L03 Clustering

D. Schneider, B. Stuhr, J. Haselberger

Gaussian Distribution

- Standard deviation and mean describe the Gaussian distribution
- In *n*-dimensional case, we use mean-vectors and covariance matrices

(Conditional) probability

- Conditional probability allows us to update probabilistic models when additional information are available
- Conditional probabilities can be used to determine the intersection of several events in a structured way

- Bayes' Theorem deals with the relationship between $P(A \mid B)$ and the inverse probability $P(B \mid A)$

$$P(A \mid B) = \frac{P(B|A)P(A)}{P(B)}$$

- Most important decision rule: Use the highest probability

Agenda

Theory (60 min) → Break (15 min) → Exercise (120 min)

L03.1 (2 min)
- Types of Machine Learning
- What is clustering

L03.2 (3 min)
- Types of cluster

L03.3 (20 min)
- Hierarchical Agglomerative Clustering
- Divisive Clustering

L03.4 (20 min)
- k-Means

L03.5 (15 min)
- DBSCAN

HA02.4 (20 min)
- Data visualization
- Gaussian & probabilities

E03.1 (15 min)
- HAC
- Dendrogram

E03.2 (15 min)
- DBSCAN

E03.3 (70 min)
- k-Means
- Cost function

L03.1 Types of Machine Learning

Supervised learning

- Learns by using labeled data
- Used for regression and classification
- With the aim to calculate outcomes and classify objects
- Real-world use cases such as risk evaluation and object detection on images

Unsupervised learning

- Learns by using unlabeled data without any guidance
- Used for knowledge discovery and pattern recognition
- With the aim to discover underlaying patterns
- Real-world use cases such as anomaly detection and cluster analytics

Reinforcement learning

- Interacts with the environment (Markov decision problem) and tries to maximize a rewards
- With the aim to choose the best action for a given state
- Real-world use cases given in NLP (natural language processing), supply chain optimization and traffic control

What is clustering?

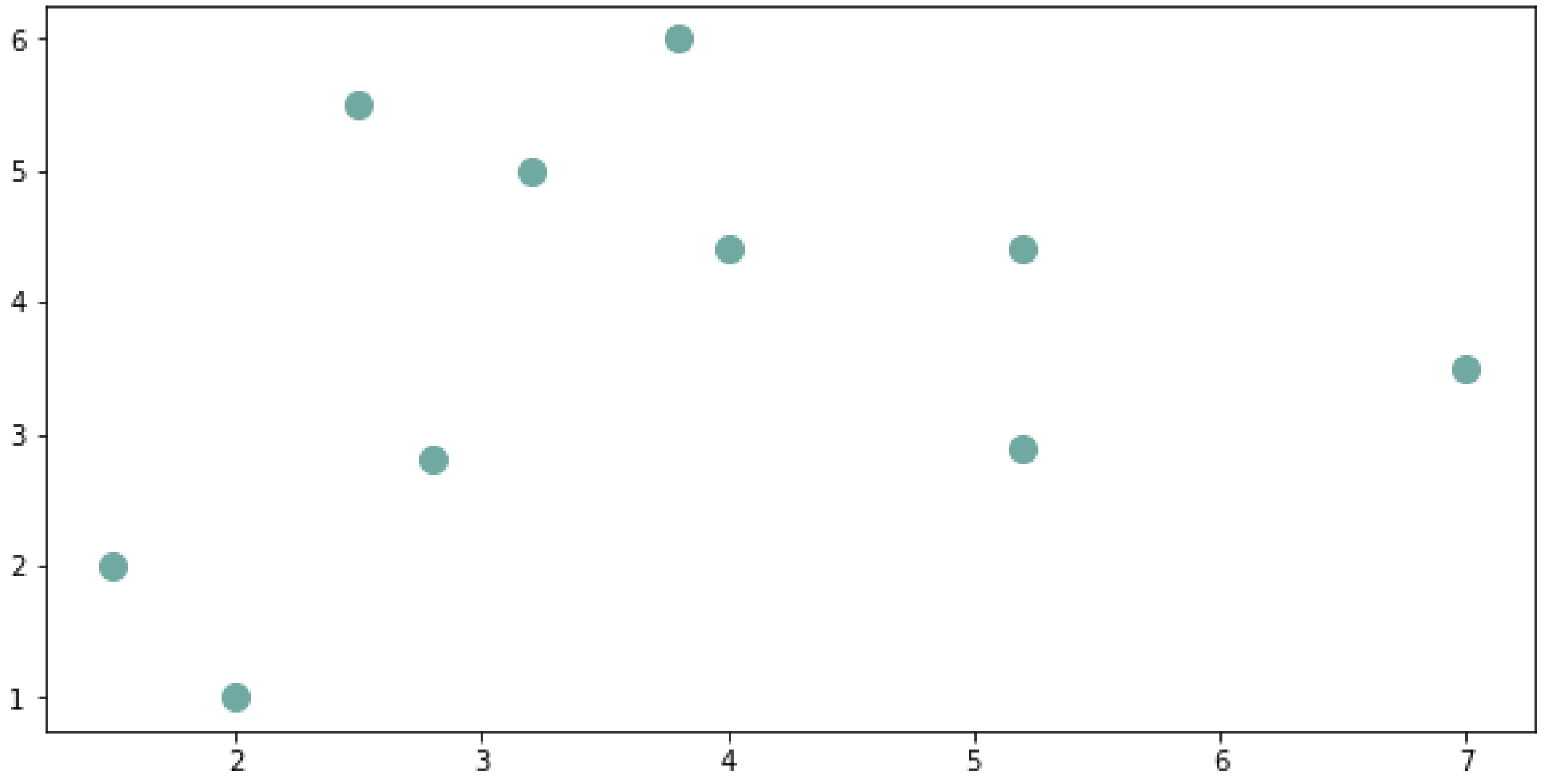
- It is about to find similar objects within a data set (clusters) and merge them together (clustering)
- The most important methodology in KDD (**k**nowledge **d**iscovery in (large) **d**atabases) is clustering

- To estimate the class conditional densities, we need a labeled data set
- Under certain circumstances the name and the exact number of classes is not known at all not known, thus:

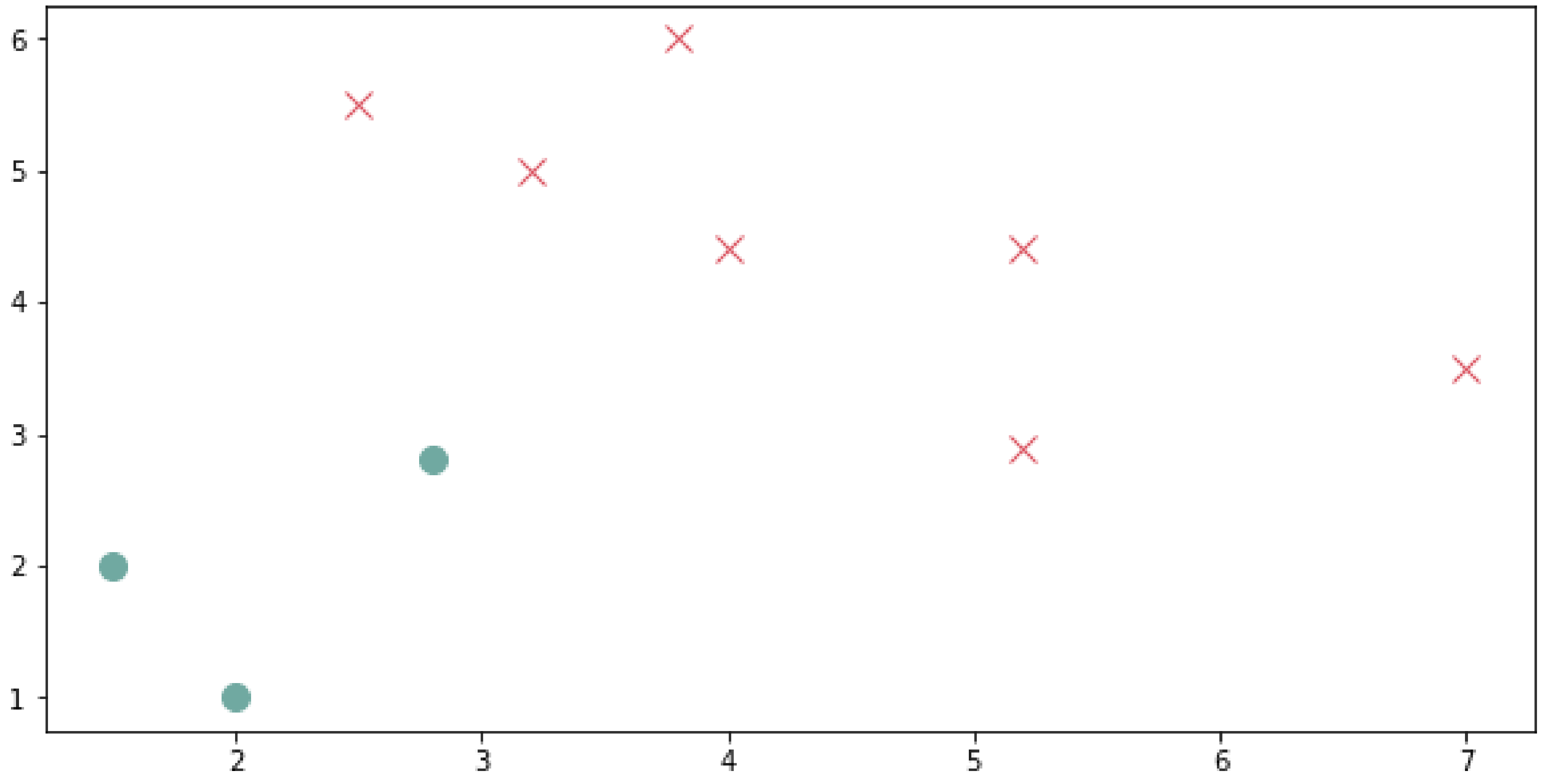
we want to have a classifier that teaches itself

- Clustering refers to procedures for discovering similarity features in (large) data sets

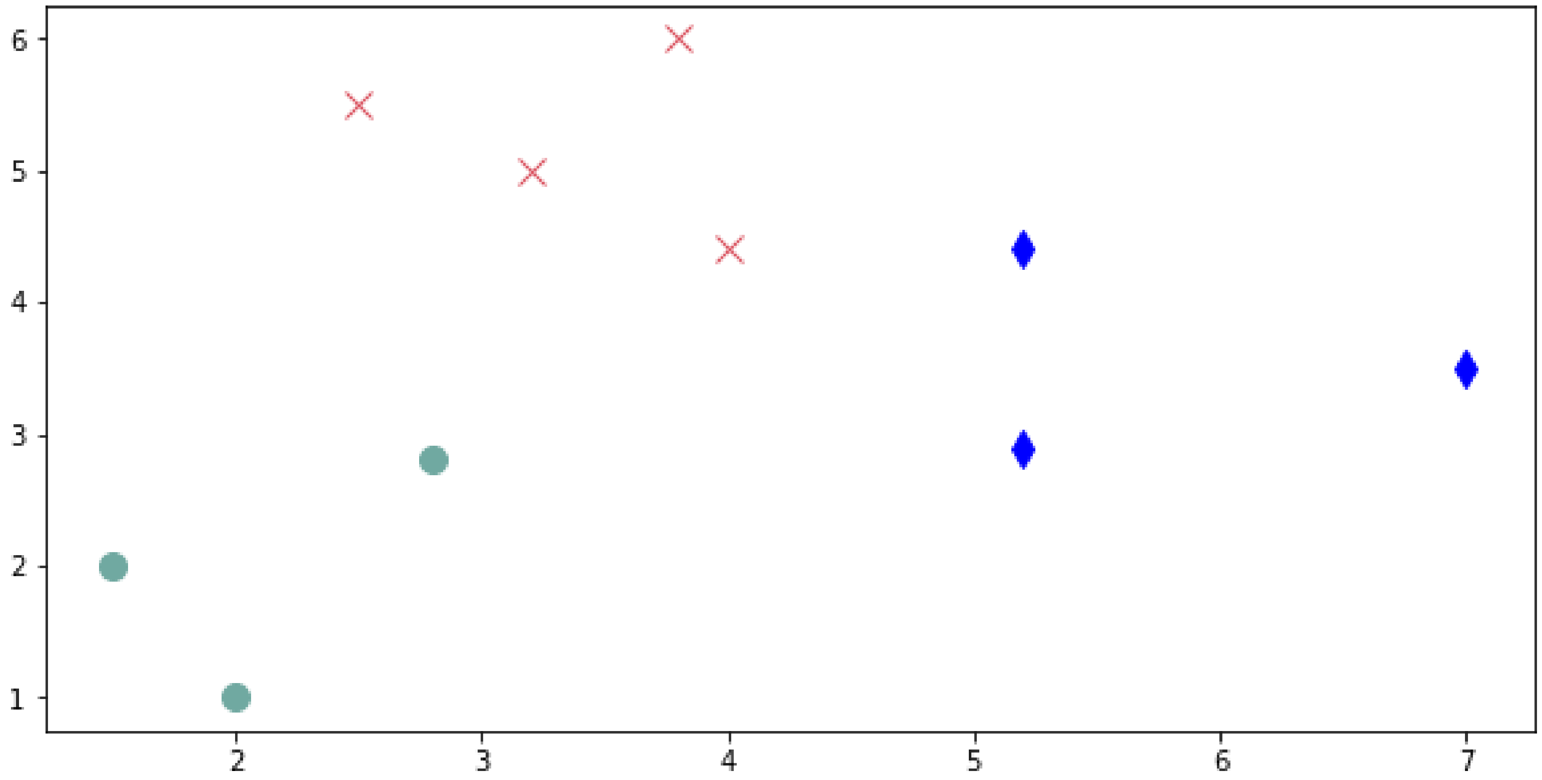
How many clusters do we have?



How many clusters do we have?



How many clusters do we have?



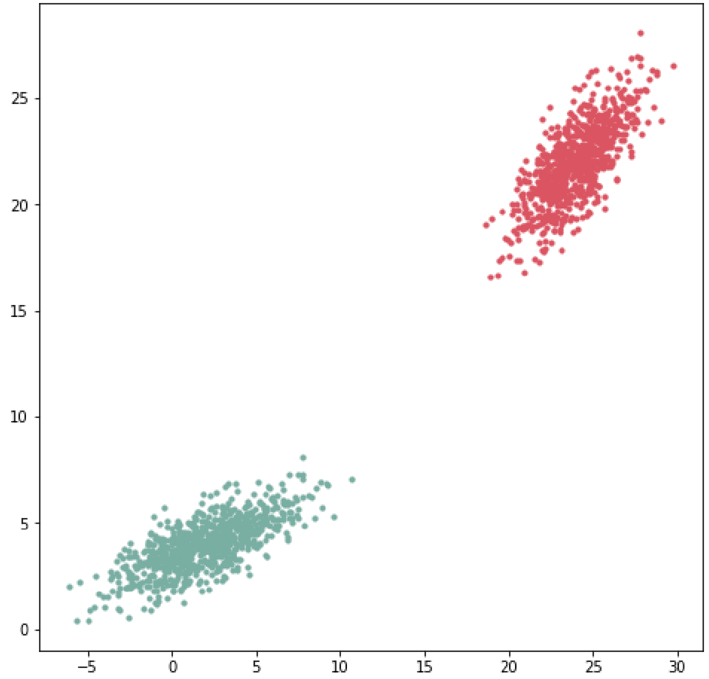
L03.2 Types of Clusters

Different types of clusters occur in real data sets

1. Well-separated clusters
2. Center-based clusters
3. Contiguous clusters
4. Density-based clusters
5. Property/Conceptual-based clusters
6. Functional described clusters

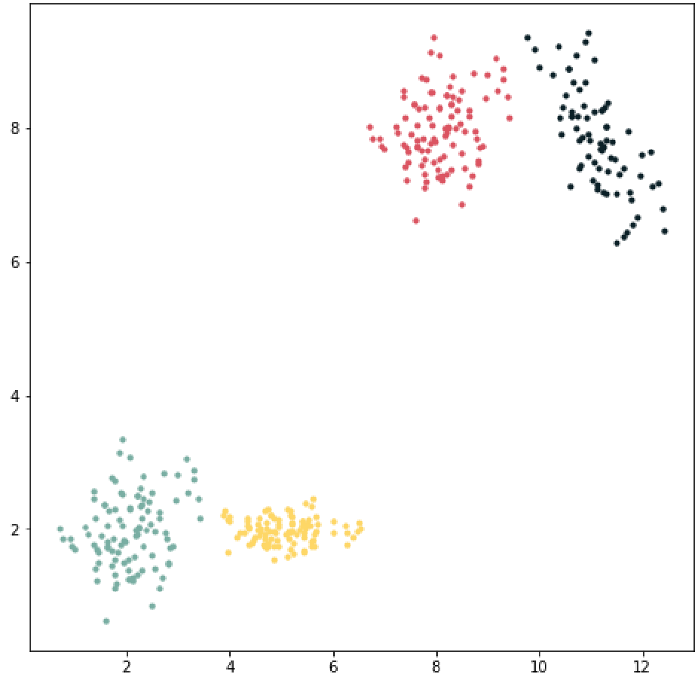
Well-separated clusters

- A cluster is a group of points where each point in a cluster is closer than to any point not in the cluster



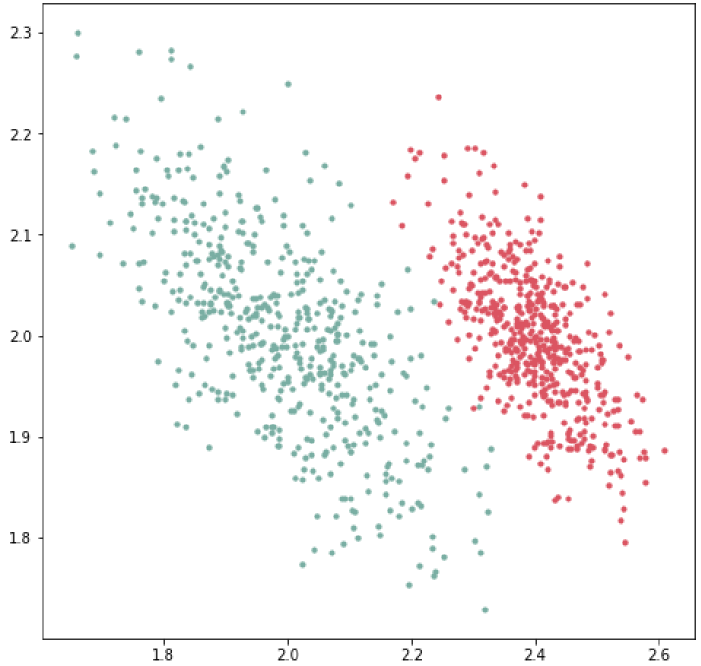
Center-based Clusters

- A cluster is a set of objects where one object in a cluster is closer to the "center" of a cluster than to the center of another cluster
- The center of the cluster is often a centroid or a medoid, the most representative point of the cluster



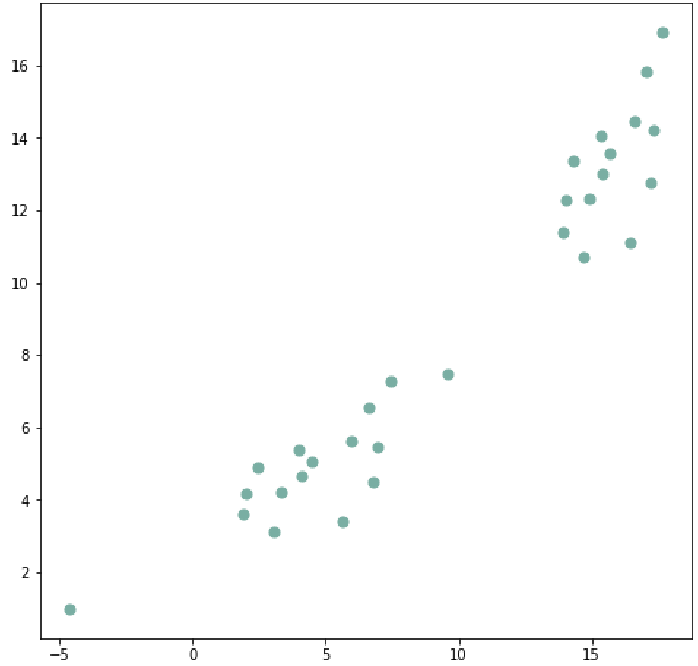
Contiguous Clusters

- A cluster is a set of objects where one object in a cluster is closer to the "center" of a cluster than to the center of another cluster



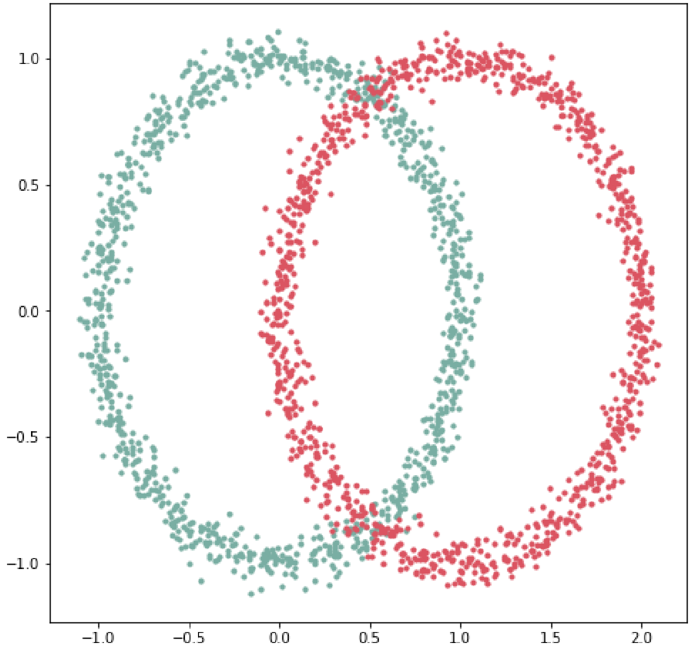
Density-based Clusters

- A cluster is a dense region of points which can be separated by low-density regions
- People often cluster through this approach



Property/Conceptual based Clusters

- A cluster represents a common property or a particular concept



Functional described clusters

- Finding clusters by minimizing or maximizing an objective function
  - Objective function could be the distance of each point to the mean of the cluster
- We can have global or local objectives
  - Hierarchical algorithms mostly use local objectives (distance between two consecutive points)
  - Partitional algorithms typically use global objectives (e. g. distance between one point and the mean)
- Mapping the cluster in different domain to solve the problem more easy
  - Transforming the points of a cluster from Cartesian coordinates into Polar coordinates
  - Create a proximity matrix to represent the distances of each point for further decision making
- The global objective function is often used to fit the data to a parameterized model

We distinguish two main groups of clustering

1. Hierarchical Clustering
   - Hierarchical clustered data, organized in so-called hierarchical tree by means of
     - Agglomerative (HAC)
     - Divisive

2. Partitional Clustering
   - Organization of large data objects into subsets (clusters) in such a way that each data object is in exactly one subset by means of
     - k-Means (and its variants)
     - Density based clustering (DBSCAN)

3. Other methods such as Fuzzy-clustering also available

L03.3 Hierarchical Agglomerative Clustering

- Hierarchical Agglomerative Clustering (HCA) is in iterative bottom-up approach
- It is also known as Agglomerative Nesting (AGNES)

- Each data point in the set is initially considered as a single cluster
- At each iteration, similar clusters are merged with other clusters until all elements belong to one cluster
- Key operation is the computation of the proximity of two clusters

- No a-priori knowledge required
- HCA is good to extract small clusters

- Dendrograms are used to visualize the correlations between all clusters

Algorithm principles

1. Assign each data point a unique cluster (id)
2. Determine the so-called proximity matrix (distance matrix) by determining the similarity between the individual data points
   - Euclidian in *n*-dimensions
   - Manhattan distance
   - …
3. Link the closest two points the same cluster ID
4. Repeat till one cluster results

Algorithm 1 Hierarchical Agglomerative Clustering

Require: $\mathbf{x} \leftarrow$ data set
Require: $\mathbf{c} \leftarrow$ clusters
Require: $\mathbf{\Omega} \leftarrow \{\mathbf{x}, \mathbf{c}\}$
Ensure: $N_c = \mathrm{len}(\mathbf{x})$ ▷ Let each point be a cluster

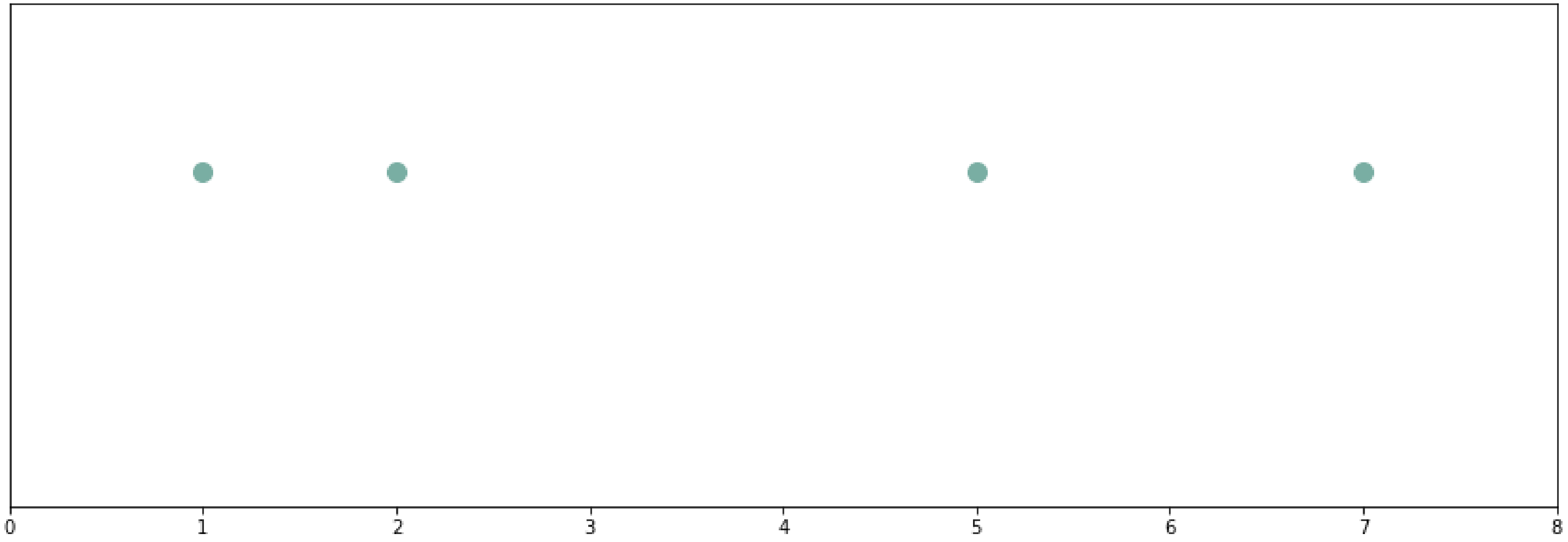
$D \leftarrow$ distance_matrix($\mathbf{\Omega}$) ▷ Compute the distance matrix
$i = 0$
while $N_c \neq 1$ **do**
  $\mathbf{\Omega} \leftarrow \{\{x|_{\arg\min[D_i]} + x|_{\arg\min[D_{i+1}]}\}, i\}$ ▷ Merge the two closest clusters
  $D \leftarrow$ distance_matrix($\mathbf{\Omega}$) ▷ Update distance matrix
  $i = i + 1$
end while

Linkage methods

- Maximum or complete linkage: Distance between two clusters as maximum value of all pairwise distances between the elements in cluster 1 and the elements in cluster 2. It tends to produce more compact clusters
- Minimum or single linkage: Distance between two clusters is the minimum value of all pairwise distances between the elements in cluster 1 and the elements in cluster 2. It tends to produce long, “loose” clusters
- Mean or average linkage: Distance between two clusters is defined as the average distance between the elements in cluster 1 and the elements in cluster 2
- Centroid linkage: Distance between two clusters is defined as the distance between the centroids of each cluster
- Ward method: It minimizes the total within-cluster variance

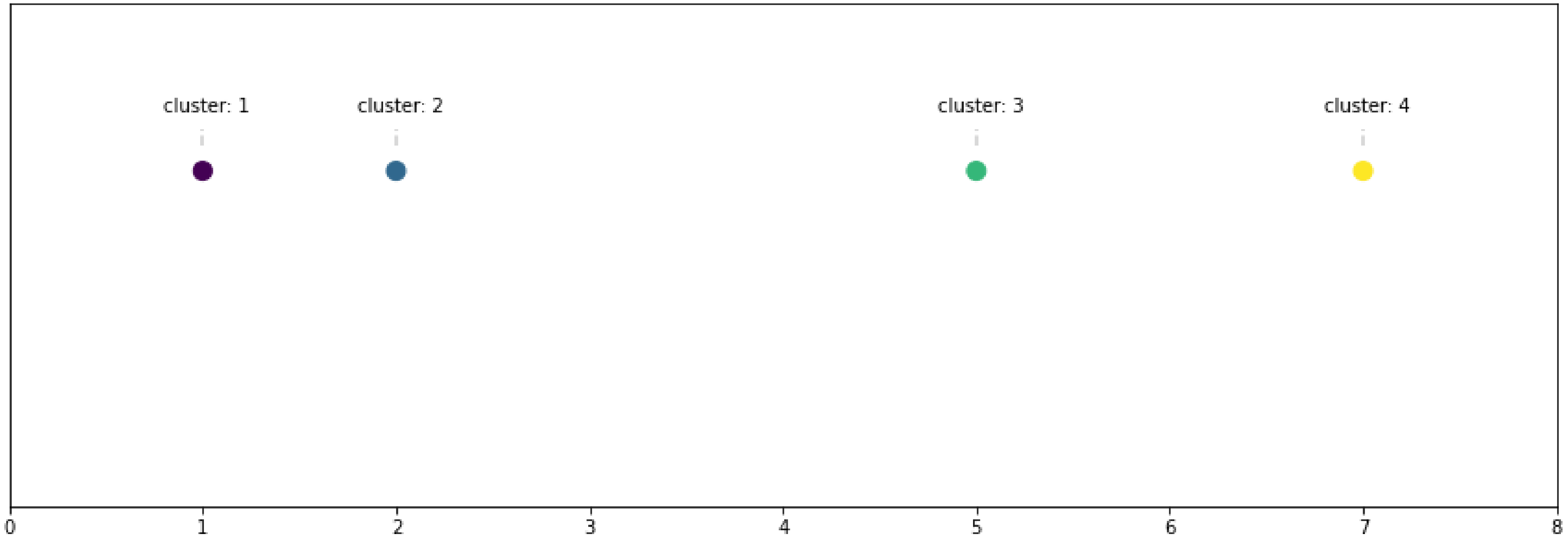
HAC example

$$\mathbf{X} = \begin{bmatrix} 1, 2, 5, 7 \\ 0, 0, 0, 0 \end{bmatrix} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \qquad \mathbf{c} = \{1,\ 2,\ 3,\ 4\}$$

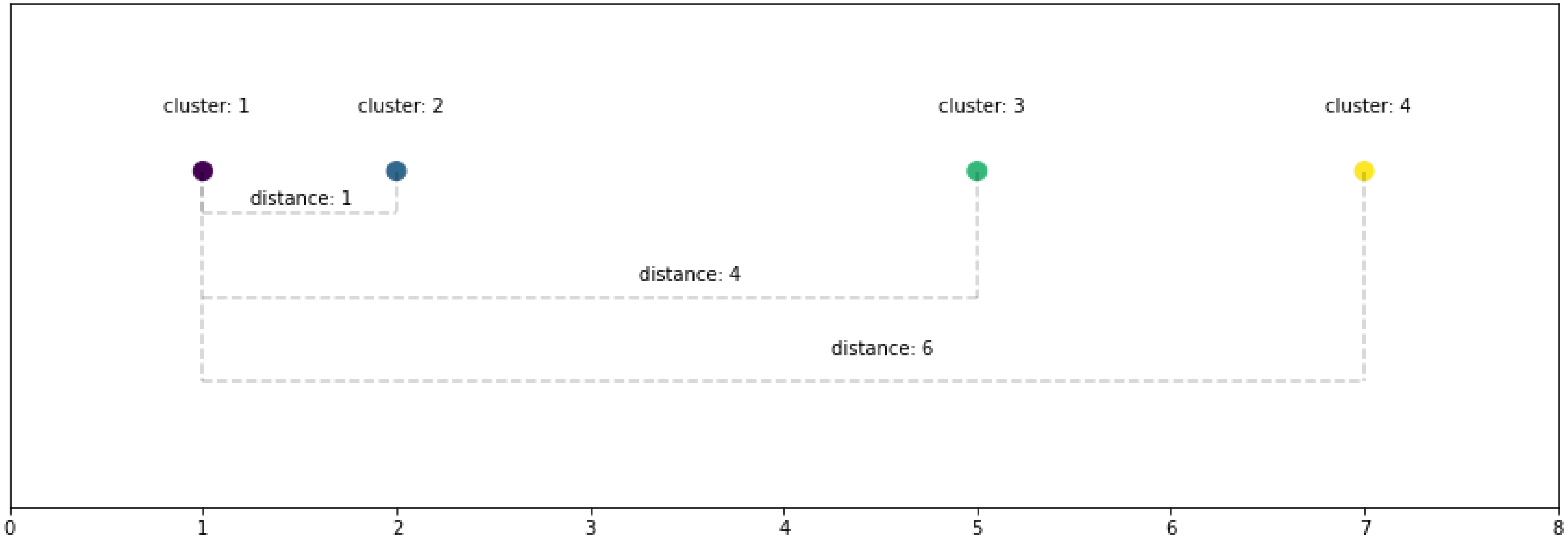


$n_c = 4$

Initial assumptions



Proximity matrix $\boldsymbol{D}$



Proximity matrix $\boldsymbol{D}$

$$
\mathbf{D} = \begin{array}{c|c|cccc}
\text{cluster} & & 1 & 2 & 3 & 4 \\
 & x & 1 & 2 & 5 & 7 \\
\hline
1 & 1 & 0 & 1 & 4 & 6 \\
2 & 2 & 1 & 0 & 3 & 5 \\
3 & 5 & 4 & 3 & 0 & 2 \\
4 & 7 & 6 & 5 & 2 & 0
\end{array}
\rightarrow
\begin{array}{c|c|cccc}
\text{cluster} & & 1 & 2 & 3 & 4 \\
 & x & 1 & 2 & 5 & 7 \\
\hline
1 & 1 & 0 & 1 & 4 & 6 \\
2 & 2 & 0 & 0 & 3 & 5 \\
3 & 5 & 0 & 0 & 0 & 2 \\
4 & 7 & 0 & 0 & 0 & 0
\end{array}
\Longrightarrow
$$

```python
def prox_mtx(x):
    return np.triu(np.abs(x[..., np.newaxis] - x)).flatten()
```

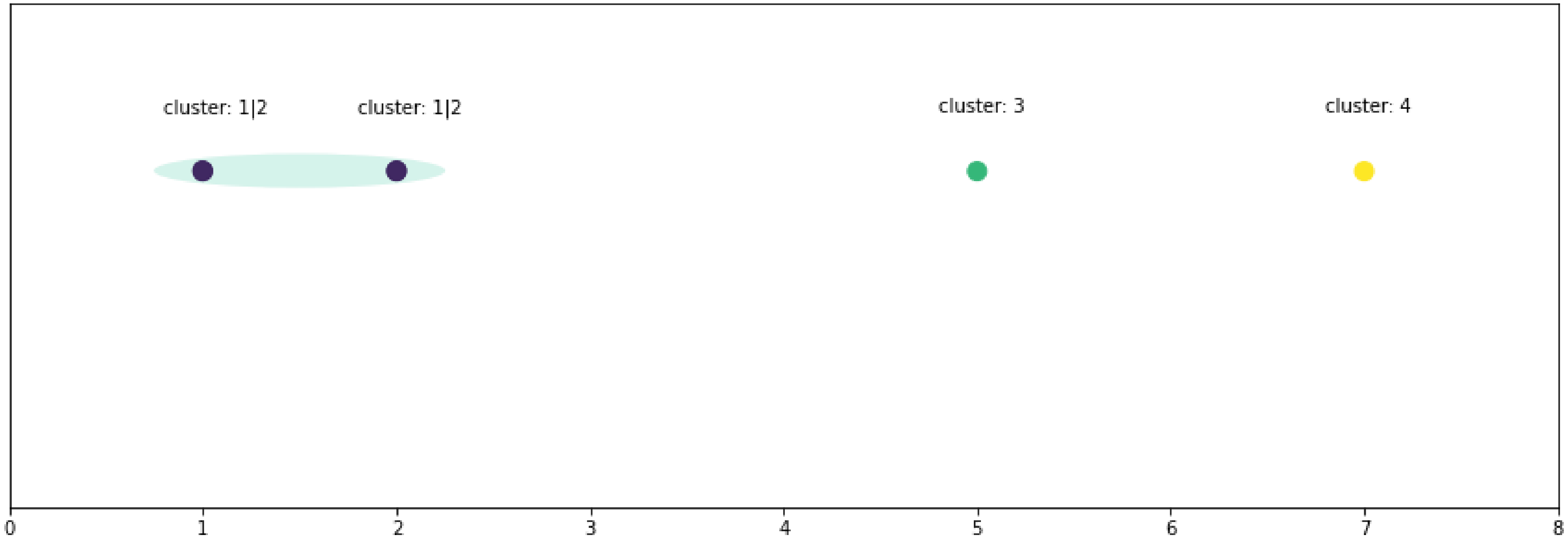
$$d(\ \ ,\ \ ) =$$

$$\min_{\mathbf{D}} =$$

$$\operatorname{argmin}_{\mathbf{D}} =$$

$n_c = 3$

Merging closest clusters (Linkage)



Update proximity matrix $\boldsymbol{D}$

- To obtain the new distance matrix, we need to remove the 1 and 2 entries and replace it by an entry for cluster “1|2"
- The proximity update depends on the linkage (centroid linkage)

- *Recap:* Distance between two clusters is defined as the distance between the centroids of each cluster
  - Cluster “1|2” consist in our case out of point 1 and point 2

$\mathbf{D} =$

cluster		1|2	1|2	3	4
	x	1	**2**	5	7
1|2	**1**	0	1	4	6
1|2	2	1	0	3	5
3	5	4	3	0	2
4	7	6	5	2	0

$\Longrightarrow$

$\mathbf{D}' =$

cluster		1|2	3	4
	x	1.5	5	7
1|2	1.5	0	3.5	5.5
3	5	0	0	2
4	7	0	0	0

$$x' = \frac{1+2}{2} = 1.5$$

$$d(1.5, 5) = 3.5$$

$$d(1.5, 7) = 5.5$$

Update proximity matrix $\boldsymbol{D}$

- To obtain the new distance matrix, we need to remove the 1 and 2 entries and replace it by an entry for cluster “1|2"
- The proximity update depends on the linkage (complete linkage)

- *Recap:* Complete linkage uses the maximum value of all pairwise distances between the elements in cluster “1|2” and the elements in the other clusters (3, 4)
  - Cluster 1 consist in our case out of point 1 and point 2

$$\mathbf{D} = \begin{array}{c|c|cccc} \text{cluster} & & 1|2 & 1|2 & 3 & 4 \\ & x & 1 & \mathbf{2} & 5 & 7 \\ \hline 1|2 & \mathbf{1} & 0 & 1 & 4 & 6 \\ 1|2 & 2 & 1 & 0 & 3 & 5 \\ 3 & 5 & 4 & 3 & 0 & 2 \\ 4 & 7 & 6 & 5 & 2 & 0 \end{array} \Longrightarrow \mathbf{D}' = \begin{array}{c|c|ccc} \text{cluster} & & 1|2 & 3 & 4 \\ & x & 1|2 & 5 & 7 \\ \hline 1|2 & 1|2 & 0 & 4 & 6 \\ 3 & 5 & 0 & 0 & 2 \\ 4 & 7 & 0 & 0 & 0 \end{array}$$

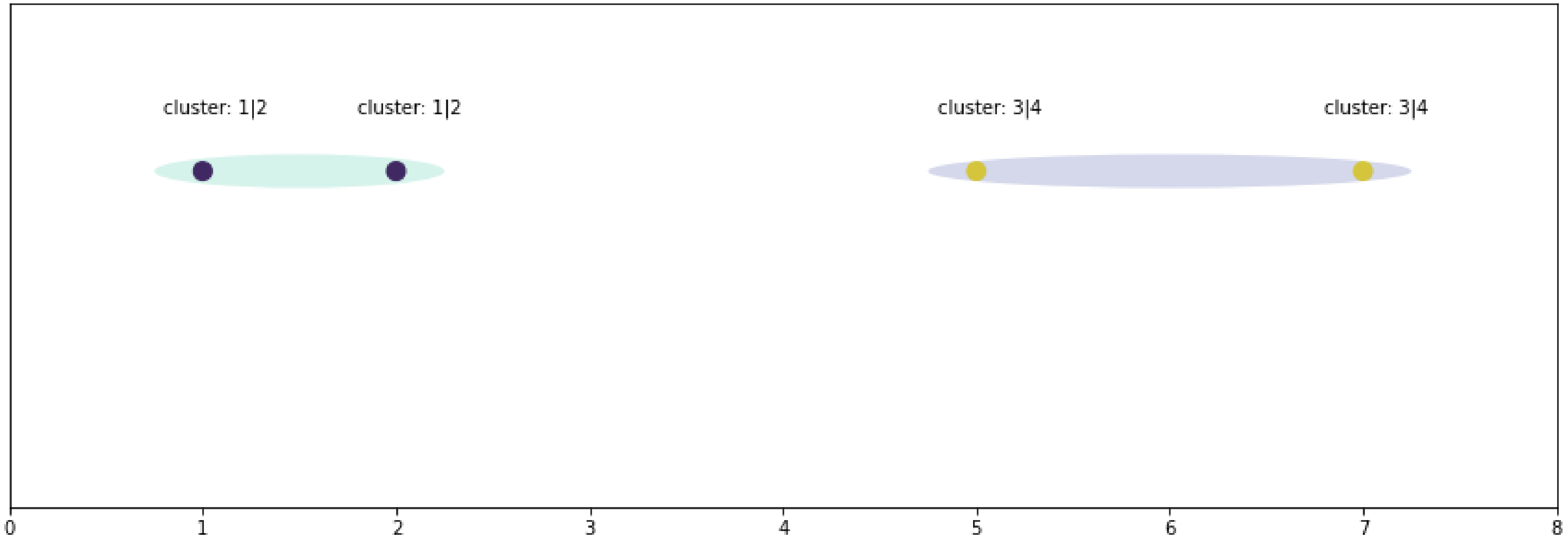
$$\max \begin{cases} d(1,5) = 4 \\ d(2,5) = 3 \end{cases}$$

$$\max \begin{cases} d(1,7) = 6 \\ d(2,7) = 5 \end{cases}$$

$n_c = 2$

Merging closest clusters

$$\min_{\mathbf{D}'} = 2$$



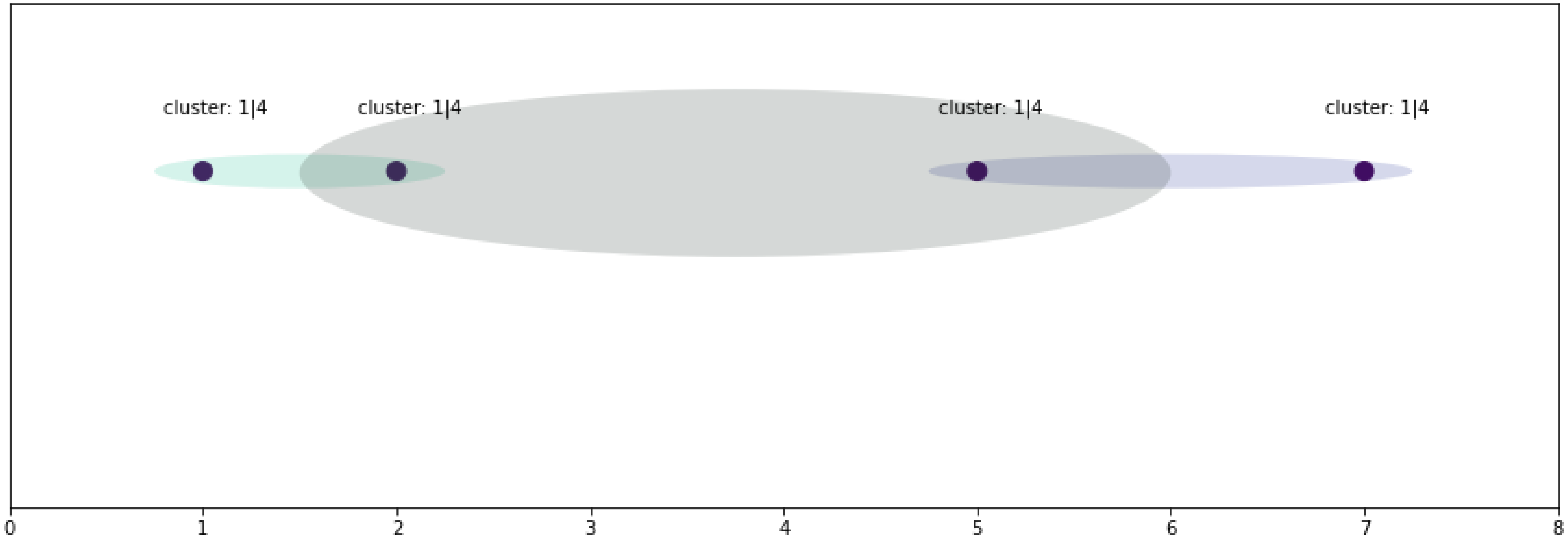
Update proximity matrix $\boldsymbol{D}$

$$\mathbf{D}' = \begin{array}{c|c|ccc} \text{cluster} & & 1|2 & 3 & 4 \\ & x & 1.5 & 5 & 7 \\ \hline 1|2 & 1.5 & 0 & 3.5 & 5.5 \\ 3 & 5 & 0 & 0 & 2 \\ 4 & 7 & 0 & 0 & 0 \end{array} \Longrightarrow \mathbf{D}'' = \begin{array}{c|c|cc} \text{cluster} & & 1|2 & 3|4 \\ & x & 1.5 & 6 \\ \hline 1|2 & 1.5 & 0 & 4.5 \\ 3|4 & 6 & 0 & 0 \end{array}$$

$n_c = 1$

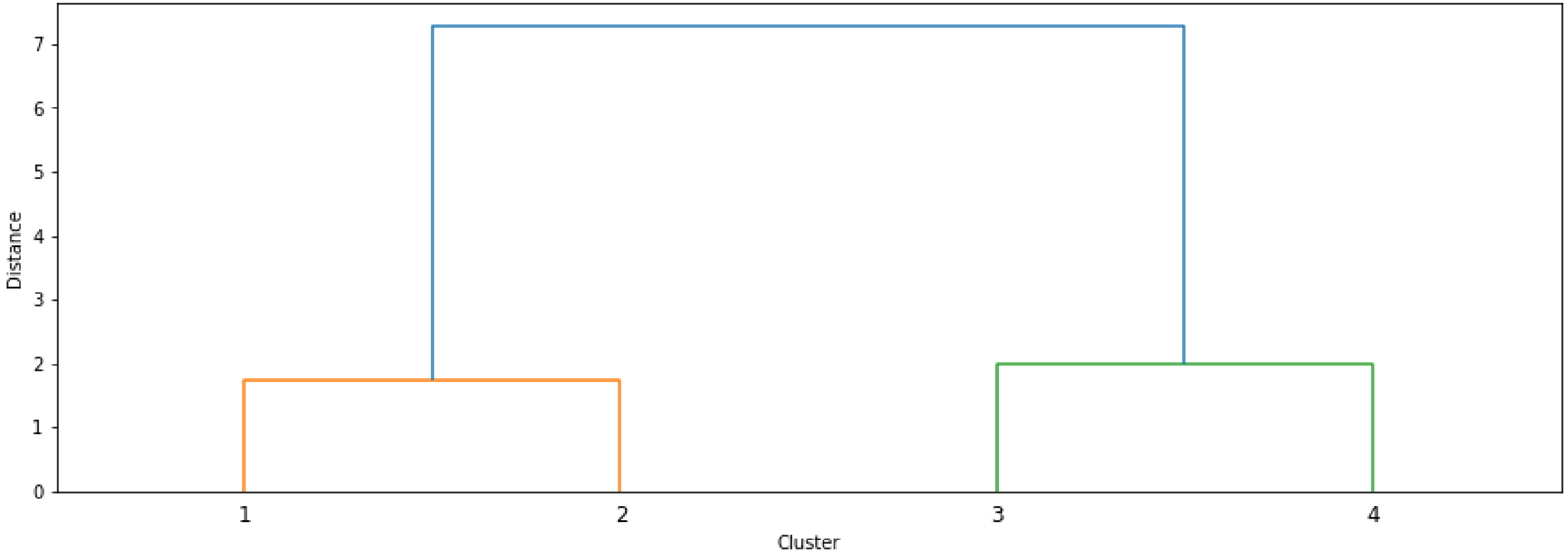
Merging closest clusters

$$\min_{\mathbf{D}'} = 4.5$$

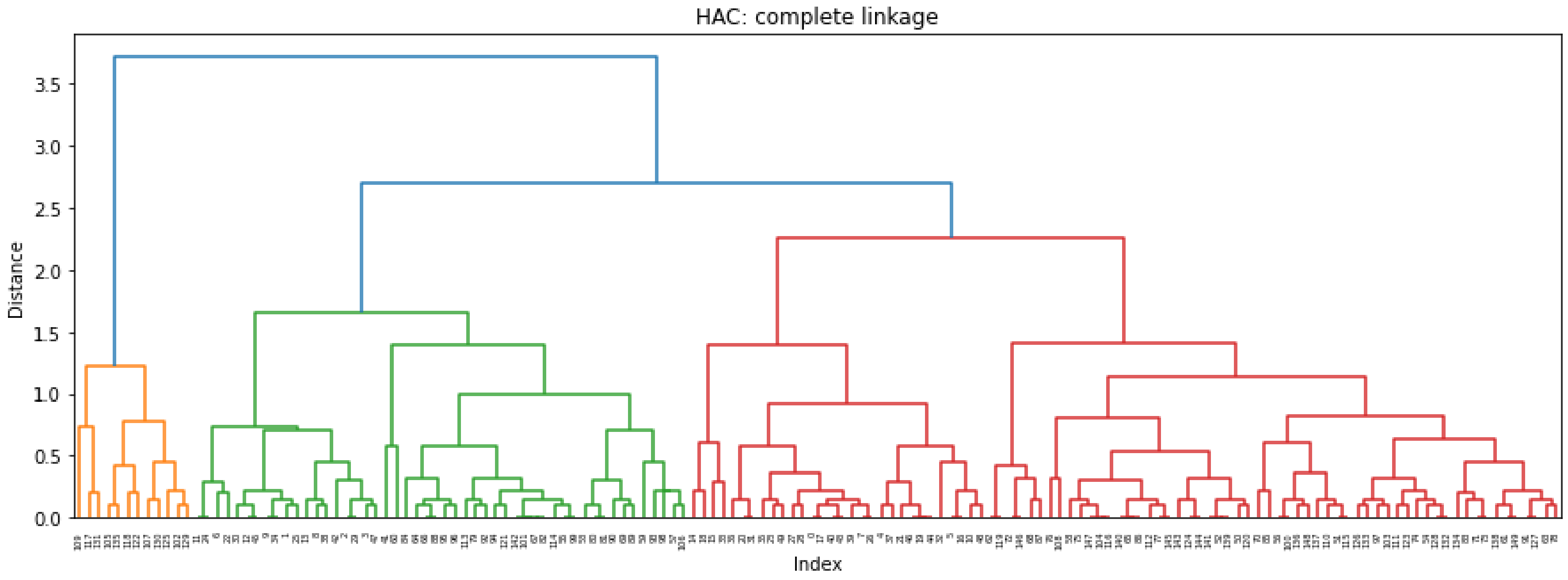


Dendrogram

```python
from scipy.cluster.hierarchy import dendrogram, linkage
from matplotlib import pyplot as plt

distances, _ = prox_mtx(data['x'])

linked = linkage(distances, 'complete')  # 'centroid', 'min', ...

fig, ax = plt.subplots(figsize=(15,5))
dendrogram(linked)
```



HAC on iris data set



We distinguish two main groups of clustering

1. Hierarchical Clustering
   - Hierarchical clustered data, organized in so-called hierarchical tree by means of
     - Agglomerative (HAC)
     - Divisive
2. Partitional Clustering
   - Organization of large data objects into subsets (clusters) in such a way that each data object is in exactly one subset by means of
     - k-Means (and its variants)
     - Density based clustering (DBSCAN)
3. Other methods such as Fuzzy-clustering also available

L03.3 Hierarchical Divisive Clustering

- Hierarchical Divisive Clustering (HDC) is in top-down approach
- It is also known as Divisive Analysis (DIANA)

- Each data point in the set is initially considered to one common cluster
- At each iteration, the cluster is partitioned into similar clusters until *n* defined clusters assigned
- Therefore, underlaying subroutine for flat clustering is required

- Compared to HAC, HDC is more accurate, since in HAC decisions are made considering local patterns or neighbor points without first considering the global distribution of the data
- These early decisions cannot be reversed, whereas in divisive clustering, the global distribution of the data is considered when top-level partitioning decisions are made
- Large computational effort if we do not restrict the clusters to be grouped together, thus:
- A-priori knowledge required

We distinguish two main groups of clustering

1. Hierarchical Clustering
   - Hierarchical clustered data, organized in so-called hierarchical tree by means of
     - Agglomerative (HAC)
     - Divisive
2. Partitional Clustering
   - Organization of large data objects into subsets (clusters) in such a way that each data object is in exactly one subset by means of
     - k-Means (and its variants)
     - Density based clustering (DBSCAN)
3. Other methods such as Fuzzy-clustering also available

L03.5 k-Means Clustering

- In *k*-Means Clustering, each cluster is represented with the centroid
- Each point in the data set is associated to cluster with the closest centroid
- *k* represents the number of clusters as a-priori knowledge

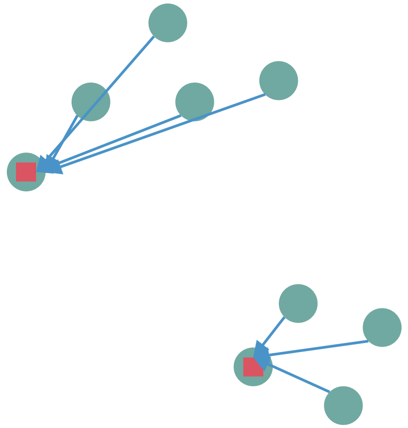
- The objective of this approach is to minimize the sum of the distances of the points to their respective centroid

<u>k-Means Objective Function:</u>

$$x_i^{(k)} = \arg\min_k ||x_i - \mu_k||^2 \; \forall i, k$$

Principle Algorithm

$k = 2$



Algorithm 1 k-Means
Require: Initialize k centroids
repeat
Minimize objective function
Update centroids
until Convergence

$$\mu_k = \frac{1}{n_k} \sum_{i \in S_k} x_i$$

$$S_k = \{i : x_i^{(k)}\}$$

$$n_k = |S_k|$$

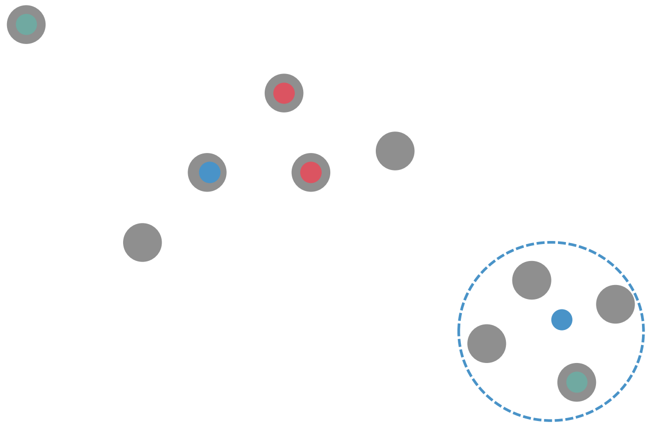
<u>k-Means cost function:</u>

$$C(x^{(k)}, \mu) = \sum_{i=0}^{n-1} ||x_i - \mu_{k_i}||^2$$

Two-step optimization:

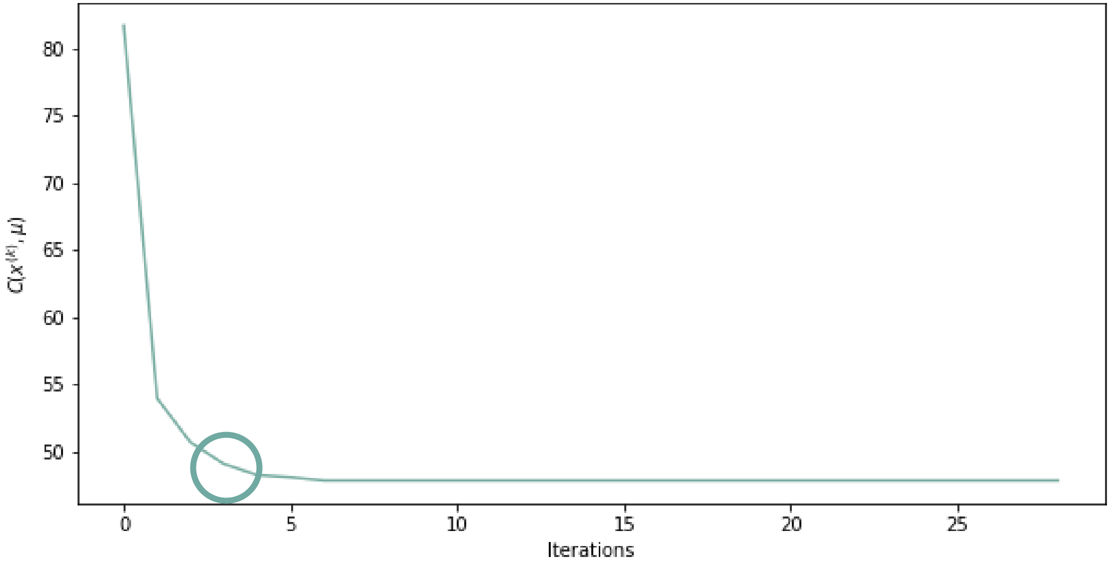
1. Over the cluster assignment $x^{(k)}$
2. Over the cluster centroids

Initialization methods



- Random
  - Choose a random data index
  - It is possible to select two neighbors accidentally
- Distance-based
  - Starting with one random point
  - Search for the nearest $k-1$ furthest points in the data set (more complex in terms of computational effort)
  - It is possible to select outliers
- Random + distance based (k-Means++ implementation) [ArtVass2007]
  - Starting with one selected point
  - Choose the next centroid by finding the furthest points combined with the probability, proportional to the squared distance

Choosing the number of clusters *k*



- Elbow-method
  - Using the "elbow" as an indicator of the number of parameters
  - Common approach in mathematical optimization to choose a point at which diminishing returns are no longer worth the additional cost

- Penalize for complexity
  - $Total = Error + Complexity$
  - Bayesian Information Criterion (BIC)
    - Interpret the error as likelihood of a multivariate Gaussian with fixed variance and unknown mean

<u>Bayesian Information Criterion:</u>

$$J(x^{(k)}, \mu) = log\left[\frac{1}{n}\sum_{i=0}^{n-1}||x_i - \mu_{k_i}||^2\right] + k\frac{logn}{n}$$

We distinguish two main groups of clustering

1. Hierarchical Clustering
    - Hierarchical clustered data, organized in so-called hierarchical tree by means of
        - Agglomerative (HAC)
        - Divisive
2. Partitional Clustering
    - Organization of large data objects into subsets (clusters) in such a way that each data object is in exactly one subset by means of
        - k-Means (and its variants)
        - Density based clustering (DBSCAN)
3. Other methods such as Fuzzy-clustering also available

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

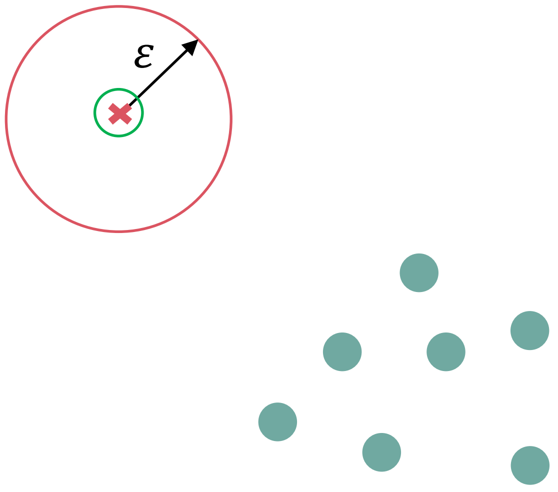
- One of the most used cluster algorithms in data mining
- Performs often better than k-Means
- Robust against outliers

- Two important parameters must be selected:
  - Radius ε
  - Minimal amount of points n_{min}

- Three different types of points are used:
  - Core point
  - Border point
  - Outlier

Principle Algorithm

$$\epsilon = 2$$
$$n_{min} = 3$$
$$i = 0$$



$n_c = 3, \geq n_{min}$
$n_c = 3, \geq n_{min}$
$n_c = 1, < n_{min}$
$n_c = 2, < n_{min}$
$n_c = 4, < n_{min}$

Algorithm 1 DBSCAN

Require: ϵ: Radius
Require: n_{min}: Density threshold
Require: $\mathbf{c} \in \mathbb{R}^{1\times n}$: Labels, initially undefined
Require: $\mathbf{X} \in \mathbb{R}^{2\times n}$: Data set
Require: $\mathbf{\Omega} : \{\mathbf{X}, \mathbf{c}\}$

```
for all x, idx in X do
    n_x ← FindPointsWithinRadius(x)                         ▷ Fast tree search
    if |n_x| = 1 then c(idx) = noise                        ▷ We have only the centroid
    else if n_x < n_min then c(idx) = border                ▷ We have a boarder point
    else c(idx) = next cluster label                        ▷ Core point
        c(idx) ← MergeClassesWithinCircle(c, idx)
    end if
end for
```

Class affiliation

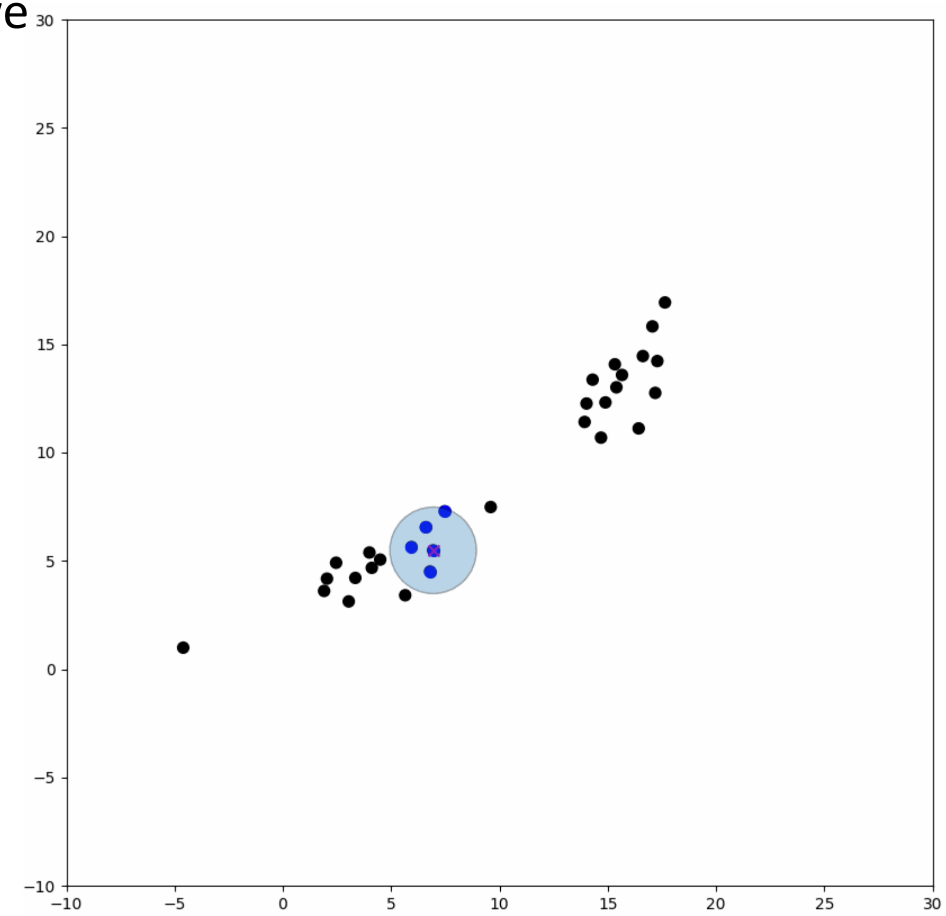
$$\epsilon = 2$$
$$n_{min} = 3$$

- Classification in core, border or noise points not sufficient, since we do not know how many different clusters included

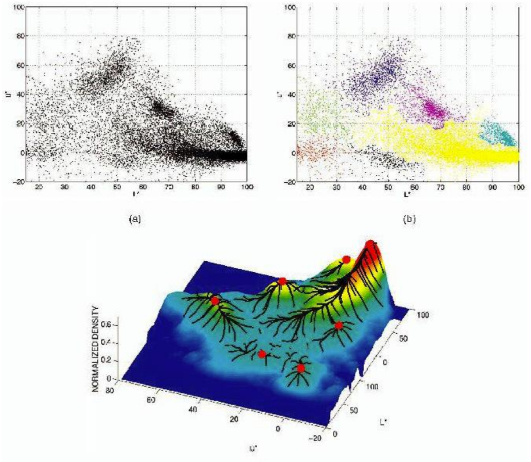
- We can assign the class affiliation while checking the point type

Algorithm 1 Class affiliation in DBSCAN

```
function MergeClassesWithinCircle(c, data set)
    for all x in circle do
        if any(x) has class ID then
            all(x) ← class ID
            Join all(x) with identical class ID in data set
        else
            all(x) ← new class ID
        end if
    end for
end function
```

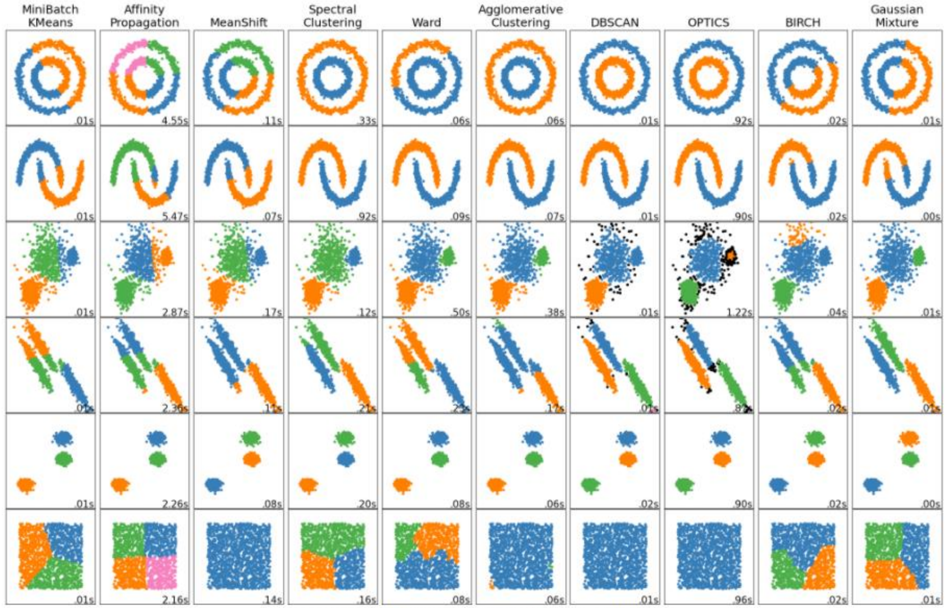


L03.7 Further Clustering



- Clustering algorithm are very common in image processing
- Segmentation (object extraction) by means of the so-called Mean-shift as for instance

- The best clustering algorithm depends on your data set
- [sklearn]() is a very powerful and well documented Python package for ML, especially for clustering



A comparison of the clustering algorithms in scikit-learn

Break

E02.1

HA02.4 – Home assignment.iypnb

IFM

Institut für Fahrerassistenz und vernetzte Mobilität

Hochschule Kempten University of Applied Sciences

www.hs-kempten.de/ifm