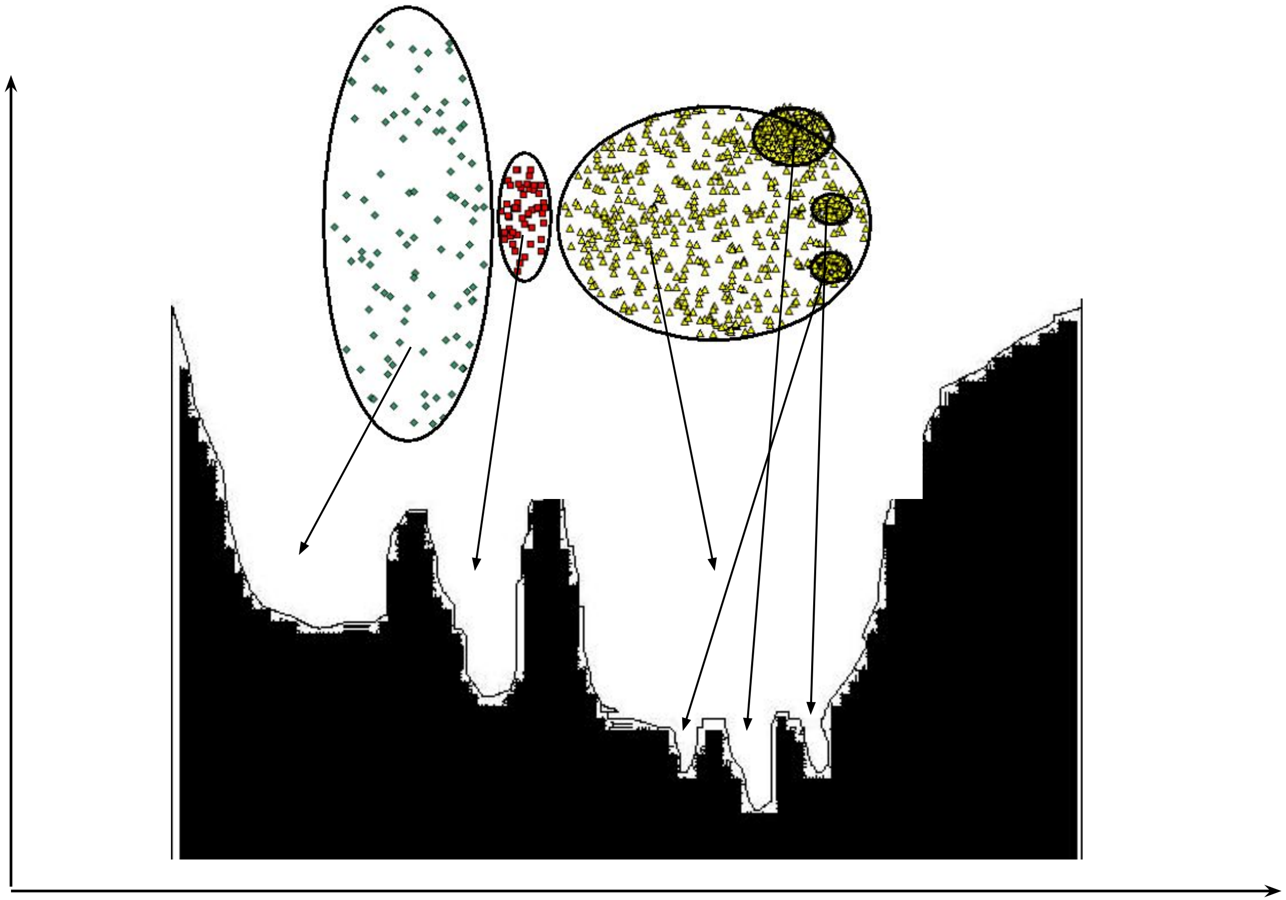
neighboring objects stay close to each other in a linear sequence.



DBSCAN VS OPTICS

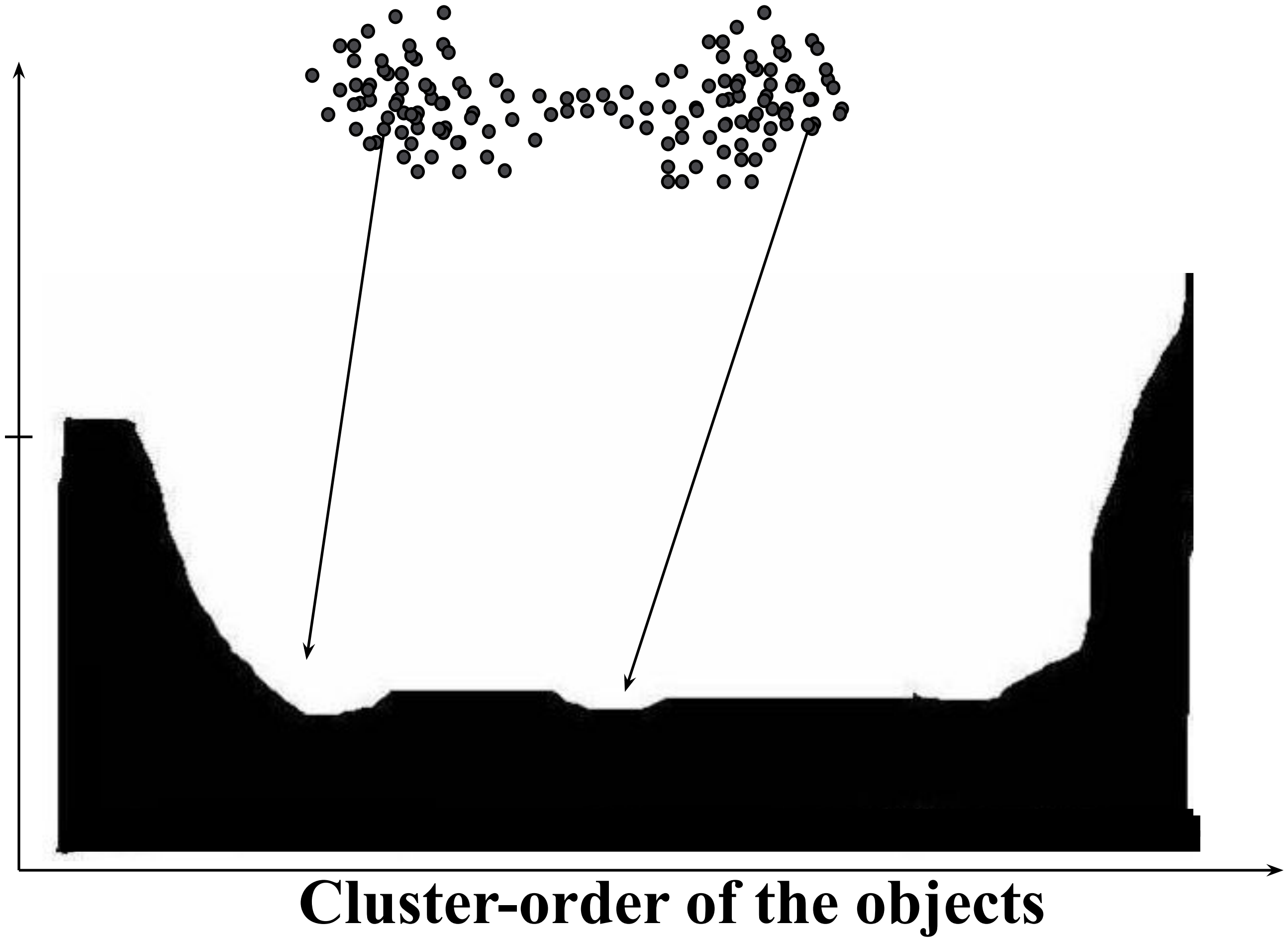
	DBSCAN	OPTICS
Density	Boolean value (high/low)	Numerical value (core distance)
Density-connected	Boolean value (yes/no)	Numerical value (reachability distance)
Searching strategy	random	greedy

When OPTICS Works Well



Cluster-order of the objects

When OPTICS Does NOT Work Well



DENCLUE: using density functions

- DENsity-based CLUstEring by Hinneburg & Keim (KDD'98)
- Major features
  - Solid mathematical foundation
  - Good for data sets with large amounts of noise
  - Allows a compact mathematical description of arbitrarily shaped clusters in high-dimensional data sets
  - Significantly faster than existing algorithm (faster than DBSCAN by a factor of up to 45)
  - But needs a large number of parameters

Denclue: Technical Essence

- Model density by the notion of influence
- Each data object exert influence on its neighborhood.
- The influence decreases with distance
- Example:
  - Consider each object is a radio, the closer you are to the object, the louder the noise
- Key: Influence is represented by mathematical function

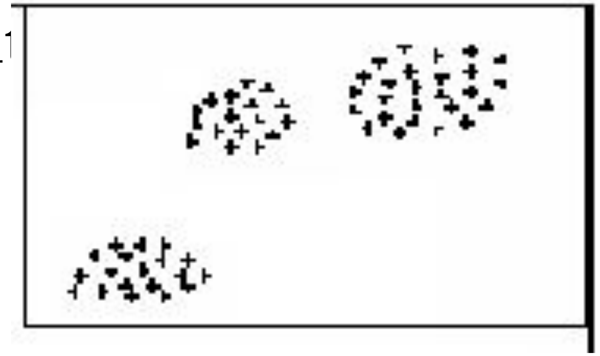
Denclue: Technical Essence

- Influence functions: (influence of y on x, σ is a user given constant)
  - Square : $f^{y}_{square}(x) = 0$, if dist(x,y) > σ,
    1, otherwise
  - Guassian:

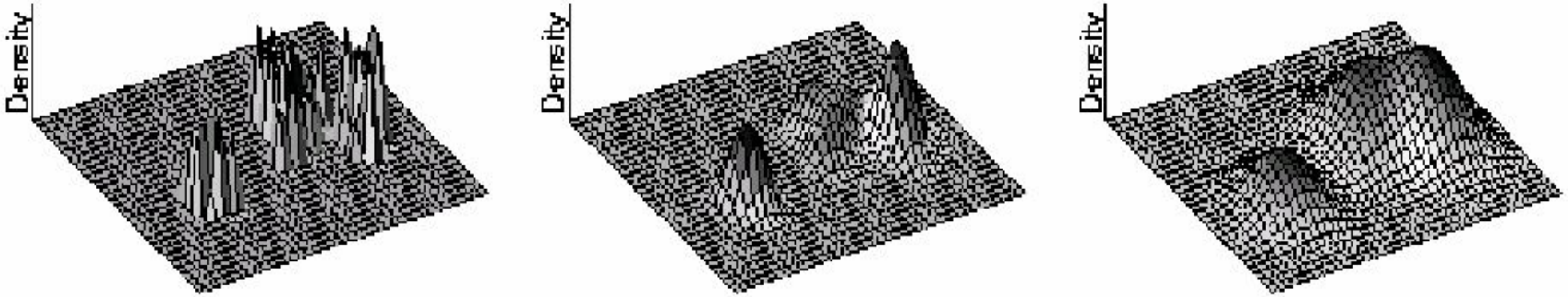
$$f^{y}_{Gaussian}(x) = e^{-\frac{d(x,y)^2}{2\sigma^2}}$$

Density Function

- Density Definition is defined as the sum of the influence functions of all data point



$$f_{Gaussian}^{D}(x) = \sum_{i=1}^{N} e^{-\frac{d(x,x_i)^2}{2\sigma^2}}$$



(a) $\sigma = 0.2$ (b) $\sigma = 0.6$ (d) $\sigma = 1.5$

Gradient: The steepness of a slope

- Example

$$f_{Gaussian}(x,y) = e^{-\frac{d(x,y)^2}{2\sigma^2}}$$

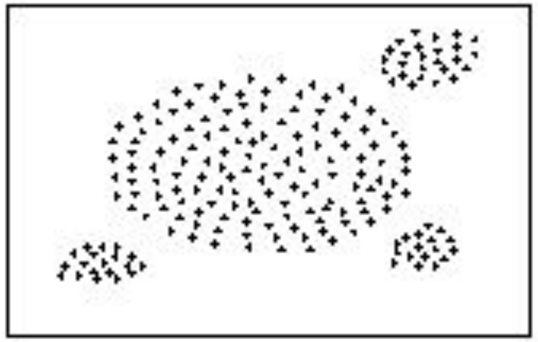
$$f_{Gaussian}^{D}(x) = \sum_{i=1}^{N} e^{-\frac{d(x,x_i)^2}{2\sigma^2}}$$

$$\nabla f_{Gaussian}^{D}(x,x_i) = \sum_{i=1}^{N} (x_i - x) \cdot e^{-\frac{d(x,x_i)^2}{2\sigma^2}}$$

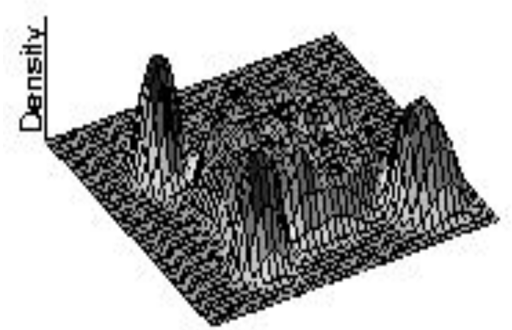
Denclue: Technical Essence

- Clusters can be determined mathematically by identifying density attractors.
- Density attractors are local maximum of the overall density function.

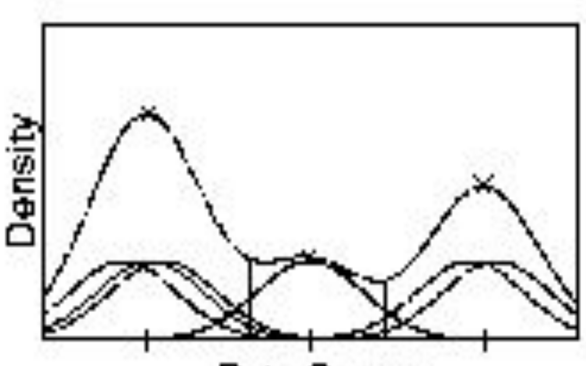
Density Attractor



(a) Data Set



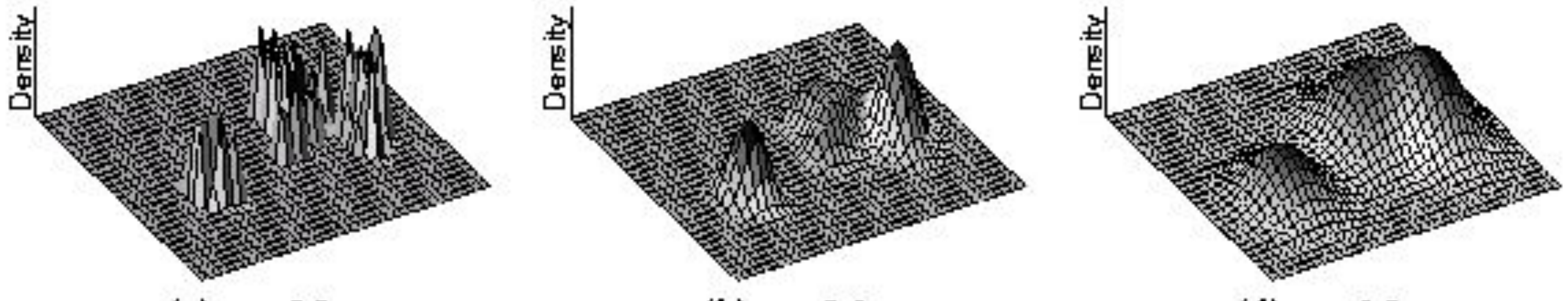
(c) Gaussian



Cluster Definition

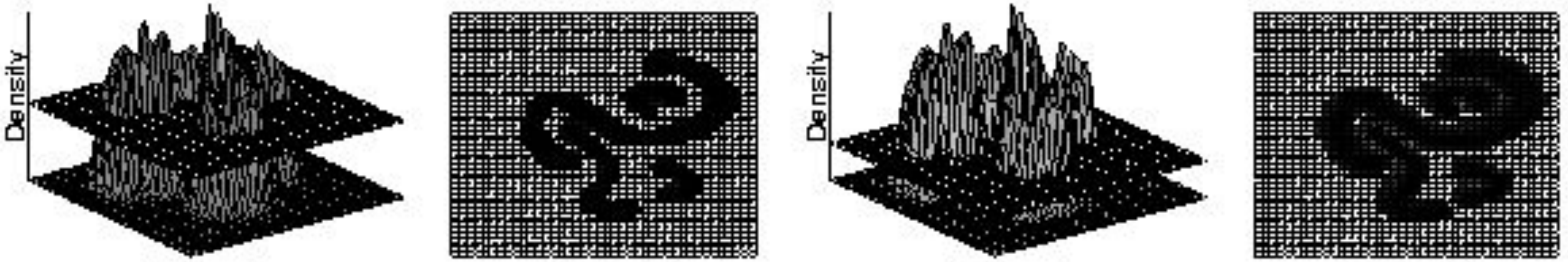
- Center-defined cluster
  - A subset of objects attracted by an attractor x
  - density(x) $\geq \xi$
- Arbitrary-shape cluster
  - A group of center-defined clusters which are connected by a path P
  - For each object x on P, density(x) $\geq \xi$.

Center-Defined and Arbitrary



(a) $\sigma = 0.2$ (b) $\sigma = 0.6$ (d) $\sigma = 1.5$

Figure 3: Example of Center-Defined Clusters for different σ



(a) $\xi = 2$ (b) $\xi = 2$ (c) $\xi = 1$ (d) $\xi = 1$

Figure 4: Example of Arbitray-Shape Clusters for different ξ

DENCLUE: How to find the clusters

- Divide the space into grids, with size 2σ
- Consider only grids that are highly populated
- For each object, calculate its density attractor using hill climbing technique
  - Tricks can be applied to avoid calculating density attractor of all points
- Density attractors form basis of all clusters

Features of DENCLUE

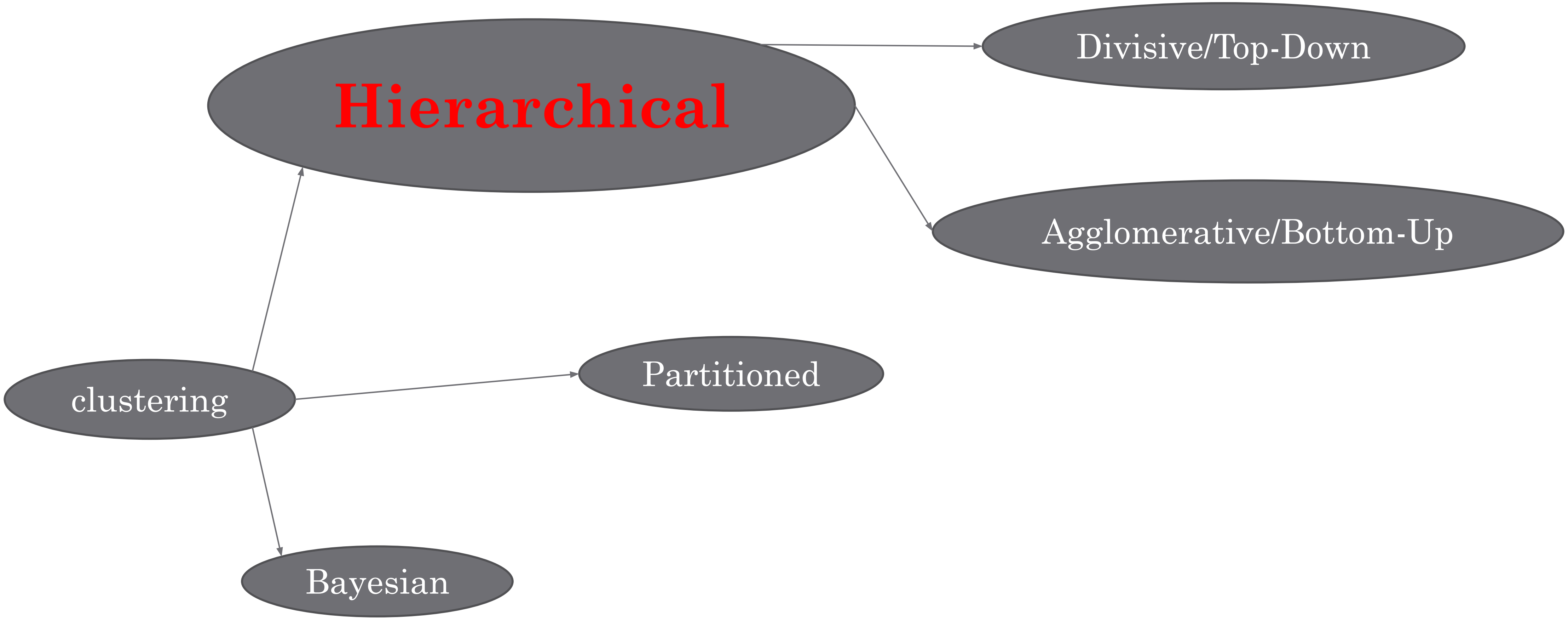
- Major features
  - Solid mathematical foundation
    - Compact definition for density and cluster
    - Flexible for both center-defined clusters and arbitrary-shape clusters
  - But needs a large number of parameters
    - σ: parameter to calculate density
    - ξ: density threshold
    - δ: parameter to calculate attractor

CS4104 Applied Machine Learning

Hierarical Clustering

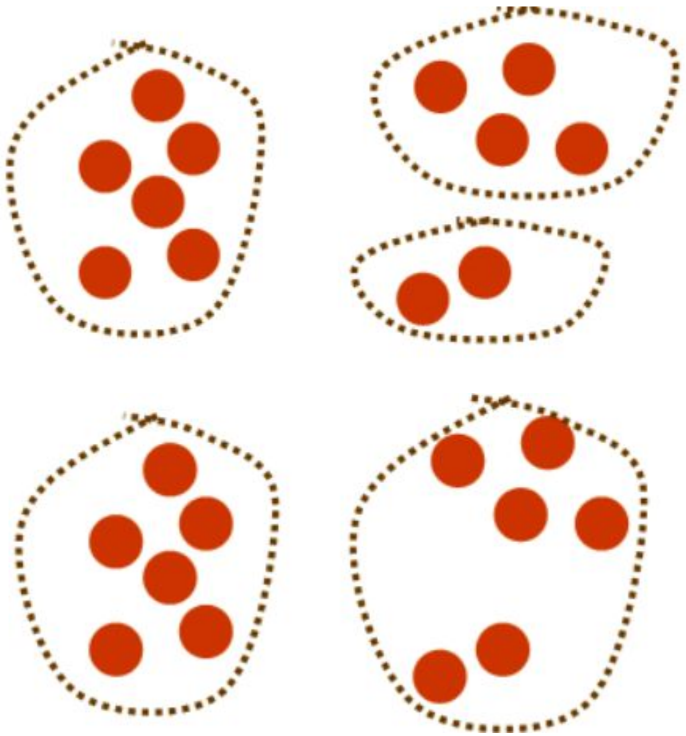
zeshan.khan@nu.edu.pk

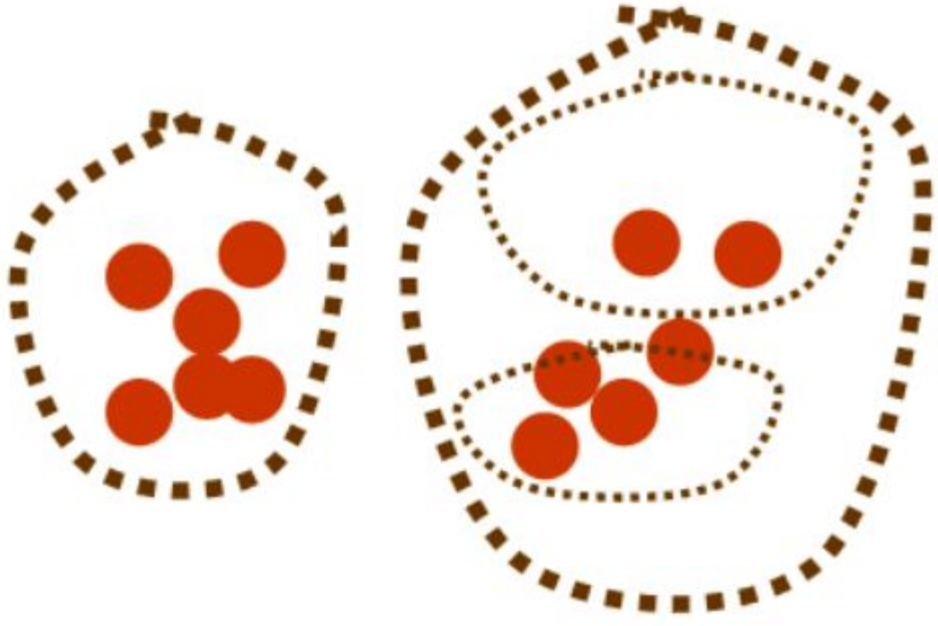
Clustering techniques



zeshan.khan@nu.edu.pk

Flat vs Hierarchical Clustering





Types of hierarchical clustering

Divisive (top down) clustering

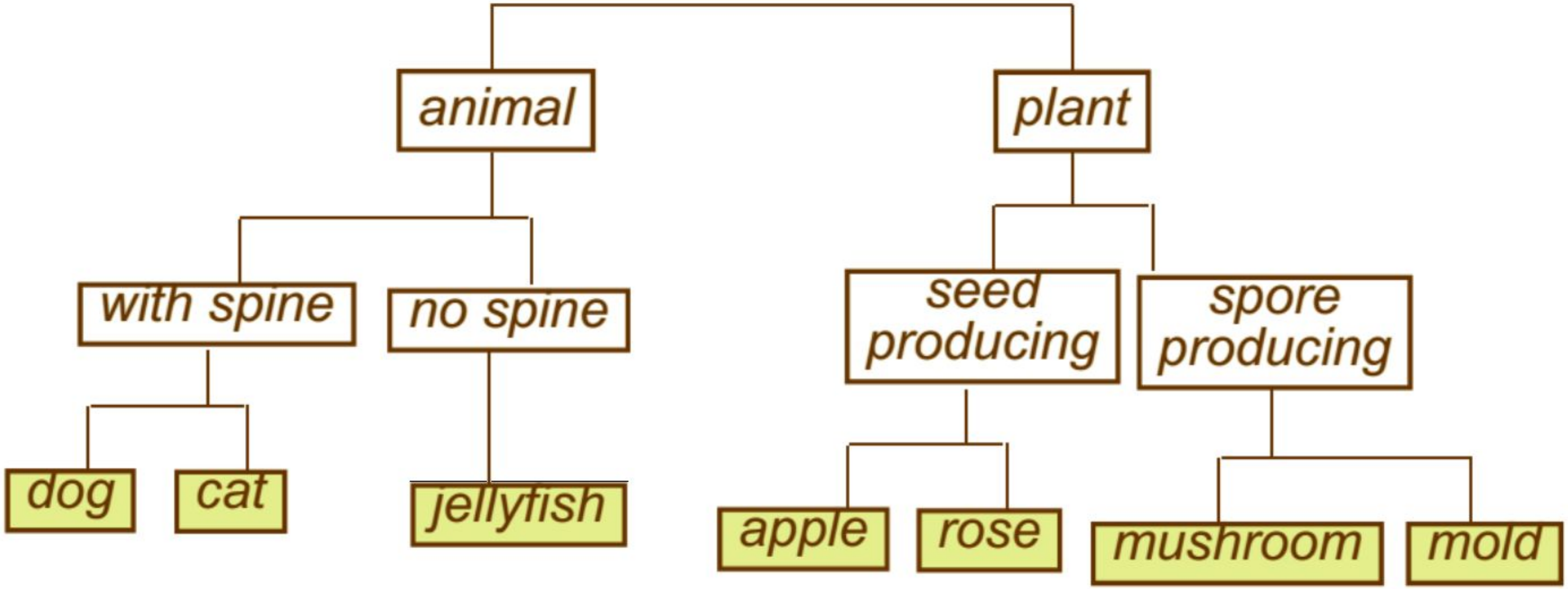
- Starts with all data points in one cluster, the root, then
  - Splits the root into a set of child clusters. Each child cluster is recursively divided further
  - stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point

Agglomerative (bottom up) clustering

- The dendrogram is built from the bottom level by
  - merging the most similar (or nearest) pair of clusters
  - stopping when all the data points are merged into a single cluster (i.e., the root cluster)

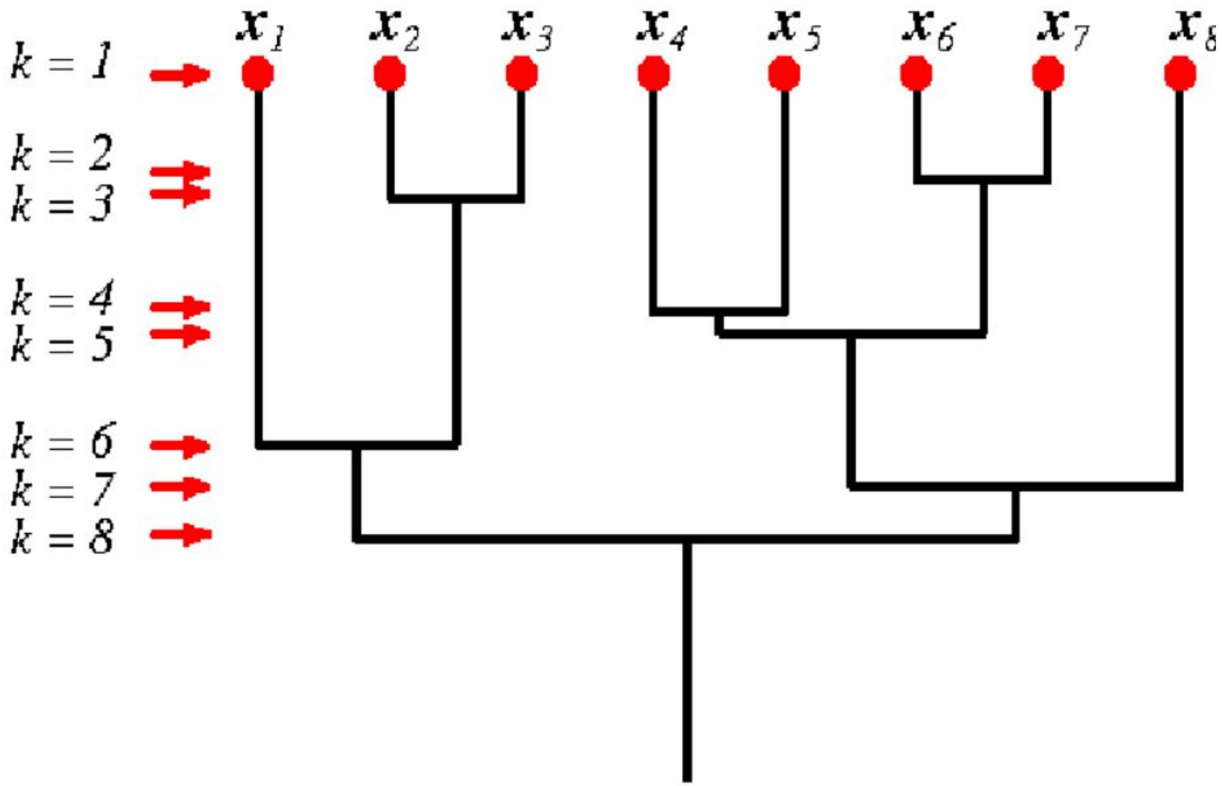
zeshan.khan@nu.edu.pk

Divisive hierarchical clustering



zeshan.khan@nu.edu.pk

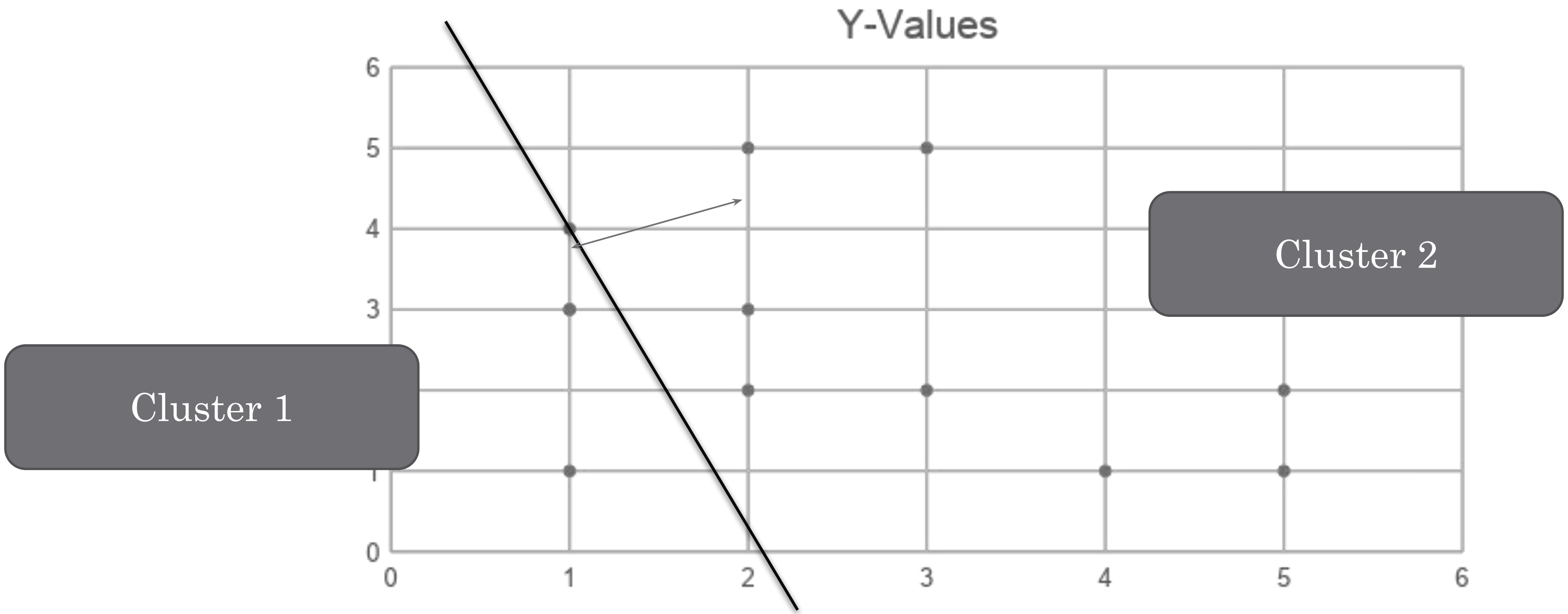
Agglomerative/Bottom-Up



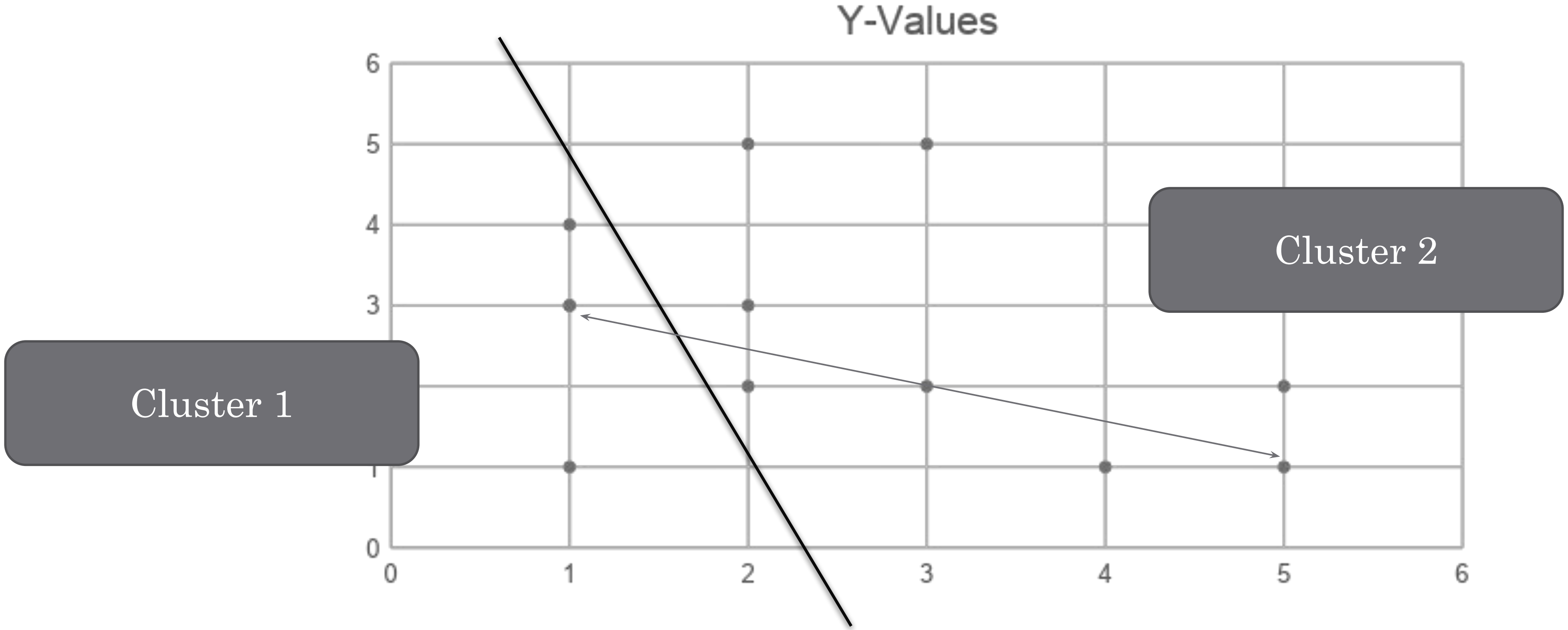
Agglomerative/Bottom-Up

- Single Link or Nearest neighbor
- Complete Link or Farthest neighbor
- Average Link or Centeroid distance

Single Link

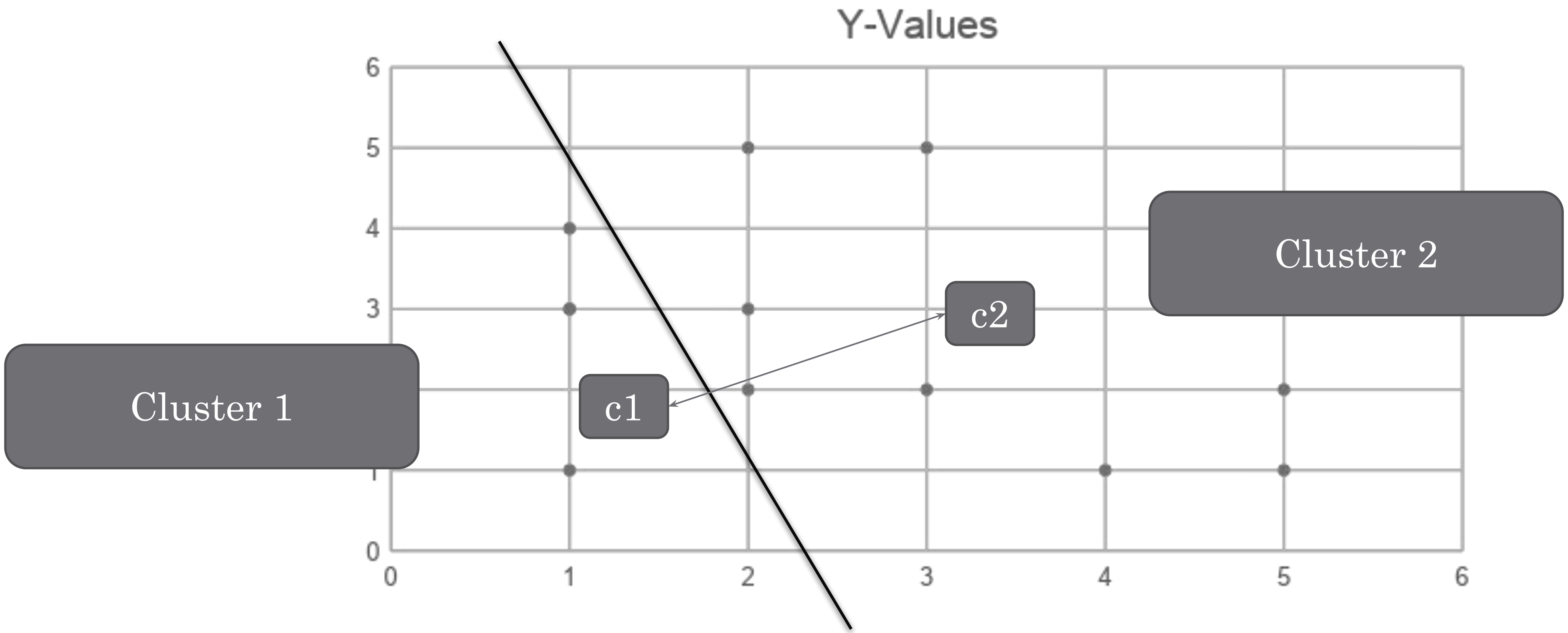


Complete Link



zeshan.khan@nu.edu.pk

Average Link

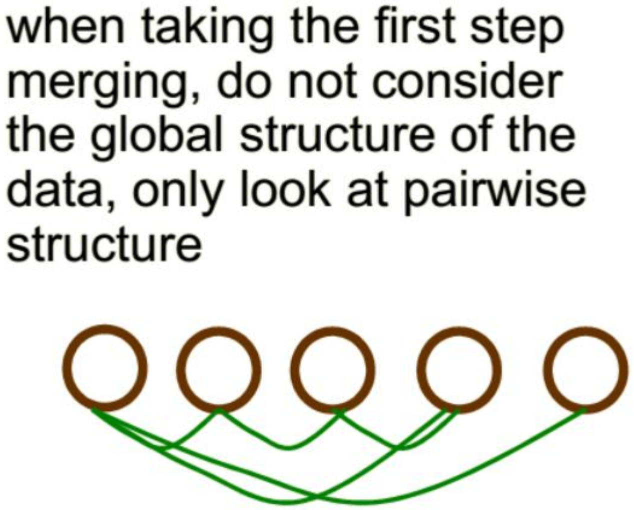


Divisive vs. Agglomerative

Divisive

Agglomerative

- Faster



Assignment

Element	X1	X2
1	1	2
2	5	2
3	3	5
4	2	4
5	8	3
6	5	7
7	2	1
8	6	7
9	2	6
10	4	3
11	4	5
12	5	4

- Apply k-means clustering for k=3 on the provided data the first three points as a seed elements are:
- seed1=Your_RollNumber%5
- seed2=(Your_RollNumber%5)+2
- seed3=(Your_RollNumber%5)+5

zeshan.khan@nu.edu.pk