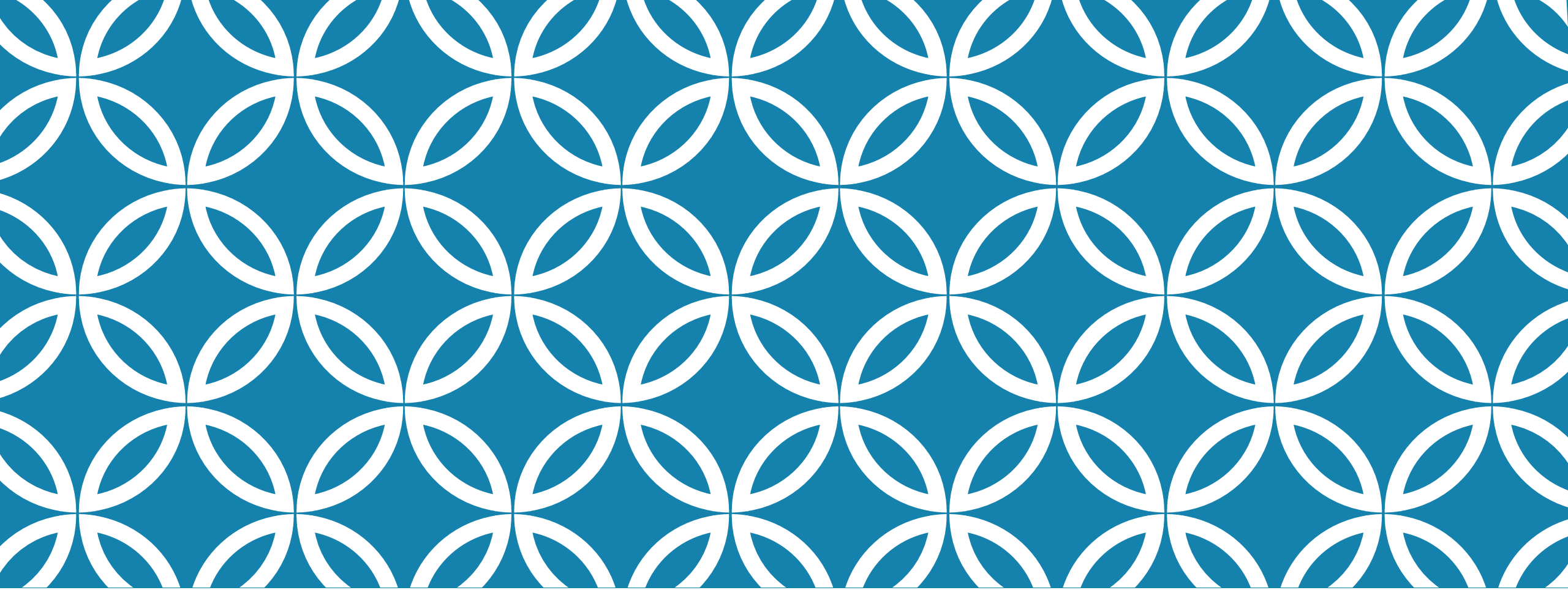


CS5228 LECTURE 3: CLUSTERING II

Bryan Hooi

School of Computing

National University of Singapore

Slide credits: Chris von der Weth, Wang Wei, Ng See Kiong, Wynne Hsu

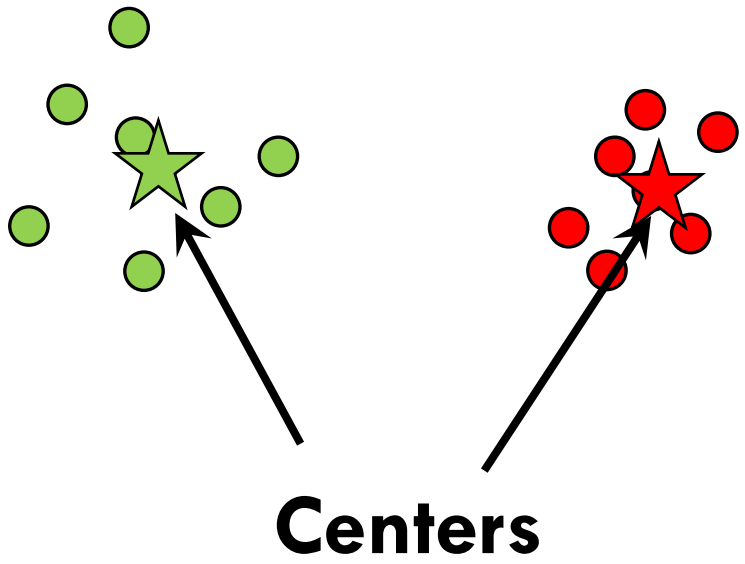
ANNOUNCEMENTS

- HW1 and Project will be released this weekend.
- Tutorial 2 held next week.

Week	Date	Topics	Tutorials	Important Dates
1	13 Jan	Introduction		
2	20 Jan	No class (public holiday)		
3	27 Jan	Clustering I	Tutorial 1	
4	3 Feb	Clustering II		Release A1 + project
5	10 Feb	Association Rules	Tutorial 2	
6	17 Feb	Regression & Classification I		
Recess		No class		
7	3 Mar	Regression & Classification II	Tutorial 3	A1 due (Sunday 11.59pm), release A2
8	10 Mar	Regression & Classification III		
9	17 Mar	Recommender Systems	Tutorial 4	
10	24 Mar	Graph Mining		
11	31 Mar	Data Stream Mining	Tutorial 5	A2 due (Sunday 11.59pm)
12	7 Apr	No class (public holiday)		
13	14 Apr	Review & Outlook		Project due (Sunday 11.59pm)

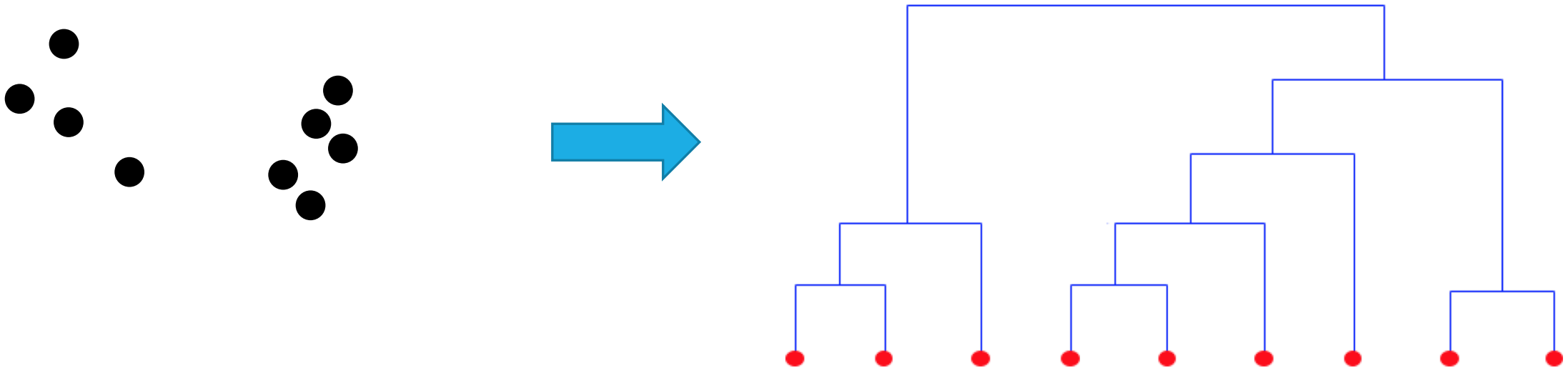
REVIEW: OVERVIEW OF CLUSTERING APPROACHES

- **Center-based:** each cluster is characterized by its center



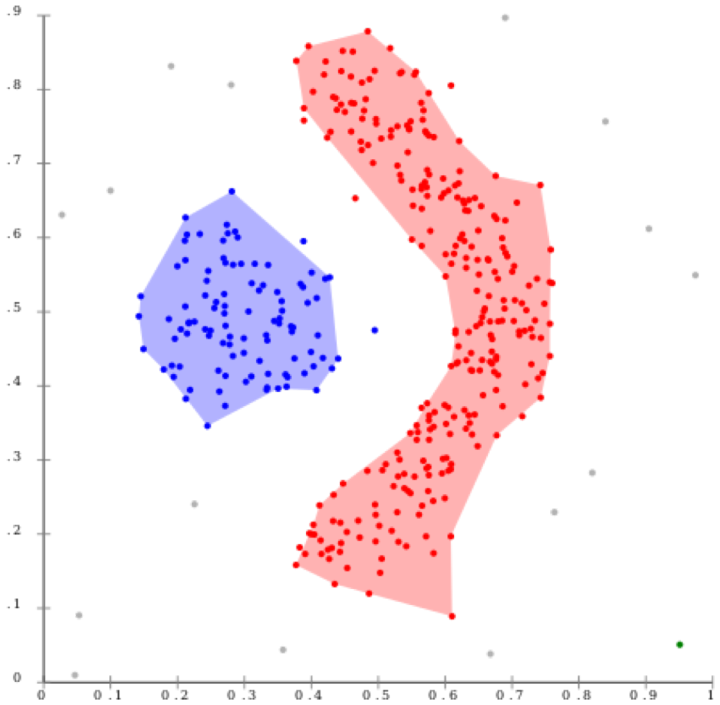
REVIEW: OVERVIEW OF CLUSTERING APPROACHES

- **Hierarchical:** points are organized according to a hierarchy (or tree structure)



REVIEW: OVERVIEW OF CLUSTERING APPROACHES

- **Density-based:** clusters are high-density regions surrounded by low-density regions



REVIEW: K-MEANS ALGORITHM: STEPS

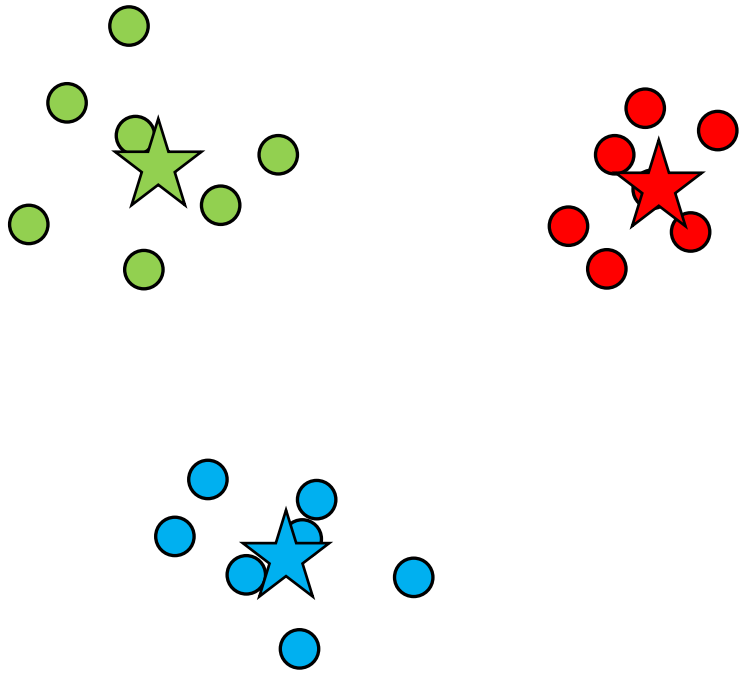
1. Initialization: Pick K random points as centers

2. Repeat:

a) Assignment: assign each point to nearest cluster

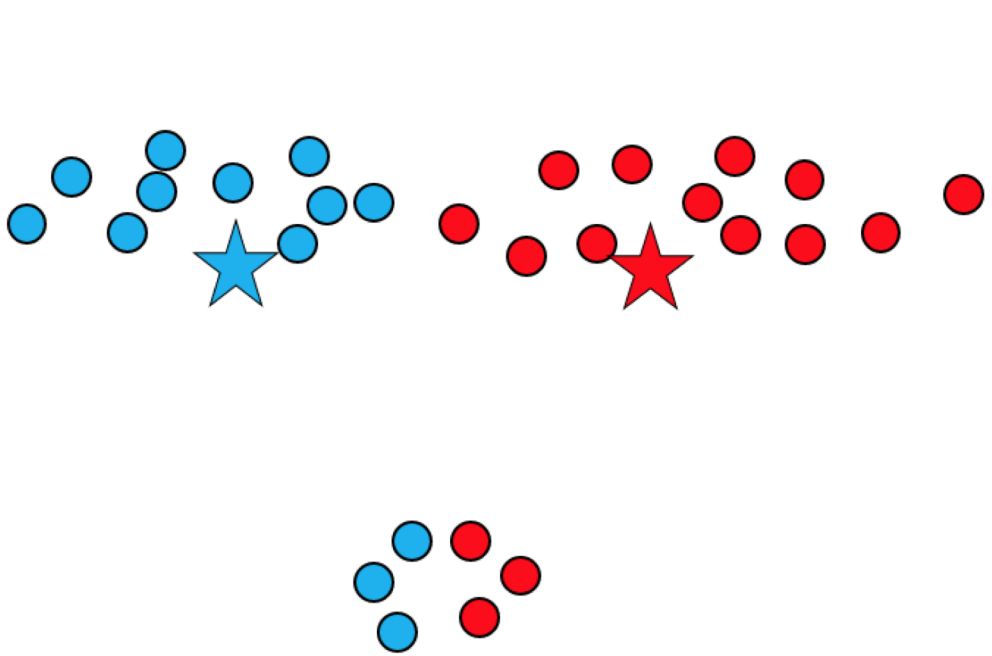
b) Update: move each cluster center to average of its assigned points

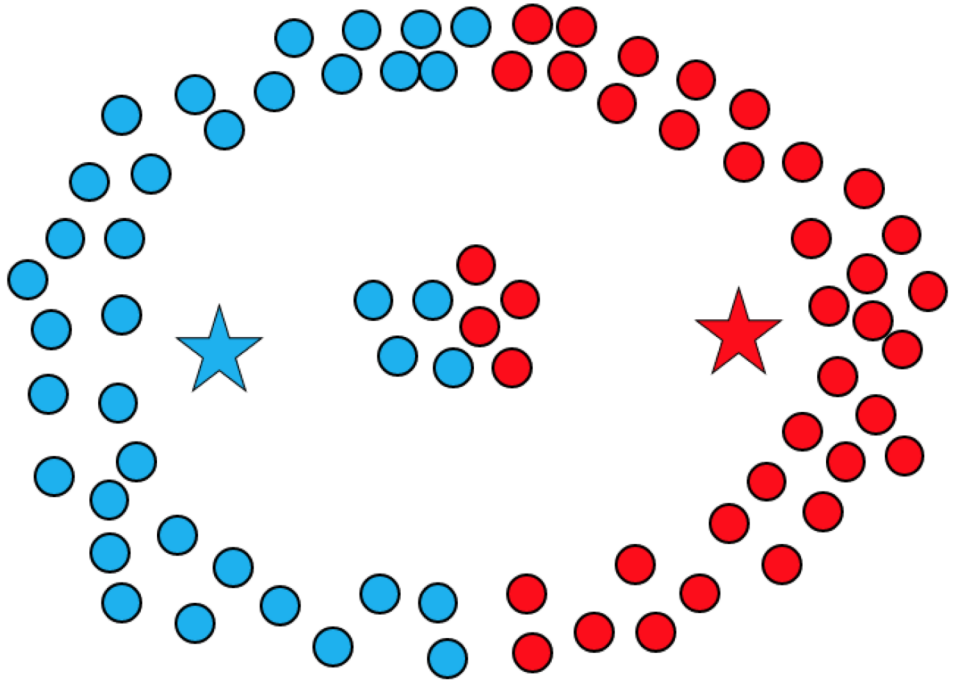
Stop if no assignments change



REVIEW: K-MEANS & NON-SPHERE-LIKE CLUSTERS

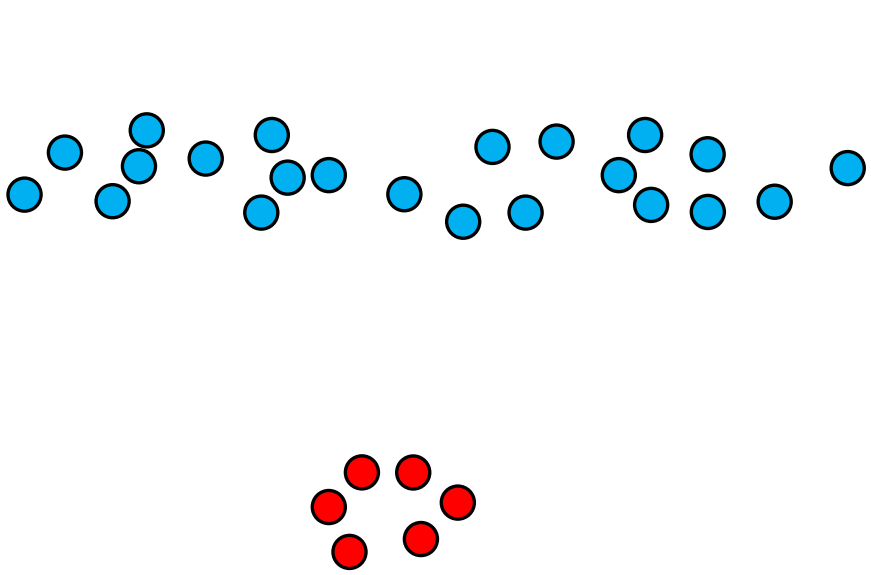
- K-means does not perform well for highly non-spherical clusters

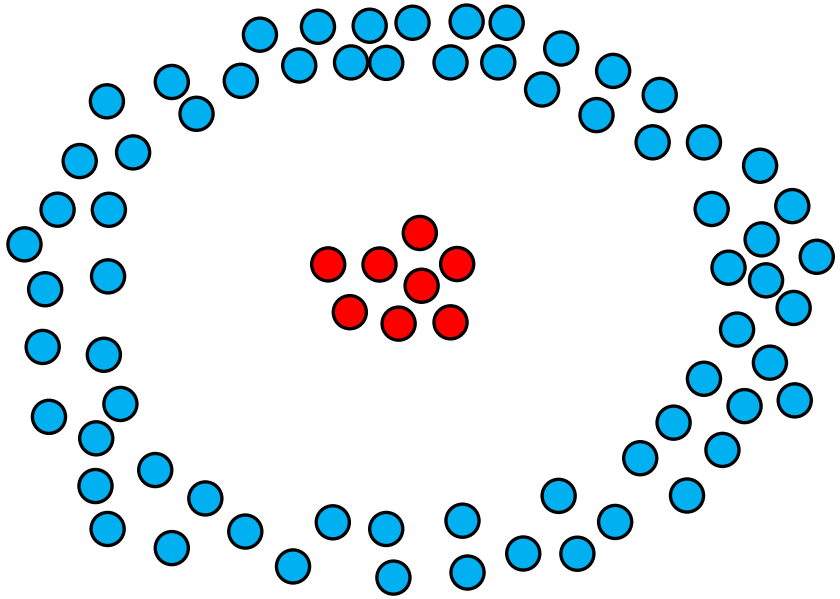




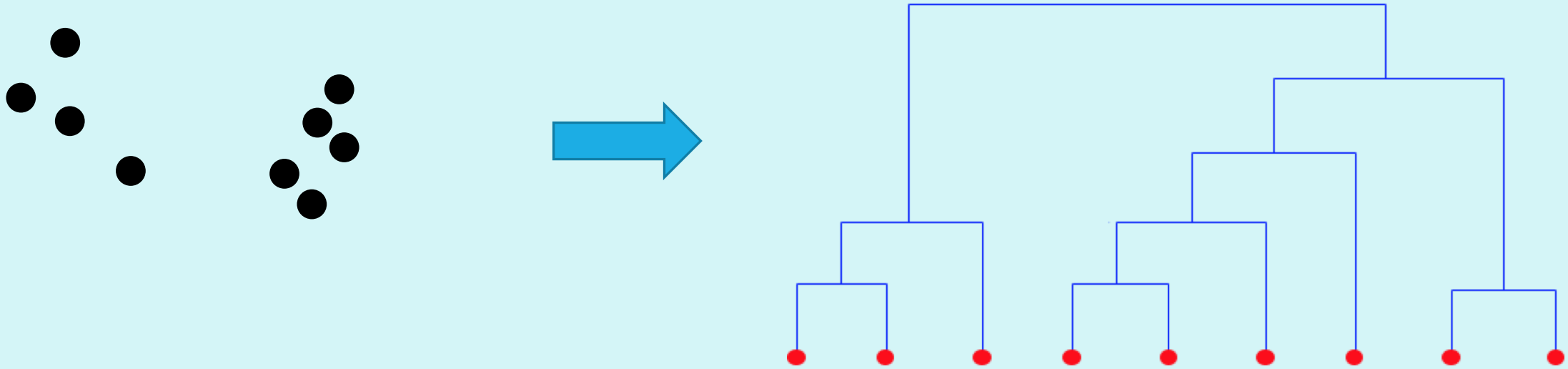
TODAY: HIERARCHICAL CLUSTERING

- Hierarchical clustering can handle unevenly shaped clusters!





- Does this mean we can give up on k-means?
  - No, hierarchical clustering has weaknesses which depend on which variant we choose; and also has higher computation time



HIERARCHICAL CLUSTERING

Bryan Hooi

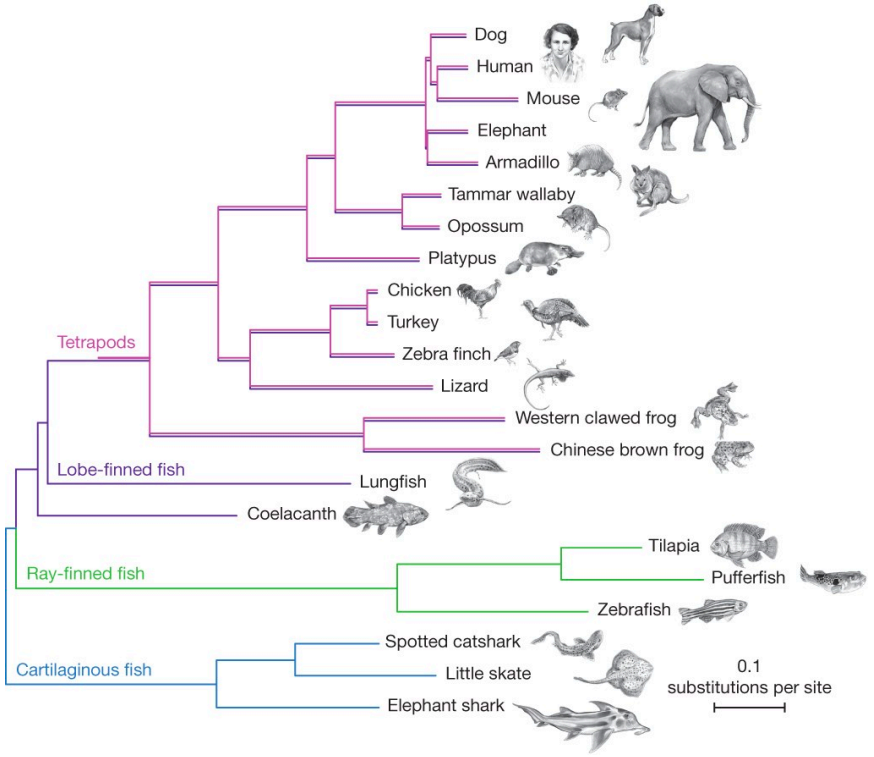
School of Computing

National University of Singapore

WHAT IS HIERARCHICAL CLUSTERING?

Constructing a **hierarchy** is a common and intuitive way to organize a set of objects:

Tree of Life



E-commerce product hierarchy

https://plato.stanford.edu/entries/phylogenetic-inference/

WHAT IS HIERARCHICAL CLUSTERING?

Clustering method that builds a **hierarchy of clusters**, i.e. multiple levels of clusters, where higher level clusters can contain lower level clusters. Example:

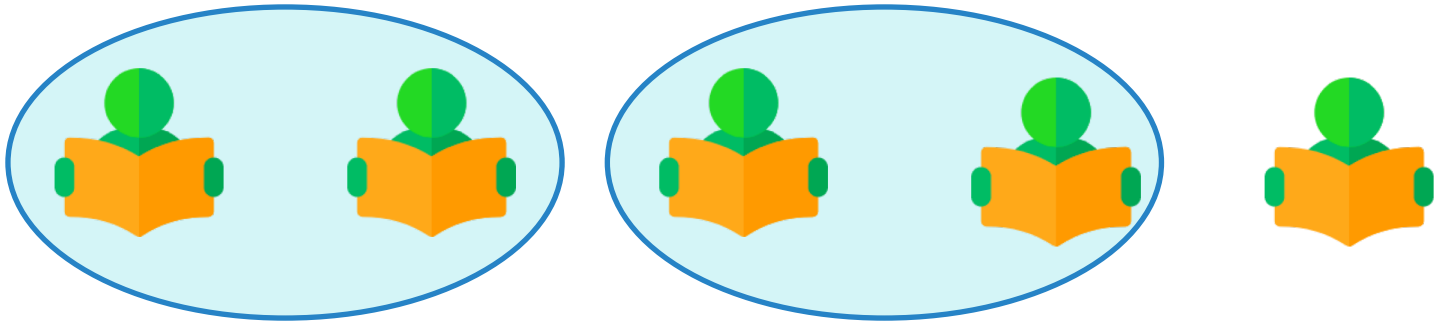
WHAT IS HIERARCHICAL CLUSTERING?

Clustering method that builds a **hierarchy of clusters**, i.e. multiple levels of clusters, where higher level clusters can contain lower level clusters. Example:

CS Majors

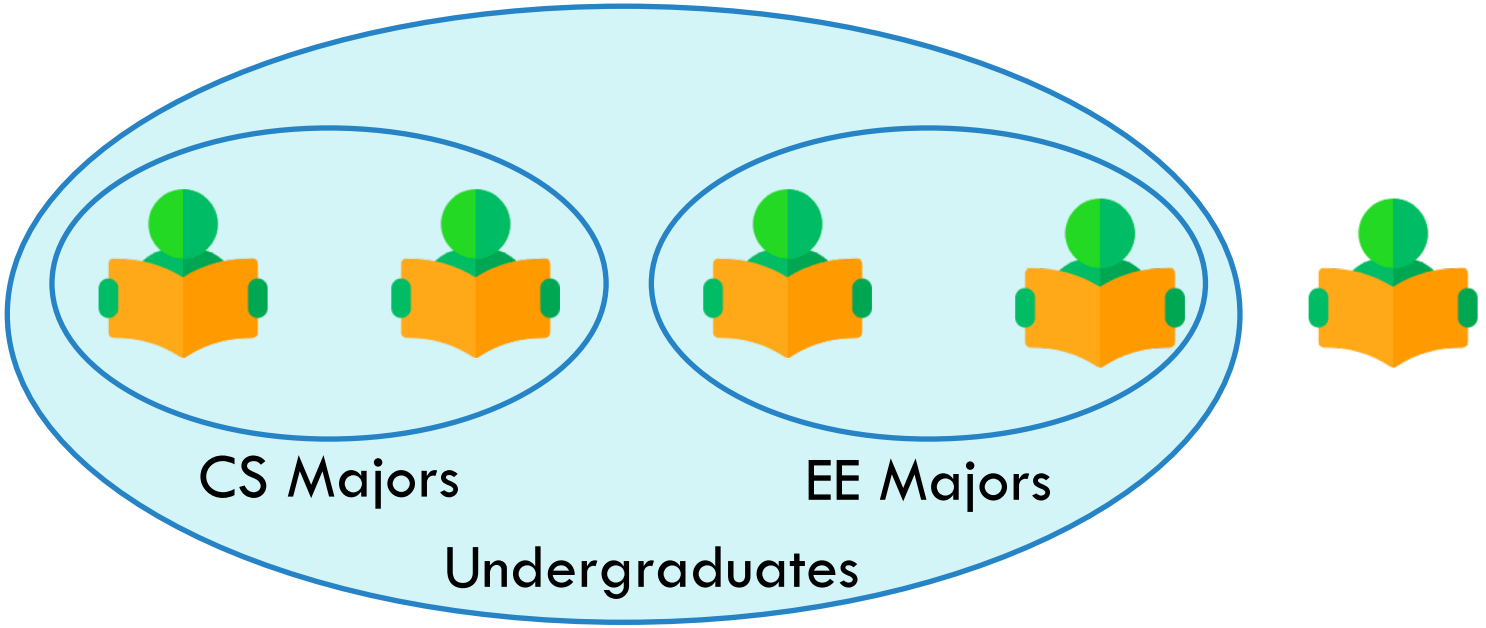
WHAT IS HIERARCHICAL CLUSTERING?

Clustering method that builds a **hierarchy of clusters**, i.e. multiple levels of clusters, where higher level clusters can contain lower level clusters. Example:



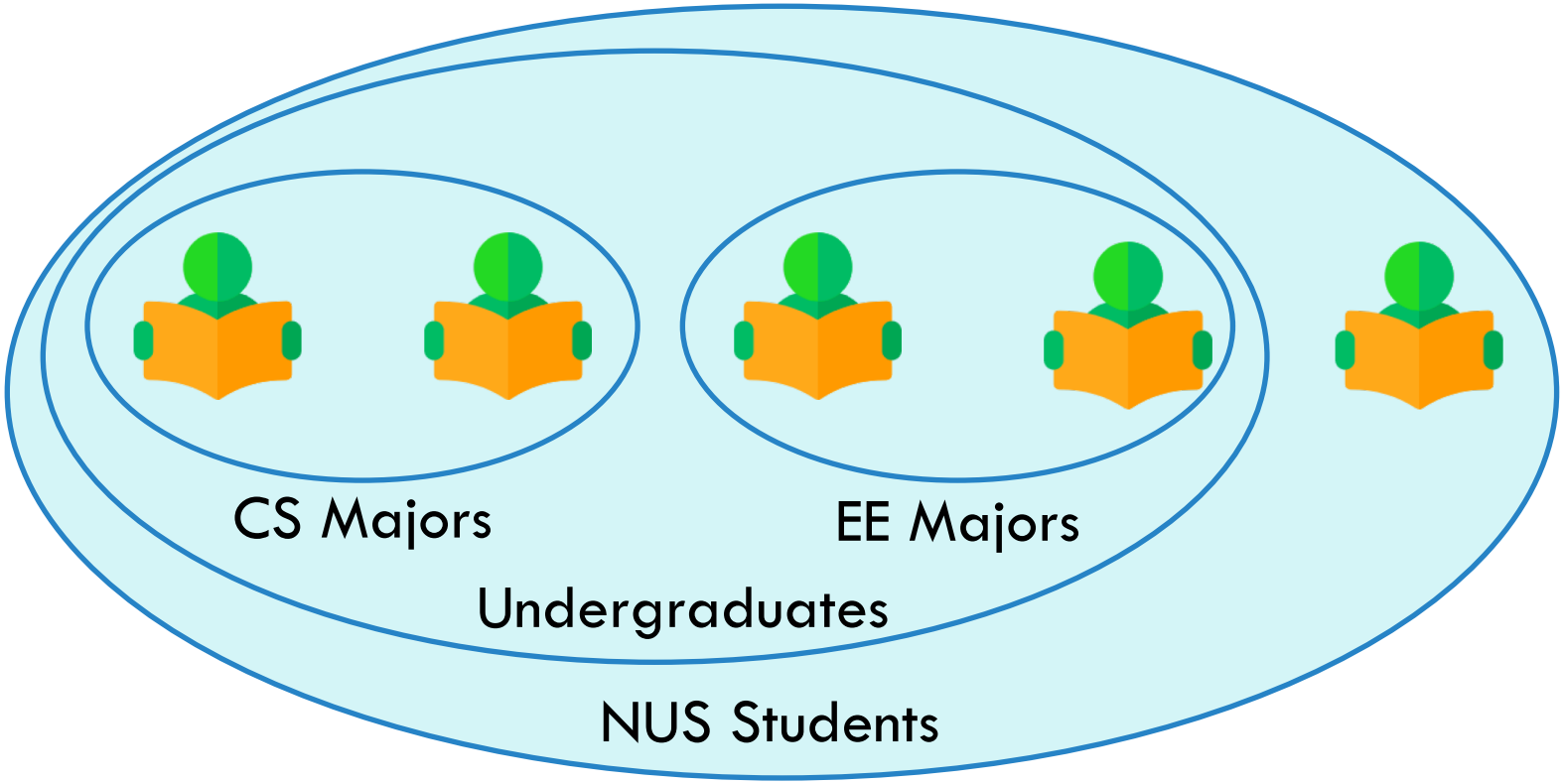
WHAT IS HIERARCHICAL CLUSTERING?

Clustering method that builds a **hierarchy of clusters**, i.e. multiple levels of clusters, where higher level clusters can contain lower level clusters. Example:



WHAT IS HIERARCHICAL CLUSTERING?

Clustering method that builds a **hierarchy of clusters**, i.e. multiple levels of clusters, where higher level clusters can contain lower level clusters. Example:



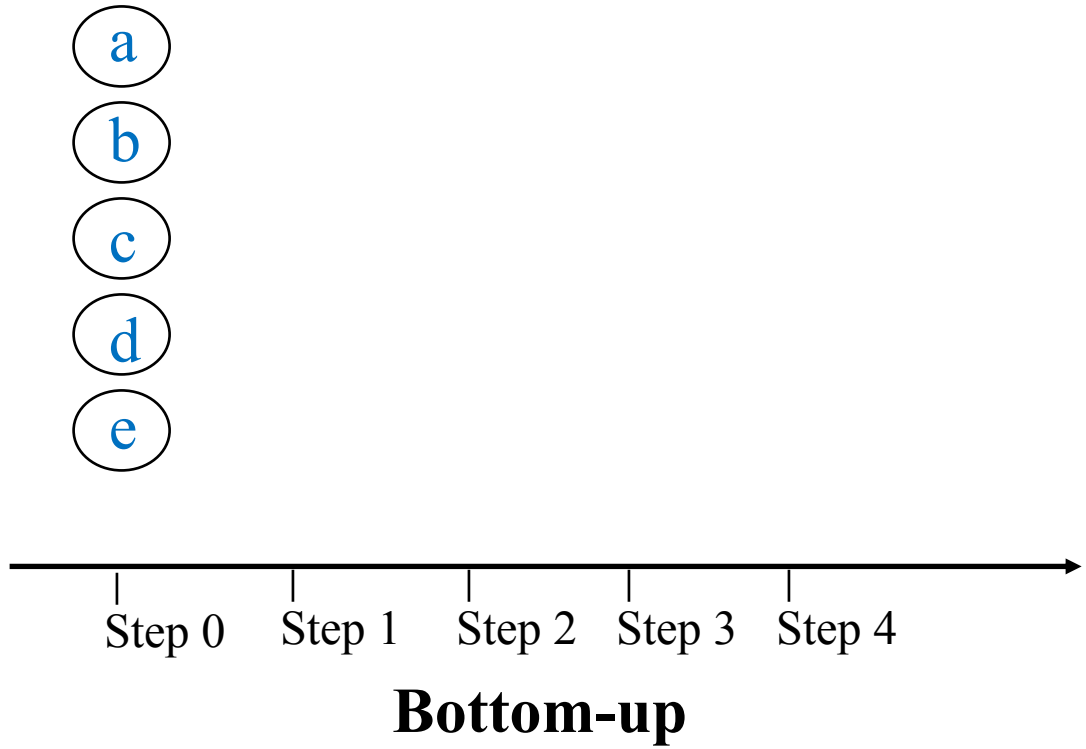
AGGLOMERATIVE (BOTTOM UP) APPROACH

1. Initialization: Each point starts out in its own cluster

2. Repeat:

a) **Merge:** combine the two most similar clusters into one

Until only a single cluster is left



AGGLOMERATIVE (BOTTOM UP) APPROACH

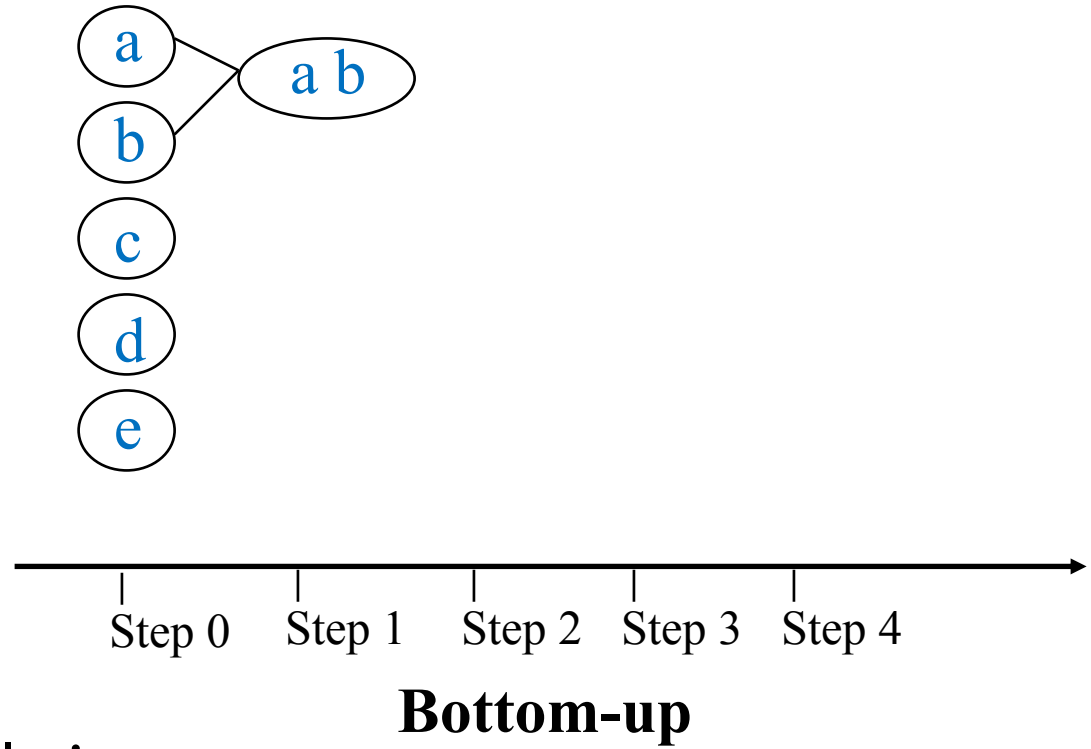
1. **Initialization**: Each point starts out in its own cluster

2. Repeat:

a) Merge: combine the two most similar clusters into one

Until only a single cluster is left

Assume we have a way to measure similarity between 2 clusters (explained later!)



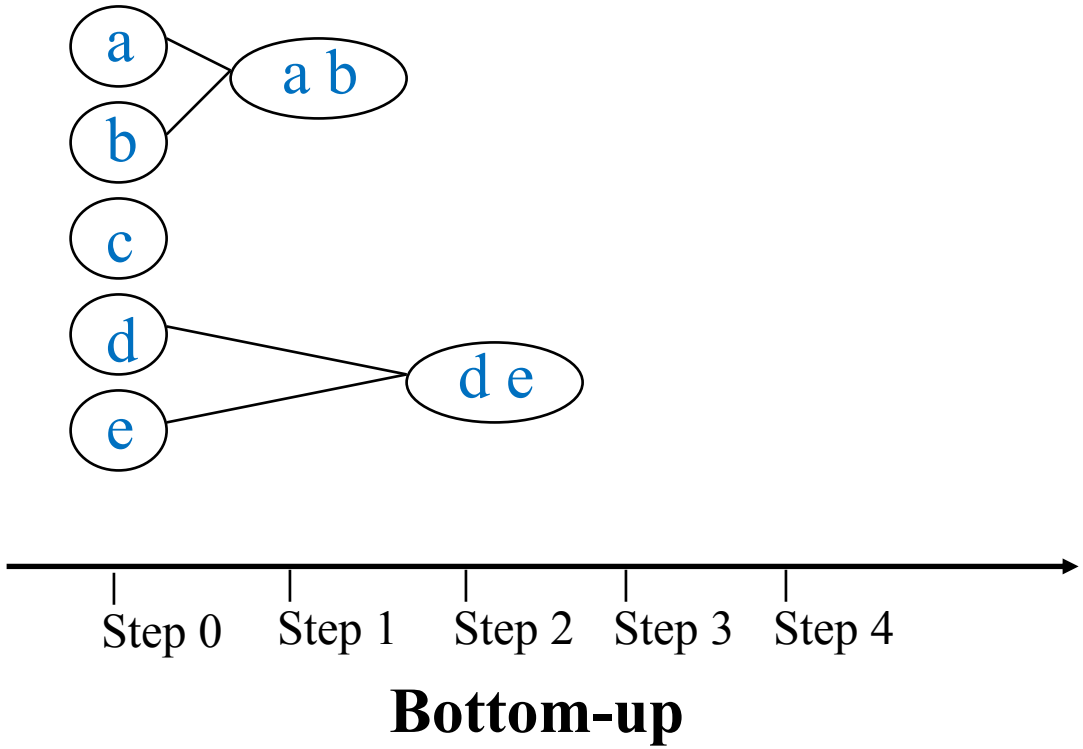
AGGLOMERATIVE (BOTTOM UP) APPROACH

1. **Initialization**: Each point starts out in its own cluster

2. Repeat:

a) Merge: combine the two most similar clusters into one

Until only a single cluster is left



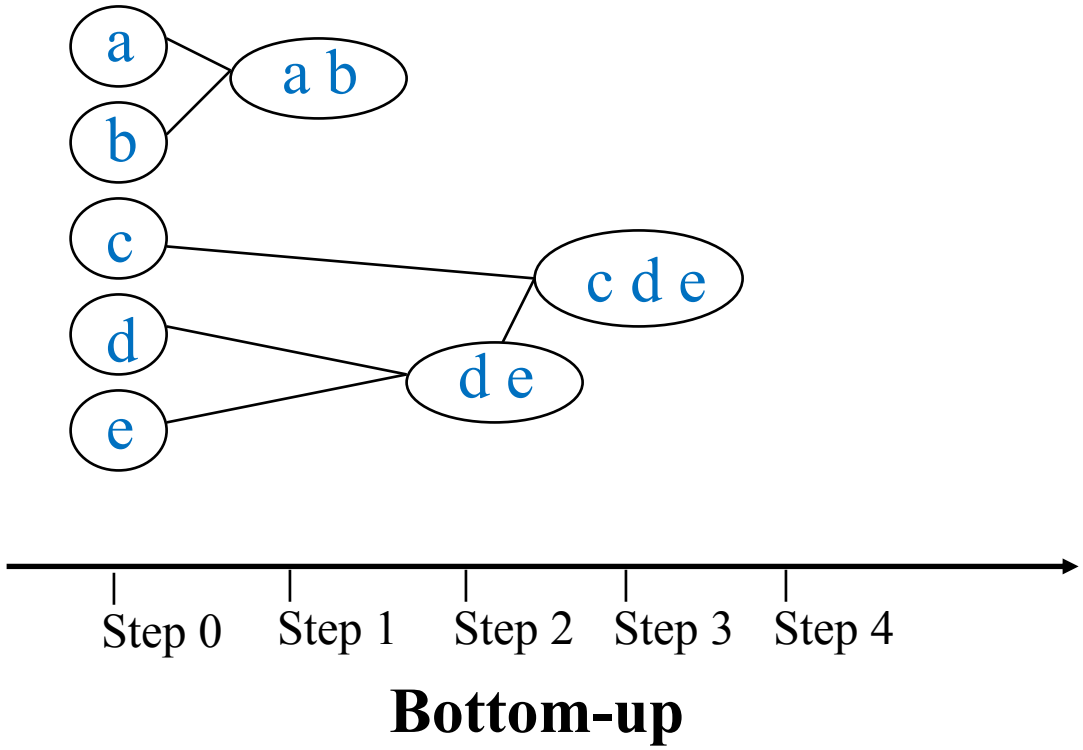
AGGLOMERATIVE (BOTTOM UP) APPROACH

1. **Initialization**: Each point starts out in its own cluster

2. Repeat:

a) Merge: combine the two most similar clusters into one

Until only a single cluster is left



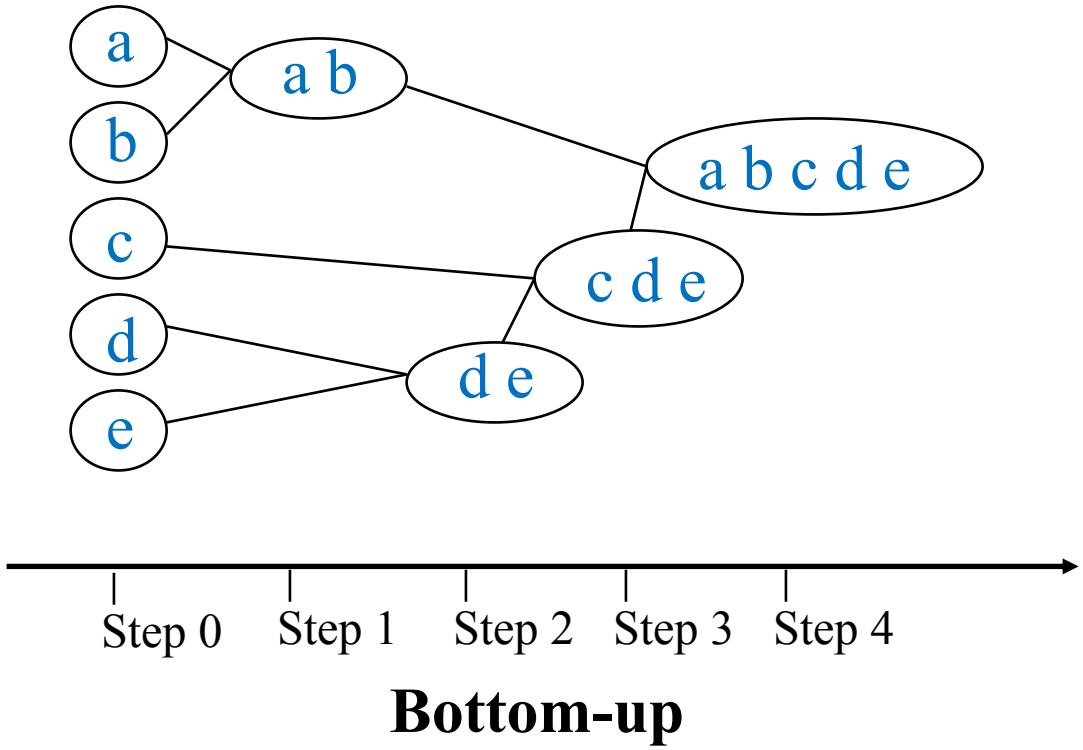
AGGLOMERATIVE (BOTTOM UP) APPROACH

1. **Initialization**: Each point starts out in its own cluster

2. Repeat:

a) Merge: combine the two most similar clusters into one

Until only a single cluster is left



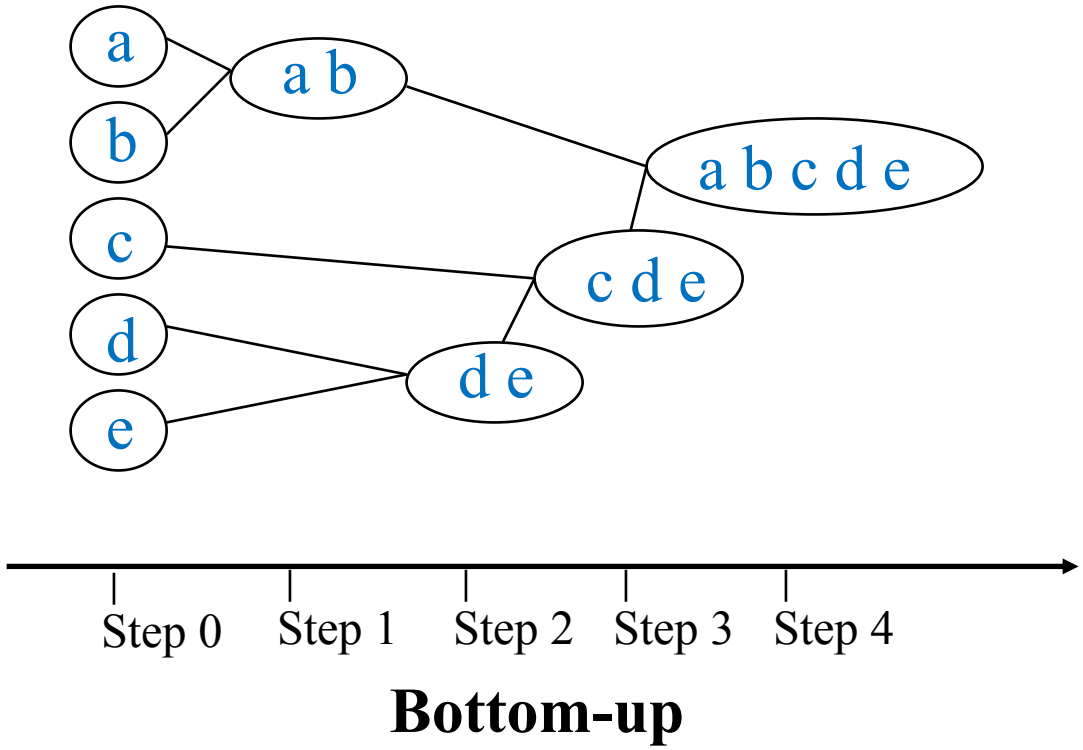
AGGLOMERATIVE (BOTTOM UP) APPROACH

1. **Initialization**: Each point starts out in its own cluster

2. Repeat:

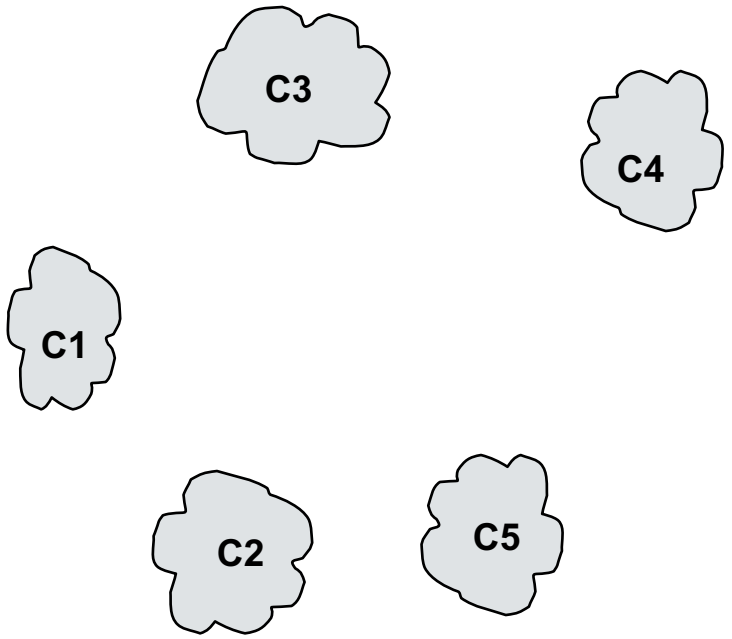
a) **Merge:** combine the two most similar clusters into one

Until only a single cluster is left



IMPLEMENTATION USING CLUSTER DISTANCE MATRIX

What is the cluster distance matrix? It keeps track of the distances between every pair of clusters as the algorithm runs

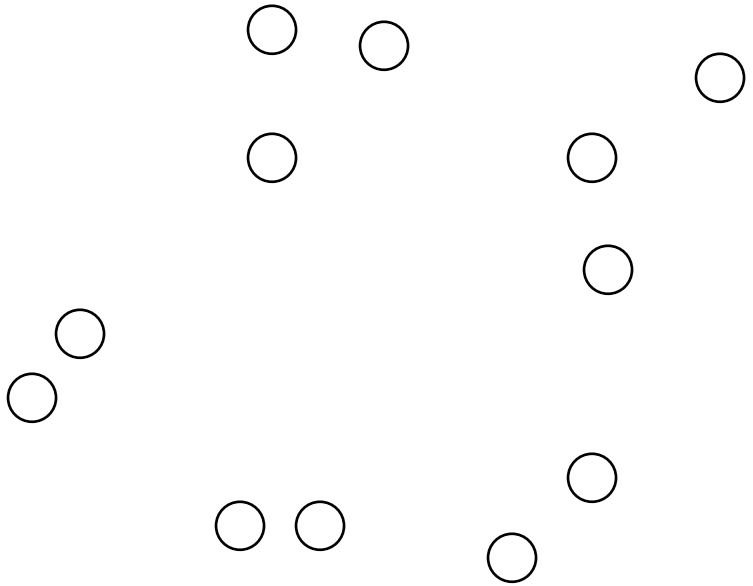


	C1	C2	C3	C4	C5
C1					
C2	0.5				
C3	0.4	0.9			
C4	0.9	0.9	0.6		
C5	0.4	0.1	0.8	0.7	

Distance Matrix

IMPLEMENTATION USING CLUSTER DISTANCE MATRIX

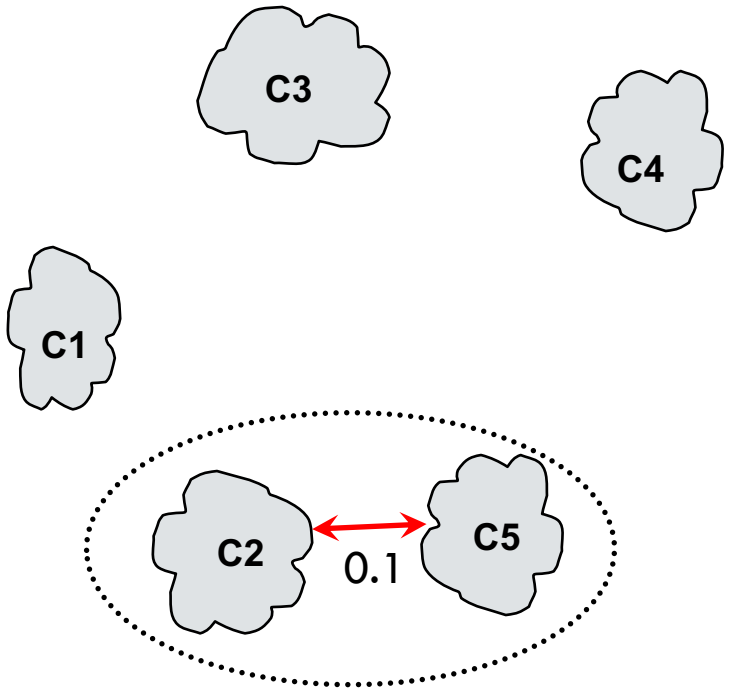
How to initialize it? Initially, our clusters are individual points. Compute distances between pairs of points, e.g. using Euclidean distance.



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

IMPLEMENTATION USING CLUSTER DISTANCE MATRIX

Update step: we have some clusters and a distance matrix between them. Find the closest pair of clusters (by finding the smallest entry in the distance matrix)



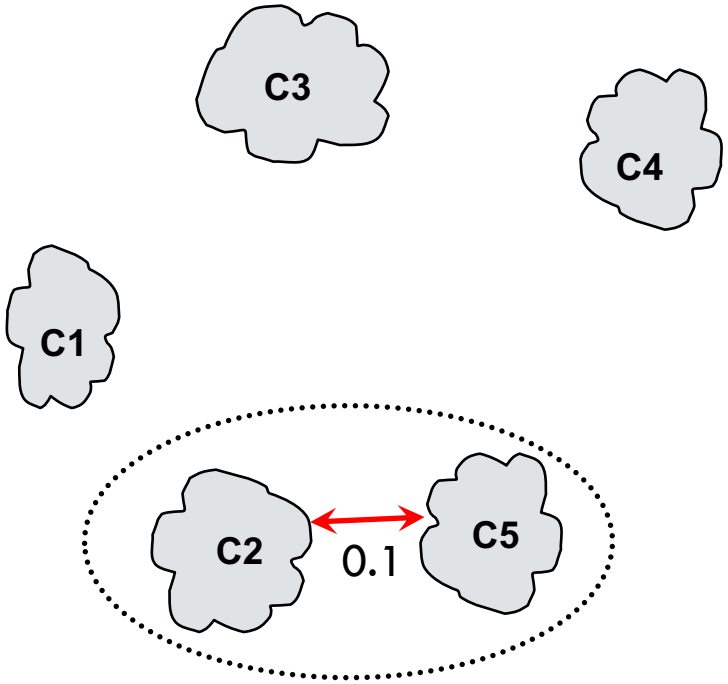
(0.1 is the minimum dist between clusters)

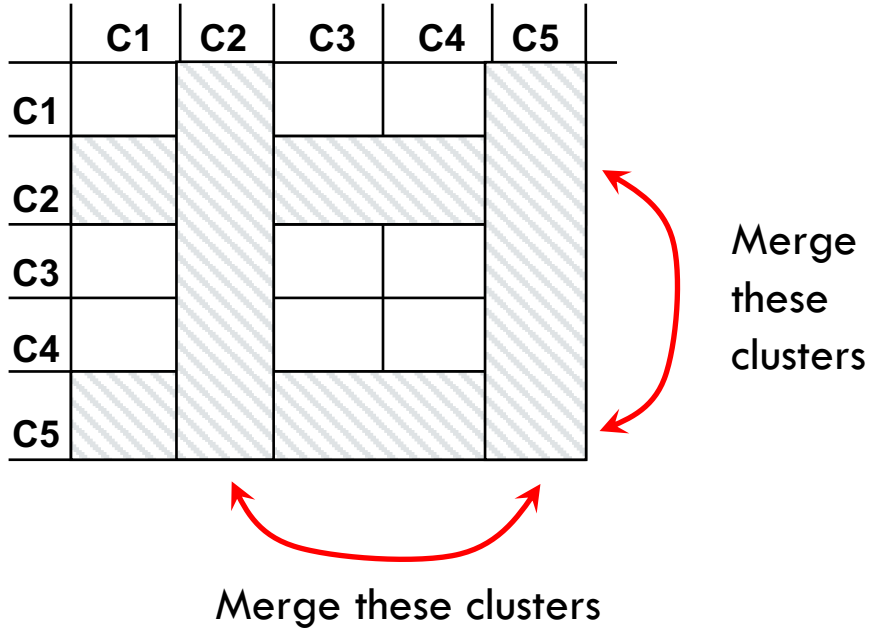
	C1	C2	C3	C4	C5
C1					
C2	0.5				
C3	0.4	0.9			
C4	0.9	0.9	0.6		
C5	0.4	0.1	0.8	0.7	

Distance Matrix

IMPLEMENTATION USING CLUSTER DISTANCE MATRIX

Update step: merge the closest 2 clusters

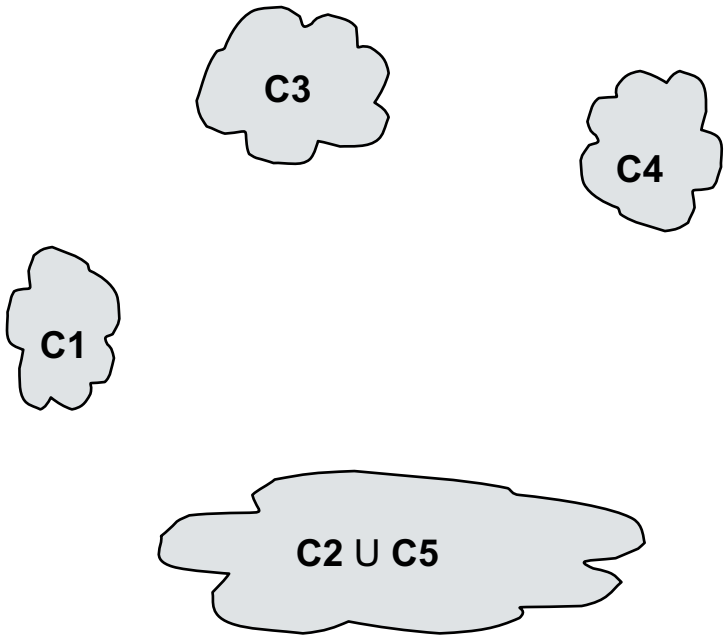




IMPLEMENTATION USING CLUSTER DISTANCE MATRIX

Update step: merge the closest 2 clusters

How we update the distance matrix depends on how we measure distance between clusters

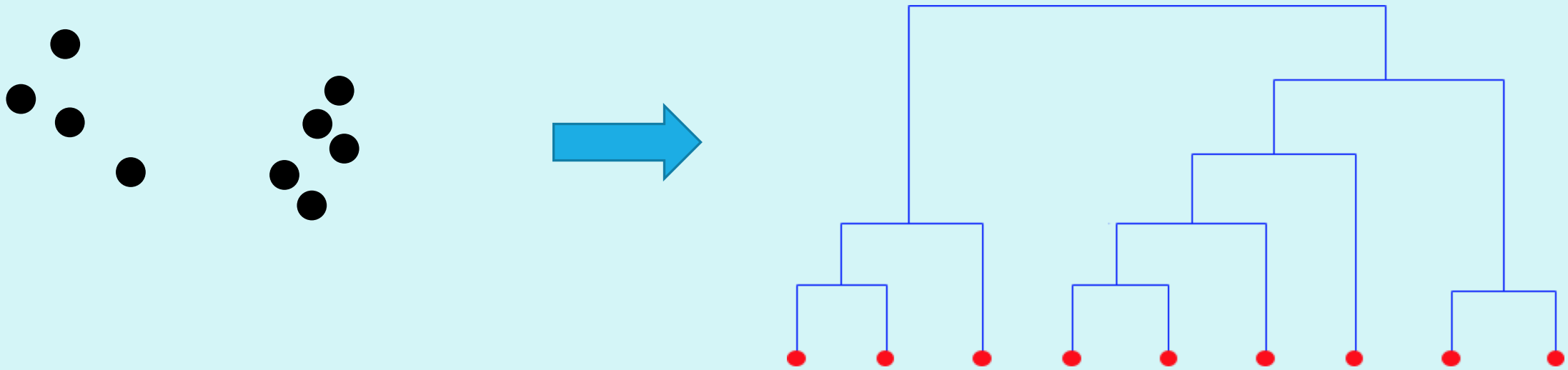


Merged cluster

C2 U C5

	C1		C3	C4
C1		?		
Merged cluster C2 U C5	?	?	?	?
C3		?		
C4		?		

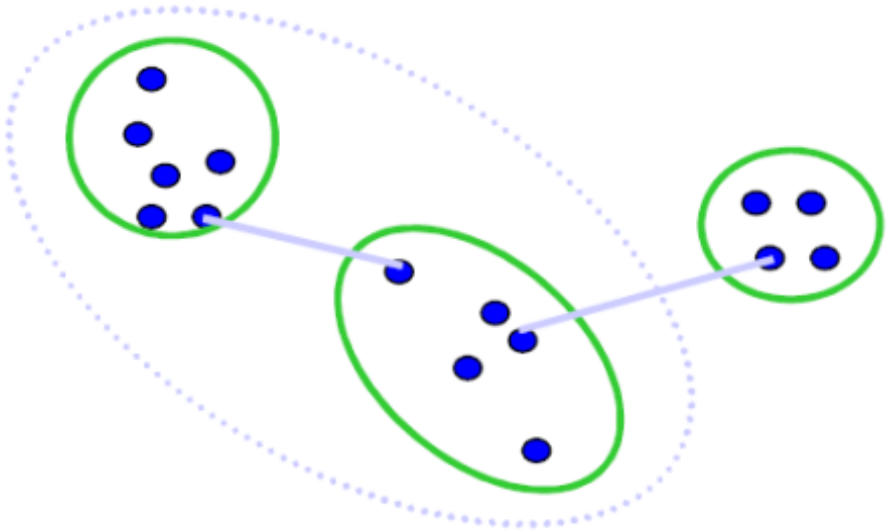
Distance Matrix



HIERARCHICAL CLUSTERING DISTANCE MEASURES (SINGLE, COMPLETE, AVERAGE LINKAGE)

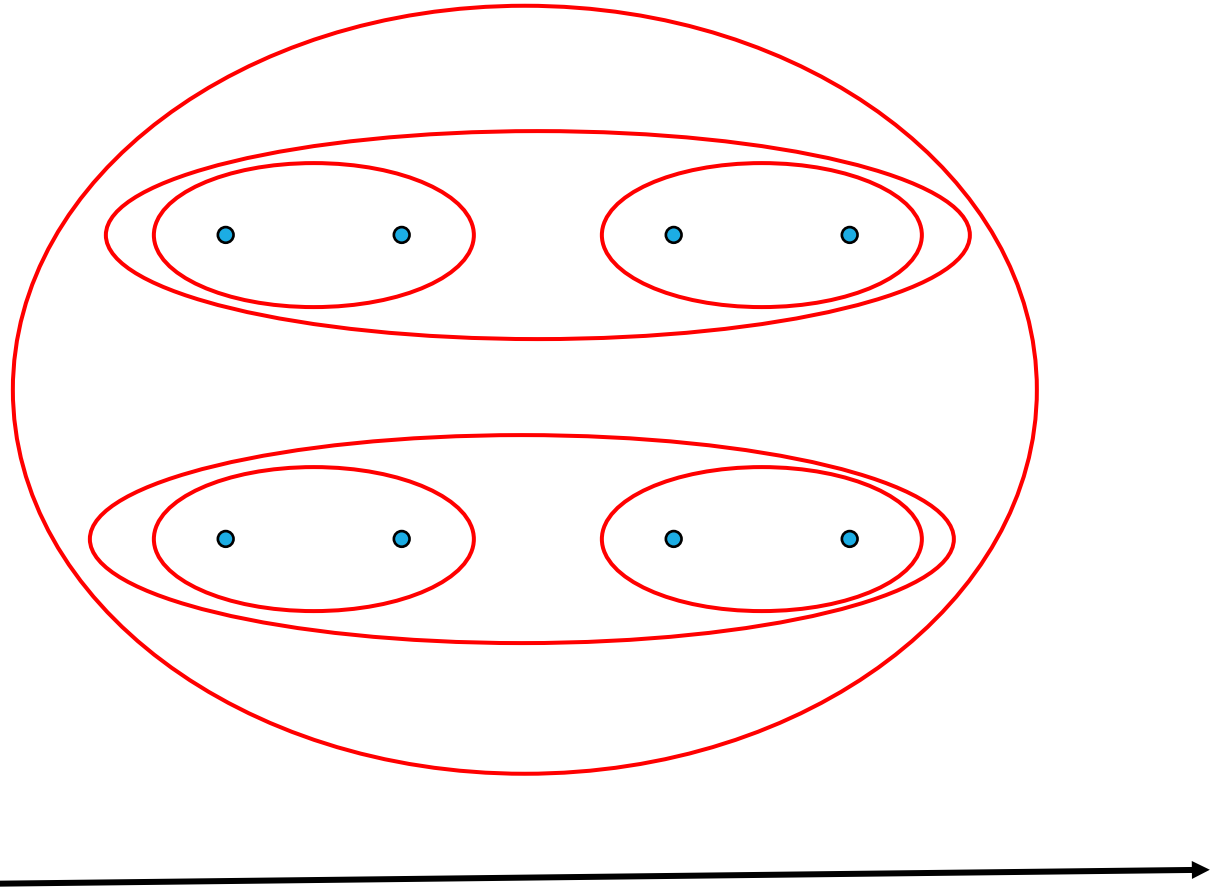
SINGLE LINKAGE CLUSTERING

Distance between clusters is defined as the **minimum** distance between two points, one from each cluster:



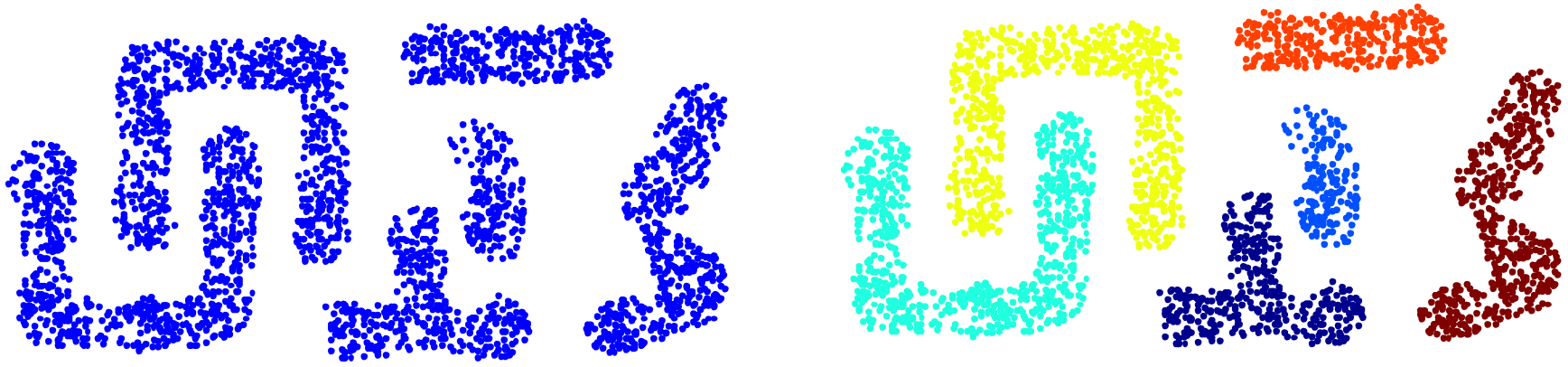
$$d_{\min}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p, q)$$

SINGLE LINKAGE EXAMPLE



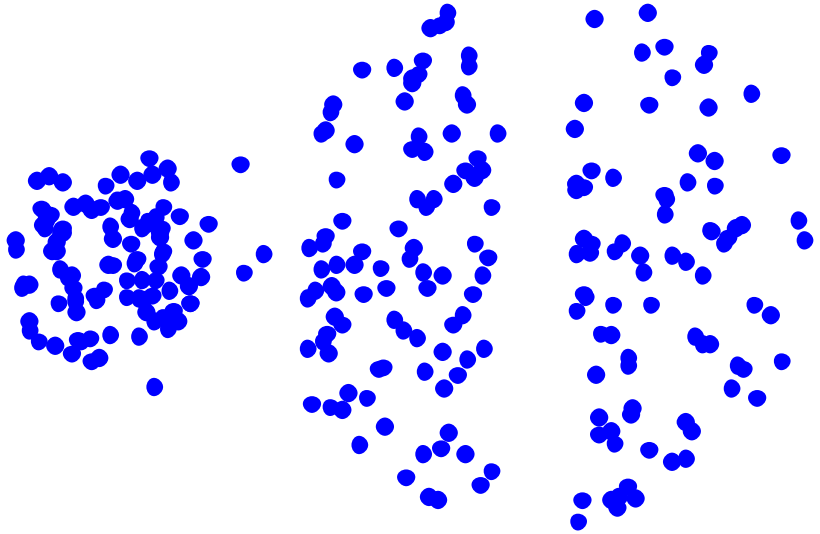
STRENGTH OF SINGLE LINKAGE

Can handle non-elliptical shapes!

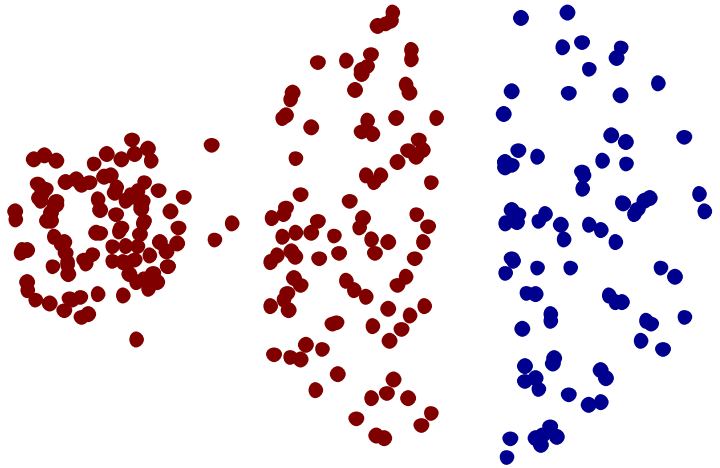


WEAKNESS OF SINGLE LINKAGE

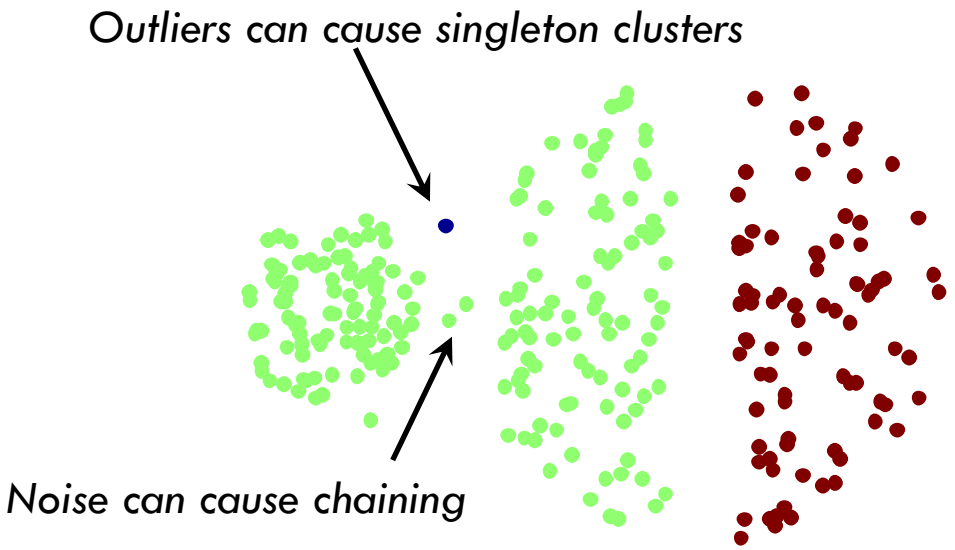
"**Chaining**": different clusters can get merged just due to a few points connecting them. Sensitive to noise / outliers.



Original Points



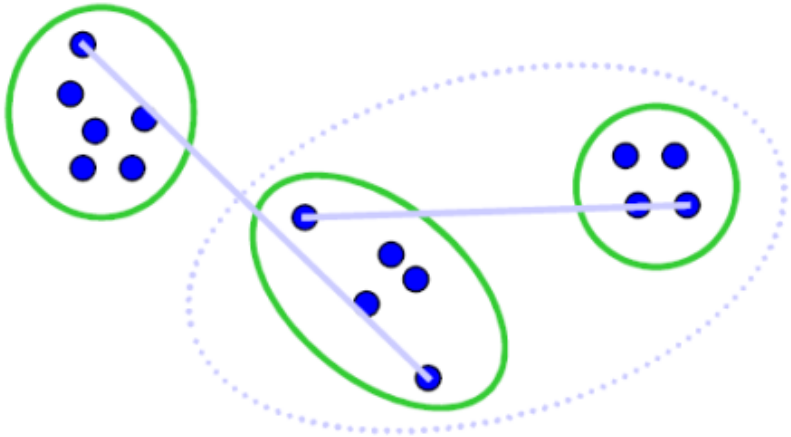
Two Clusters



Three Clusters

COMPLETE LINKAGE CLUSTERING

Distance between clusters is defined as the **maximum** distance between two points, one from each cluster:

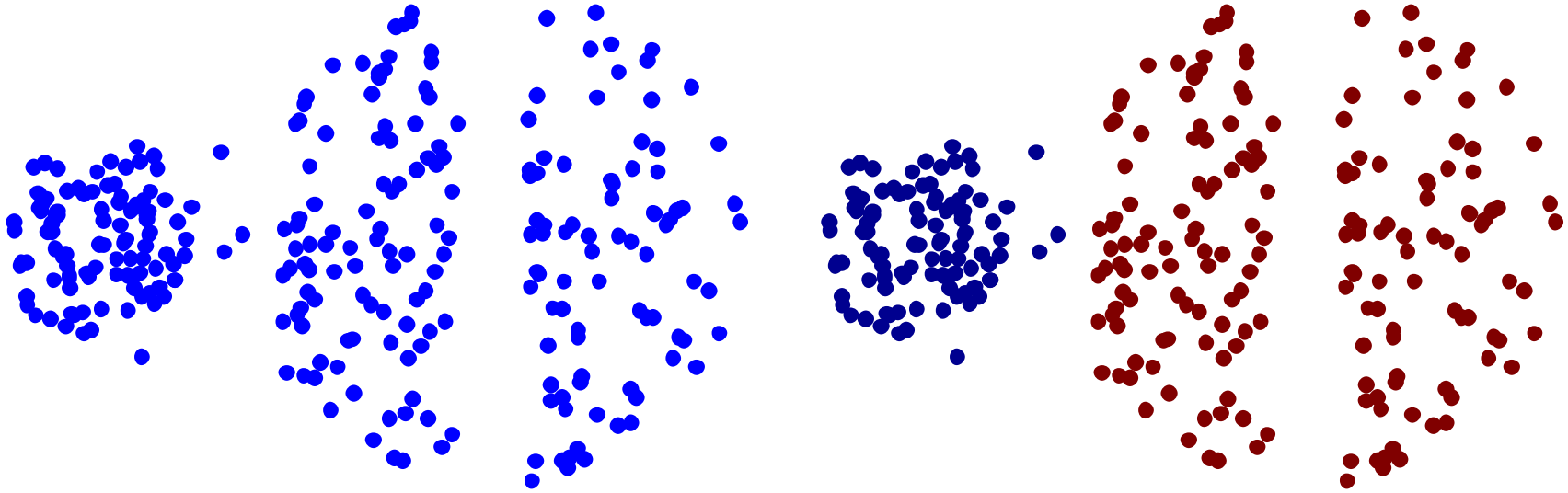


$$d_{\max}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)$$

Leads to “tight” clusters: two clusters are only merged if **all** their members are (relatively) similar

STRENGTH OF COMPLETE LINKAGE

Less susceptible to noise / outliers

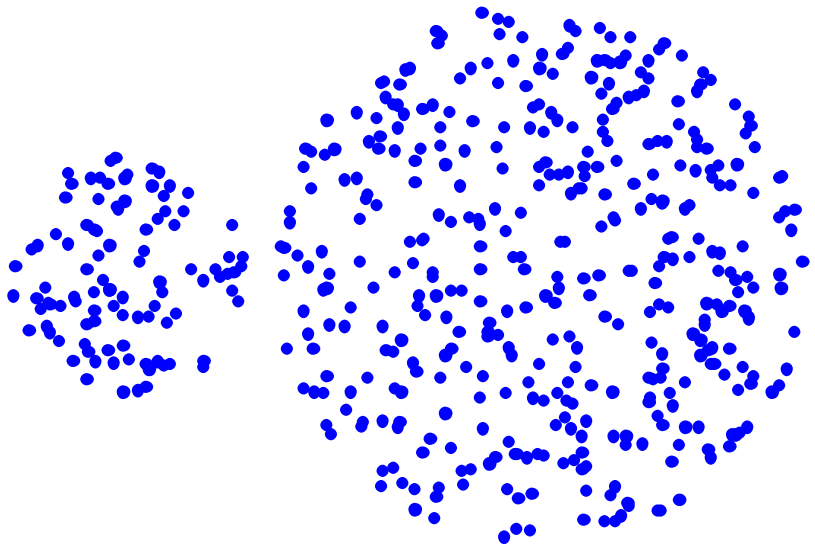


Original Points **Two Clusters**

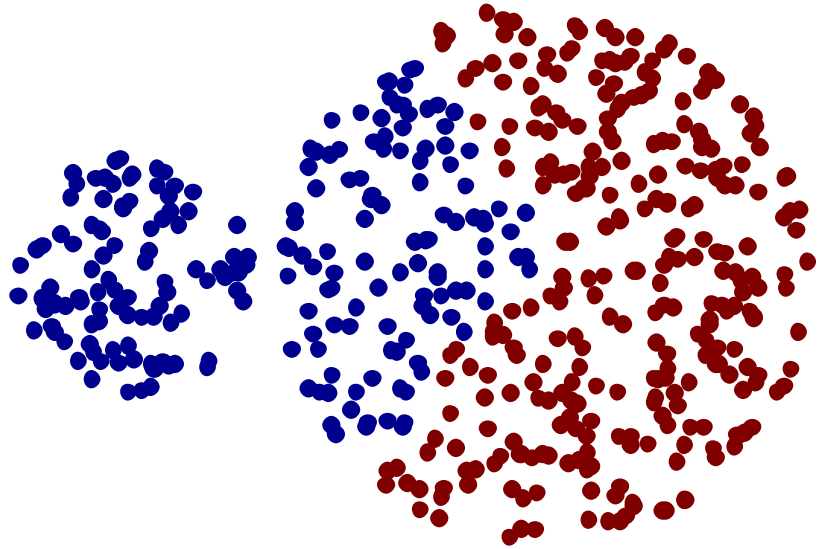
WEAKNESS OF COMPLETE LINKAGE

Tends to break large clusters

Biased towards globular clusters



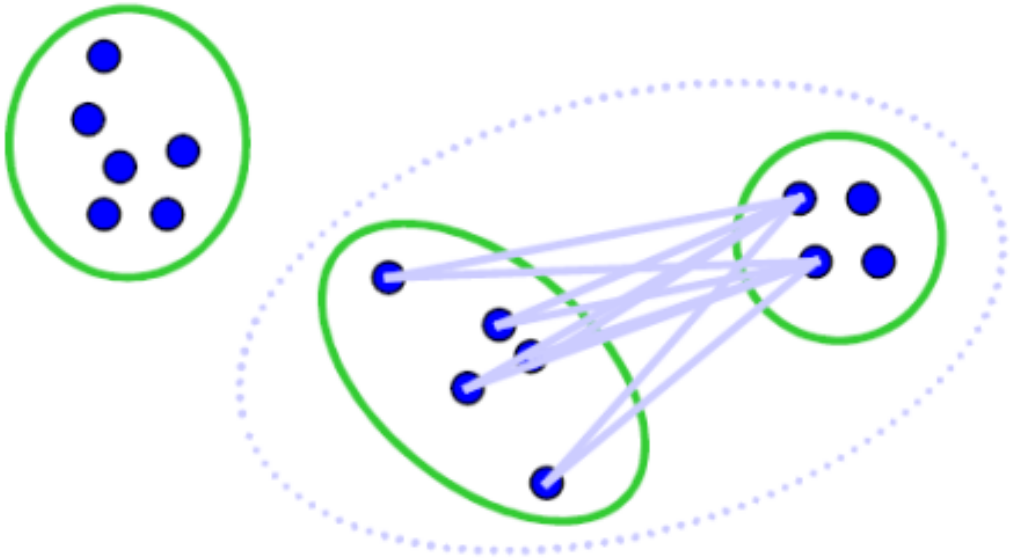
Original Points



Two Clusters

AVERAGE LINKAGE CLUSTER SIMILARITY

Distance between clusters is defined as the **average** distance between two points, one from each cluster:



$$d_{\text{avg}}(C_i, C_j) = \underset{p \in C_i, q \in C_j}{avg} \; d(p,q)$$

'Compromise' between single and complete linkage

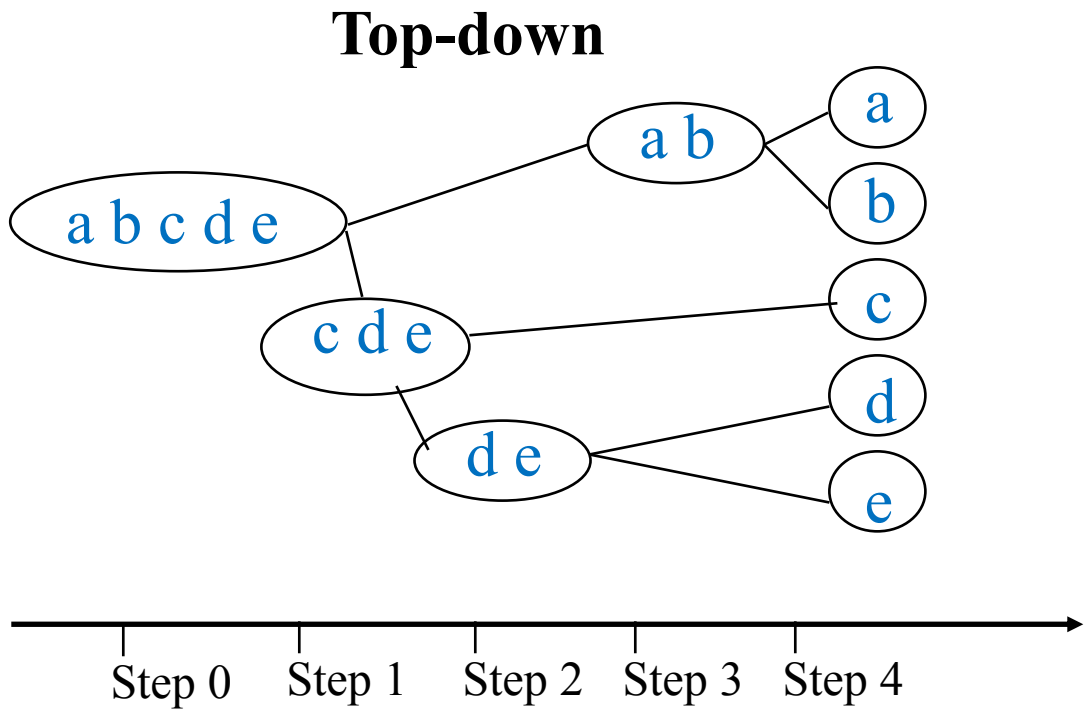
Robust against noise / outliers, but biased towards globular clusters

Optional

DIVISIVE (TOP-DOWN) HIERARCHICAL CLUSTERING

1. Start with all objects in the same cluster
2. Recursively split one of the clusters, until all clusters have size 1

- This requires heuristics for splitting clusters, e.g. DIANA (DIvisive ANAlysis)
- We won't go into the heuristics used, as they are not as commonly used, and quite complicated / slow

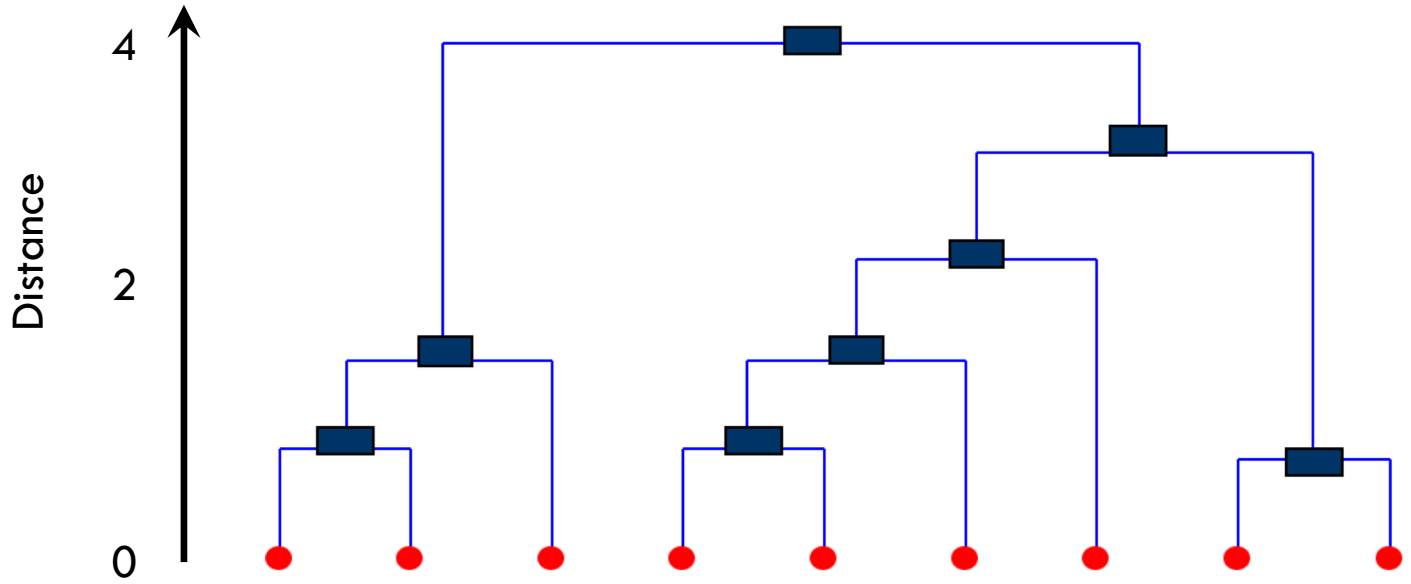


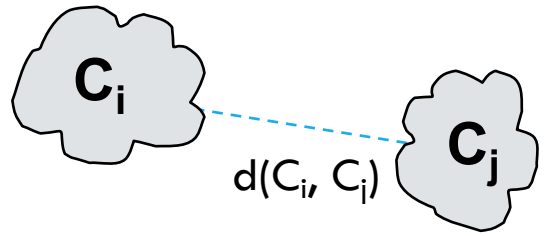
DENDROGRAMS

A binary tree that shows how clusters are merged/split hierarchically.

Each node on the tree is a cluster; each leaf node is a singleton cluster.

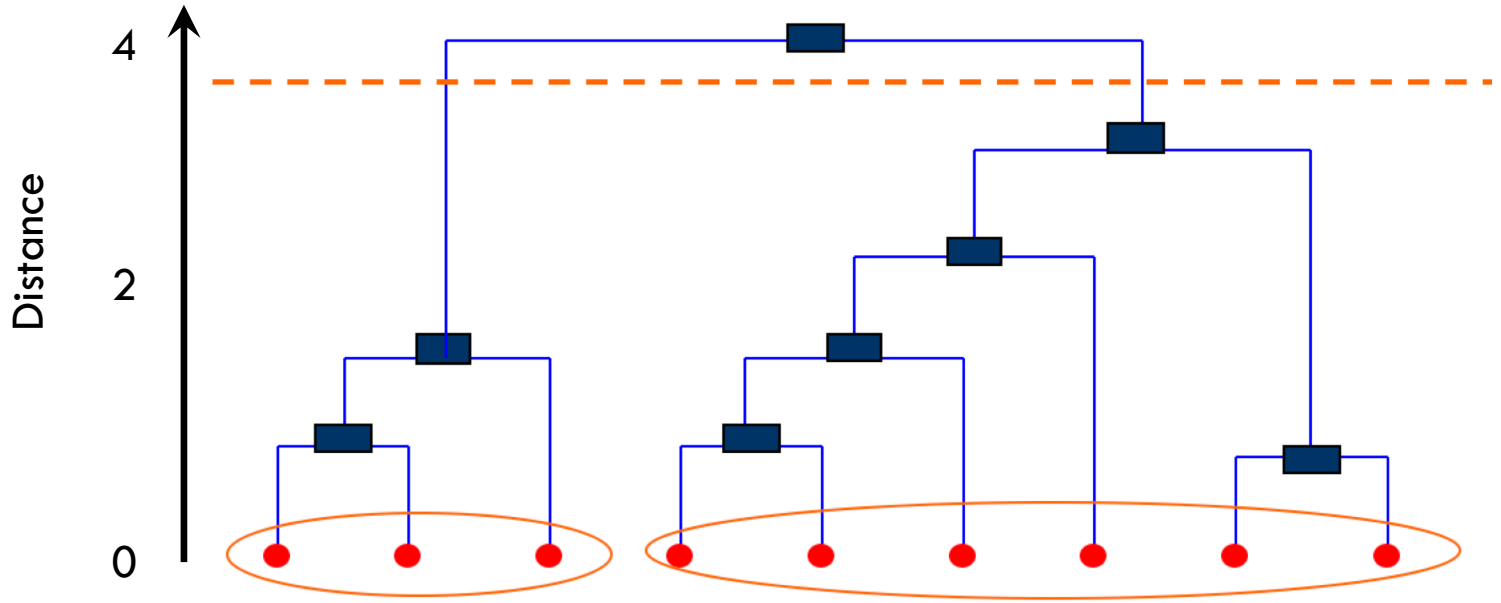
Y-axis indicates the distance level $d(C_i, C_j)$ at which each merger took place





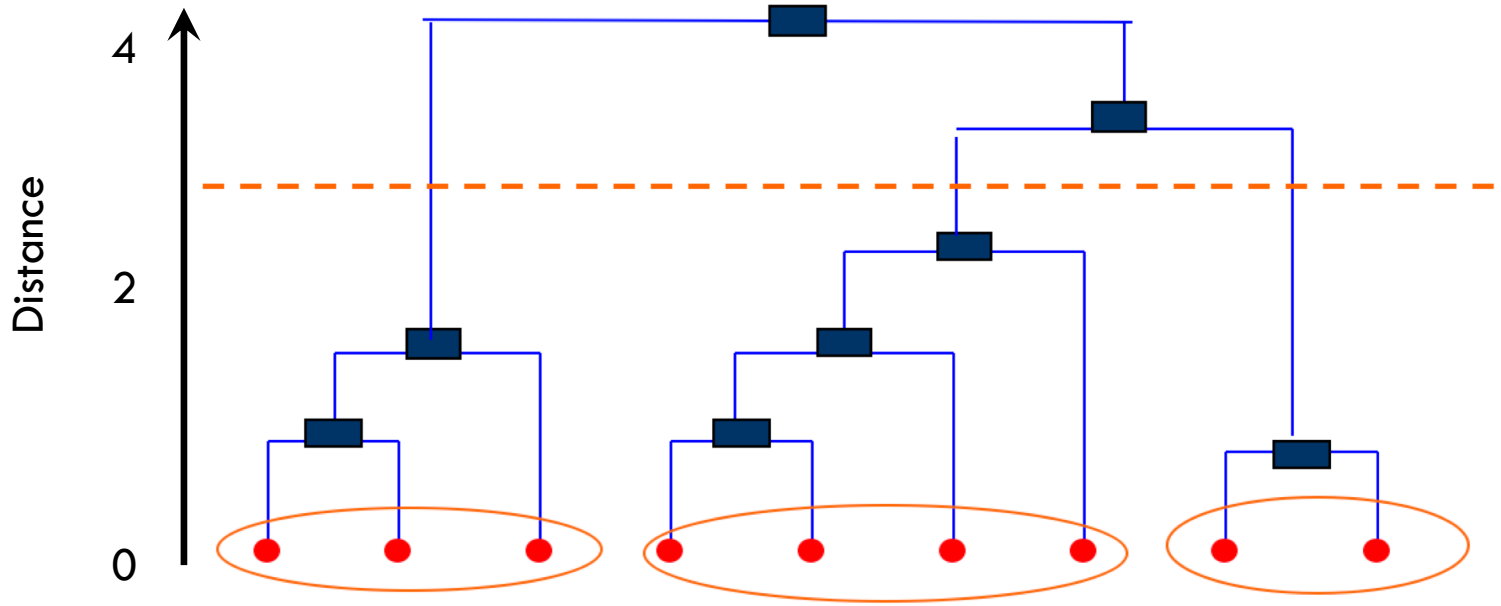
DENDROGRAMS

A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster

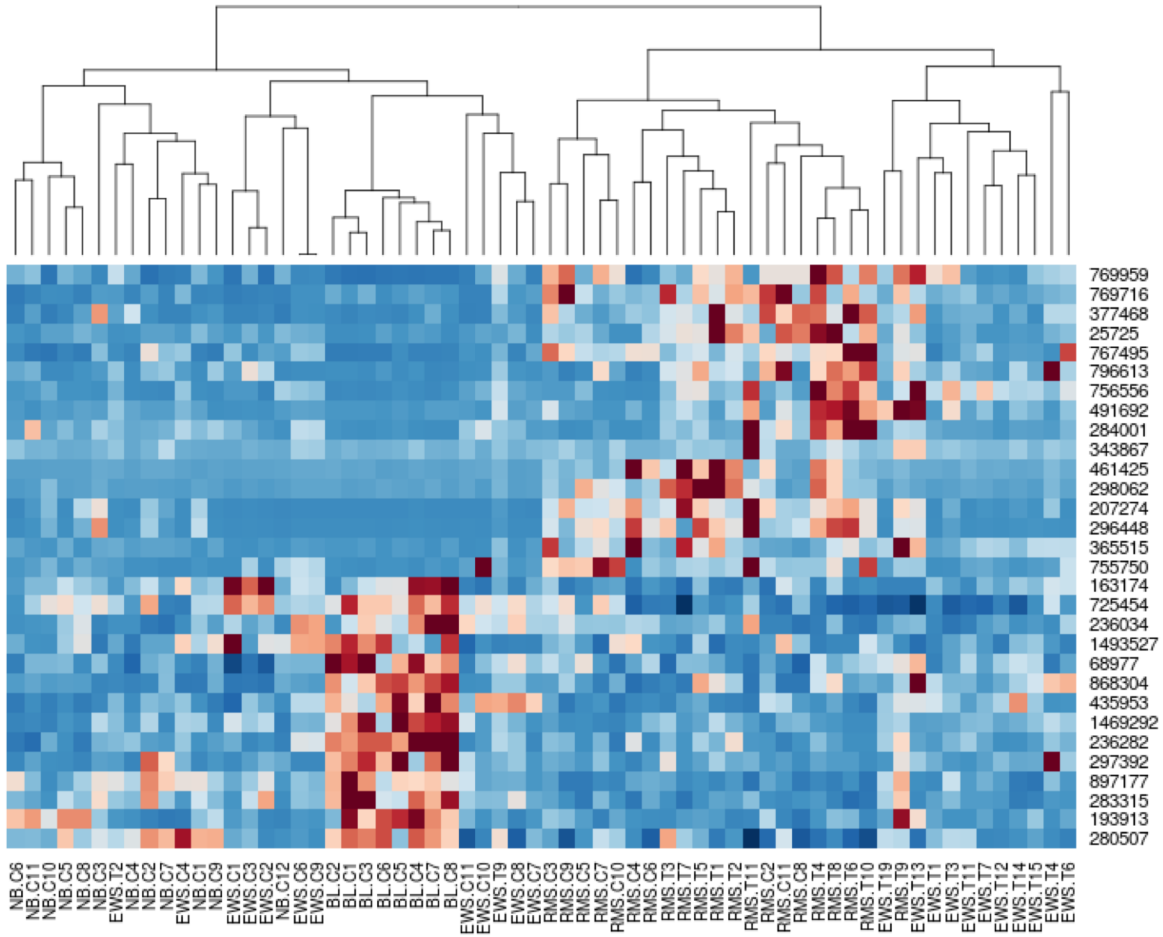


DENDROGRAMS

A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster



EXAMPLE OF HIERARCHICAL CLUSTERING OUTPUT



Source: https://stackoverflow.com/questions/17924828/differences-in-heatmap-clustering-defaults-in-r-heatplot-versus-heatmap-2

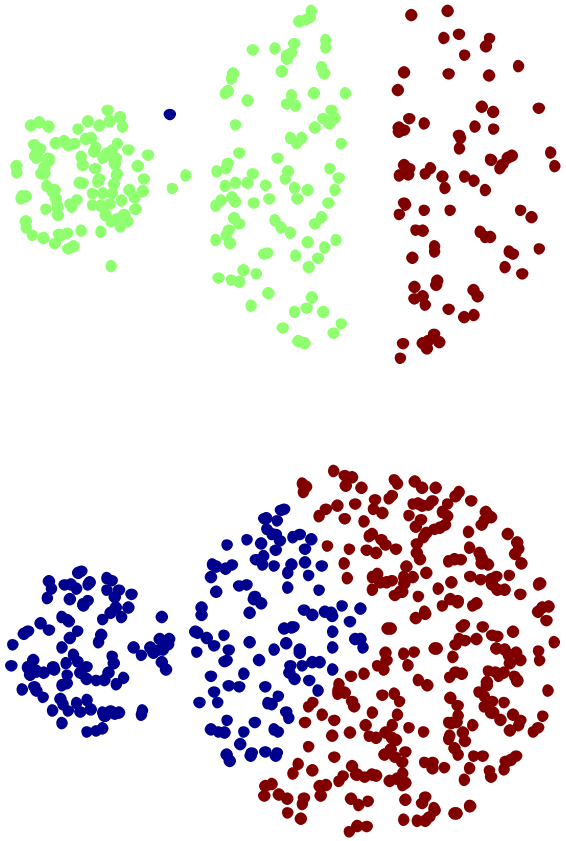
ADVANTAGES AND DISADVANTAGES

Advantages

- Hierarchies may correspond to meaningful taxonomies (e.g. animal kingdom, biological taxonomies)
- No need to determine number of clusters k

Disadvantages

- Different schemes have problems with one or more of the following:
  - Chaining: sensitivity to noise and outliers
  - Difficulty handling clusters of different sizes and non-globular shapes
  - Breaking large clusters



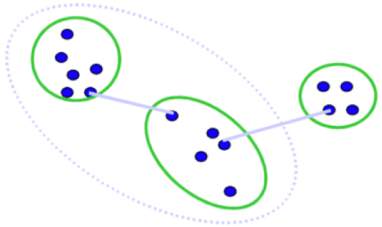
Q: HIERARCHICAL CLUSTERING AND TRANSFORMATIONS

Q: Suppose you run hierarchical clustering, then run it again but squaring the distance between each pair of points. Which of the following methods will always be unchanged in terms of the final cluster output?

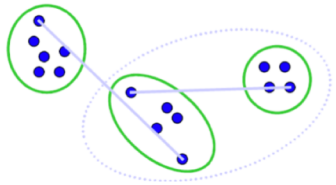
(a) Single Linkage

(b) Complete Linkage

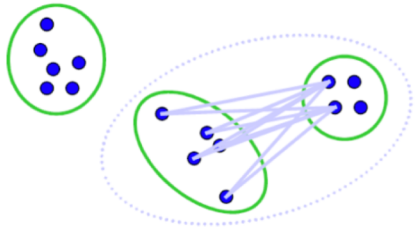
(c) Average Linkage



$$d_{\min}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p, q)$$



$$d_{\min}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)$$



$$d_{\text{avg}}(C_i, C_j) = \underset{p \in C_i, q \in C_j}{avg} d(p, q)$$

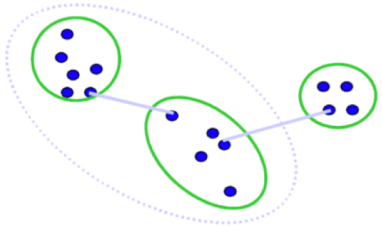
Q: HIERARCHICAL CLUSTERING AND TRANSFORMATIONS

Q: Suppose you run hierarchical clustering, then run it again but squaring the distance between each pair of points. Which of the following methods will always be unchanged in terms of the final cluster output?

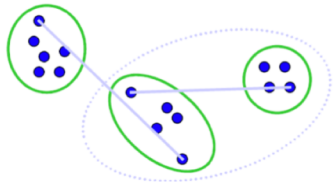
(a) Single Linkage

(b) Complete Linkage

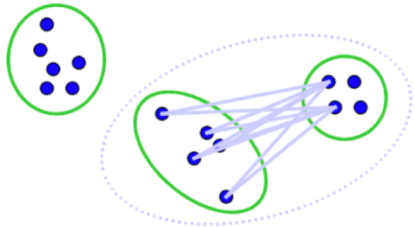
(c) Average Linkage



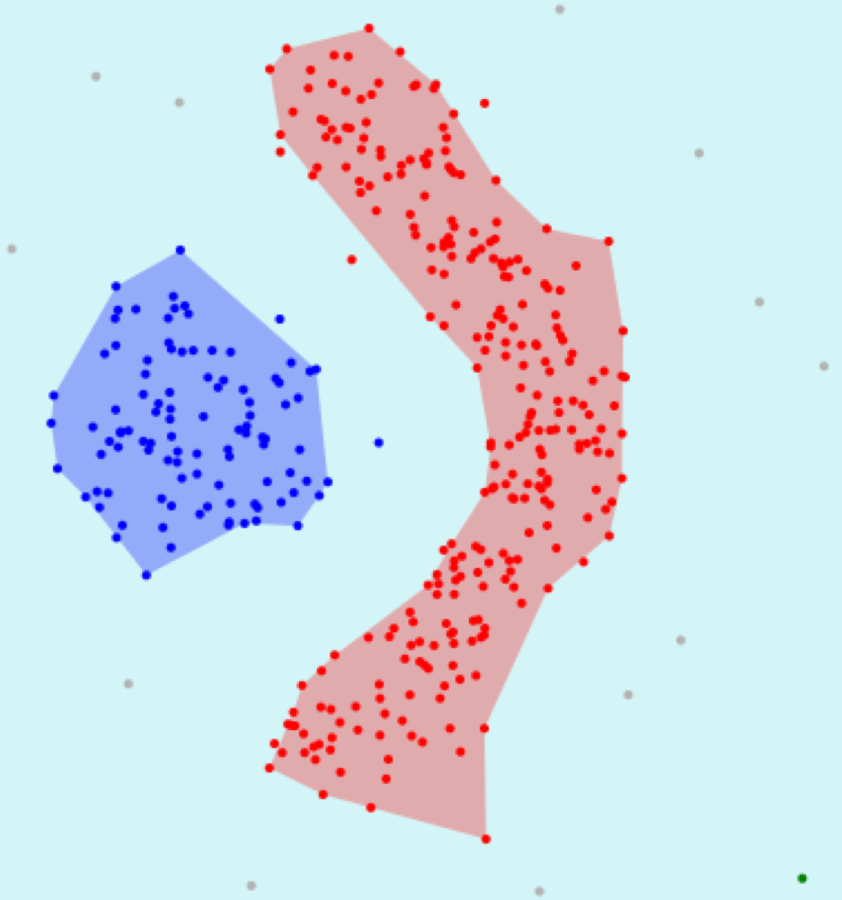
$$d_{\min}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p, q)$$



$$d_{\min}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)$$



$$d_{\text{avg}}(C_i, C_j) = \underset{p \in C_i, q \in C_j}{avg} d(p, q)$$

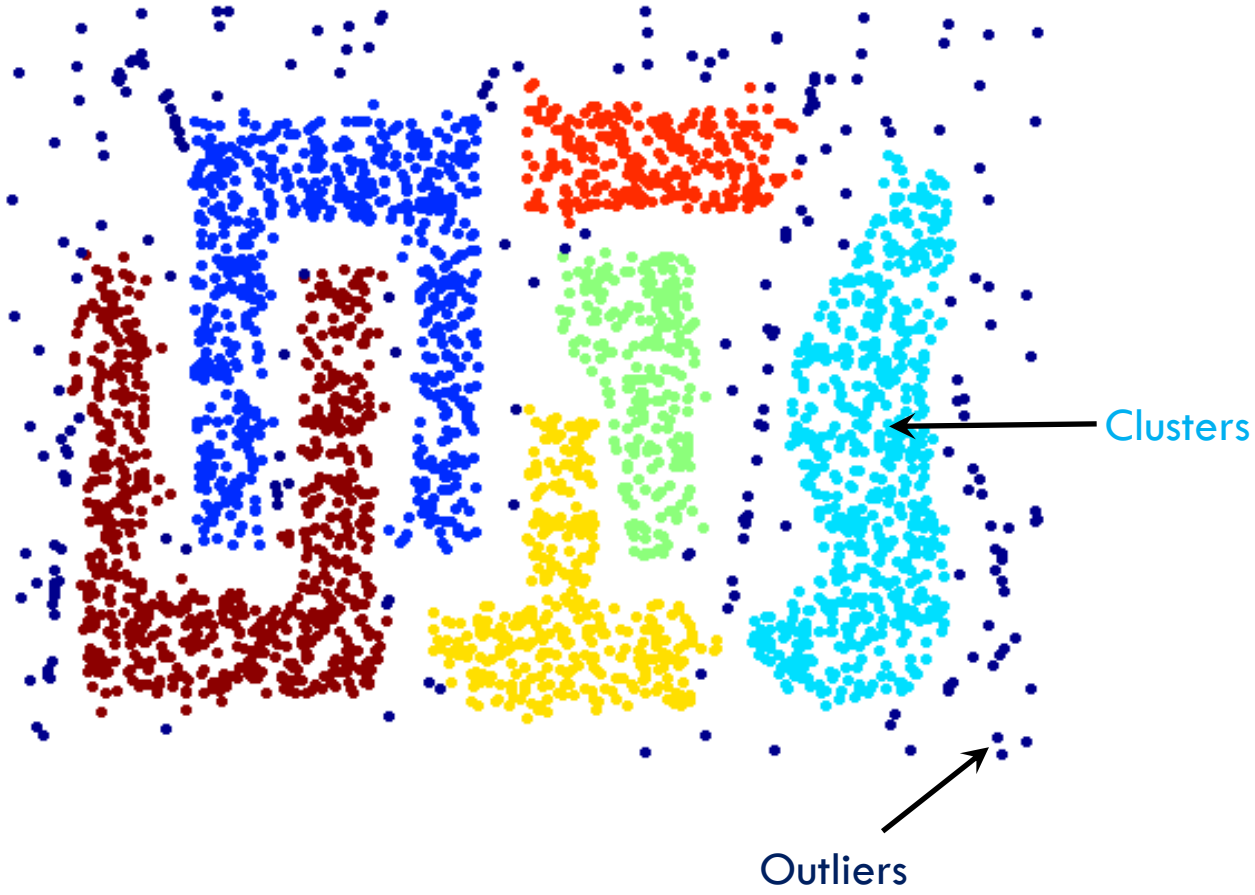


DENSITY-BASED CLUSTERING

DENSITY-BASED CLUSTERING

Basic ideas:

- *Clusters* can be of any shape
- They are dense regions surrounded by regions of lower density
- *Outliers* are isolated points that don't belong to a cluster



DBSCAN

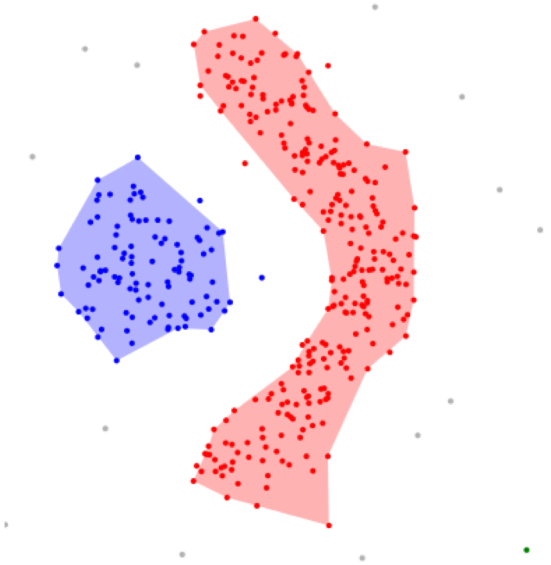
Density-Based Spatial Clustering of Applications with Noise

Intuition

- First find the points of 'high density' (**core points**)
- Then connect together core points which are nearby each other, to form clusters

Input Parameters

- ε – radius defining each point's **neighborhood**
- ***MinPts*** – minimum number of points in each point's neighborhood



CORE, BORDER AND OUTLIER POINTS

Core points are points with at least ***MinPts*** neighbors within their radius of ε.

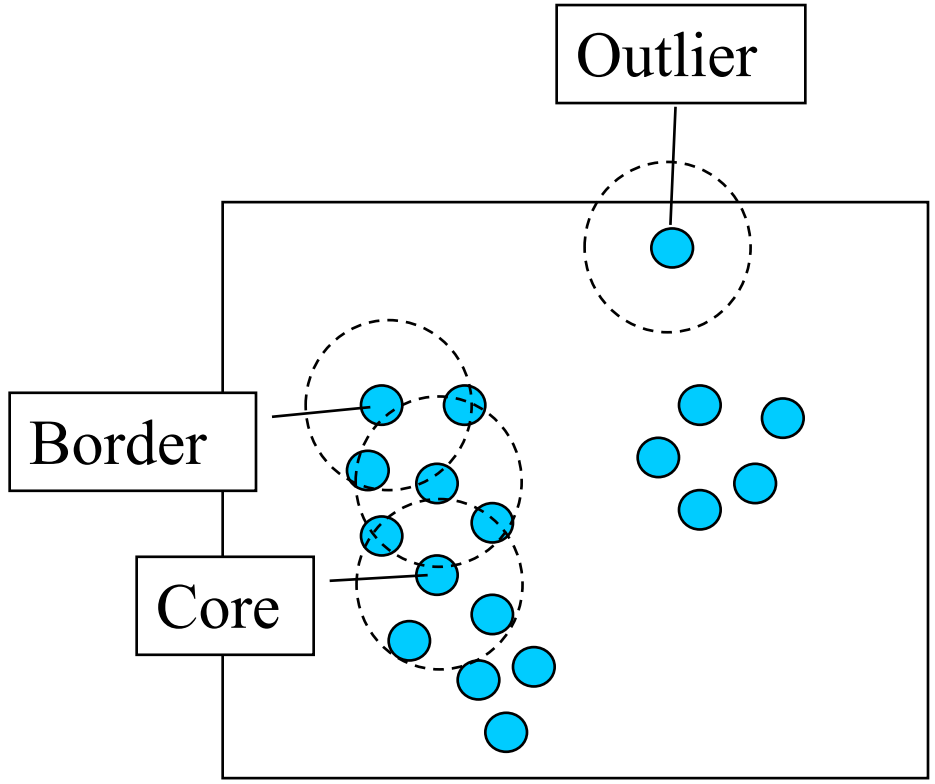
- (these will form the 'interior' of clusters)

Border points are non-core points with at least 1 core point in its neighborhood.

- (these will form the 'border' of clusters)

Outliers are all other points.

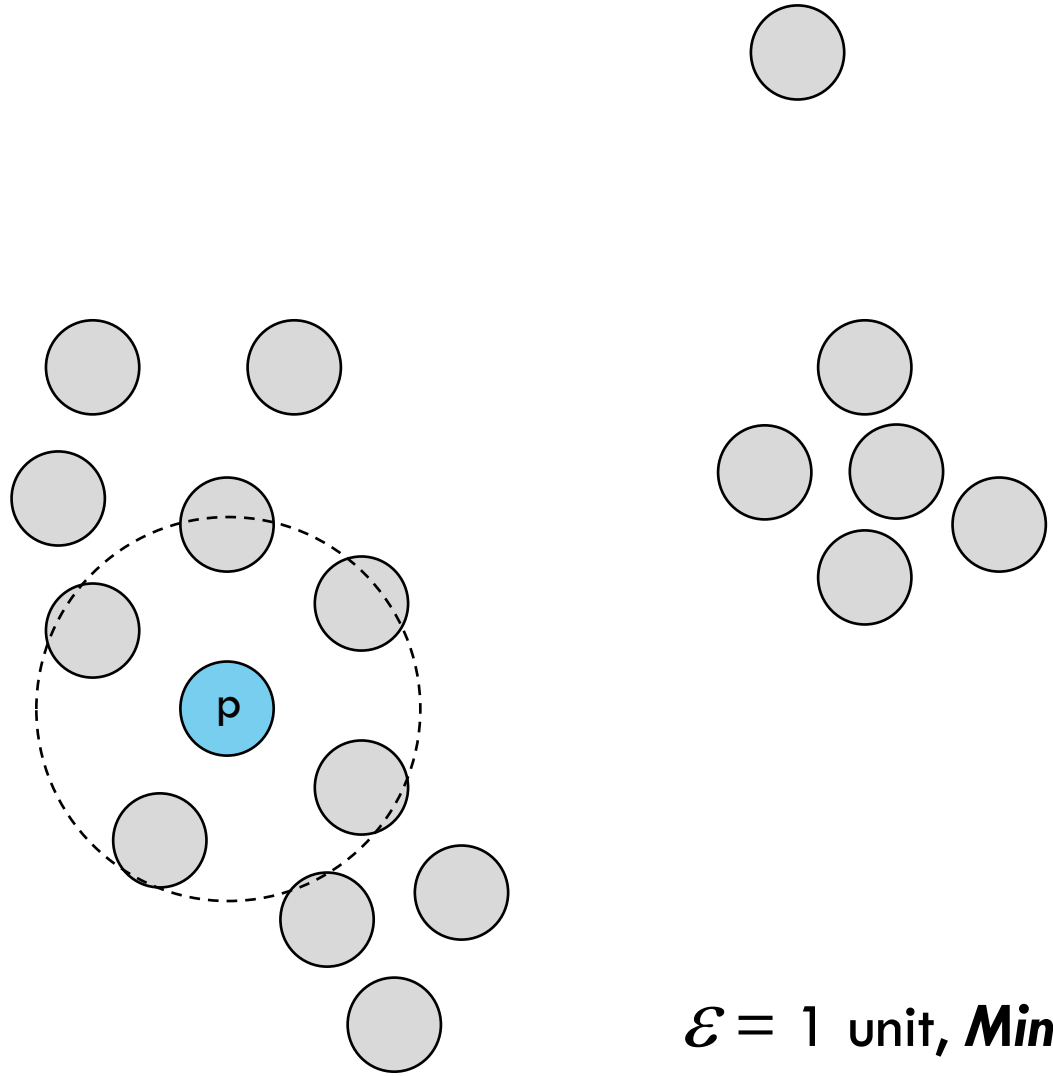
- (these will not belong to any cluster)



ε = 1 unit, ***MinPts*** = 5

DBSCAN ALGORITHM

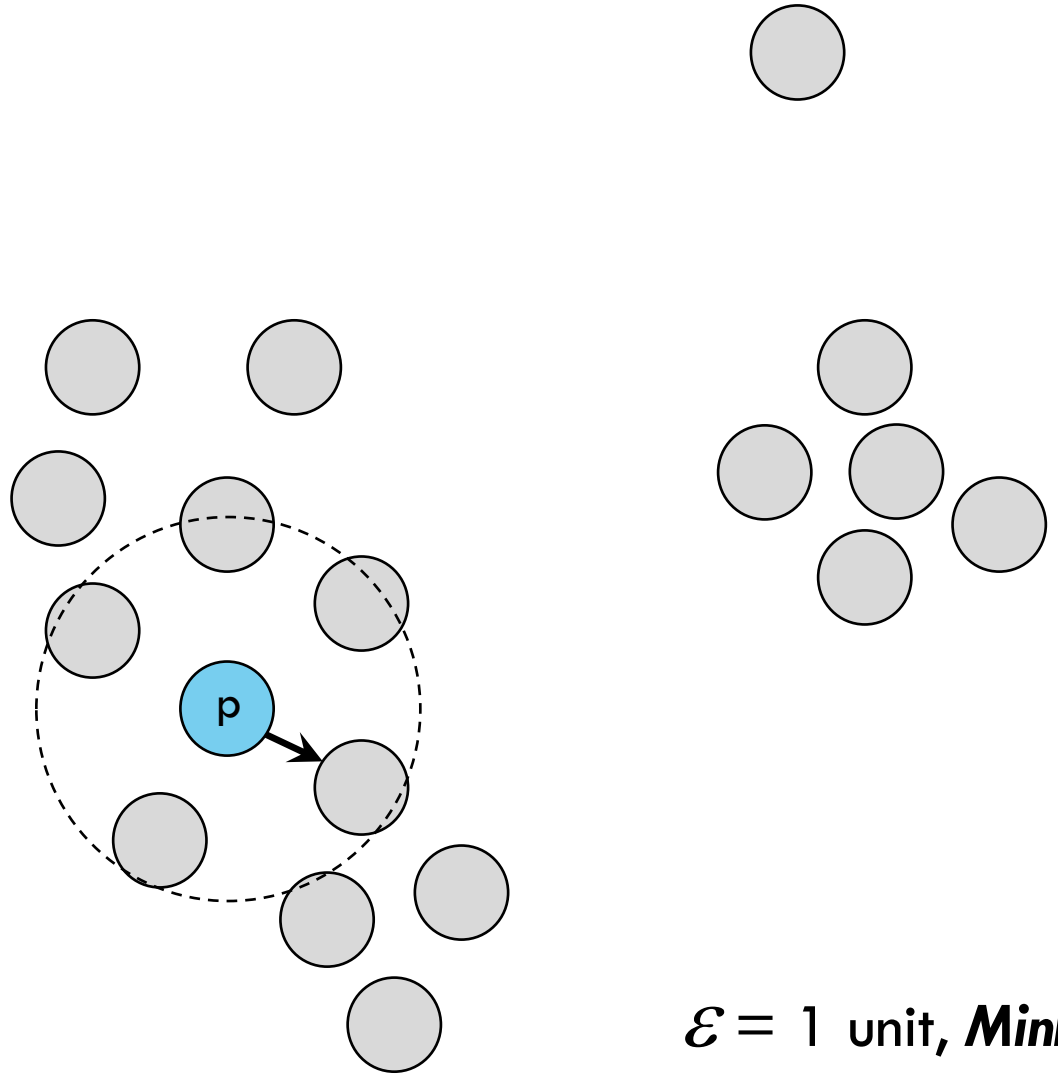
- Arbitrarily select a unexplored core point *p*.



$\mathcal{E}$ = 1 unit, ***MinPts*** = 5

DBSCAN ALGORITHM

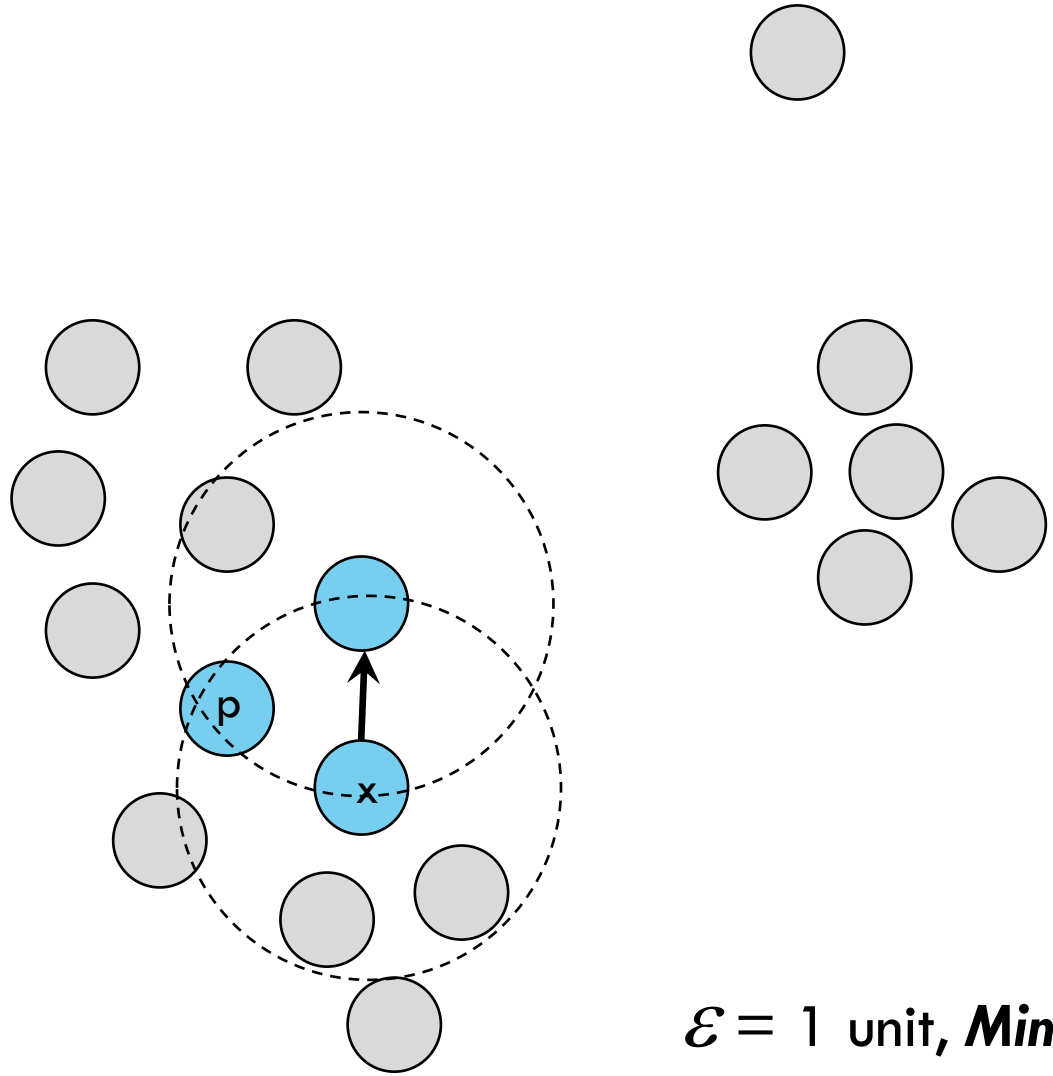
- Arbitrarily select a unexplored core point *p*.
- Recursively **`explore`** starting at p.



$\mathcal{E} = 1$ unit, ***MinPts*** $= 5$

DBSCAN ALGORITHM

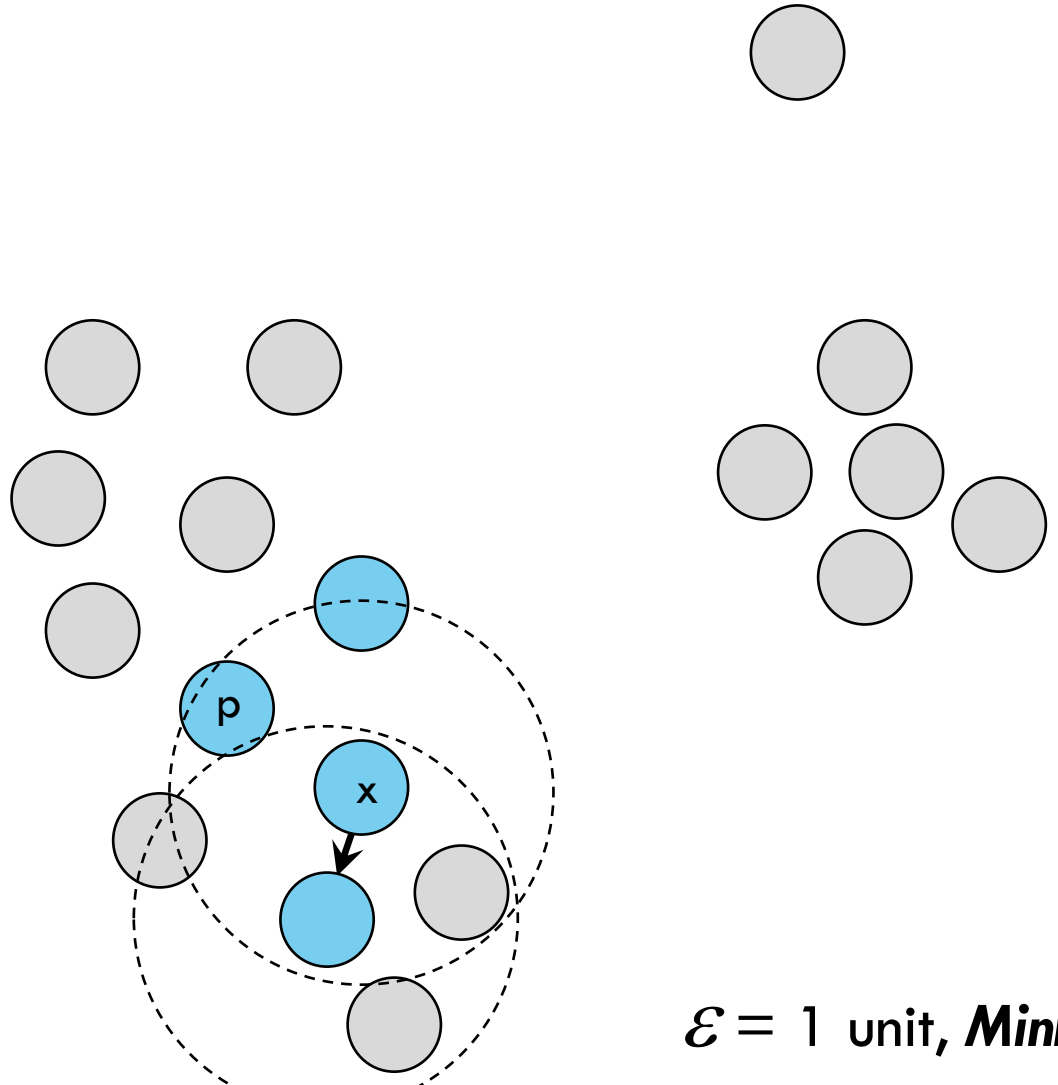
- Arbitrarily select a unexplored core point *p*.
- Recursively **`explore`** starting at p.
- When **`exploring`** a node x:
  - Add all points in x's neighborhood to the current cluster
  - Recursively **`explore`** all unexplored **core** points in x's neighborhood



$\mathcal{E} = 1$ unit, ***MinPts*** $= 5$

DBSCAN ALGORITHM

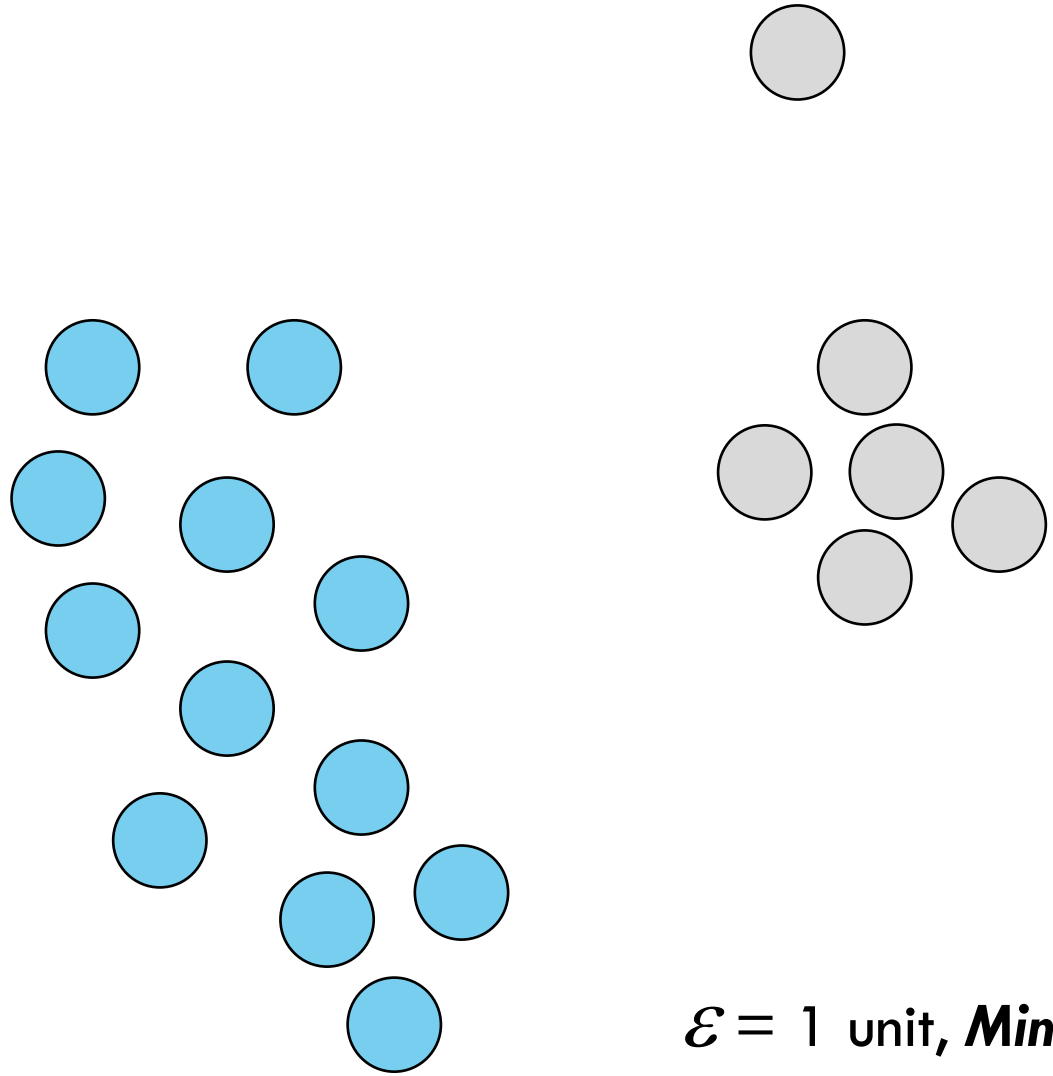
- Arbitrarily select a unexplored core point *p*.
- Recursively **`explore`** starting at p.
- When **`exploring`** a node x:
  - Add all points in x's neighborhood to the current cluster
  - Recursively **`explore`** all unexplored **core** points in x's neighborhood



$\mathcal{E} = 1$ unit, ***MinPts*** $= 5$

DBSCAN ALGORITHM

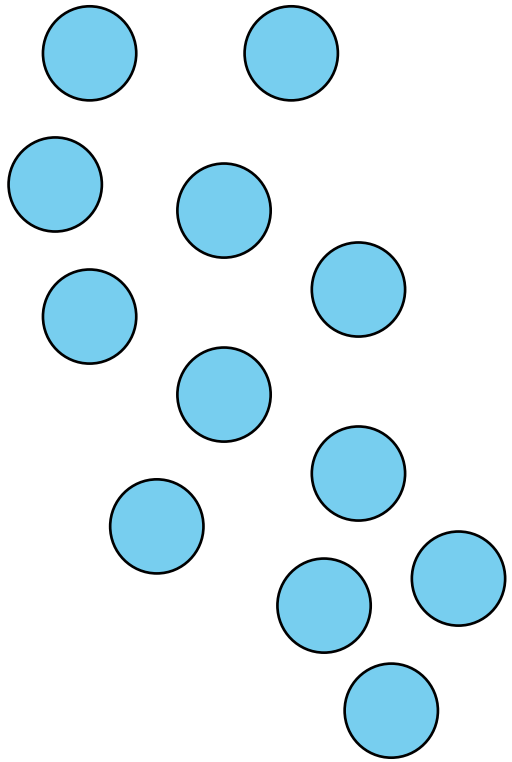
- Arbitrarily select a unexplored core point *p*.
- Recursively **`explore`** starting at p.
- When **`exploring`** a node x:
  - Add all points in x's neighborhood to the current cluster
  - Recursively **`explore`** all unexplored **core** points in x's neighborhood

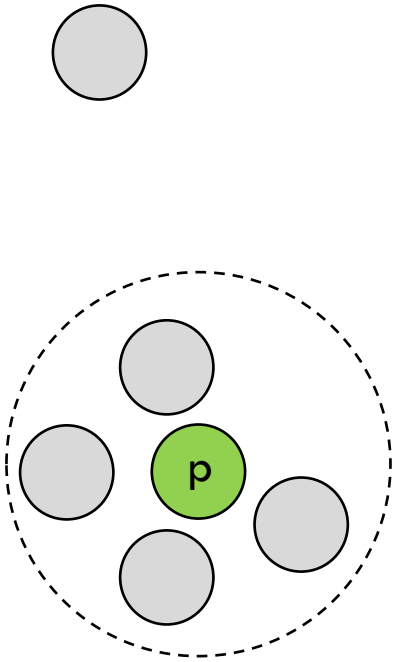


$\mathcal{E} = 1$ unit, ***MinPts*** $= 5$

DBSCAN ALGORITHM

- Arbitrarily select a unexplored core point *p*.
- Recursively **`explore`** starting at p.
- When **`exploring`** a node x:
  - Add all points in x's neighborhood to the current cluster
  - Recursively **`explore`** all unexplored **core** points in x's neighborhood

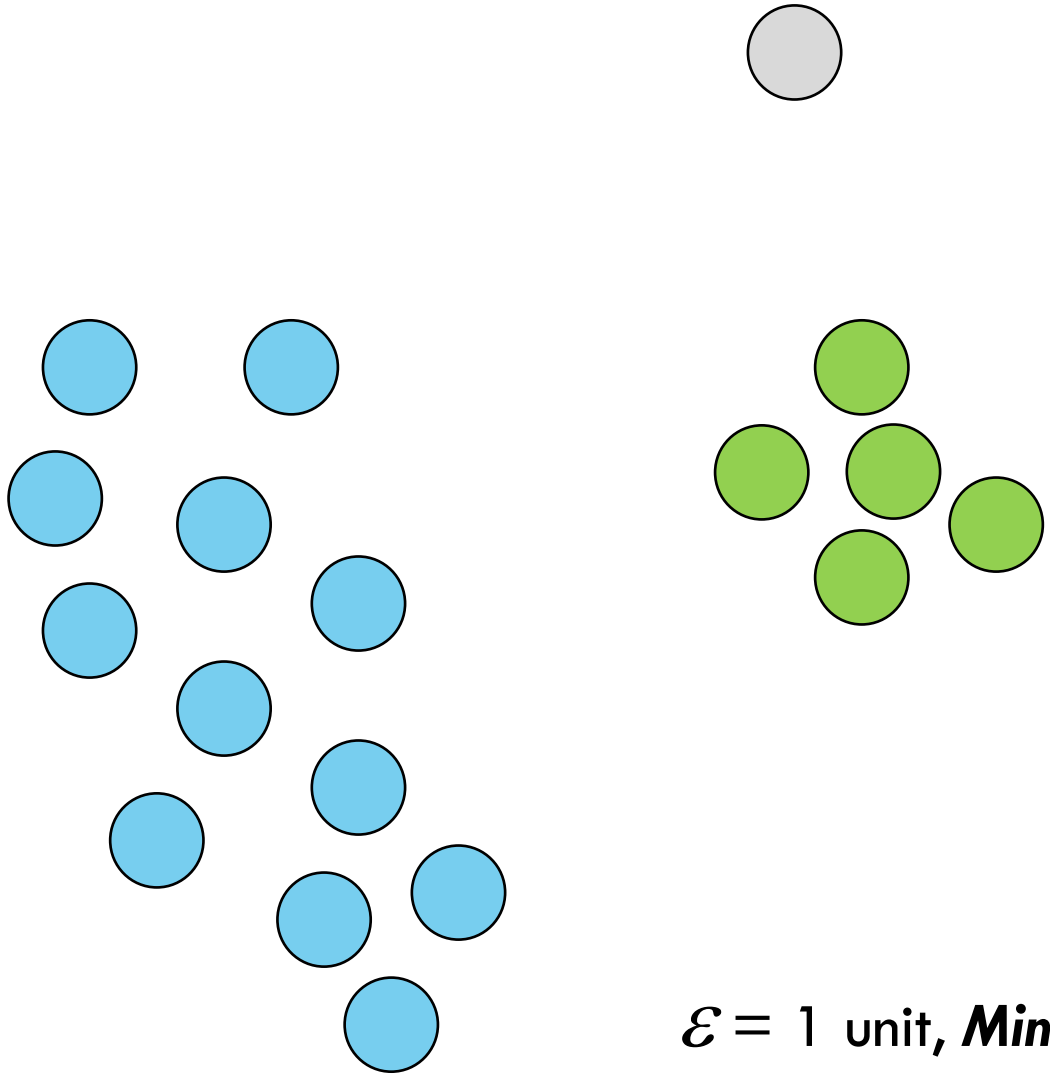




$\mathcal{E} = 1$ unit, ***MinPts*** $= 5$

DBSCAN ALGORITHM

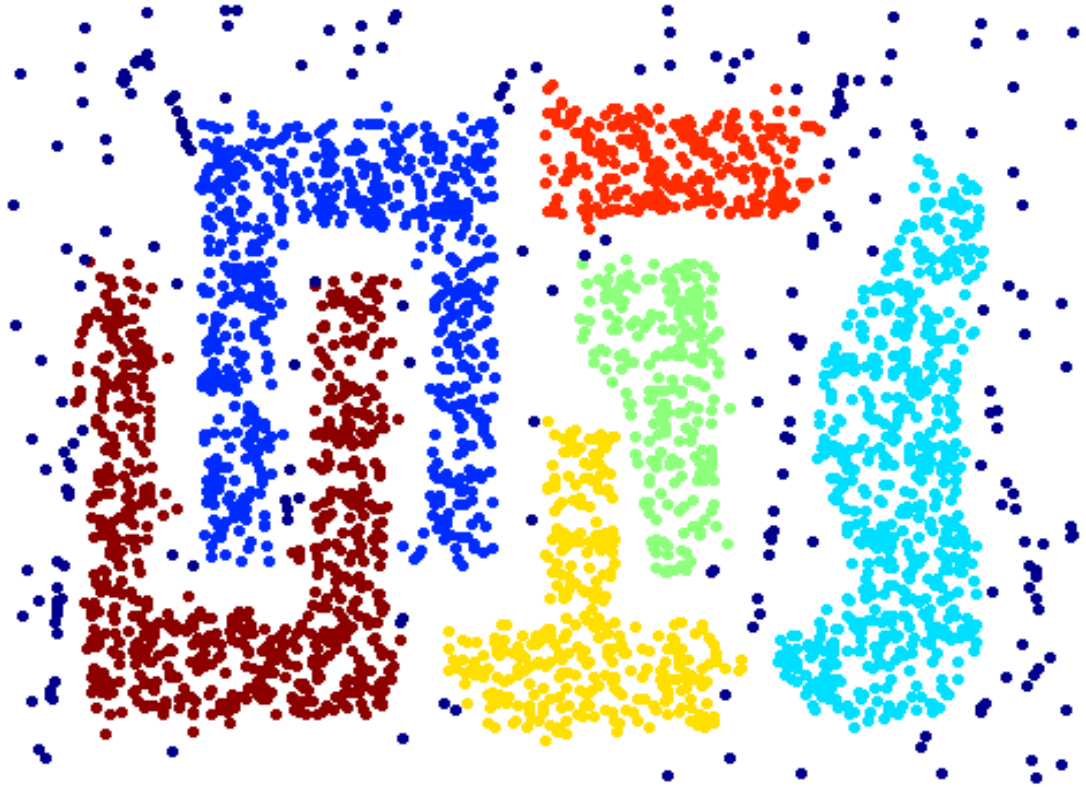
- Arbitrarily select a unexplored core point *p*.
- Recursively **`explore`** starting at p.
- When **`exploring`** a node x:
  - Add all points in x's neighborhood to the current cluster
  - Recursively **`explore`** all unexplored **core** points in x's neighborhood
- When this recursion terminates, continue to the next cluster



$\mathcal{E} = 1$ unit, ***MinPts*** $= 5$

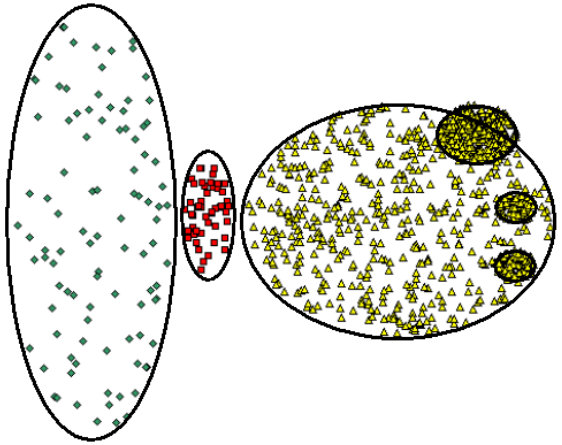
DBSCAN: STRENGTHS

- Automatically finds no. of clusters
- Handles clusters of different shapes and sizes
- Resistant to outliers
- Resistant to chaining

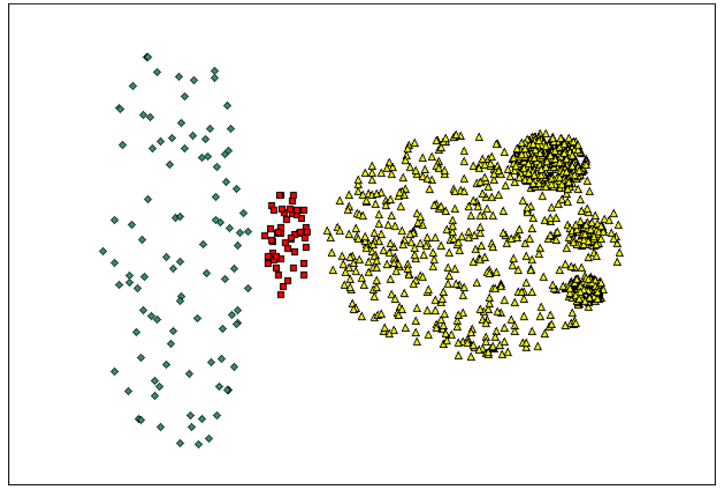


DBSCAN: WEAKNESSES

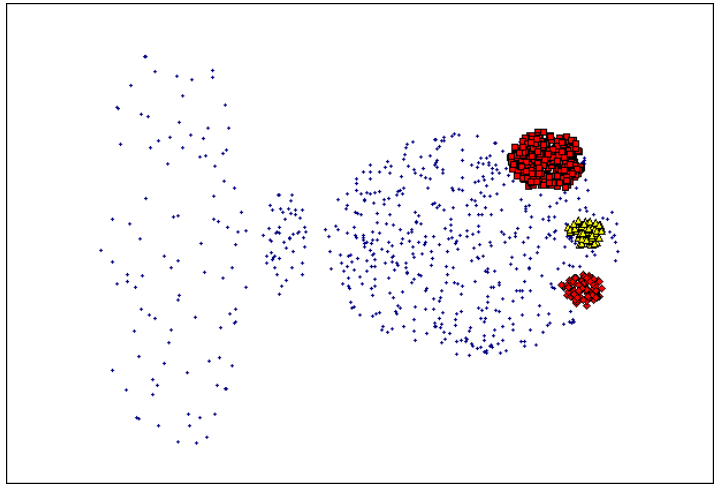
- Sensitive to parameters (ε and ***MinPts***)



Original Points



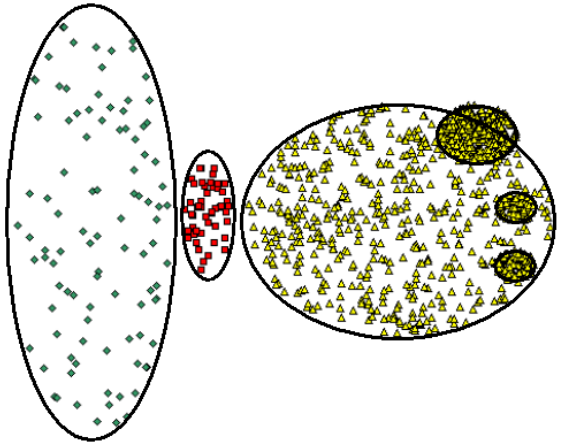
(MinPts=4, Eps=9.92).



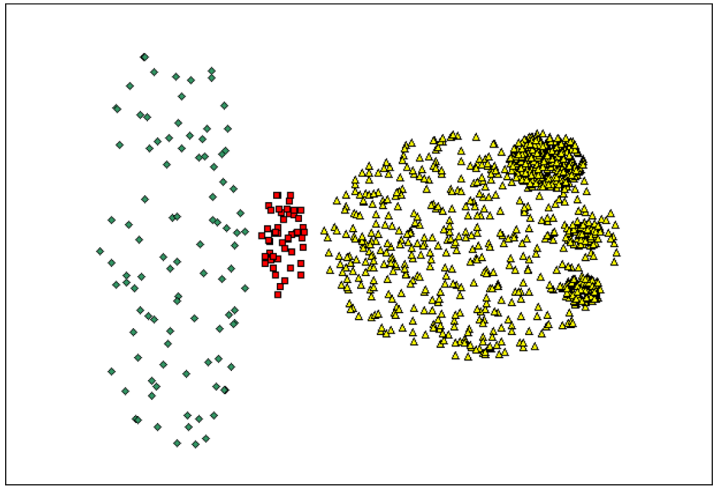
(MinPts=4, Eps=9.75)

DBSCAN: WEAKNESSES

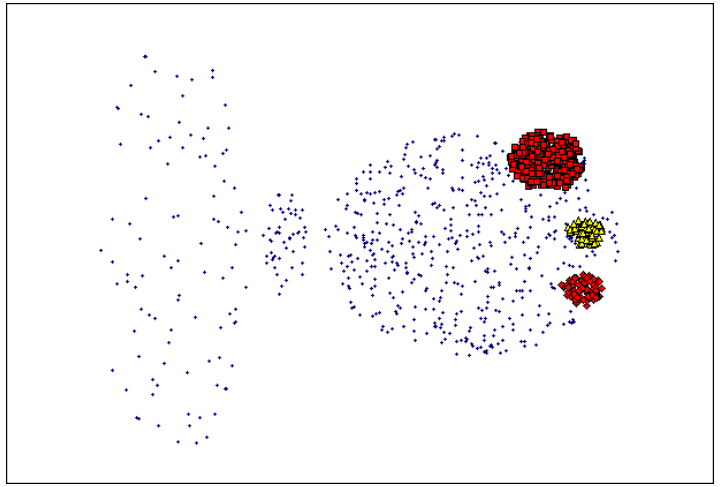
- Sensitive to parameters (ε and ***MinPts***)
- Cannot model clusters of different density



Original Points



(MinPts=4, Eps=9.92).



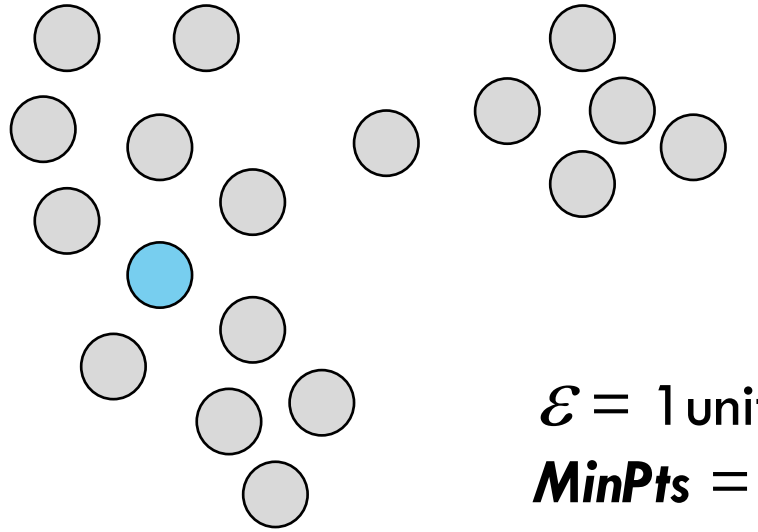
(MinPts=4, Eps=9.75)

IS DBSCAN DETERMINISTIC?

Q: Is DBSCAN deterministic? I.e. Does it always ends at the same result regardless of which nodes the algorithm happened to start at?

(a) Yes

(b) No



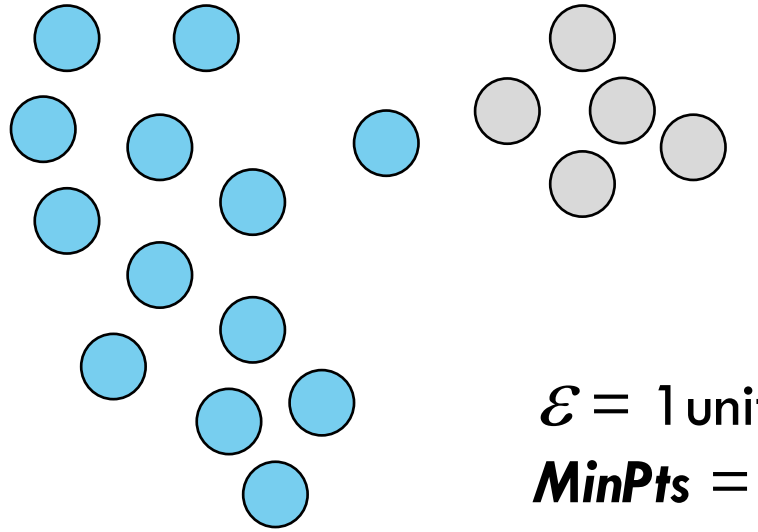
$\varepsilon = 1$unit,
MinPts $= 5$

IS DBSCAN DETERMINISTIC?

Q: Is DBSCAN deterministic? I.e. Does it always ends at the same result regardless of which nodes the algorithm happened to start at?

(a) Yes

(b) No

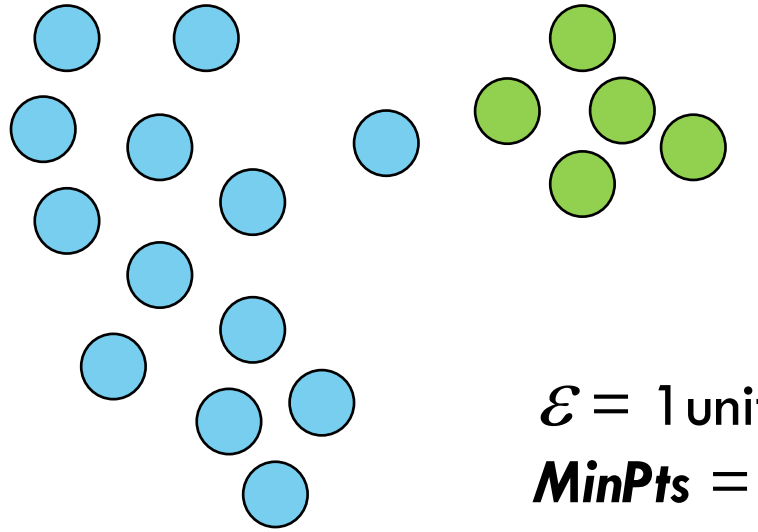


IS DBSCAN DETERMINISTIC?

Q: Is DBSCAN deterministic? I.e. Does it always ends at the same result regardless of which nodes the algorithm happened to start at?

(a) Yes

(b) No



$\varepsilon = 1$ unit,
MinPts $= 5$

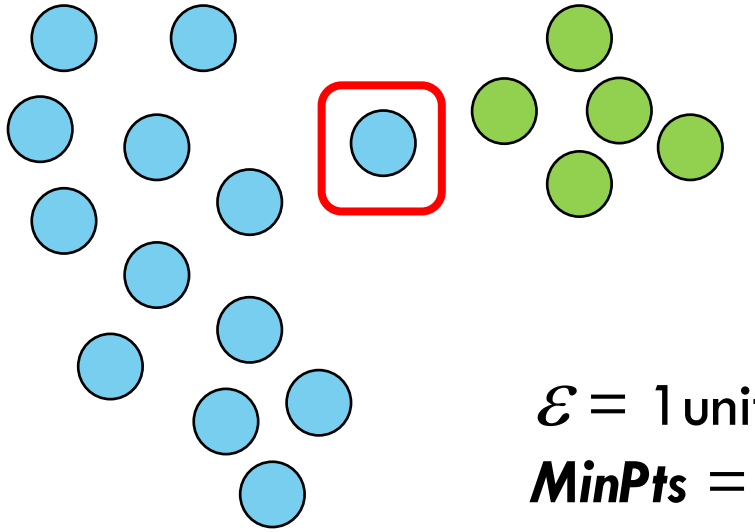
IS DBSCAN DETERMINISTIC?

Q: Is DBSCAN deterministic? I.e. Does it always ends at the same result regardless of which nodes the algorithm happened to start at?

(a) Yes

(b) No

(Some border points can have multiple possibilities for which cluster they end up in. Aside from this, the output of DBSCAN is deterministic).



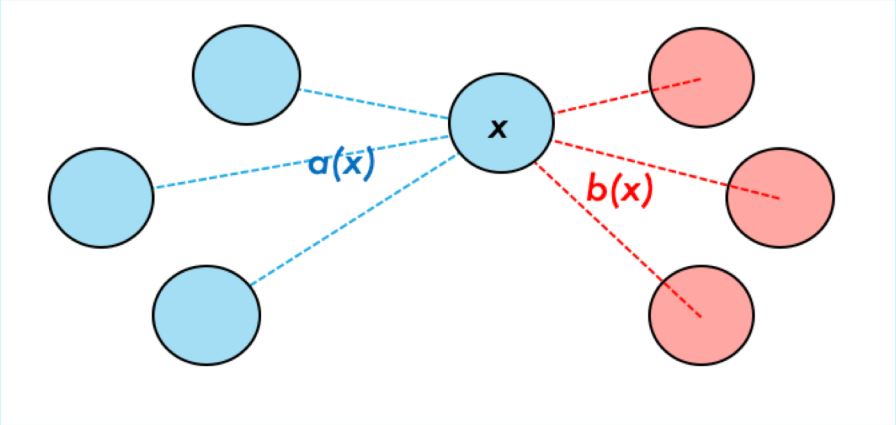
$\varepsilon = 1$ unit,
MinPts $= 5$

Optional

COMPARISON OF CLUSTERING METHODS

	Algorithm	Suitable for	Time Complexity (in n)	Explainability
Center-based	K-means (K-means++, X-means)	Near-spherical clusters	Linear in practice[1]	Cluster centers[2]
	K-medoids	Near-spherical clusters around data points; Complex data types	Quadratic	Cluster representatives
Hierarchical	Single Linkage	Complex-shaped clusters; low noise setting	Quadratic	-
	Complete Linkage	Near-spherical clusters	Quadratic	-
	Average Linkage	Near-spherical clusters	Cubic[3]	-
Density-based	DBSCAN	Clusters of high density surrounded by low density	Linear in practice	-

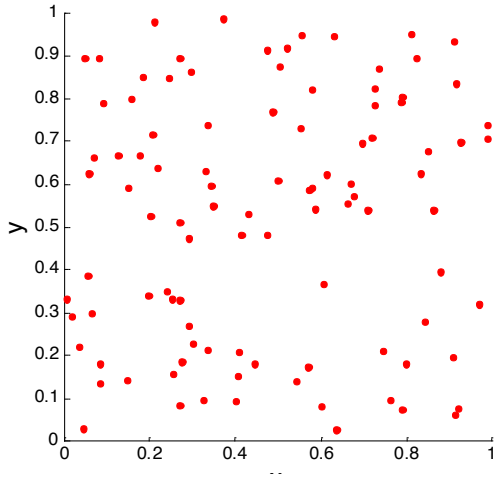
1: K-Means and DBScan tend to run in linear (O(n)) time in practice, though in theory they are not exactly linear. However, hierarchical clustering is $O(n^2)$ or slower. The table refers to time complexity with respect to the data size n. With respect to dimensions, all these methods are close to linear, since the time to compute each distance scales linearly with dimension size.
2: We consider "explainable" models as those whose predictions can be understood based on a easy-to-understand / low-parametric representation.
3: Quadratic-time $O(n^2 \log n)$ algorithms also exist, but at the cost of quadratic memory, which is often prohibitive.

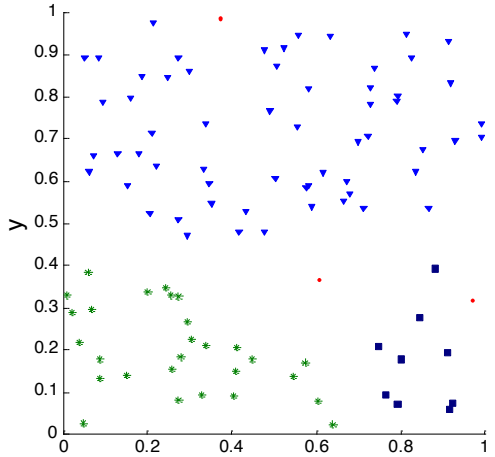


CLUSTER EVALUATION

CLUSTERS FOUND IN RANDOM DATA

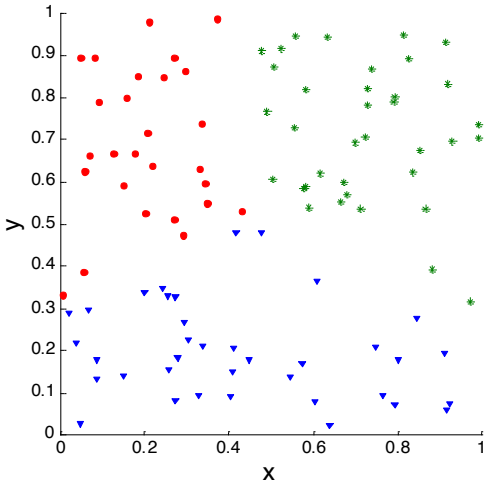
Random Points

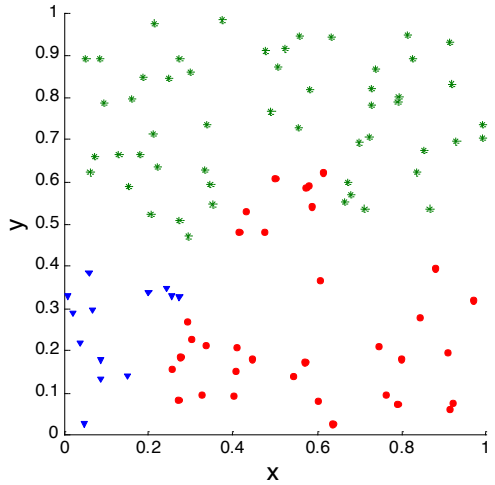




DBSCAN

K-means





Complete Linkage

CLUSTER QUALITY

How do we know how good is the clustering generated by an algorithm?

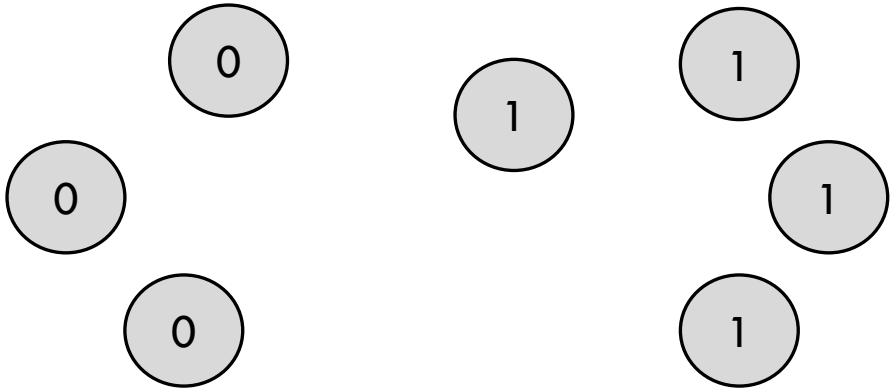
How do we know that the clusters discovered are “real”, i.e. not just occurring by chance?

INTERNAL VS EXTERNAL QUALITY MEASURES

External quality measures evaluate a clustering against a set of **ground truth** (i.e. a set of externally known labels)

Internal quality measures evaluate a clustering without ground truth, from the data itself

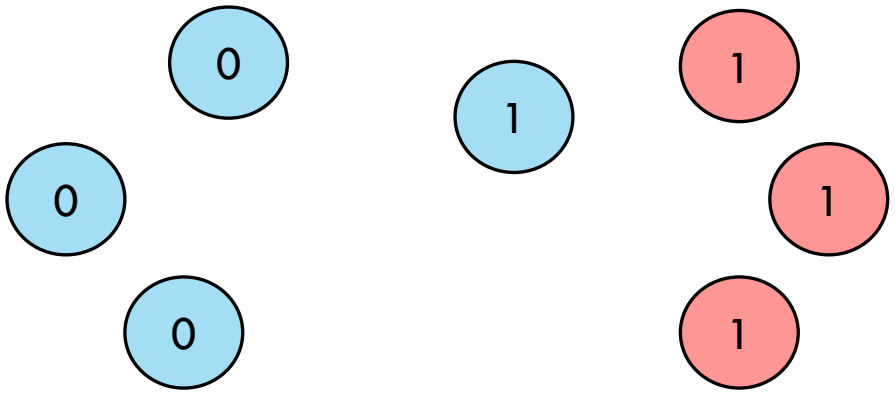
EXTERNAL QUALITY MEASURES



Numbers: ground truth labels

Label
0
0
0
1
1
1
1

EXTERNAL QUALITY MEASURES



Numbers: ground truth labels

Colors: output of clustering algorithm

Cluster	Label
Blue	0
Blue	0
Blue	0
Blue	1
Red	1
Red	1
Red	1

CLUSTER PURITY

Assign each cluster to the label that has highest overlap with this cluster

Then compute the fraction of matches between the assigned labels and ground truth

For blue cluster: the label with highest overlap is 0

Cluster	Label
Blue	0
Blue	0
Blue	0
Blue	1
Red	1
Red	1
Red	1

For red cluster: the label with highest overlap is 1

Purity = 6 / 7

CLUSTER PURITY

Assign each cluster to the label that has highest overlap with this cluster

Then compute the fraction of matches between the assigned labels and ground truth

For blue cluster: the label with highest overlap is 0

Cluster	Label
Blue → 0	0
Blue → 0	0
Blue → 0	0
Blue → 0	1
Red → 1	1
Red → 1	1
Red → 1	1

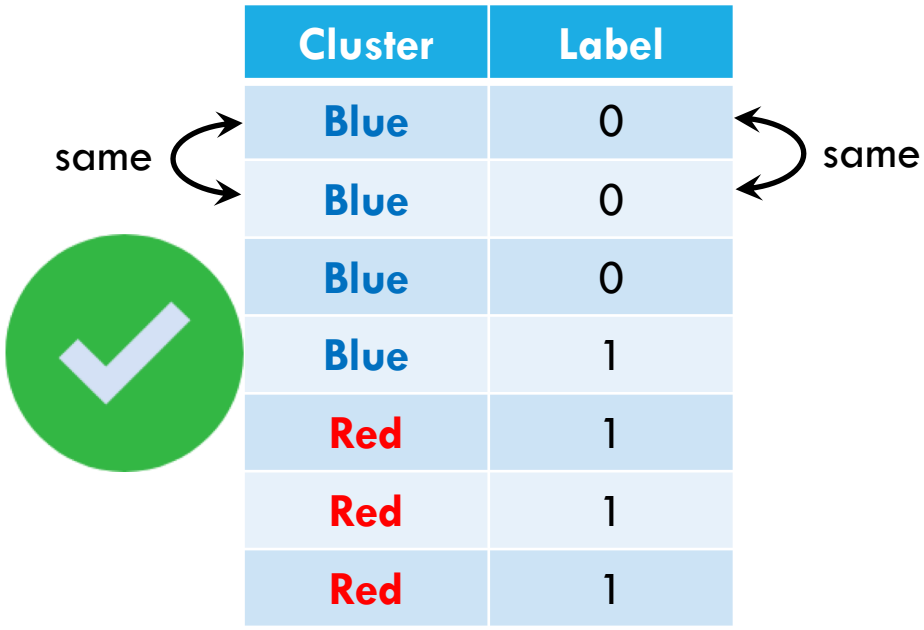
For red cluster: the label with highest overlap is 1

Purity = 6 / 7

RAND INDEX

Based on the clustering, each pair of points is either ‘same cluster’ or ‘different cluster’

Rand index is the fraction of such decisions that match the corresponding decisions in ground truth

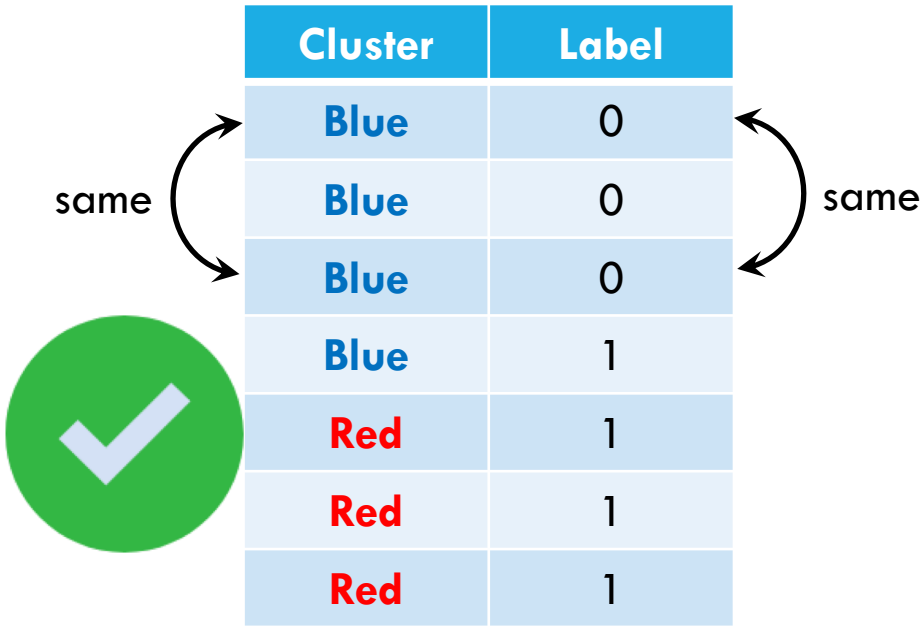


Cluster	Label
Blue	0
Blue	0
Blue	0
Blue	1
Red	1
Red	1
Red	1

RAND INDEX

Based on the clustering, each pair of points is either ‘same cluster’ or ‘different cluster’

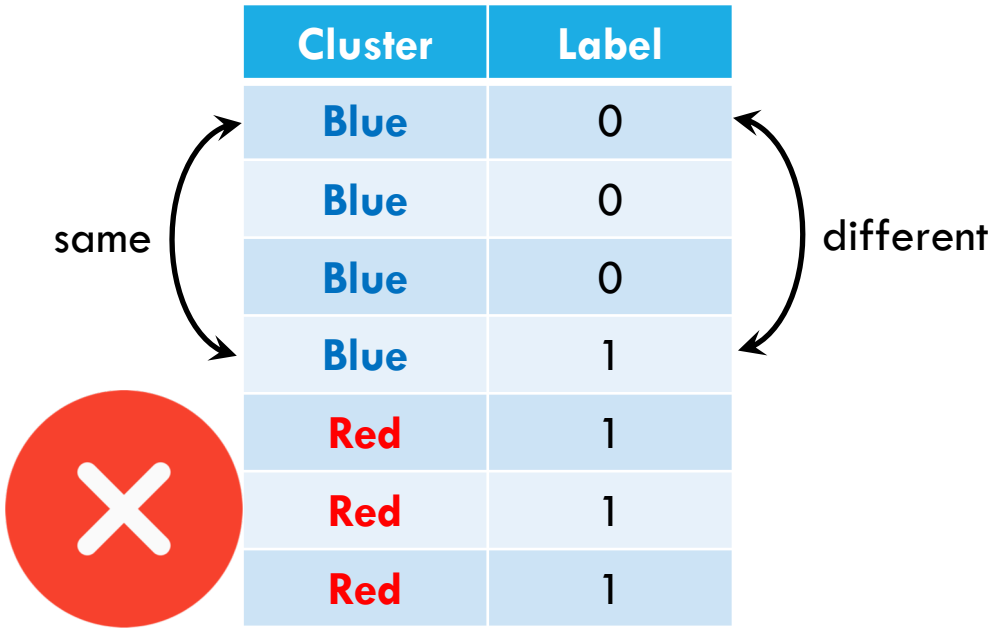
Rand index is the fraction of such decisions that match the corresponding decisions in ground truth



RAND INDEX

Based on the clustering, each pair of points is either ‘same cluster’ or ‘different cluster’

Rand index is the fraction of such decisions that match the corresponding decisions in ground truth



RAND INDEX

Based on the clustering, each pair of points is either ‘same cluster’ or ‘different cluster’

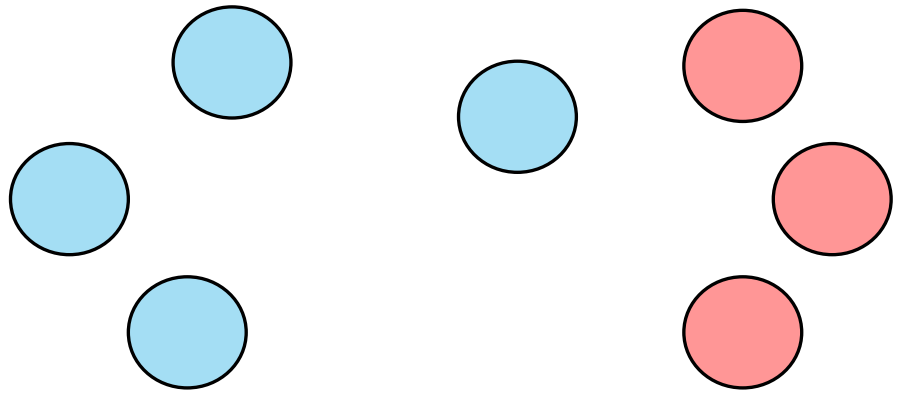
Rand index is the fraction of such decisions that match the corresponding decisions in ground truth

Cluster	Label
Blue	0
Blue	0
Blue	0
Blue	1
Red	1
Red	1
Red	1

Rand Index = 15 / 21

INTERNAL QUALITY MEASURES

If ground truth is unavailable, only internal quality measures (i.e. based on the data itself) can be used



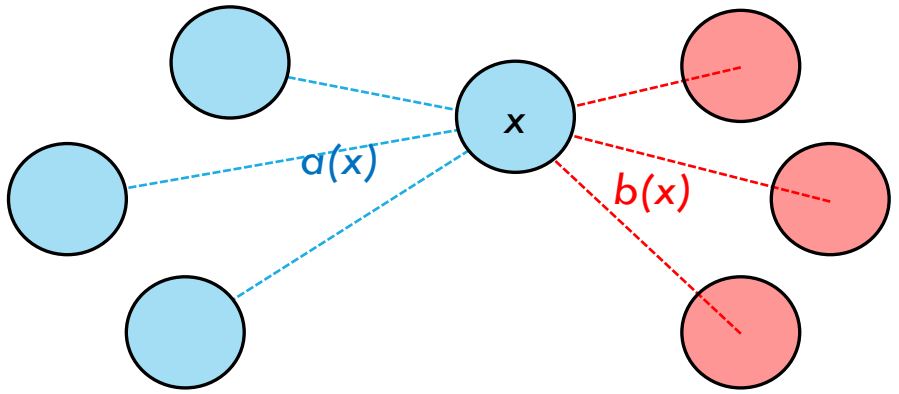
SILHOUETTE COEFFICIENT

Intuition: we want high inter-cluster distance, and low intra-cluster distance

Often used for choosing the number of clusters

For each point x, define

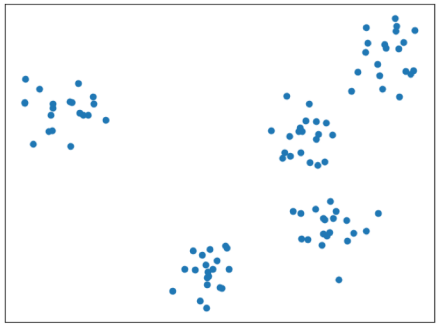
- **Cohesion** $a(x)$: average distance to points in the same cluster
- **Separation** $b(x)$: average distance to points in another cluster. Take the minimum over clusters.
- **Silhouette** $s(x) = \dfrac{b(x) - a(x)}{\max\{a(x), b(x)\}}$
- $-1 \leq s(x) \leq 1$; -1 = bad, 0 = indifferent, 1 = good



Silhouette coefficient: $SC = \dfrac{1}{N}\sum_{i=1}^{N} s(x_i)$

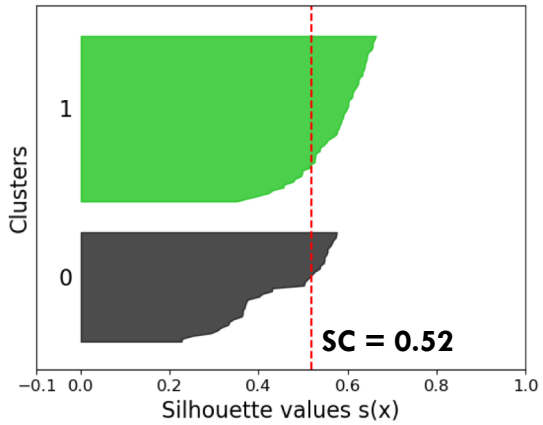
SILHOUETTE PLOT & SELECTING NO. OF CLUSTERS

For each K, we run K-means, then create the below "silhouette plots" which show the distribution of silhouette values s(x) in each cluster

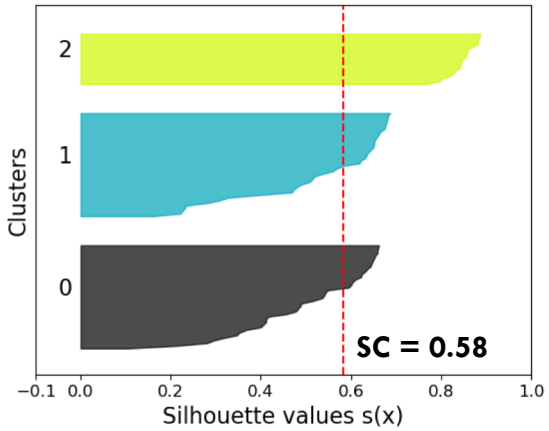


input data
(100 data points)

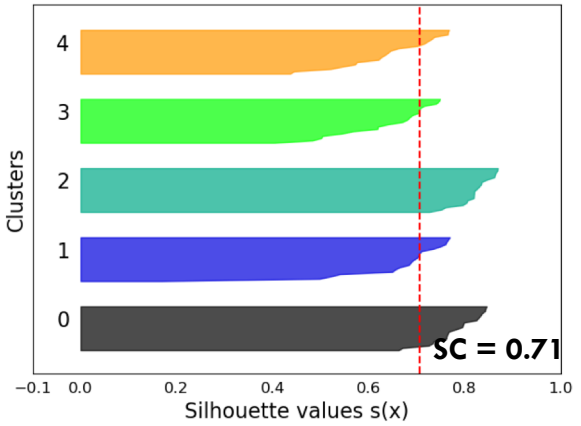
K = 2



K = 3



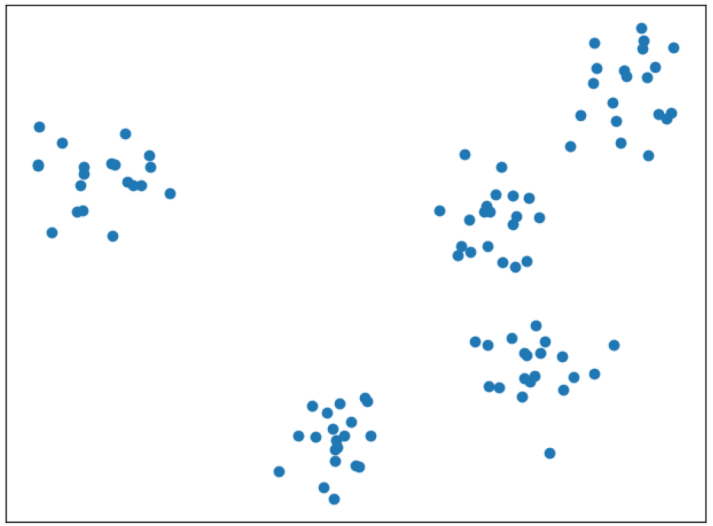
K = 5



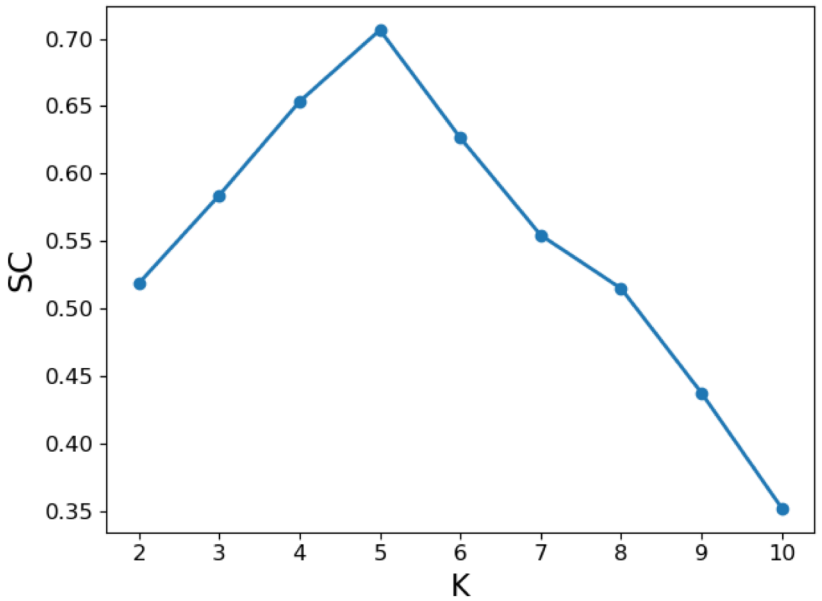
SILHOUETTE PLOT & SELECTING NO. OF CLUSTERS

K can be selected by maximizing the silhouette coefficient:

input data (100 data points)



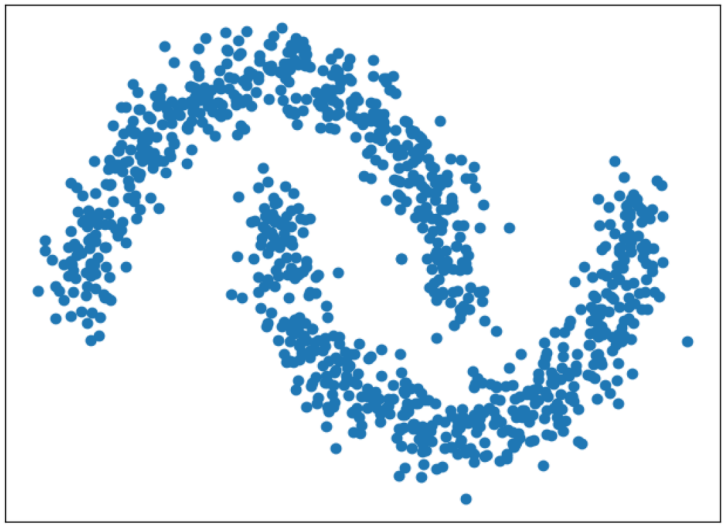
Silhouette Coefficients



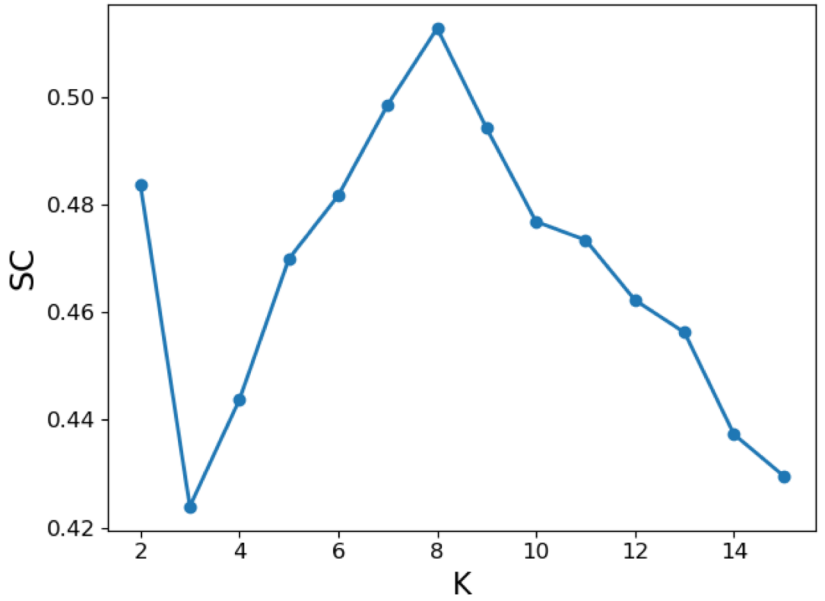
SILHOUETTE PLOT & SELECTING NO. OF CLUSTERS

K can be selected by maximizing the silhouette coefficient:

input data (100 data points)



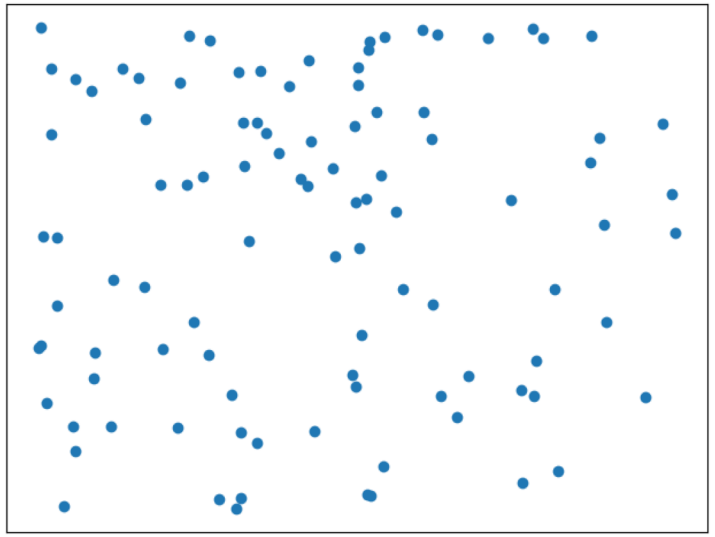
Silhouette Coefficients



SILHOUETTE PLOT & SELECTING NO. OF CLUSTERS

K can be selected by maximizing the silhouette coefficient:

random noise



SC for increasing K