

# Clustering Algorithms

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

Summer term 2025



# Lecture Overview

- I Machine Learning Introduction
- II Optimization Techniques
- III Bayesian Decision Theory
- IV Non-parametric Density Estimation
- V Probabilistic Graphical Models
- VI Linear Regression
- VII Logistic Regression
- VIII Deep Learning
- IX Evaluation
- X Decision Trees
- XI Support Vector Machines
- XII Clustering
- XIII Principal Component Analysis
- XIV Reinforcement Learning
- XV Advanced Regression

# Agenda for this Unit

① KMeans

② Hierarchical Clustering

③ DBSCAN

④ Mean-Shift Clustering

⑤ Wrap-Up

## Section: **KMeans**

- Introduction to Clustering and Clustering Strategies
- Introduction to KMeans
- KMeans Algorithm
- Use Case: Image Compression
- Problems of KMeans and Solutions

# Clustering Introduction

- **Clustering** belongs to the category of **unsupervised learning**
- A clustering algorithm tries to **find structure** in the data
- Once the clusters are found, they first have to be interpreted...
- ...and can then be used for prediction purposes

A cluster should be **internally homogeneous**, but simultaneously **externally heterogeneous**, i. e. elements of one cluster should be similar to each other, but should differ significantly from elements belonging to other clusters.

# Example Use Cases for Clustering

- **Behavioral segmentation**
  - Customer segmentation (e.g. **sinus milieus**)
  - Creating profiles based on activity monitoring
- **Sorting sensor measurements**
  - Image grouping
  - Detection of activity types in motion sensors
- **Inventory categorization**
  - Grouping inventory by sales activity
  - Grouping inventory by manufacturing metrics

# Clustering Strategies

There are different types of clustering algorithms. Most prominent are:

- ① **EM-based clustering**, e. g.: KMeans
- ② **Hierarchical clustering**, e. g.: agglomerative clustering, divisive clustering
- ③ **Affinity-based/Density-based clustering**,  
e. g.: DBSCAN, mean-shift clustering, spectral clustering

# KMeans: Procedure

- The algorithm is an instance of **vector quantization**
  - It represents data points by a single vector (**centroid**) which is close to them
  - This is useful for **data compression!**
- **How to:** Create  $K$  partitions (i. e. clusters)  $\mathbf{X}_k$  ( $1 \leq k \leq K$ ) of the dataset  $\mathbf{X}$ , such that the sum of squared deviations from the cluster centroids is **minimal**:

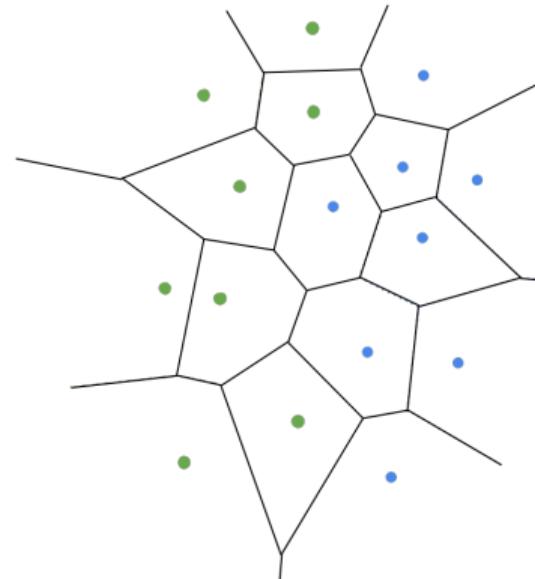
$$\min_{\mu^k, \mathbf{X}_k} \sum_{k=1}^K \sum_{\mathbf{x} \in \mathbf{X}_k} \|\mathbf{x} - \mu^k\|^2 \quad (1)$$

- $\mathbf{X}_k$  is the  $k$ -th cluster / partition and  $\mu^k$  its centroid

# Result: Voronoi Diagram

## Voronoi diagram:

- The dots represent cluster centroids
- The lines visualize the **cluster boundaries**
- For a new data point we can easily determine which cluster it has to be assigned to



# KMeans Algorithm

**Input:**  $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N\}$ , number of clusters  $K$

- 1 Set  $t \leftarrow 0$  and randomly choose  $K$  means (cluster centroids)  $\mu^{1,t}, \mu^{2,t}, \dots, \mu^{K,t}$
- 2 **while** *not converged* **do**
- 3     **E-step:** Assign each  $\mathbf{x} \in \mathbf{X}$  to the closest cluster:  
$$\mathbf{X}_{k,t} \leftarrow \left\{ \mathbf{x} \in \mathbf{X} : \|\mathbf{x} - \mu^{k,t}\|^2 \leq \|\mathbf{x} - \mu^{j,t}\|^2 ; \forall j \in \{1, 2, \dots, K\} \setminus \{k\} \right\}$$
- 4     **M-step:** Update all cluster centroids  $\mu^{k,t}$ :  
$$\mu^{k,t+1} \leftarrow \frac{1}{|\mathbf{X}_{k,t}|} \sum_{\mathbf{x} \in \mathbf{X}_{k,t}} \mathbf{x}$$
- 5      $t \leftarrow t + 1$
- 6 **end**

# KMeans Algorithm (Ctd.)

- The algorithm **might need some iterations** until the result is satisfactory
- **Caveat:** The algorithm **might get stuck in local optima**, i. e. several restarts might be required

# Use Case: Image Compression

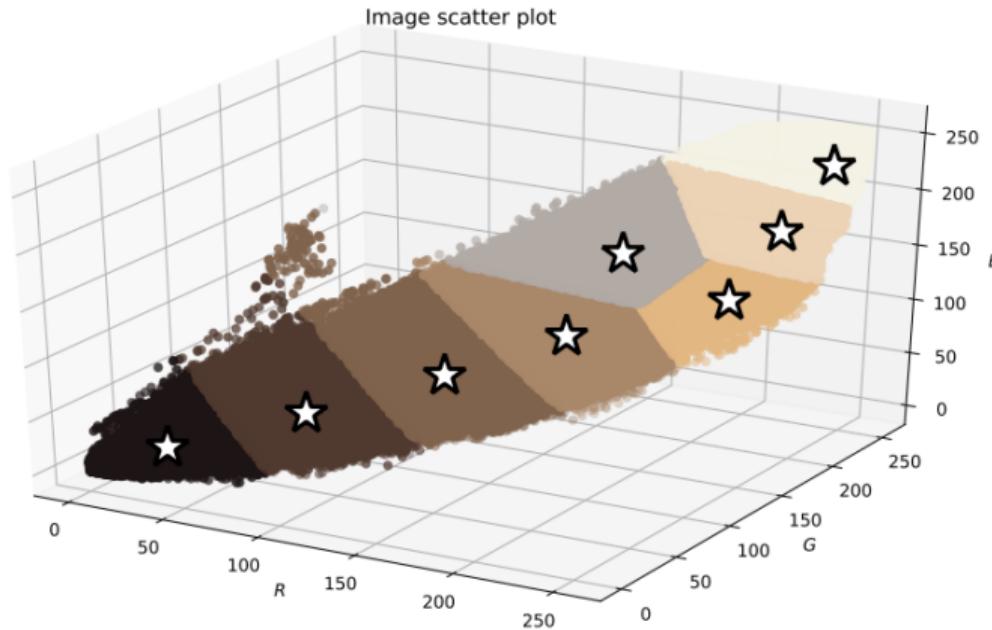
Original image



Compressed image



# Use Case: Image Compression (Ctd.)



# KMeans Issues

The algorithm has some downsides:

- ① Several restarts might be required as the algorithm **can get stuck in local optima** (*computation can turn out to be expensive*)
- ② The algorithm **assumes all clusters to be spherical** (*remedies?*)
- ③ It does **not have a notion of outliers** (*unlike DBSCAN*)
- ④ **Choice of  $K$ :** What is the correct value for  $K$ ?

**But:** The algorithm is simple and intuitive

# Choice of $K$

## What is the correct value for $K$ ?

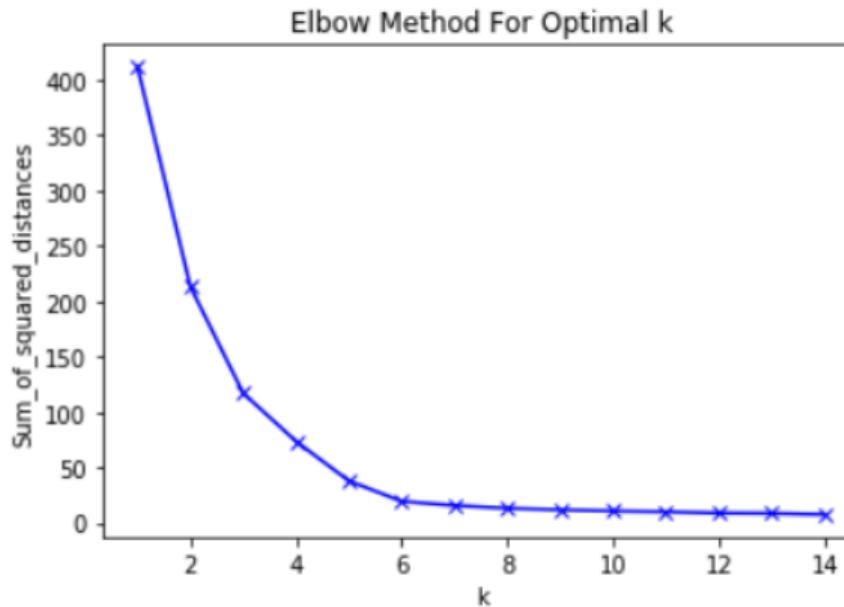
① We can use the **Elbow-method**:

- Compute the average of the sum of squared distances from all data points to their cluster centers for different values of  $K$
- Create a line plot based on the result
- Search for the **elbow point**

② Remember the **BAYESian Information Criterion (BIC)**?

- Try out several values for  $K$  and record the BIC values
- Choose  $K$  which minimizes the BIC

# Elbow Method



## Section: Hierarchical Clustering

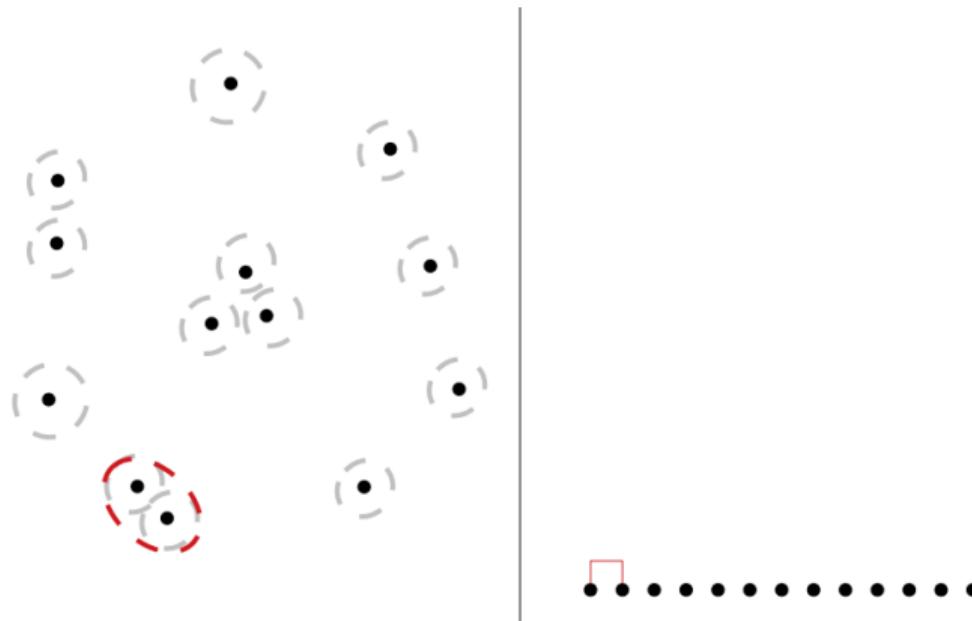
Agglomerative Clustering Algorithm  
Agglomerative Clustering: Example  
Distance Metrics between Clusters

# Agglomerative Clustering Algorithm

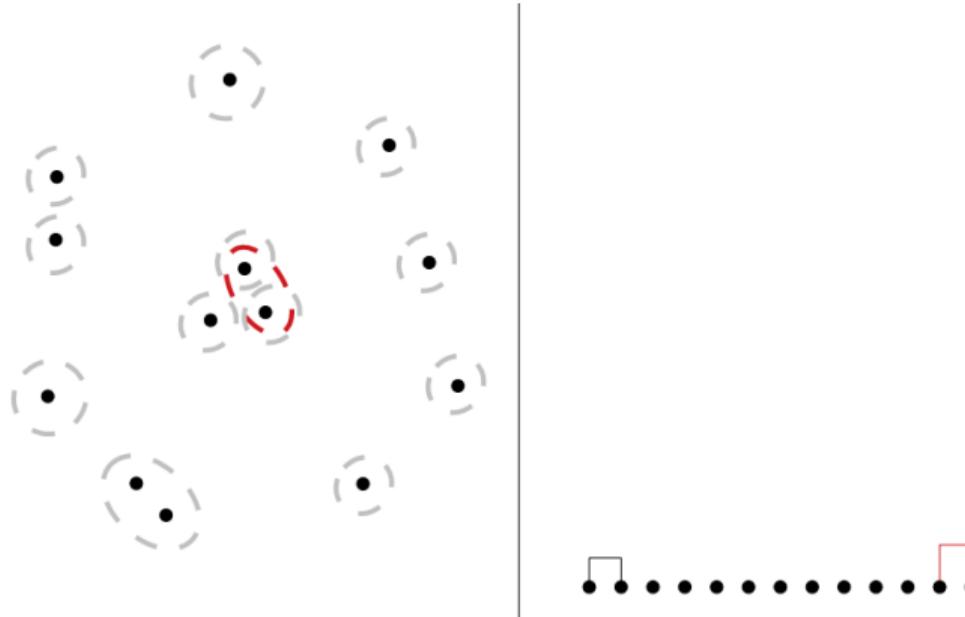
**Input:**  $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N\}$ , distance metric  $d$ , number of clusters  $K$

- 1 Start with one cluster for each instance:  $\mathfrak{C} := \{\{\mathbf{x}\} : \mathbf{x} \in \mathbf{X}\}$
- 2 **while**  $|\mathfrak{C}| > K$  **do**
- 3     Compute the distance  $d(C_i, C_j)$  between all pairs of clusters  $C_i, C_j \in \mathfrak{C}$
- 4     Merge the clusters  $C_i$  and  $C_j$  with minimum distance into a new cluster  $\tilde{C}$ :  
$$\tilde{C} := \{C_i, C_j\}$$
$$\mathfrak{C} \leftarrow (\mathfrak{C} \setminus \{\{C_i\}, \{C_j\}\}) \cup \{\tilde{C}\}$$
- 5 **end**
- 6 **return** the set of clusters  $\mathfrak{C}$

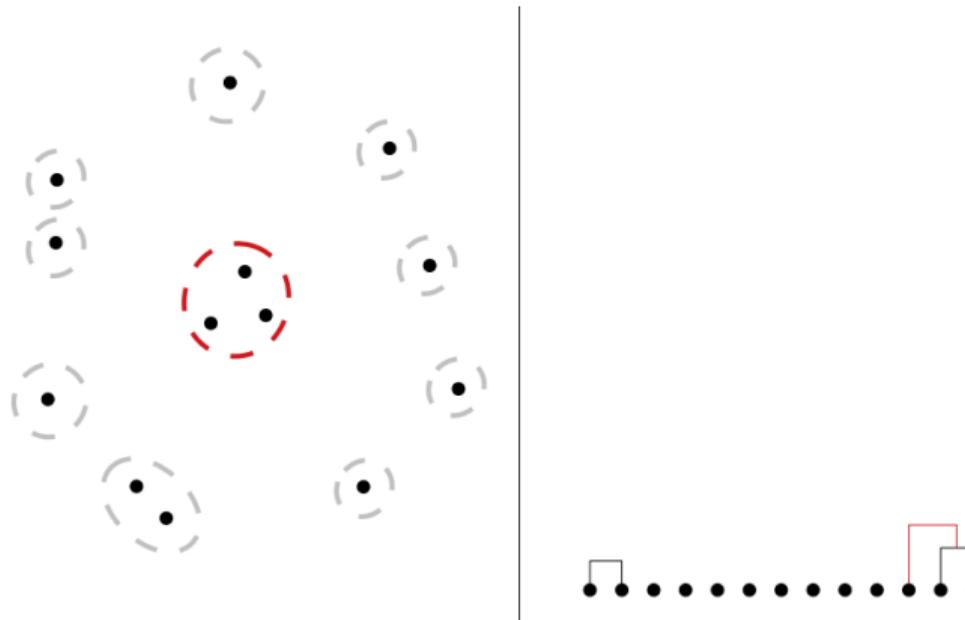
# Agglomerative Clustering: Example



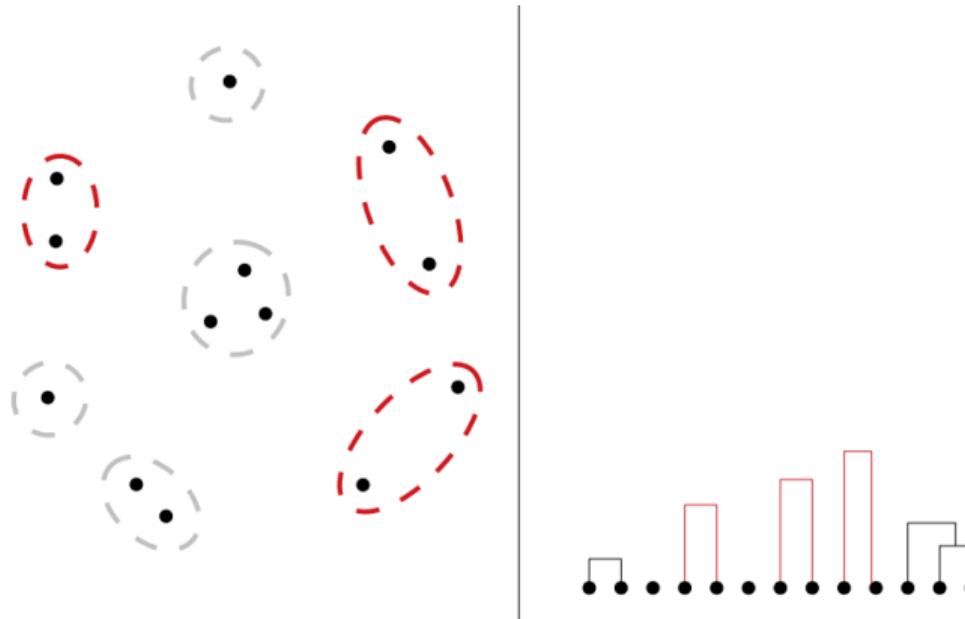
# Agglomerative Clustering: Example (Ctd.)



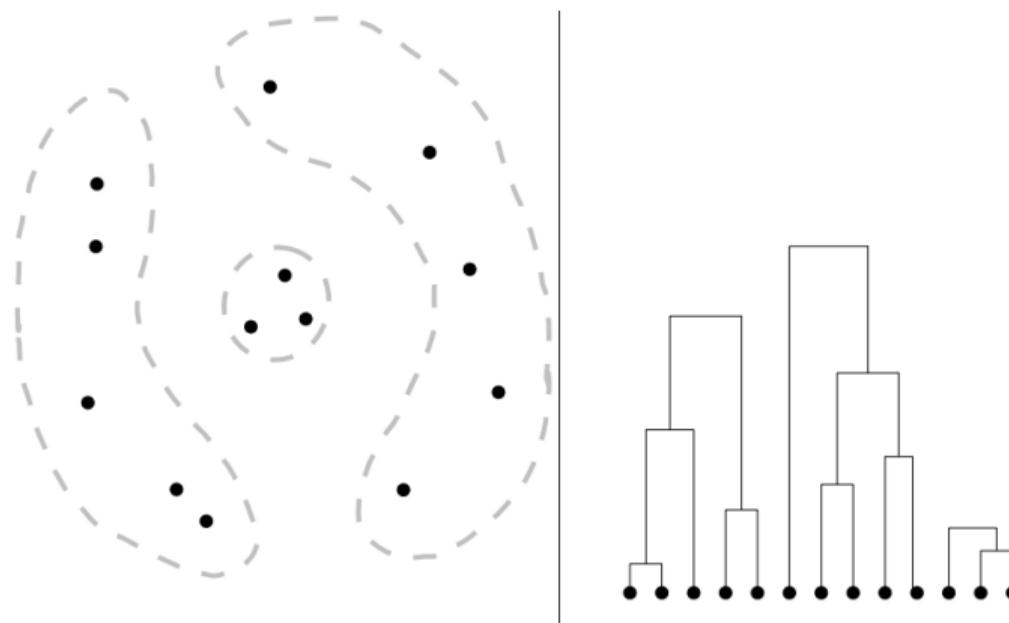
# Agglomerative Clustering: Example (Ctd.)



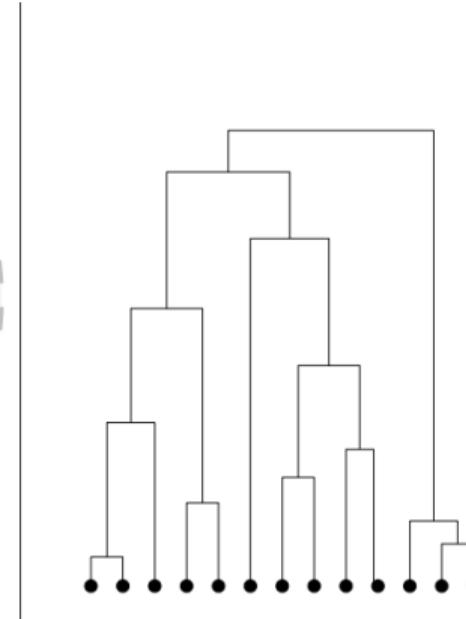
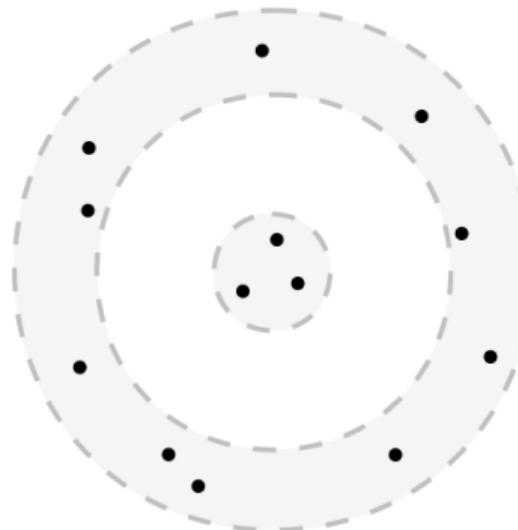
# Agglomerative Clustering: Example (Ctd.)



# Agglomerative Clustering: Example (Ctd.)



# Agglomerative Clustering: Example (Ctd.)

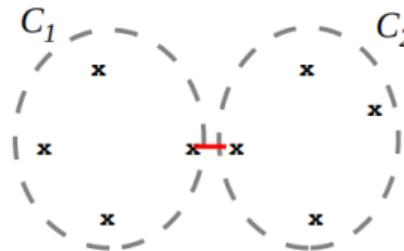


This is a  
**dendrogram**

# Single Linkage

- How to compute the distance between two clusters  $C_1$  and  $C_2$ ?
- **Single linkage:**

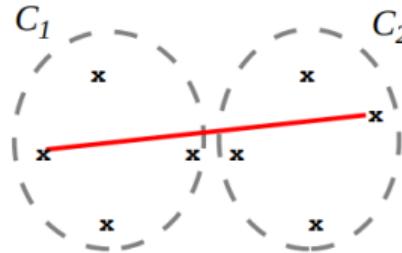
$$d(C_1, C_2) := \min \left\{ d(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in C_1, \mathbf{y} \in C_2 \right\}$$



# Complete Linkage

- How to compute the distance between two clusters  $C_1$  and  $C_2$ ?
- **Complete linkage:**

$$d(C_1, C_2) := \max \left\{ d(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in C_1, \mathbf{y} \in C_2 \right\}$$



## Section: **DBSCAN**

Introduction to DBSCAN  
Comparison of KMeans and DBSCAN  
An illustrative Example

# General Idea of DBSCAN

- **DBSCAN** is short for **Density-Based Spatial Clustering of Applications with Noise**
- The algorithm is a density-based method
- **General idea:**
  - Group together points that are closely packed in high-density regions
  - Mark points as outliers which lie alone in low-density regions

## Advantages:

- The algorithm does **not assume the clusters to be of a certain shape**
- **Concept of outliers:** Not all data points have to be assigned to a cluster

# DBSCAN Hyperparameters

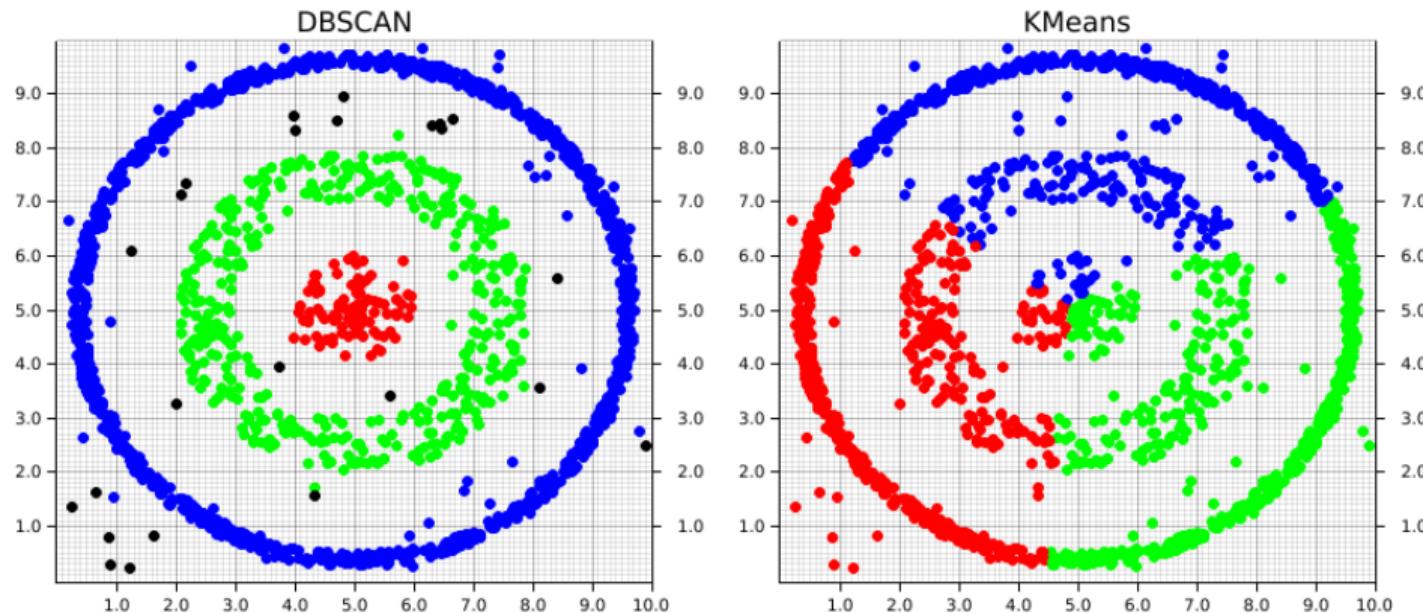
- DBSCAN has two hyperparameters, `eps` and `minPts`
- DBSCAN distinguishes three different types of points:

**Core point:** A point  $p$  is a **core point**, if at least `minPts` points are within distance `eps` of  $p$  (*including  $p$  itself*)

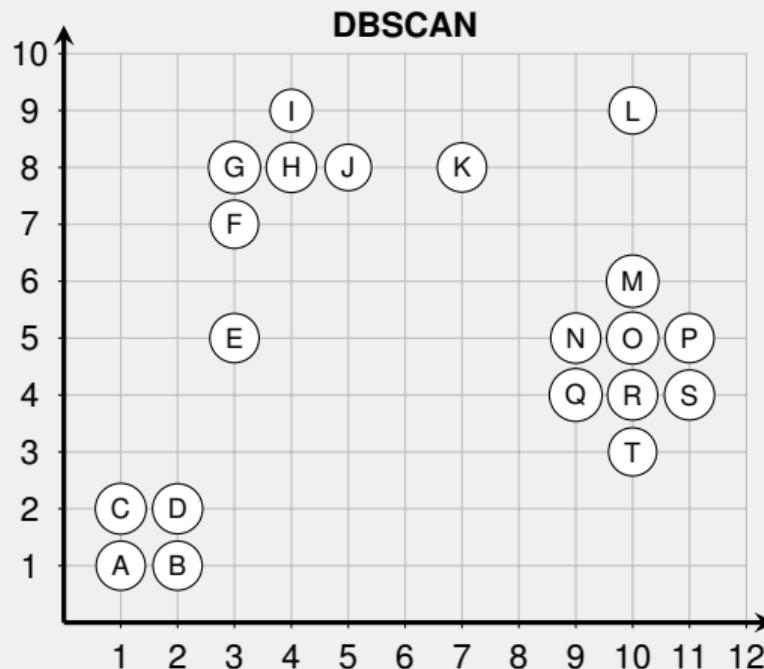
**Border point:** A non-core point  $q$  is a **border point**, if there is a core point  $p$  within distance `eps` from  $q$

**Outlier:** Points which are neither core points nor border points are said to be **outliers**

# KMeans vs. DBSCAN



# Example: DBSCAN



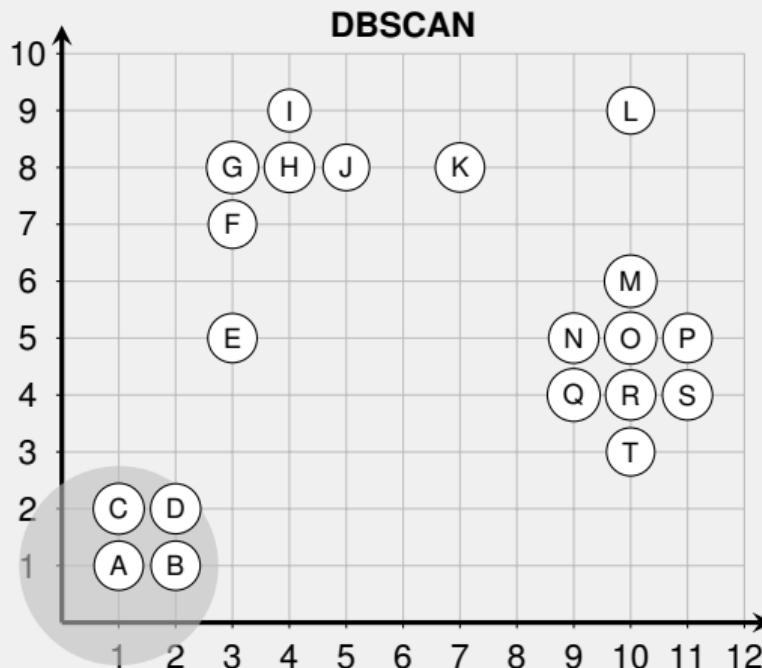
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



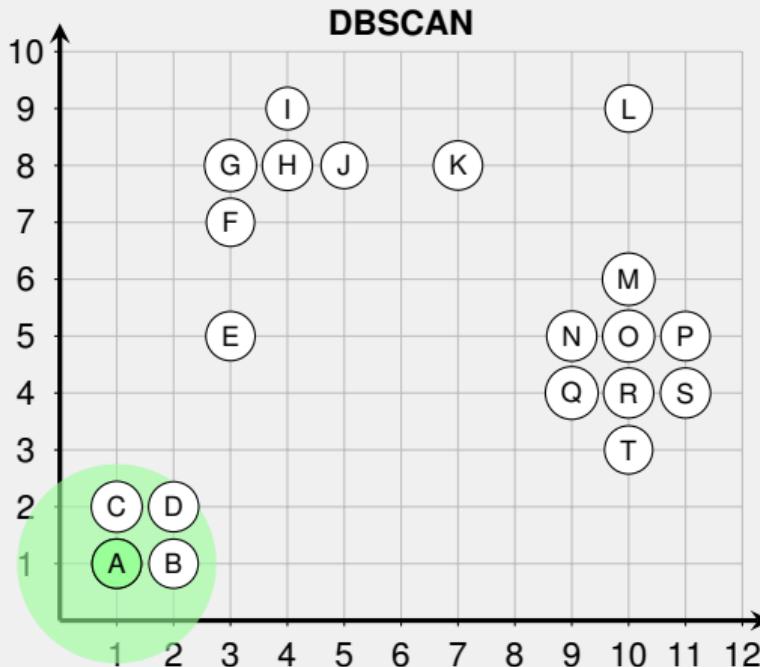
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



**Hyperparameters:**

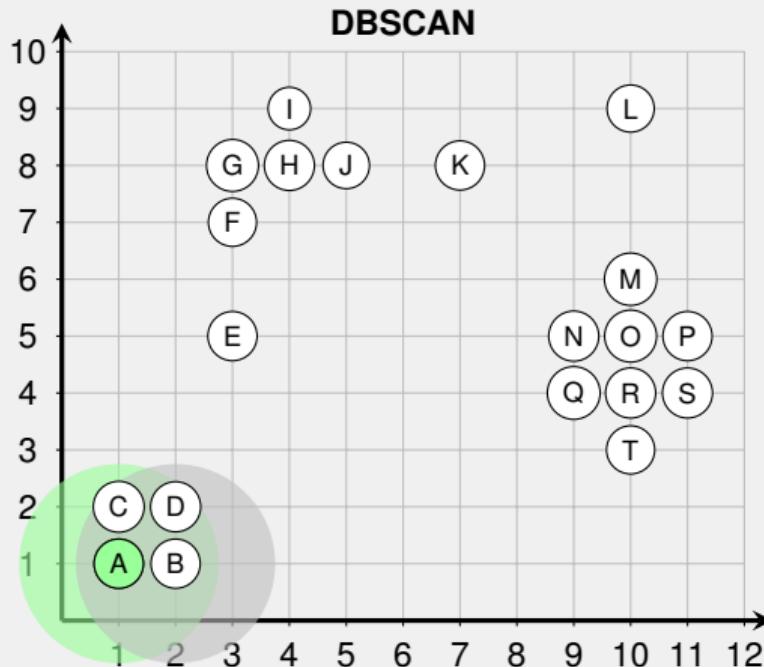
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

A B C D

# Example: DBSCAN



**Hyperparameters:**

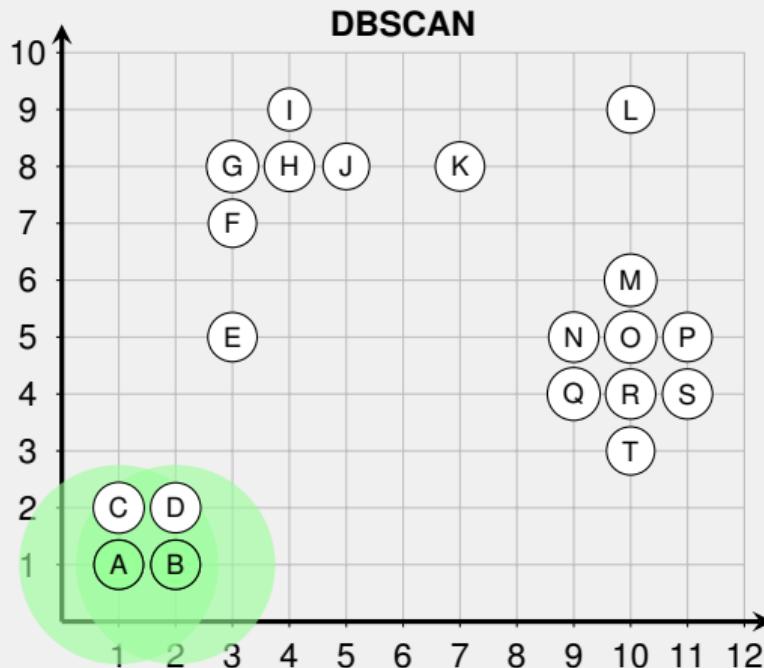
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

A B C D

# Example: DBSCAN



**Hyperparameters:**

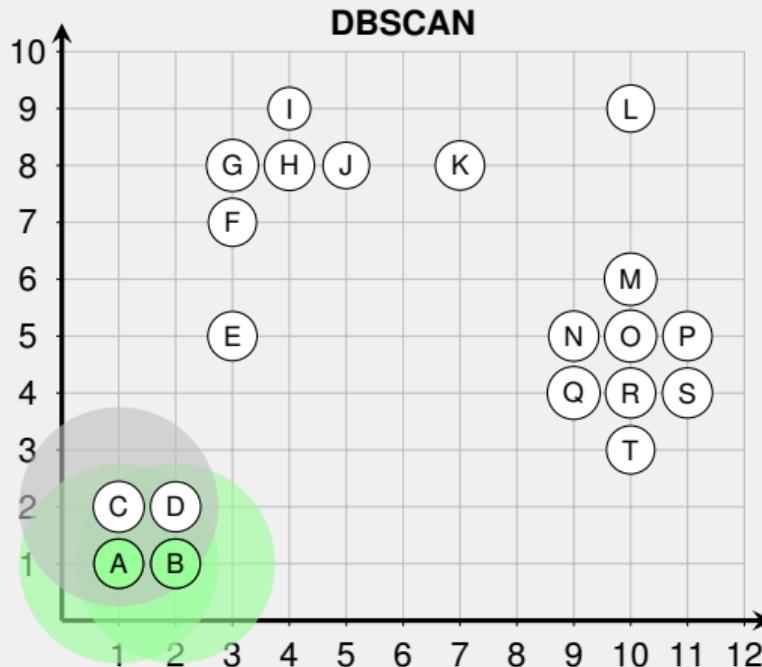
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

A B C D

# Example: DBSCAN



**Hyperparameters:**

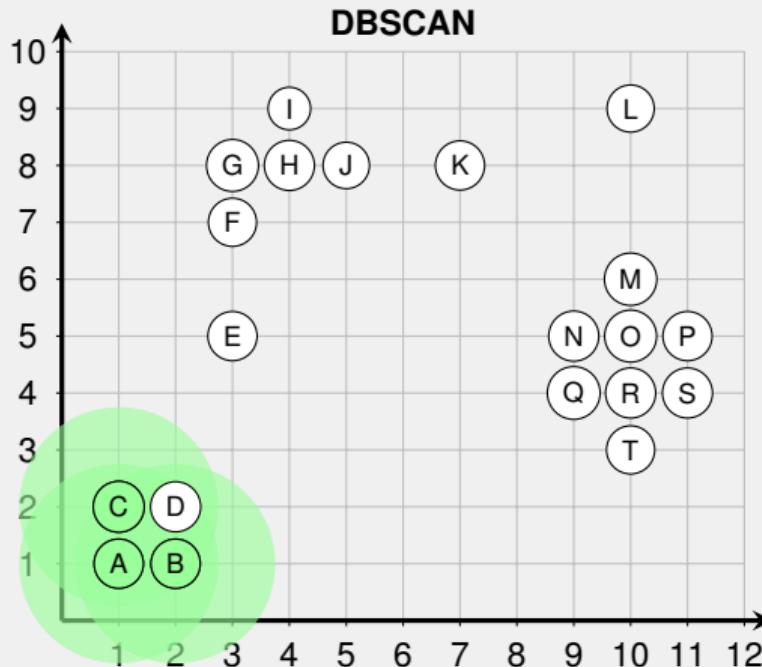
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

A B C D

# Example: DBSCAN



**Hyperparameters:**

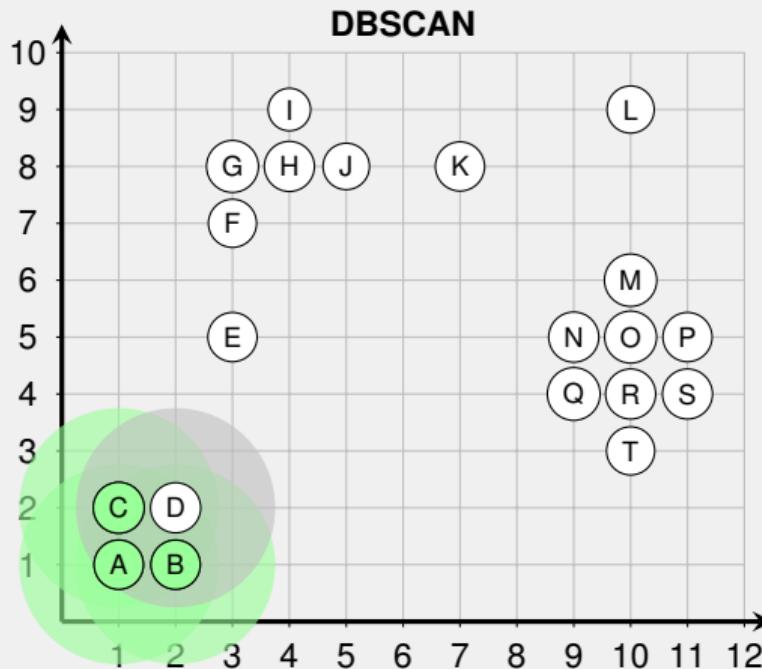
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

A B C D

# Example: DBSCAN



**Hyperparameters:**

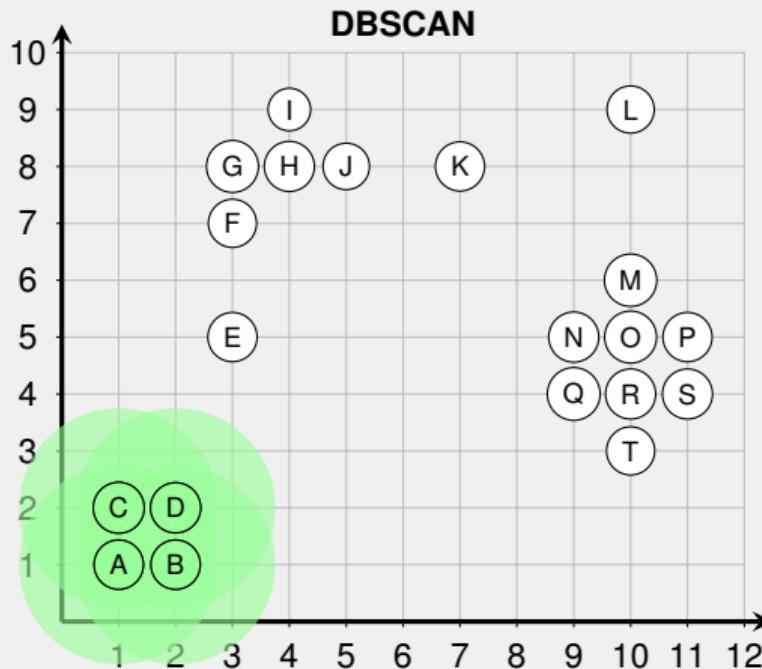
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

A B C D

# Example: DBSCAN



**Hyperparameters:**

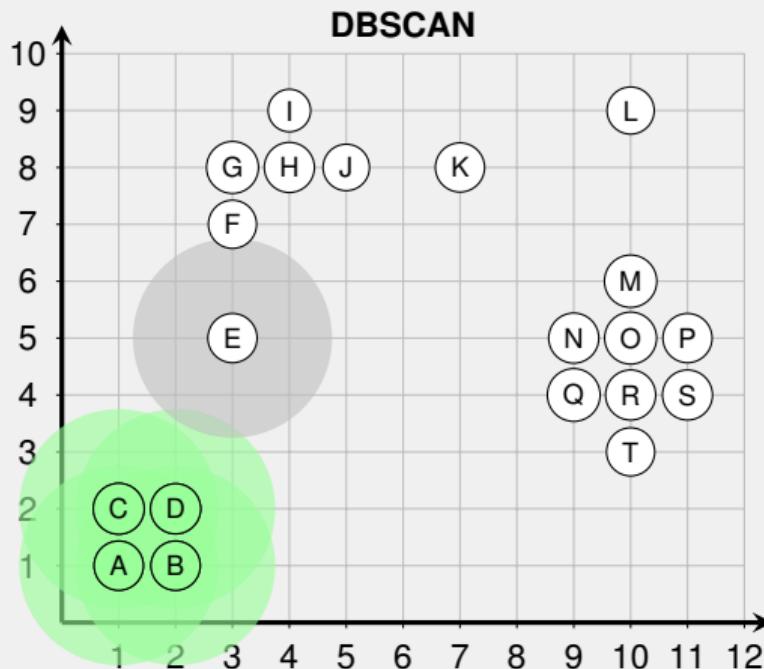
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

A B C D

# Example: DBSCAN



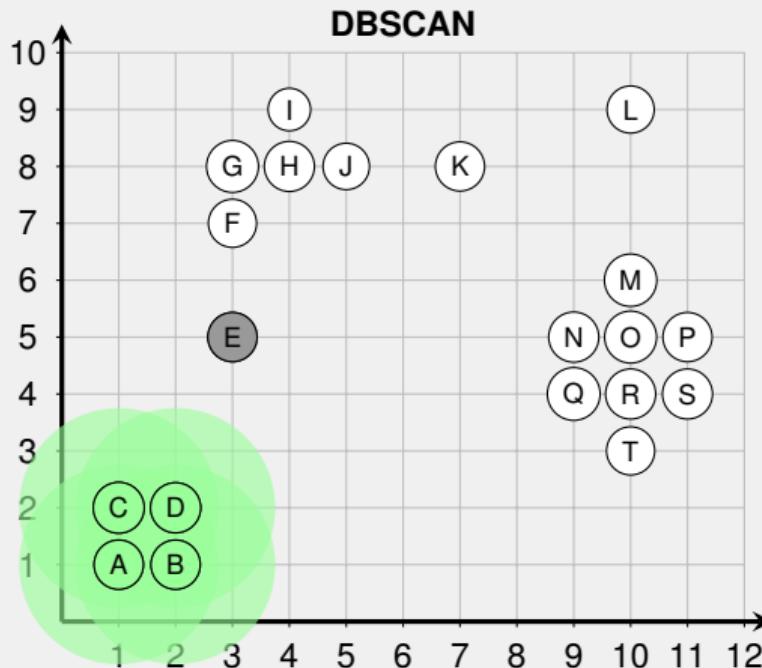
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



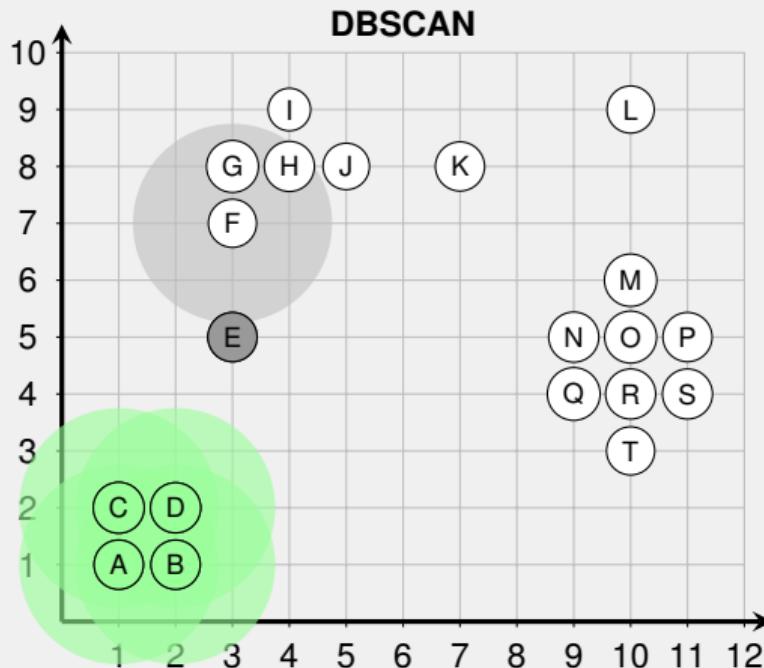
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



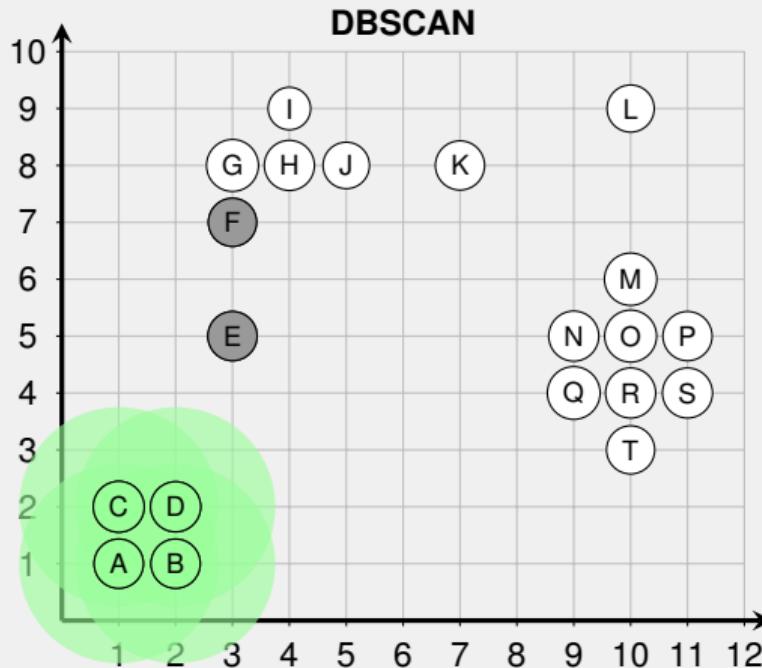
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



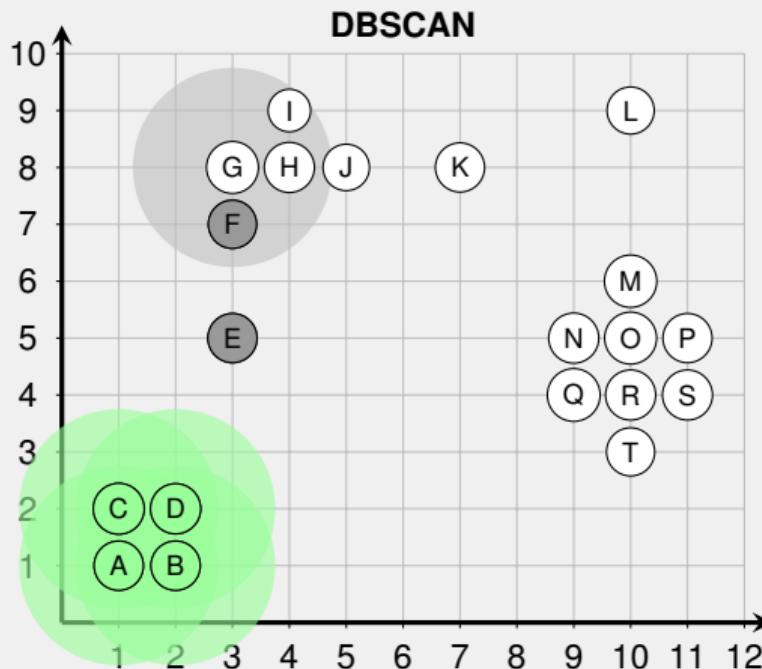
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



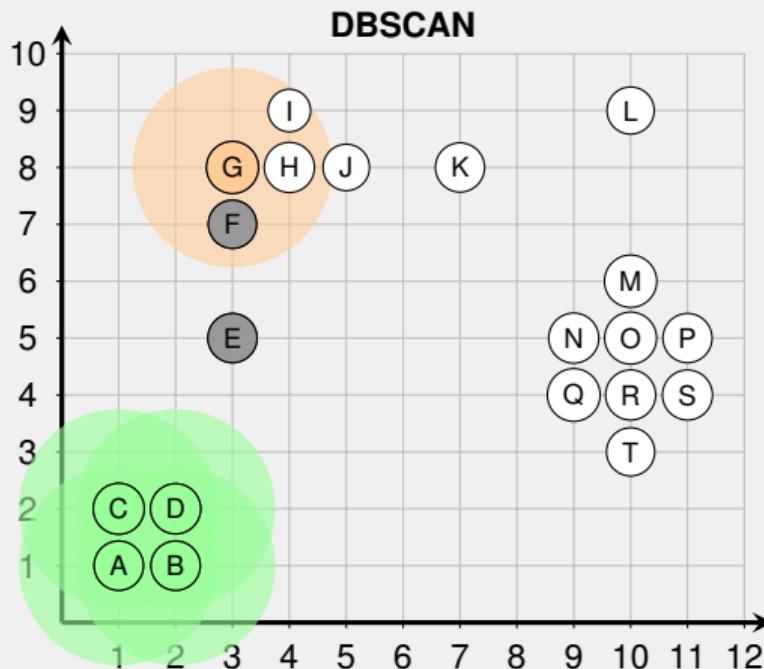
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



**Hyperparameters:**

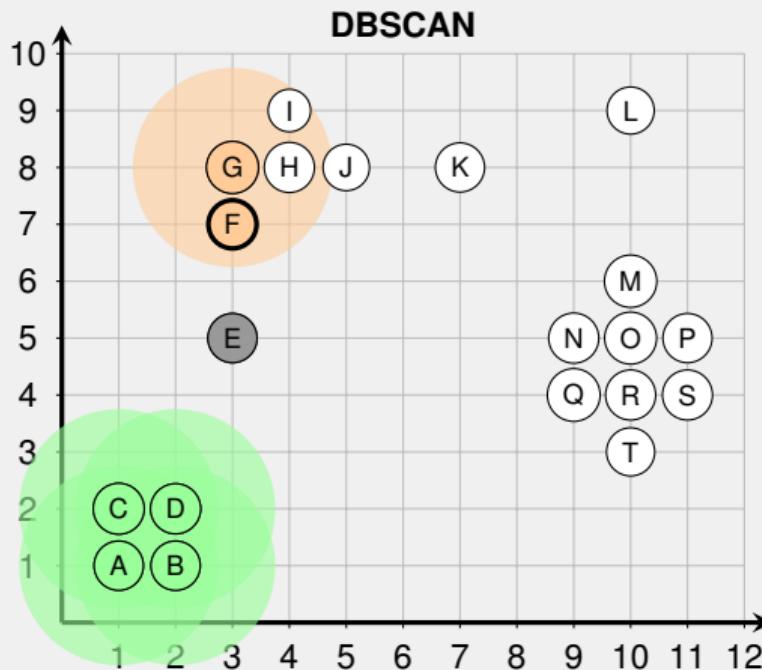
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

F G H I

# Example: DBSCAN



**Hyperparameters:**

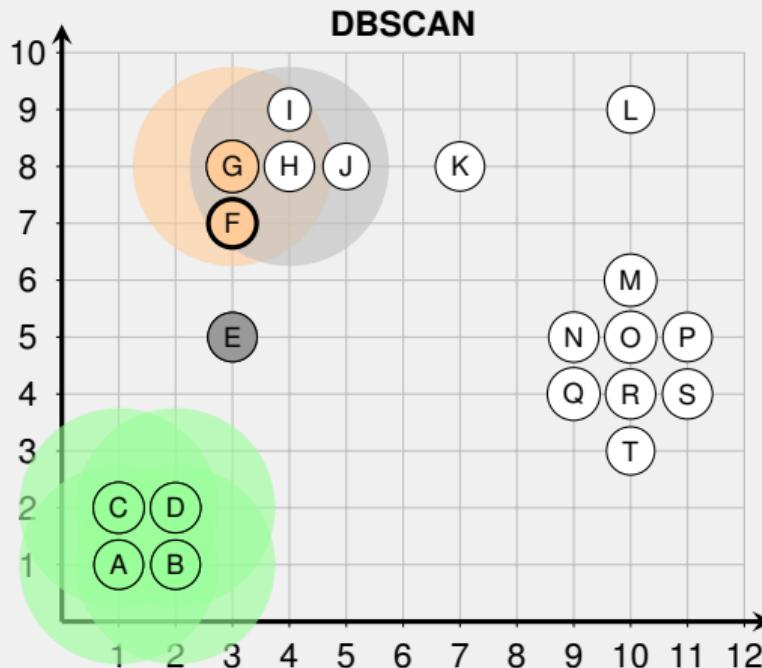
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

F G H I

# Example: DBSCAN



**Hyperparameters:**

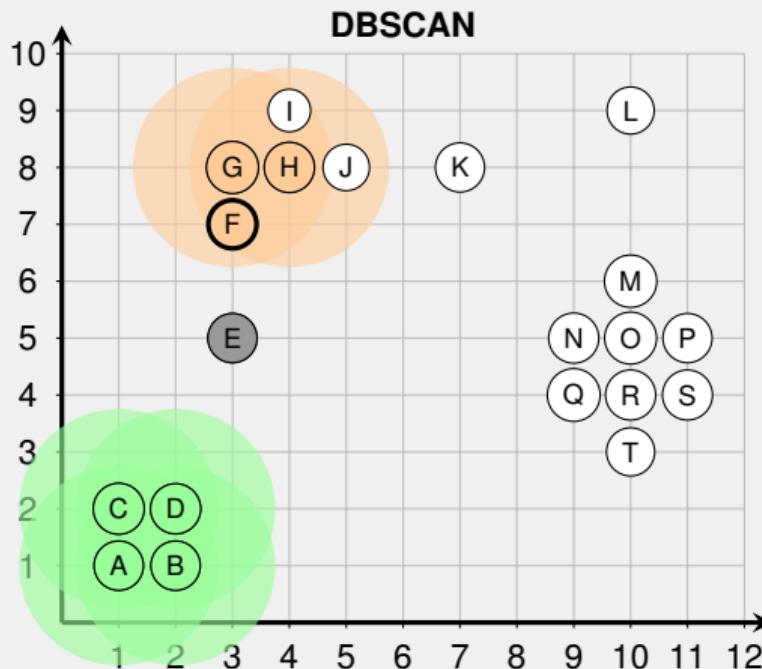
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

F G H I

# Example: DBSCAN



**Hyperparameters:**

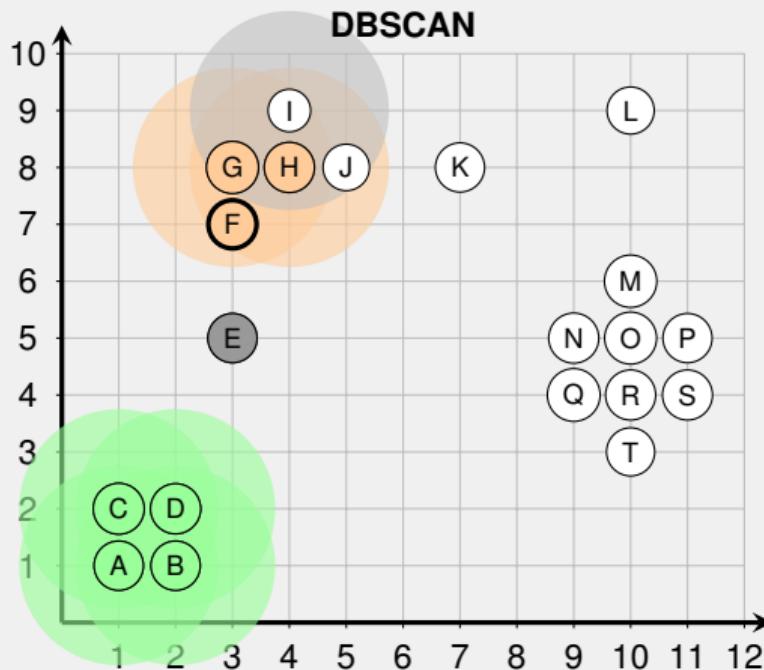
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

F G H I J

# Example: DBSCAN



**Hyperparameters:**

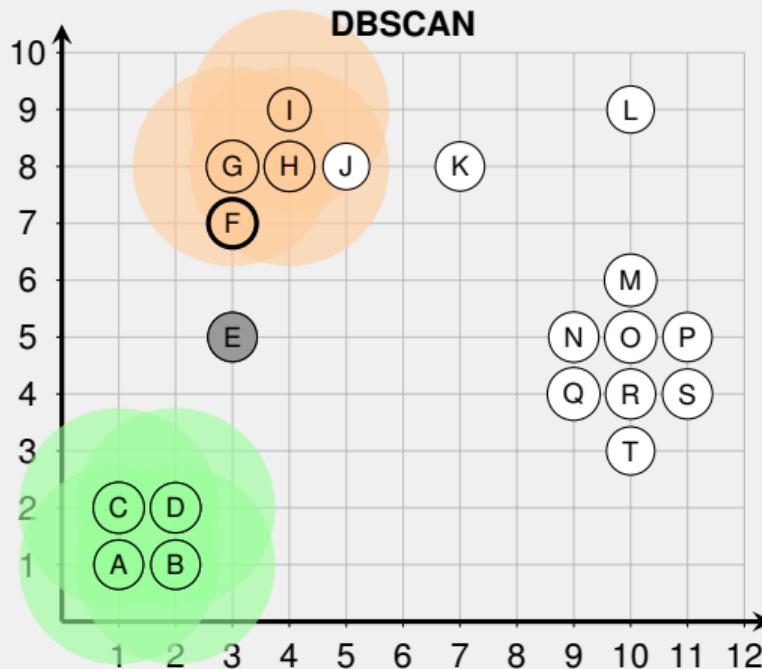
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

F G H I J

# Example: DBSCAN



**Hyperparameters:**

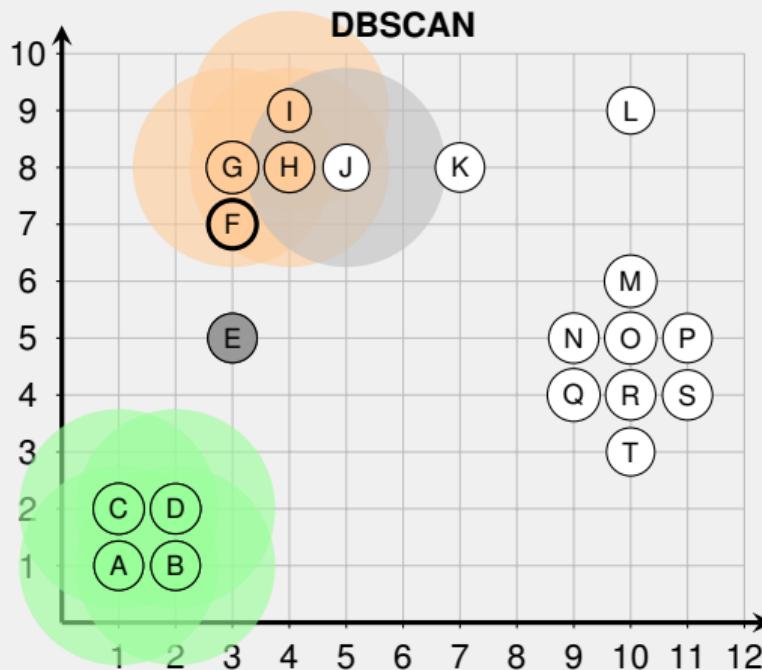
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

F G H I J

# Example: DBSCAN



**Hyperparameters:**

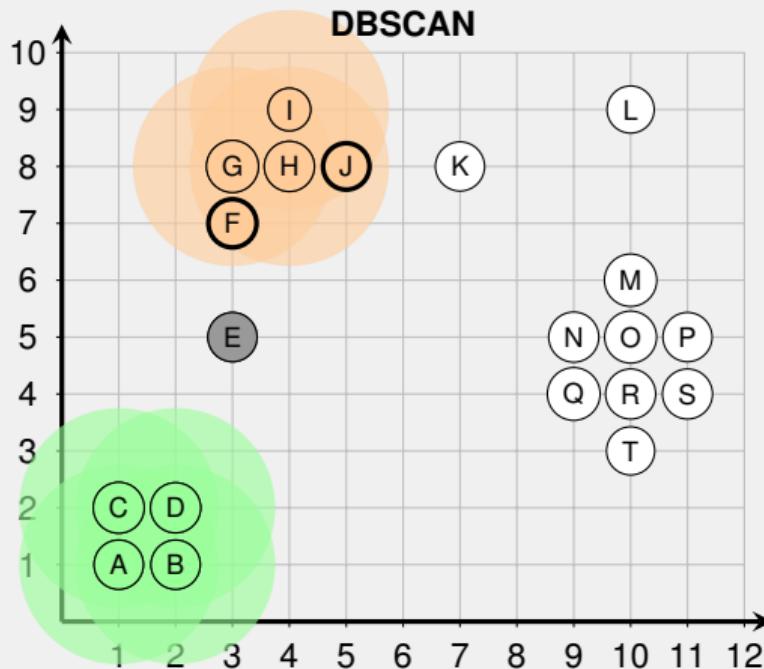
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

F G H I J

# Example: DBSCAN



**Hyperparameters:**

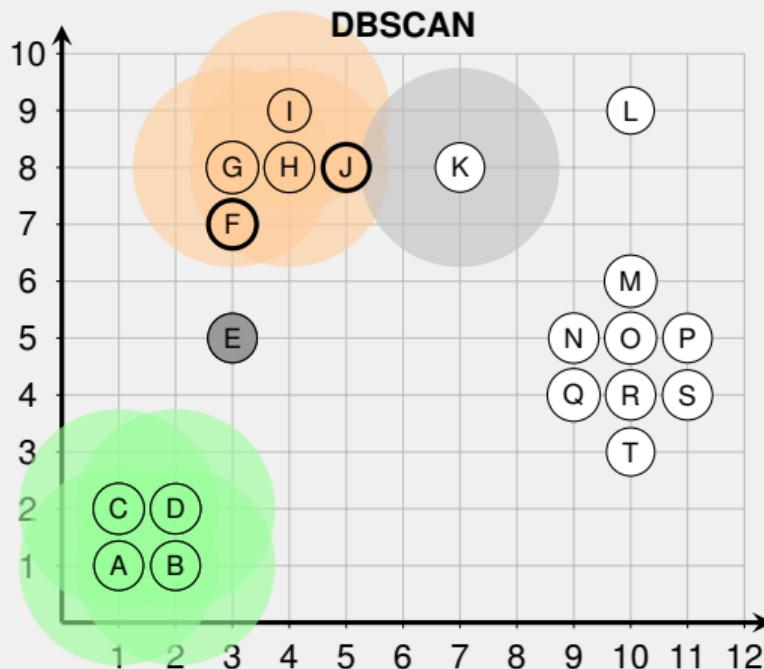
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

F G H I J

# Example: DBSCAN



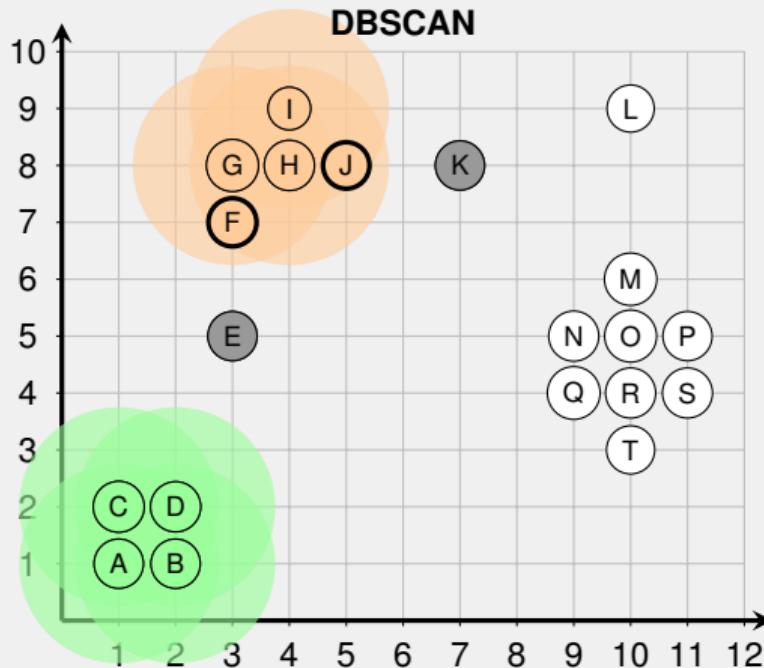
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



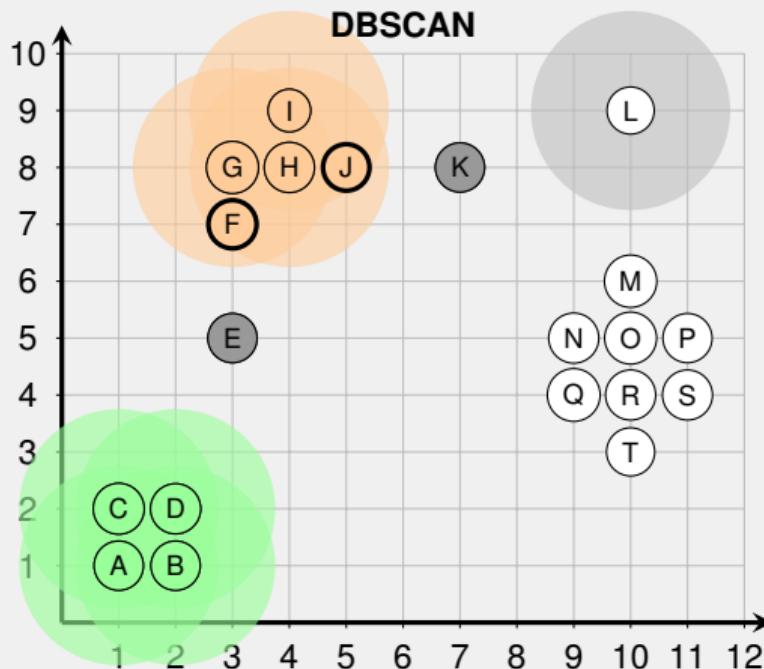
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



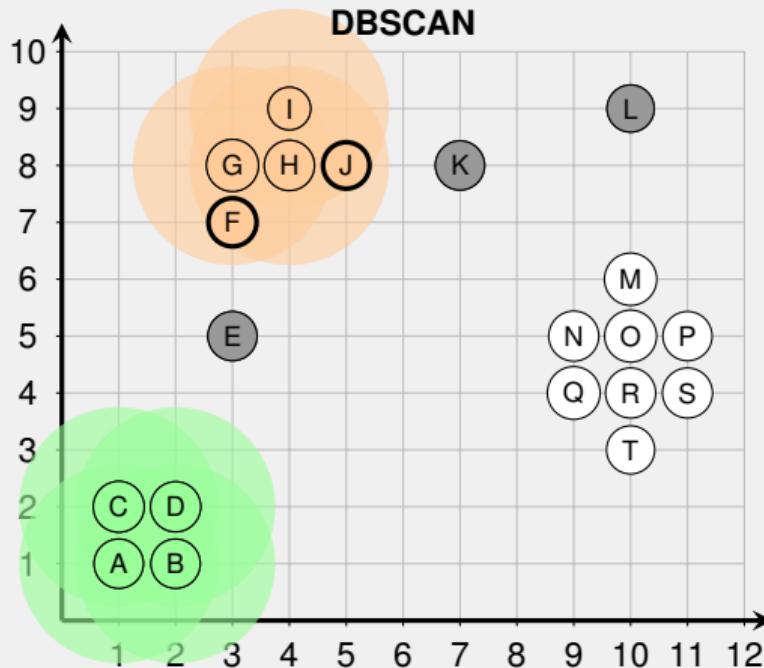
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



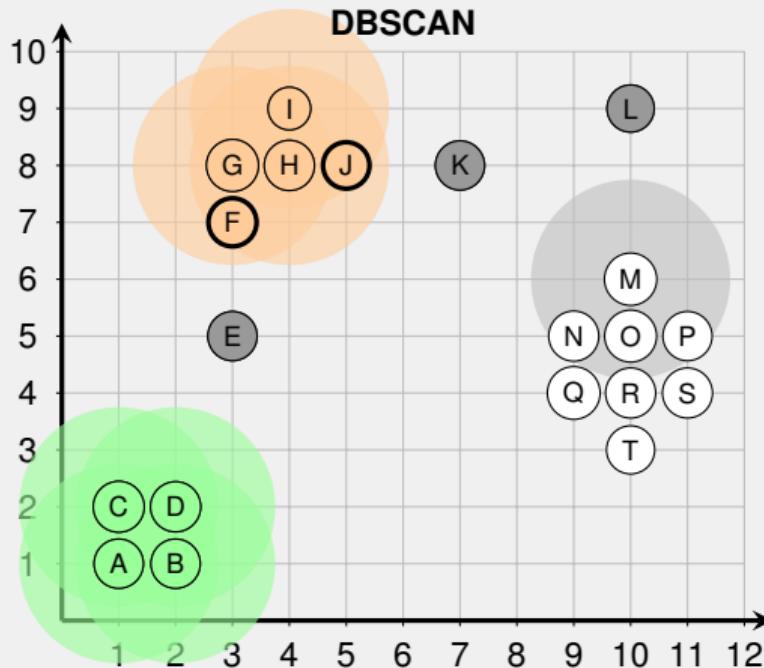
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



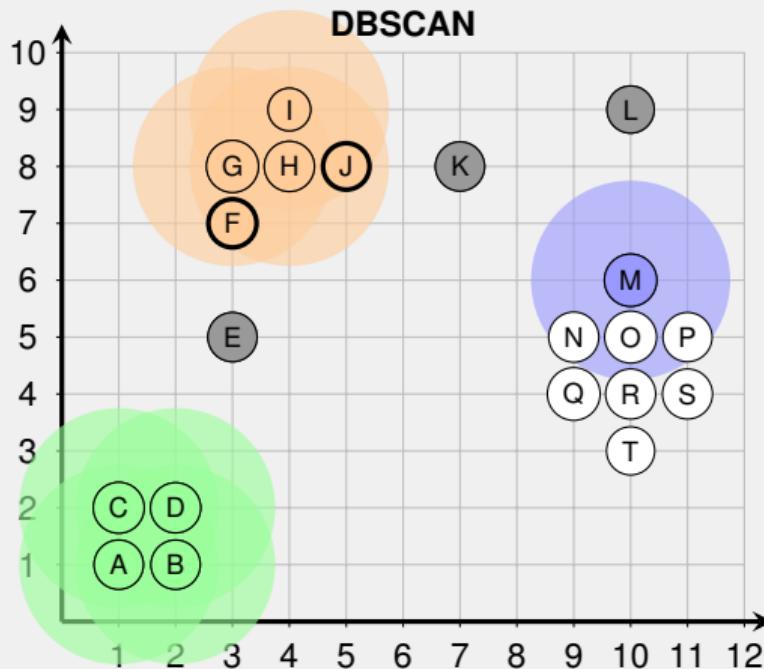
**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

# Example: DBSCAN



**Hyperparameters:**

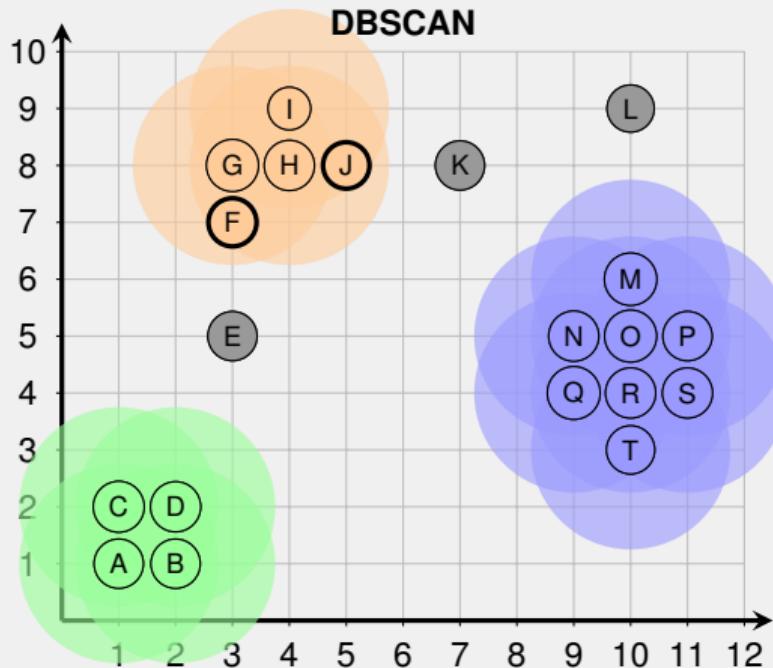
$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

M N O P

# Example: DBSCAN



**Hyperparameters:**

$\text{eps} = 1.75$

$\text{minPts} = 4$

**Seed list:**

M N O P Q R S T

## Section: Mean-Shift Clustering

General Idea of the Algorithm  
Recap: Kernel Density Estimation  
Derivation of the Mean-Shift Vector  
Final Algorithm

# Introduction to Mean-Shift Clustering

- The **mean-shift** algorithm is a **non-parametric** clustering technique
- **Basic idea of the algorithm:**
  - Compute the density at each data point using a **kernel density estimator**
  - Move (a copy) each data point into the direction of the maximum increase of the density
  - All data points ending up in (roughly) the same spot form a cluster

## Advantages:

- We do **not need prior knowledge of the number of clusters**
- The algorithm does **not constrain the shape of the clusters**

# Recap: Kernel Density Estimation

- **Recall:** The non-parametric density is given by:

$$p(\mathbf{x}) = \frac{1}{Nh^D} \sum_{n=1}^N K\left(\frac{\mathbf{x} - \mathbf{x}^n}{h}\right) \quad (2)$$

- $K$  is the kernel function, e.g. the Gaussian kernel:

$$K(\mathbf{u}) := \frac{1}{(\sqrt{2\pi})^D} \exp\left(-\frac{\|\mathbf{u}\|^2}{2}\right) \quad (3)$$

# Profile of a Kernel

- A kernel  $K$  can be written in terms of its **profile**  $k$ :  
*( $c_D$  is a normalization constant which ensures that the kernel integrates to 1)*

$$K(\mathbf{u}) := c_D k(\|\mathbf{u}\|^2) \quad (4)$$

- **Example:** GAUSSian kernel – equation (3):

$$k(x) := \exp\left(-\frac{1}{2}x\right) \quad (5)$$

$$c_D := \frac{1}{(\sqrt{2\pi})^D} \quad (6)$$

# Kernel Density Estimator with Kernel Profile

The kernel density estimation becomes:

$$\begin{aligned} p(\mathbf{x}) &= \frac{1}{Nh^D} \sum_{n=1}^N K\left(\frac{\mathbf{x} - \mathbf{x}^n}{h}\right) \\ &= \frac{c_D}{Nh^D} \sum_{n=1}^N k\left(\left\|\frac{\mathbf{x} - \mathbf{x}^n}{h}\right\|^2\right) \end{aligned} \tag{7}$$

We have to compute the gradient of  $p(\mathbf{x})$  to know in which direction the density increases the most!

# Gradient of the Kernel Density

- The gradient of  $p(\mathbf{x})$  is

$$\begin{aligned}\nabla p(\mathbf{x}) &= \frac{c_D}{Nh^D} \sum_{n=1}^N k' \left( \left\| \frac{\mathbf{x} - \mathbf{x}^n}{h} \right\|^2 \right) \cdot 2 \cdot \frac{\mathbf{x} - \mathbf{x}^n}{h} \cdot \frac{1}{h} \\ &= \frac{2c_D}{Nh^{D+2}} \sum_{n=1}^N (\mathbf{x} - \mathbf{x}^n) k' \left( \left\| \frac{\mathbf{x} - \mathbf{x}^n}{h} \right\|^2 \right)\end{aligned}\tag{8}$$

- As a next step, we introduce  $g(\mathbf{x}) := -k(\mathbf{x})$  (the profile of the **shadow kernel**  $G$ ) and further rewrite the gradient

## Gradient of the Kernel Density (Ctd.)

$$\begin{aligned}
 \nabla p(\mathbf{x}) &= \frac{2c_D}{Nh^{D+2}} \sum_{n=1}^N (\mathbf{x}^n - \mathbf{x}) g\left(\left\|\frac{\mathbf{x} - \mathbf{x}^n}{h}\right\|^2\right) \\
 &= \frac{2c_D}{Nh^{D+2}} \left[ \sum_{n=1}^N g\left(\left\|\frac{\mathbf{x} - \mathbf{x}^n}{h}\right\|^2\right) \right] \underbrace{\left[ \frac{\sum_{n=1}^N \mathbf{x}^n g\left(\left\|\frac{\mathbf{x} - \mathbf{x}^n}{h}\right\|^2\right)}{\sum_{n=1}^N g\left(\left\|\frac{\mathbf{x} - \mathbf{x}^n}{h}\right\|^2\right)} - \mathbf{x} \right]}_{=: \mathbf{m}(\mathbf{x})} \quad (9)
 \end{aligned}$$

We call  $\mathbf{m}(\mathbf{x})$  the **mean-shift vector** (*points into the direction of maximum increase of the density*)

# Mean-Shift Algorithm

**Input:**  $\mathbf{X} = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^N\}$

**Remark:**  $\mathbf{x}^{n,t}$  refers to the  $n$ -th data point at timestep  $t$

- 1 Set  $t \leftarrow 0$  and  $\mathbf{x}^{1,t} := \mathbf{x}^1, \dots, \mathbf{x}^{N,t} := \mathbf{x}^N$
- 2 **while** *not converged* **do**
- 3     **for**  $n \leftarrow 1$  **to**  $N$  **do**
- 4         Compute the mean-shift vector  $\mathbf{m}(\mathbf{x}^{n,t}) \leftarrow \frac{\sum_{i=1}^N \mathbf{x}^i g\left(\left\|\frac{\mathbf{x}^{n,t} - \mathbf{x}^i}{h}\right\|^2\right)}{\sum_{i=1}^N g\left(\left\|\frac{\mathbf{x}^{n,t} - \mathbf{x}^i}{h}\right\|^2\right)} - \mathbf{x}^{n,t}$
- 5         Update the data point:  $\mathbf{x}^{n,t+1} \leftarrow \mathbf{x}^{n,t} + \mathbf{m}(\mathbf{x}^{n,t})$
- 6     **end**
- 7      $t \leftarrow t + 1$
- 8 **end**
- 9 Create a cluster comprising all data points shifted to the same local maximum (*with some tolerance*)
- 10 **return** the set of clusters  $\mathcal{C}$

# Visualization: Mean-Shift Clustering

## Section: Wrap-Up

- Summary
- Recommended Literature
- Self-Test Questions
- Lecture Outlook

# Summary

- Clustering belongs to the category of **unsupervised learning**
- With clustering we try to find **structure in the data**
- Different algorithms make **different assumptions** about the resulting clusters
- **Clustering Strategies:**
  - EM-based clustering (e. g. KMeans)
  - Hierarchical clustering
  - Affinity-based clustering (e. g. DBSCAN, mean-shift clustering)

# Recommended Literature

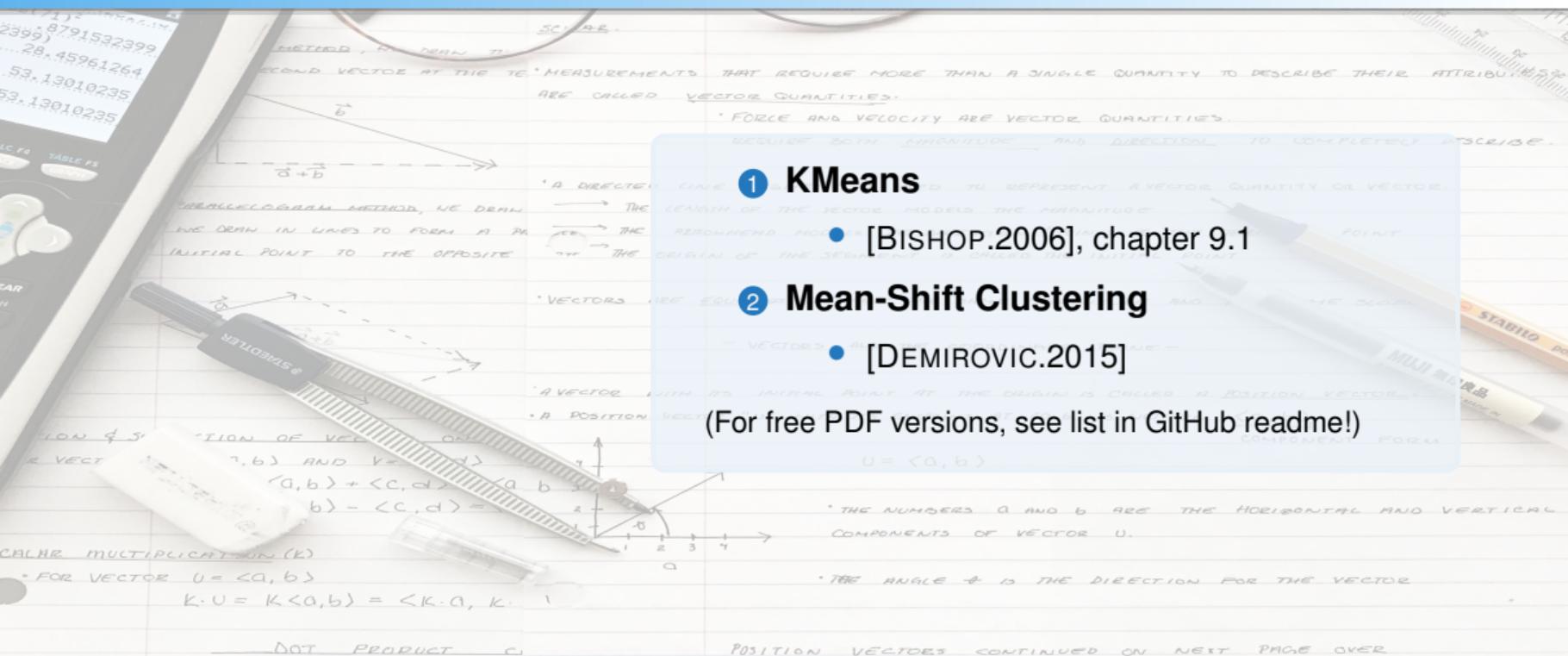
## 1 KMeans

- [BISHOP.2006], chapter 9.1

## 2 Mean-Shift Clustering

- [DEMIROVIC.2015]

(For free PDF versions, see list in GitHub readme!)



# Self-Test Questions

- ① What is clustering?
- ② What is the definition of a cluster. Which properties should it have?
- ③ Describe the general procedure of KMeans. What are disadvantages?
- ④ What is a dendrogram?
- ⑤ Describe what DBSCAN works!
- ⑥ What is affinity-based clustering? How does it differ from KMeans?
- ⑦ What is the main idea of mean-shift clustering?

# What's next...?

- |   |  |
|---|--|
| <b>I</b> Machine Learning Introduction      | <b>IX</b> Evaluation                       |
| <b>II</b> Optimization Techniques           | <b>X</b> Decision Trees                    |
| <b>III</b> Bayesian Decision Theory         | <b>XI</b> Support Vector Machines          |
| <b>IV</b> Non-parametric Density Estimation | <b>XII</b> Clustering                      |
| <b>V</b> Probabilistic Graphical Models     | • <b>XIII</b> Principal Component Analysis |
| <b>VI</b> Linear Regression                 | <b>XIV</b> Reinforcement Learning          |
| <b>VII</b> Logistic Regression              | <b>XV</b> Advanced Regression              |
| <b>VIII</b> Deep Learning                   |  |

# Thank you very much for the attention!

\* \* \* Artificial Intelligence and Machine Learning \* \* \*

**Topic:** Clustering Algorithms

**Term:** Summer term 2025

**Contact:**

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

[daniel.wehner@sap.com](mailto:daniel.wehner@sap.com)

Do you have any questions?