1

🧠 Affinity Propagation Clustering – Definition

Affinity Propagation (AP) is a clustering algorithm that **finds clusters by sending messages between data points**.

Instead of specifying the number of clusters (like in K-Means), **it automatically decides how many clusters to form** based on the data.

It works by finding **“exemplars”** — representative data points that best describe each cluster.

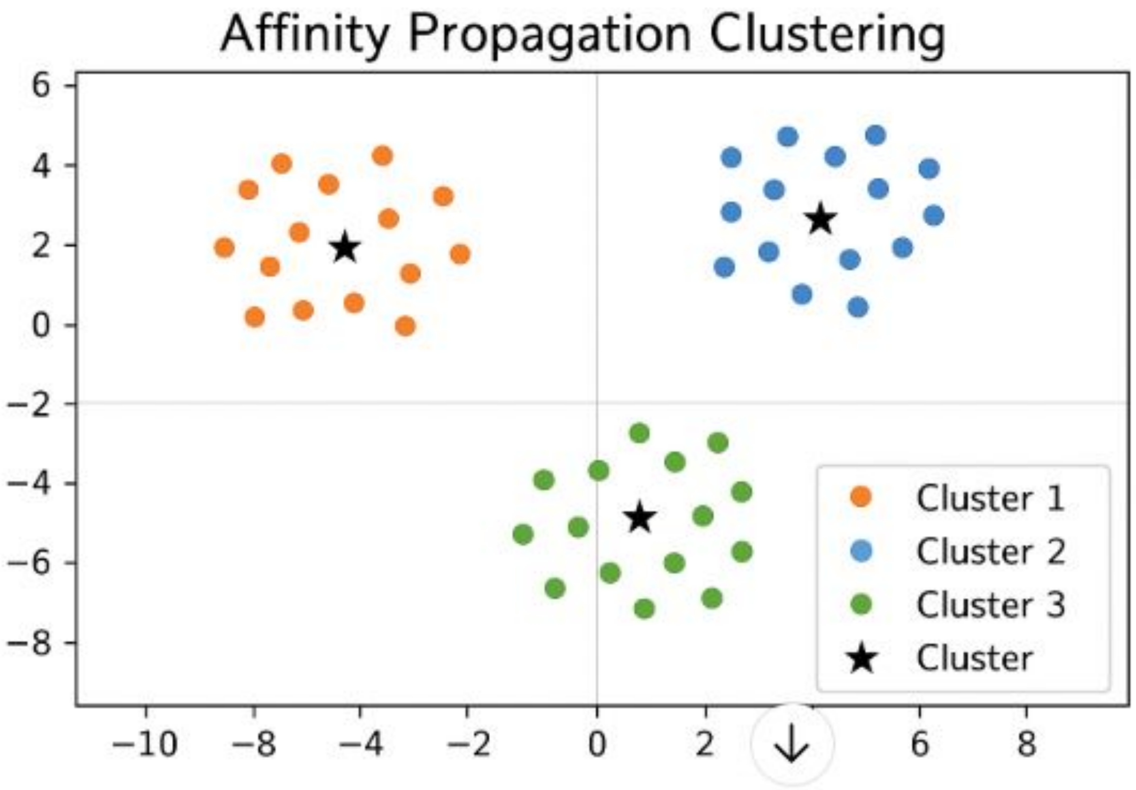
✅ Advantages

1. **No need to specify number of clusters** (unlike K-Means).
2. Can **find clusters of different sizes and shapes**.
3. Works well when there are **clear exemplar (center) points**.
4. Handles **non-metric similarities** (custom similarity measures).

❌ Disadvantages

1. **Slow** for large datasets (since it uses all pairwise similarities).
2. Needs careful tuning of parameters (`damping` , `preference`).
3. Sometimes results can be **unstable or hard to interpret**.
4. **Memory intensive**, since it stores similarity matrix for all points.

From sklearn.cluster import AffinityPropagation



🧩 Summary Table

Feature	Description
Type	Unsupervised clustering
Requires #clusters?	❌ No
Output	Cluster labels and exemplars
Key Parameters	`damping`, `preference`
Best for	Medium-sized datasets with clear cluster centers

2

🧠 Agglomerative Clustering – Definition

Agglomerative Clustering is a **bottom-up hierarchical clustering** method.

👉 It starts by treating **each data point as its own cluster**,

then **merges the two closest clusters** step by step

until all points belong to one big cluster or until the desired number of clusters is reached.

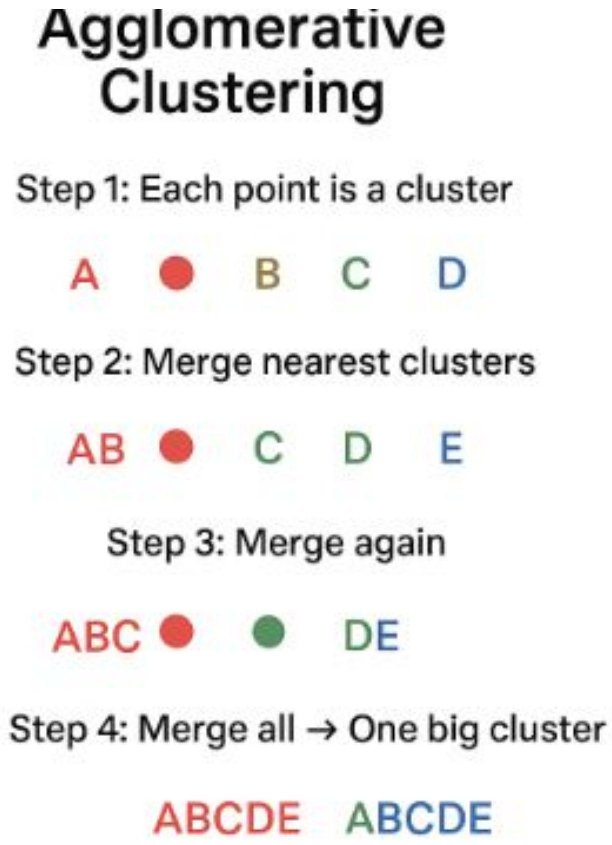
✅ Advantages

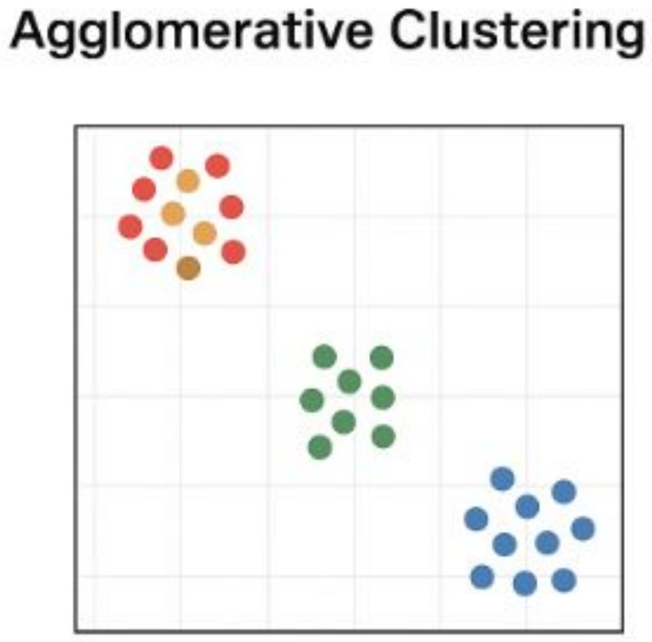
1. **No need to specify number of clusters initially** (you can decide later by cutting the dendrogram).
2. **Simple and easy to understand.**
3. Works well with **small datasets**.
4. Can handle **non-spherical clusters** (unlike K-Means).

❌ Disadvantages

1. **Computationally expensive** for large datasets.
2. **Once merged, clusters cannot be split** again.
3. **Choice of distance metric** can affect results.
4. **Sensitive to noise and outliers.**

from sklearn.cluster import AgglomerativeClustering





3

🧠 BIRCH Clustering – Definition

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) is a **hierarchical clustering algorithm** designed to handle **very large datasets efficiently**.

It builds a **tree structure (CF Tree)** that summarizes the data, and then clusters those summaries instead of all individual points — this makes it **fast and memory-efficient**.

✅ Advantages

1. ⚡ **Very fast** — works well on **large datasets**.
2. 🧩 **Automatically reduces data size** using CF Tree.
3. 💾 **Memory-efficient**, doesn't need to load all data at once.
4. 📉 Can be combined with other clustering methods (like K-Means).

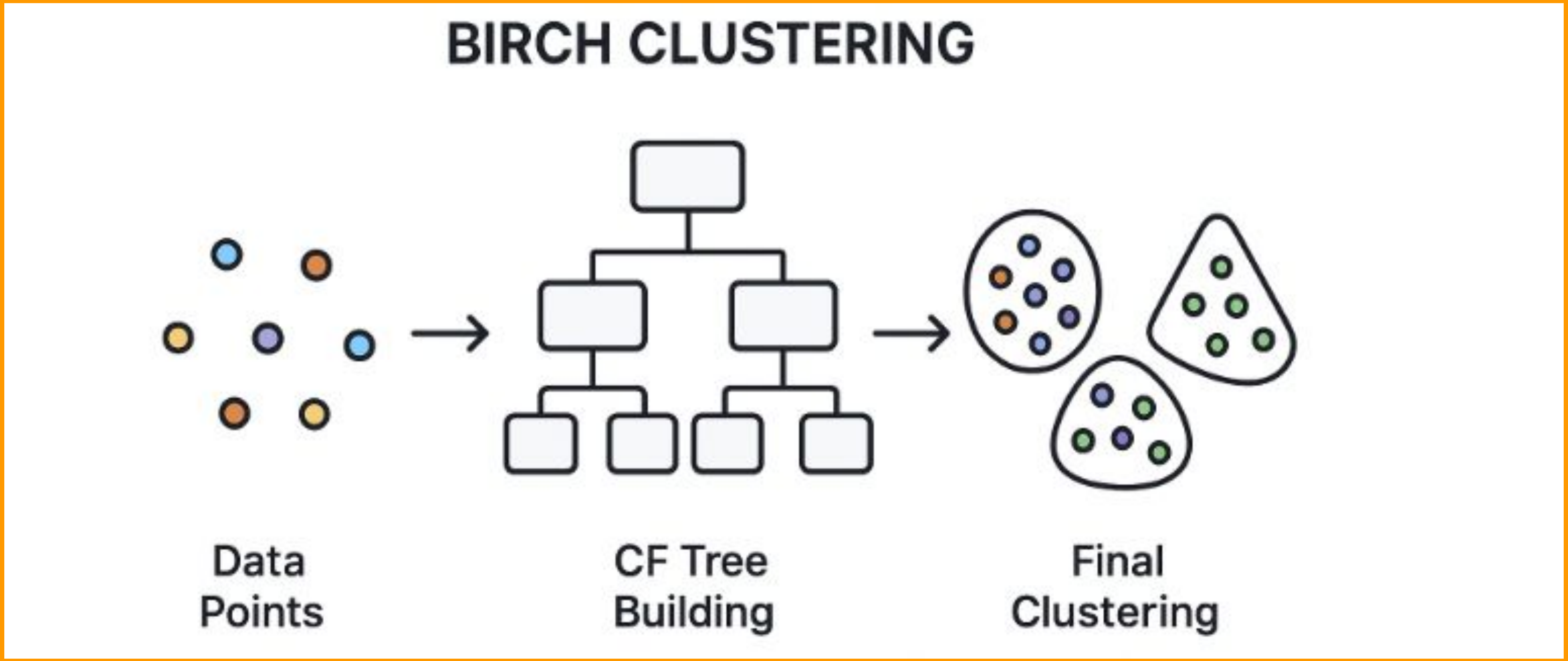
❌ Disadvantages

1. ❗ Works best with **spherical clusters** (like K-Means).
2. ❌ Doesn't always perform well with **non-uniform cluster sizes**.
3. 🧮 Needs careful choice of **threshold value** for CF Tree.
4. 💢 Can lose some information during data compression.

from sklearn.cluster import Birch

📊 Summary Table

Feature	Description
Full Form	Balanced Iterative Reducing and Clustering using Hierarchies
Type	Hierarchical + Summarization
Handles Large Data?	✅ Yes
Best for	Large datasets with spherical clusters
Output	Cluster labels
Key Parameter	`threshold` (controls cluster size)


BIRCH CLUSTERING
Data Points
CF Tree Building
Final Clustering

🧠 Bisecting K-Means – Definition

Bisecting K-Means is a **hierarchical version of K-Means**.

It starts with **all data points in one cluster**, and then **repeatedly splits (bisects)** one cluster into two using K-Means, until the desired number of clusters is formed.

👉 Think of it as **"K-Means + Divide & Conquer"** method.

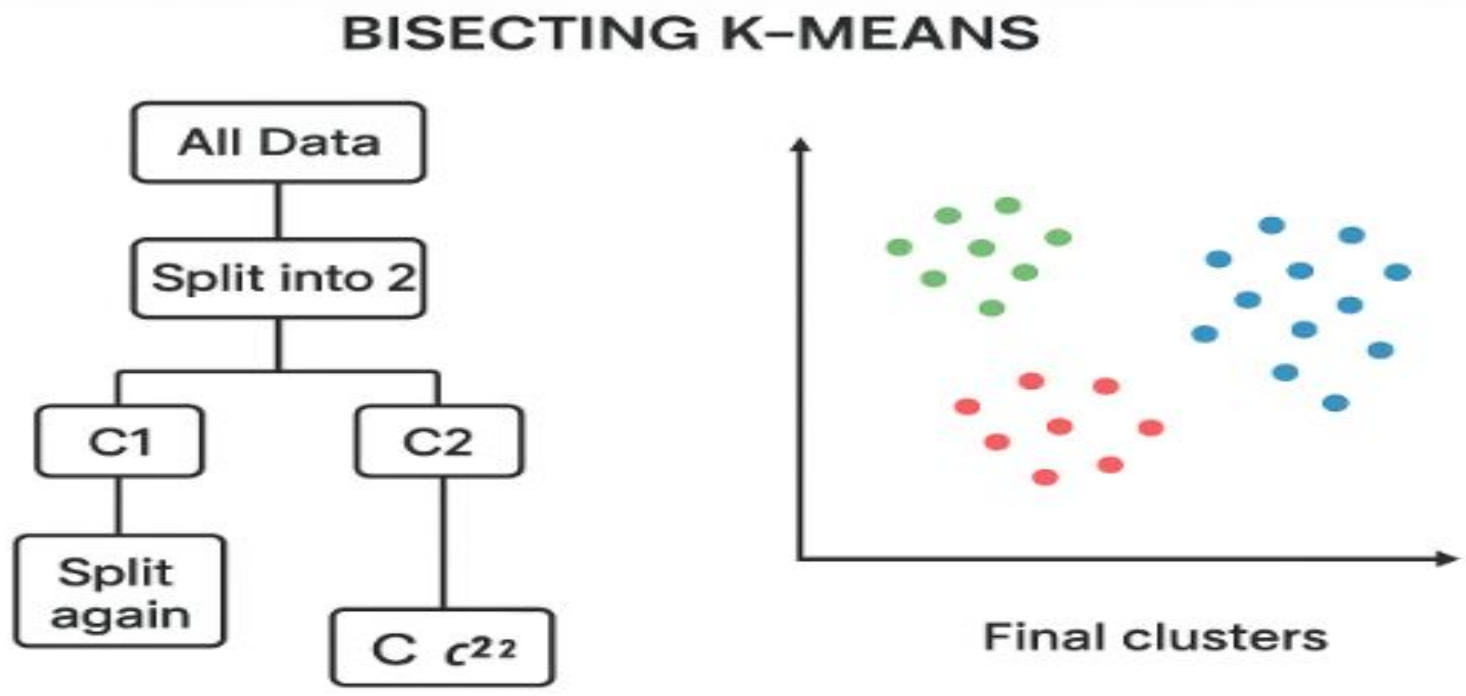
✅ Advantages

1. ✅ **Better accuracy** than regular K-Means.
2. ⚡ **Faster** than full hierarchical clustering.
3. 🧩 Works well with **large datasets**.
4. 🚀 Helps **avoid poor K-Means initialization**.
5. 📊 Creates **hierarchical cluster structure**.

❌ Disadvantages

1. ❗ Still needs to specify **number of clusters (k)**.
2. 💢 **Sensitive to outliers** (like K-Means).
3. 🧮 Can be **computationally expensive** for very large data.
4. 🧠 Only works well for **spherical-shaped clusters**.

from sklearn.cluster import BisectingKMeans



📊 Summary Table

Feature	Description
Type	Hierarchical + Partitional
Base Algorithm	K-Means
Input Required	Number of clusters (k)
Best for	Large datasets with roughly spherical clusters
Output	Cluster labels + hierarchy
Speed	Faster than standard hierarchical, slower than plain K-Means

5

🧠 DBSCAN – Definition

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a **density-based clustering** algorithm.

It groups together **closely packed points (dense regions)** and marks points that lie alone in low-density areas as **outliers**.

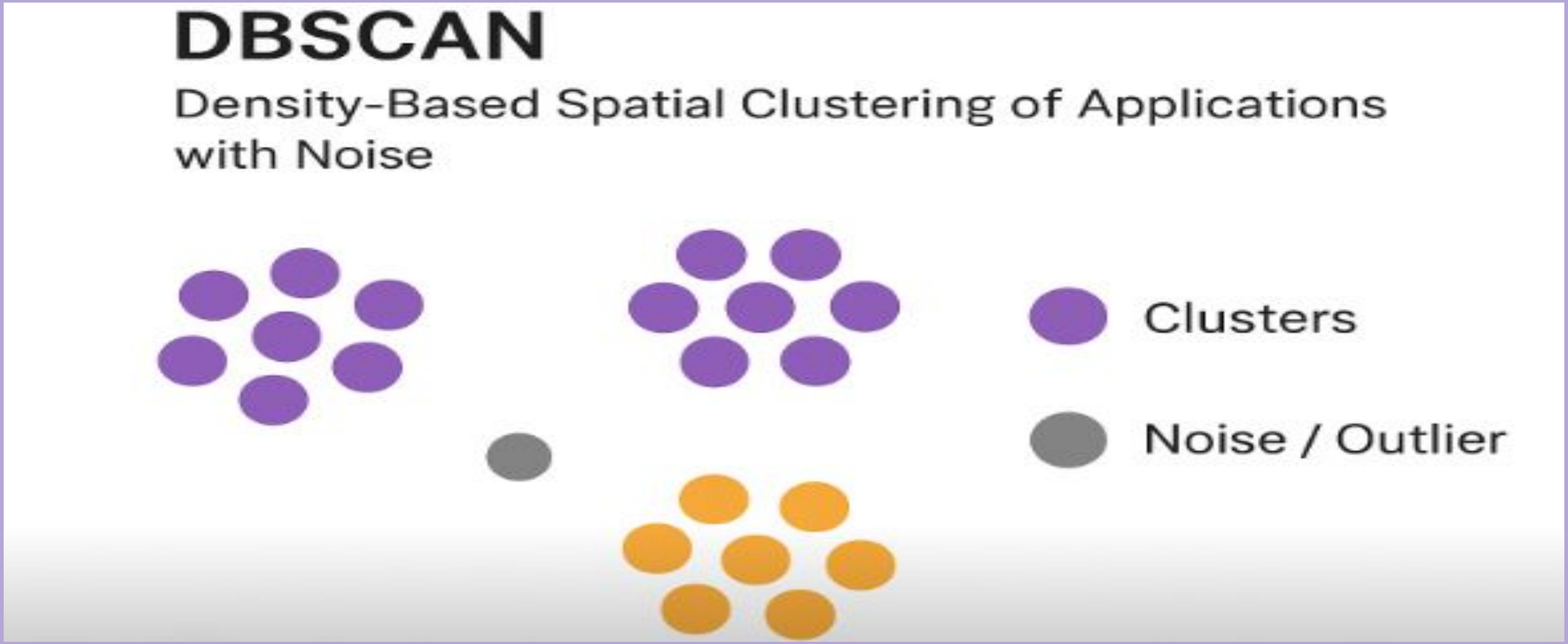
✅ Advantages:

1. **No need to specify number of clusters** beforehand (unlike K-Means).
2. Can find **arbitrarily shaped clusters** (not just circular).
3. Can **detect outliers/noise** easily.
4. Works well when clusters have different shapes and sizes.

❌ Disadvantages:

1. **Choosing ε and MinPts** can be tricky.
2. Struggles with **clusters of varying density**.
3. Not suitable for **very high-dimensional data**.

from sklearn.cluster import DBSCAN



6

🧠 **Definition:**

HDBSCAN is an **advanced version of DBSCAN** that uses **hierarchical density-based clustering**.

It automatically finds clusters of **different densities** and identifies **noise points** (outliers).

In simple terms:

➡️ DBSCAN groups dense points but struggles when densities vary.

➡️ **HDBSCAN** fixes that by building a **hierarchy (tree)** of clusters and selecting the most stable ones.

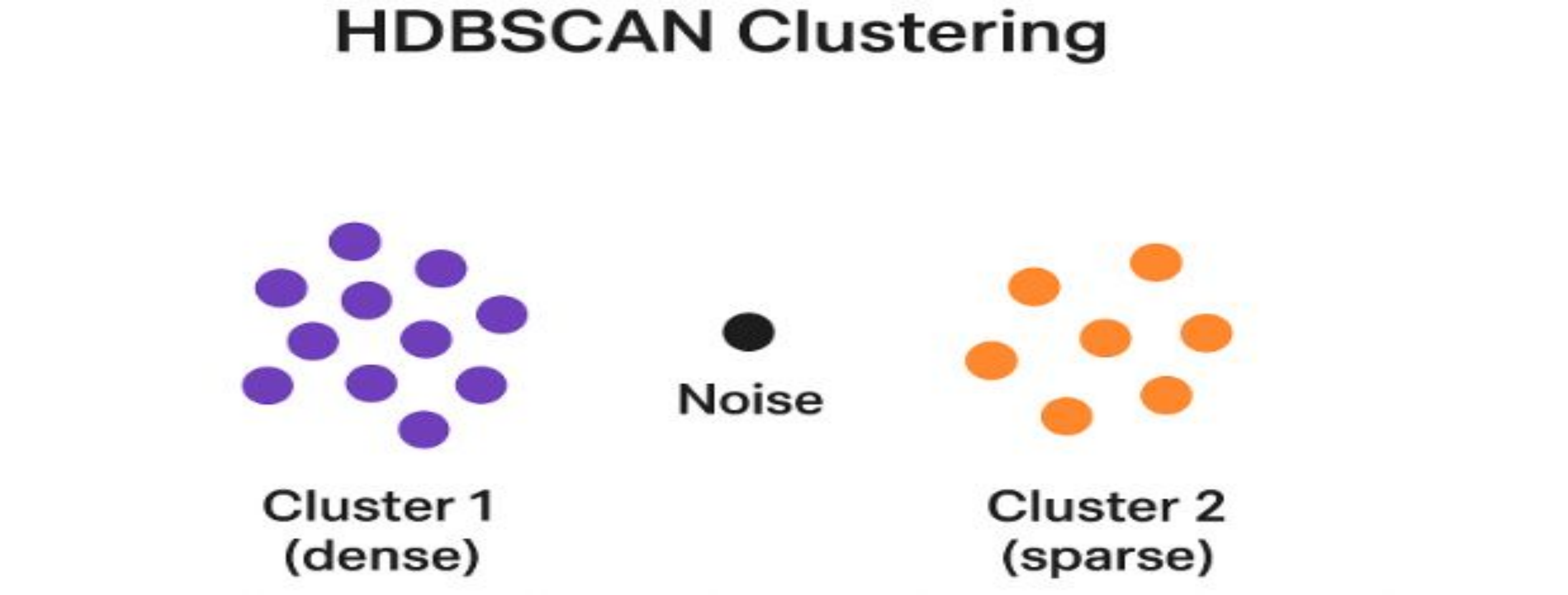
✅ Advantages:

- Handles **varying densities** easily
- **No need to pick eps** (automatic)
- Detects **noise/outliers**
- Finds **natural clusters**

❌ Disadvantages:

- Slower and **more complex** than DBSCAN
- Requires **extra library (hdbscan)**
- Harder to tune parameters

from sklearn.cluster import HDBSCAN



7

🧠 **Definition:**

K-Means Clustering is an **unsupervised learning algorithm** that groups data into **K number of clusters** based on how close the data points are to each other.

Each cluster has a **center point (centroid)**, and every data point belongs to the cluster with the **nearest centroid**.

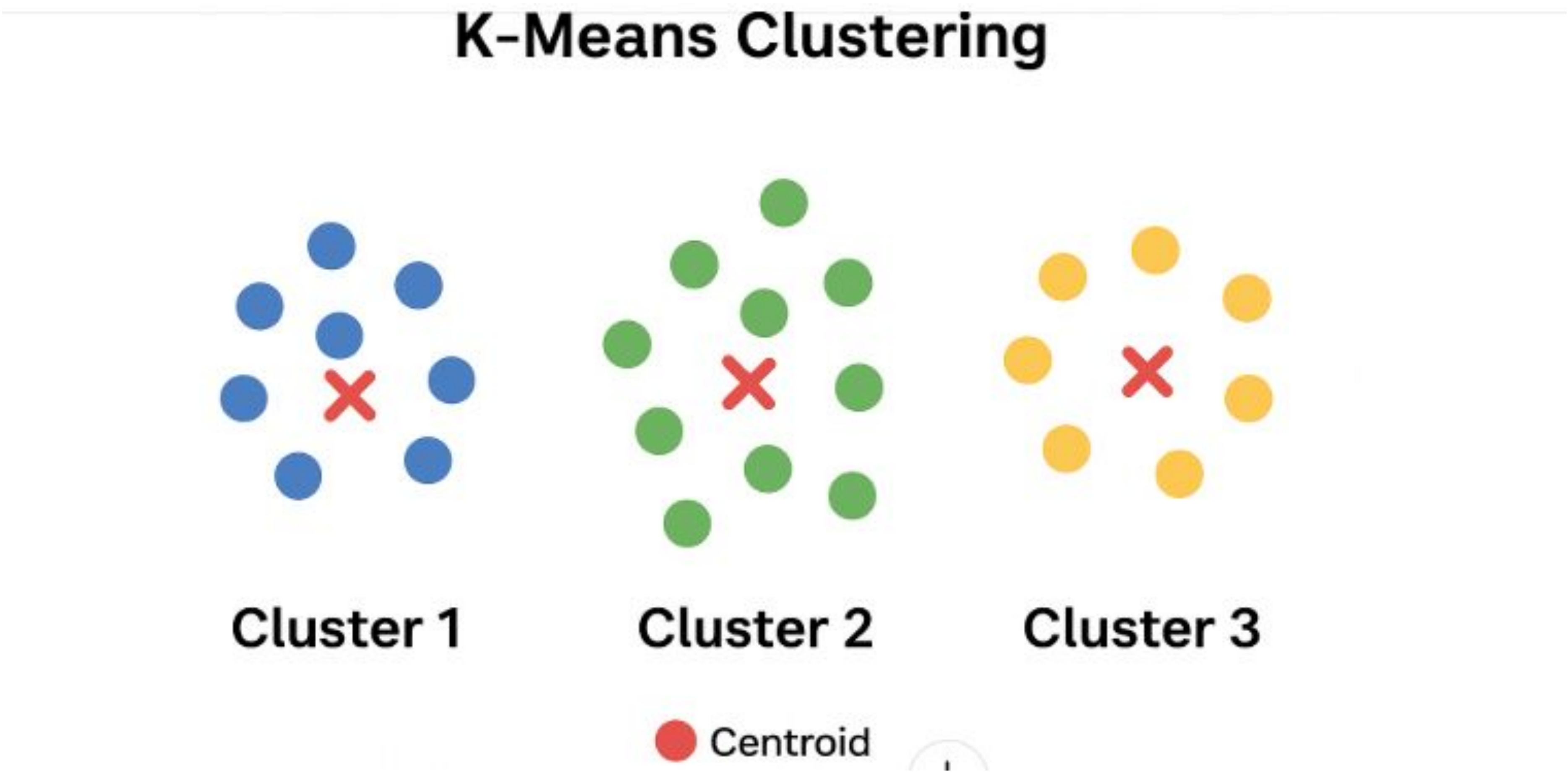
✅ **Advantages:**

1. ✅ **Simple and fast** to use.
2. ✅ Works well with **large datasets**.
3. ✅ Easy to **understand and interpret**.
4. ✅ Works well when clusters are **clearly separated**.

❌ Disadvantages:

1. ❌ You must **choose K (number of clusters)** beforehand.
2. ❌ Doesn't work well with **uneven cluster sizes** or **non-spherical shapes**.
3. ❌ Sensitive to **outliers** (they can pull centroids away).
4. ❌ Different starting points can give **different results**.

from sklearn.cluster import KMeans



8

Definition:

Mean Shift is an **unsupervised clustering algorithm** that **finds clusters by locating the densest regions of data points**. It does not require you to predefine the number of clusters. The algorithm works by **shifting data points toward areas of higher density iteratively** until convergence.

Advantages:

1. **No need to predefine clusters:** Unlike K-Means, you don't have to decide the number of clusters beforehand.
2. **Can find arbitrarily shaped clusters:** Works well with clusters that are not circular.
3. **Robust to outliers:** Less sensitive to outliers compared to some other clustering algorithms.

Disadvantages:

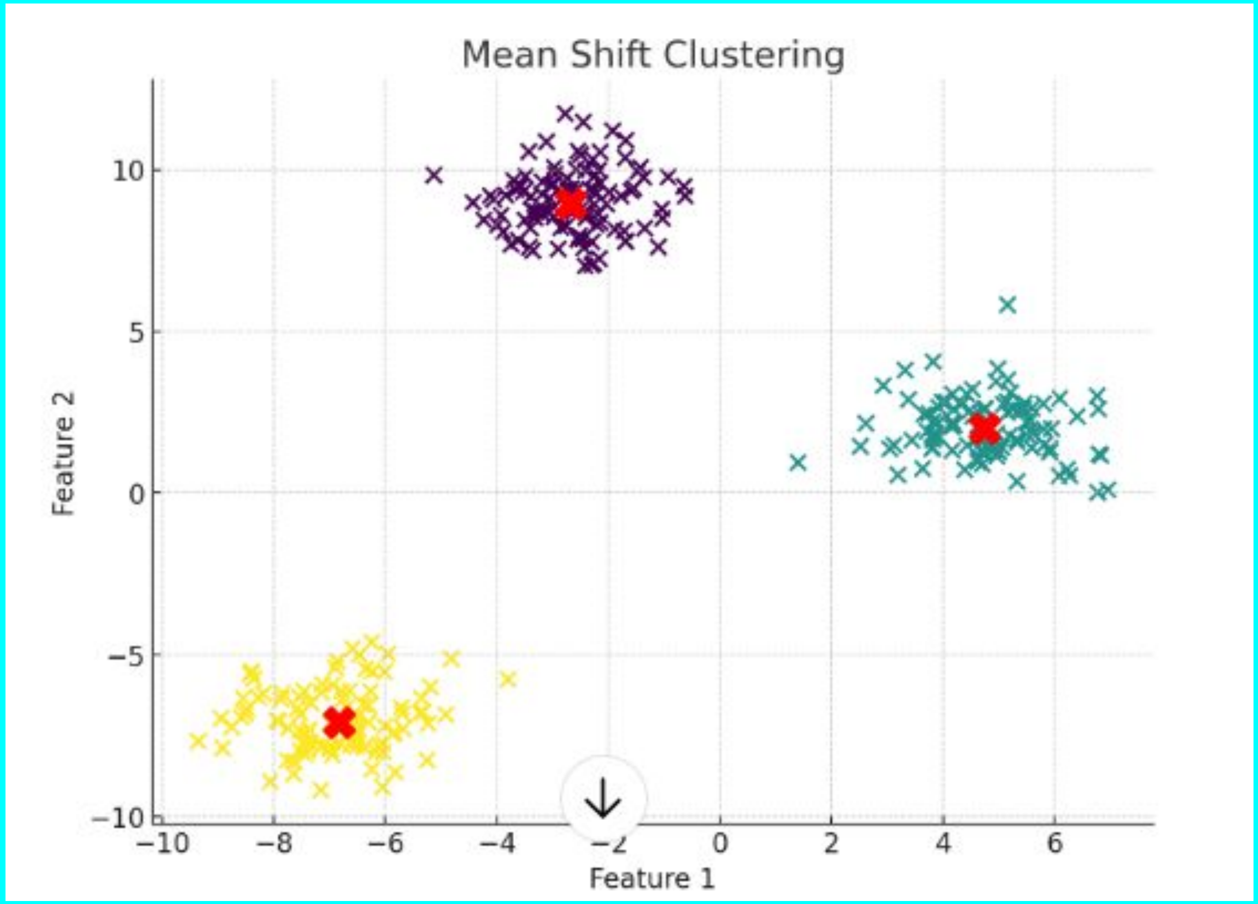
1. **Computationally expensive:** Especially with large datasets, because it calculates distances repeatedly.
2. **Bandwidth selection is tricky:** The window size (bandwidth) affects results a lot; too small → too many clusters, too large → clusters merge.
3. **Not suitable for very high-dimensional data:** Performance drops when dimensions increase.

from sklearn.cluster import Mean Shift Clustering

Colored points represent the data points.

Different colors show different clusters discovered by the algorithm.

Red Xs are the **cluster centers** found by Mean Shift.



Definition

OPTICS stands for **Ordering Points To Identify the Clustering Structure**.

It is a **density-based clustering algorithm**, similar to DBSCAN, but more flexible.

- Unlike DBSCAN, which needs a fixed density threshold (`eps`) to form clusters, OPTICS can detect clusters **with varying densities**.
- Instead of producing a strict cluster assignment, it produces an **ordering of points** and a **reachability distance**, which can then be used to extract clusters at different density levels.

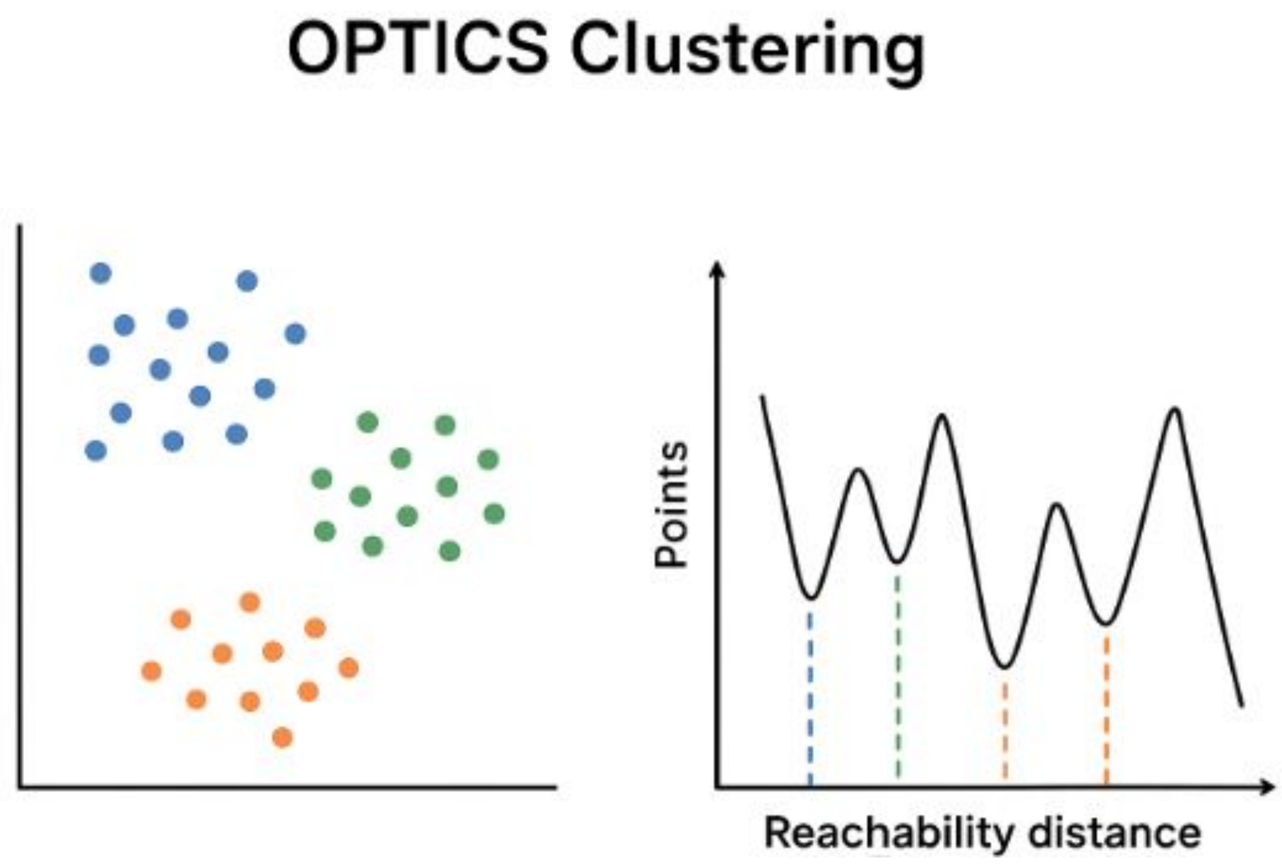
Advantages

1. **Handles varying densities** – unlike DBSCAN, it doesn't require a single `eps`.
2. **Detects arbitrary shapes** – works for circular, elongated, or irregular clusters.
3. **Noise handling** – identifies outliers naturally.
4. **Flexible cluster extraction** – you can extract clusters at multiple density levels.

Disadvantages

1. **More complex than DBSCAN** – harder to understand and implement.
2. **Slower** – especially for large datasets (complexity: $O(n \log n)$ with indexing, $O(n^2)$ without).
3. **Visualization required** – to extract clusters, you often need a reachability plot.
4. **Parameters still needed** – `minPts` (minimum points for a dense region) must be set correctly.

from sklearn.cluster import OPTICS



Definition

OPTICS stands for **Ordering Points To Identify the Clustering Structure**.

It is a **density-based clustering algorithm**, similar to DBSCAN, but more flexible.

- Unlike DBSCAN, which needs a fixed density threshold (`eps`) to form clusters, OPTICS can detect clusters **with varying densities**.
- Instead of producing a strict cluster assignment, it produces an **ordering of points** and a **reachability distance**, which can then be used to extract clusters at different density levels.

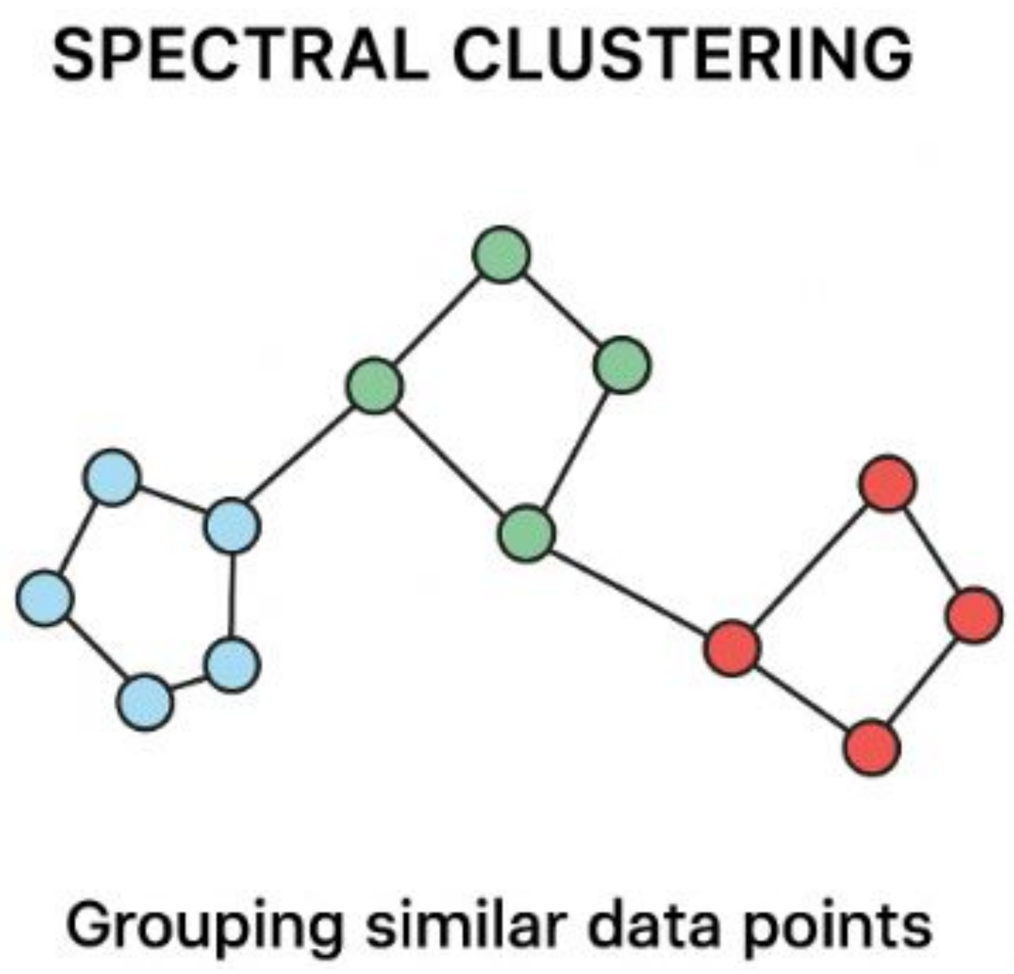
Advantages:

1. Can detect **non-convex clusters** that K-Means cannot.
2. Works well with **complex cluster structures**.
3. Doesn't assume clusters are spherical.
4. Flexible: you can choose different similarity measures.

Disadvantages:

1. Computationally expensive for **large datasets** (needs eigen decomposition).
2. Sensitive to **choice of similarity function** and parameters.
3. Number of clusters must often be specified beforehand.
4. Not very scalable to **millions of points**.

from sklearn.cluster import SpectralCustering



🧠 Definition:

The **Silhouette Score** is a **measure of how well data points fit within their assigned clusters**.

It tells us how **separated** and **well-formed** the clusters are.

💡 Typical Good Values:

- **0.7 – 1.0** → Excellent clustering
- **0.5 – 0.7** → Reasonable
- **0.25 – 0.5** → Weak
- **< 0.25** → Poor clustering

🧠 Definition:

The **Davies–Bouldin Index** is a **measure used to evaluate clustering quality**.

It checks **how well-separated** and **compact** the clusters are.

Davies–Bouldin Index	Meaning
0	Perfect clustering (clusters are very distinct)
Lower value	Better clustering
Higher value	Poor clustering (clusters overlap)

🧠 Definition:

The **Calinski–Harabasz Index**, also called the **Variance Ratio Criterion**, is a **measure of how well the data is clustered**.

It checks how **compact** each cluster is and how **separated** the clusters are from each other.

📏 Range of Values:

Calinski–Harabasz Score	Meaning
Higher value	Better clustering (well-separated and compact)
Lower value	Poor clustering (overlapping or scattered clusters)

There's **no fixed range**, but **higher is always better**.

Clustering Name	Silhouette Score	Davies-Bouldin Index	Calinski-Harabasz Index	Better_Clustering?
AffinityPropagation_clustering	0.432	0.753	264.4707125	
Agglomerative_Clustering	0.553	0.578	243.0714289	
Birch_clustering	0.323	0.288	1050.99371	
BisectingKMeans_Clustering	0.446	0.823	136.2364581	
DBSCAN_Clustering	0.096	6.71	2.020445139	
HDBSCAN_Clustering	0.405	1.794	57.35299571	
MeanShift_Clustering	0.271	0.216	1378.828452	Better_Clustering
Optics_Clustering	0.292	1.623	14.29378177	
SpectralClustering	0.340	1.333	18.96466618	