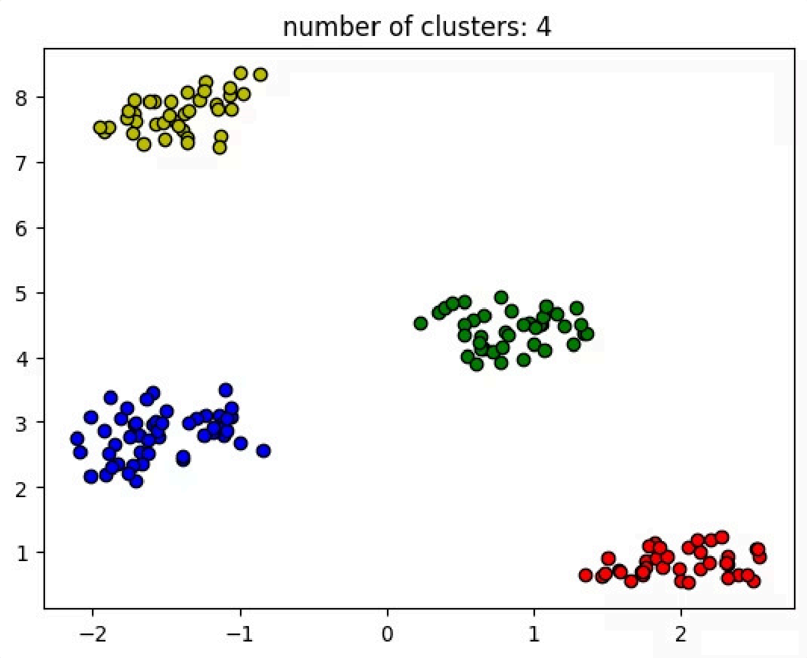


Clustering: K-Means and DBSCAN

NDL Lab 5 - 11/9/2025

Attendance: "Clustering"

Eboard Applications

What is Clustering?

Unsupervised Grouping

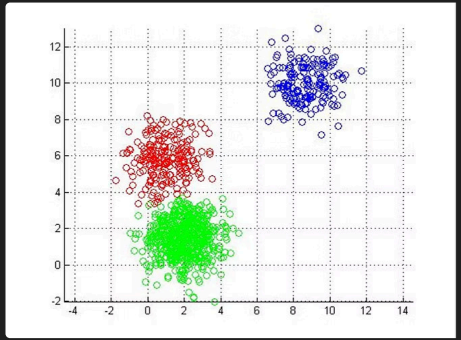
Clustering is an unsupervised learning technique used to group similar data points together based on inherent patterns.

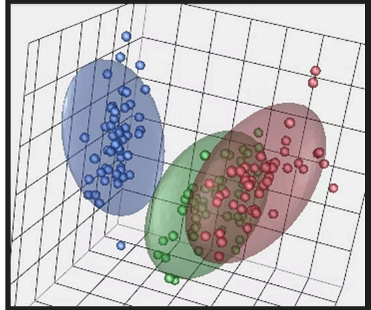
Key Applications

It is fundamental for pattern discovery, effective data segmentation, and identifying anomalies or outliers within a dataset.

Real-World Impact

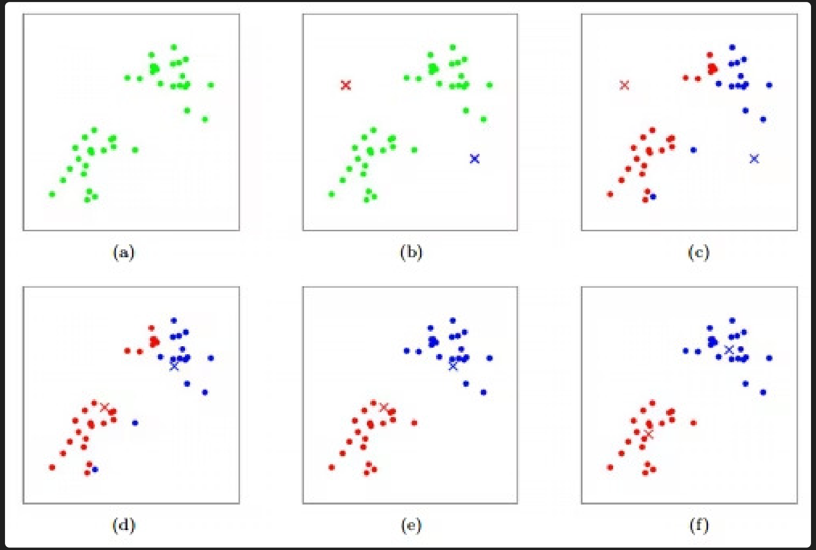
- Customer Segmentation
- Image Recognition & Compression
- Bioinformatics (Gene analysis)





Introducing K-Means Clustering

K-Means is one of the simplest and most popular partition-based clustering algorithms. It aims to partition **N** observations into **K** distinct, non-overlapping clusters.



- **The Goal:** Minimize the variance within each cluster, ensuring high intra-cluster similarity.
- **The Process:** It follows an iterative refinement approach: assign data points to the nearest centroid, then update the centroid position. This repeats until cluster assignments stabilize.

K-Means is intuitive and highly scalable for massive datasets with a clear structure.

How K-Means Works: Step-by-Step

The K-Means algorithm systematically refines cluster definitions through iteration until convergence is reached.

1

Define K

Choose the number of clusters (K) that the data must be partitioned into.

2

Initialize Centroids

Randomly select K initial centroids, representing the center of each cluster.

3

Assign Points

Assign every data point to its closest centroid, typically using Euclidean distance.

4

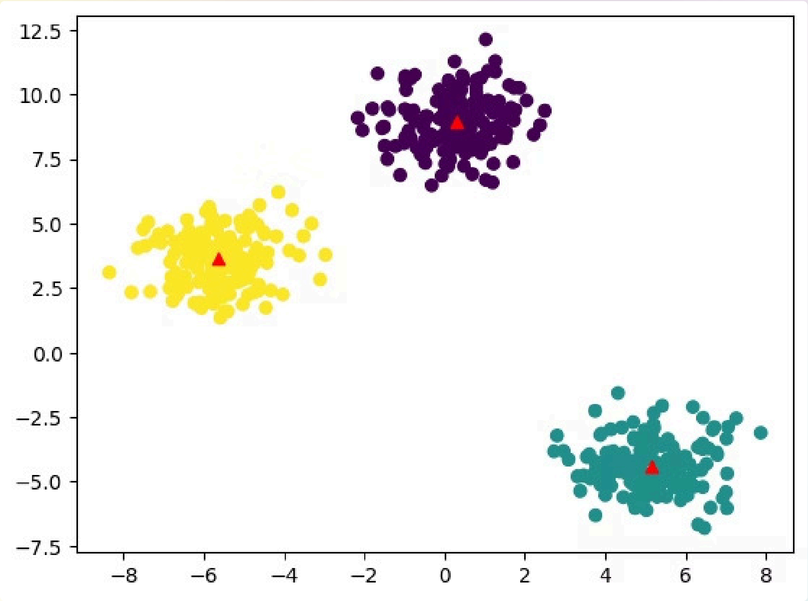
Recalculate Centroids

Compute the mean of all points assigned to a cluster; this new mean becomes the updated centroid.

5

Repeat

Continue the assignment and recalculation steps until the centroids no longer move significantly (convergence).



K-Means: Strengths and Limitations

Strengths

Speed & Efficiency

Fast and computationally efficient, making it excellent for very large datasets.

Simplicity

Easy to understand and implement, with results that are straightforward to interpret.

Limitations

Parameter Sensitivity

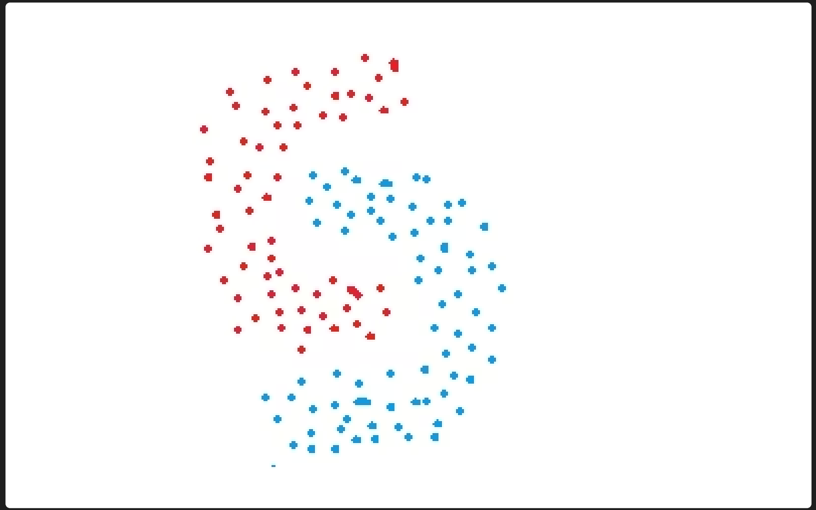
Dependent on number of clusters and initial positions of centroids

Shape Assumption

Performs poorly when clusters are not spherical, unevenly sized, or have varying densities or outliers

Colab

Introducing DBSCAN: Density-Based Clustering



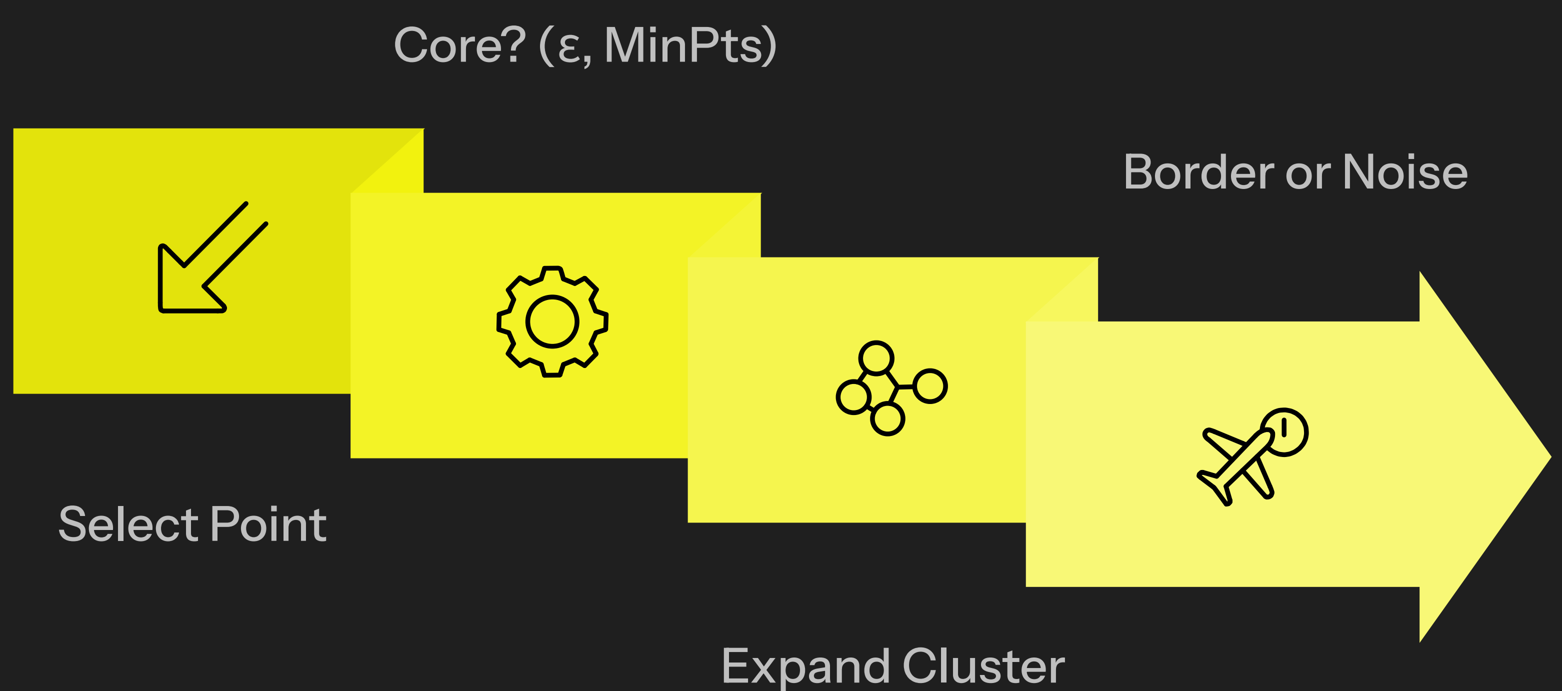
DBSCAN (Density-Based Spatial Clustering of Applications with Noise) approaches clustering from a different perspective, focusing on the density of data points.

> It identifies clusters as areas of high density, separated by areas of low density, allowing it to discover clusters of **arbitrary shape**.

- Groups points based on what other points they are close to
- Automatically separates points into **Core**, **Border**, and **Noise** (outliers).
- Excels where K-Means fails: recognizing non-spherical clusters and explicitly handling noise.

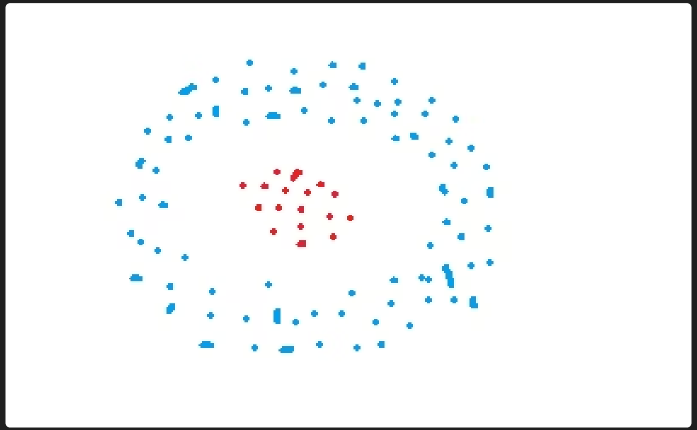
How DBSCAN Works: Step-by-Step

DBSCAN builds clusters by iteratively connecting neighboring points.



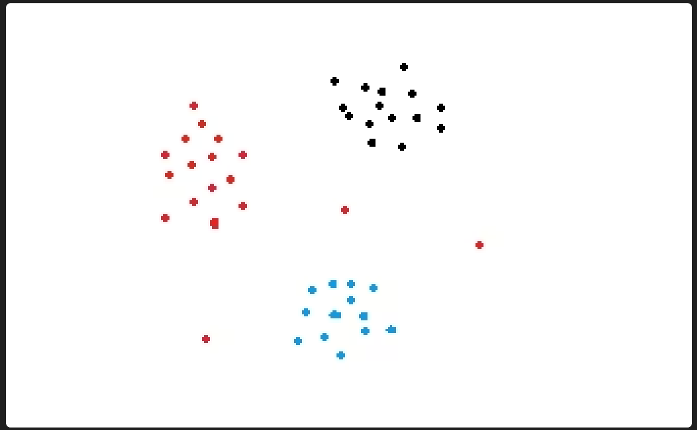
Core Point	Border Point	Noise Point
A point with at least **MinPts** neighbors (other points) within the distance **ε** (epsilon).	A point that is within **ε** of a Core Point but has fewer than **MinPts** neighbors (not a Core Point)	A point that is neither a Core Point nor a Border Point. These are considered outliers.

DBSCAN: Strengths and Limitations



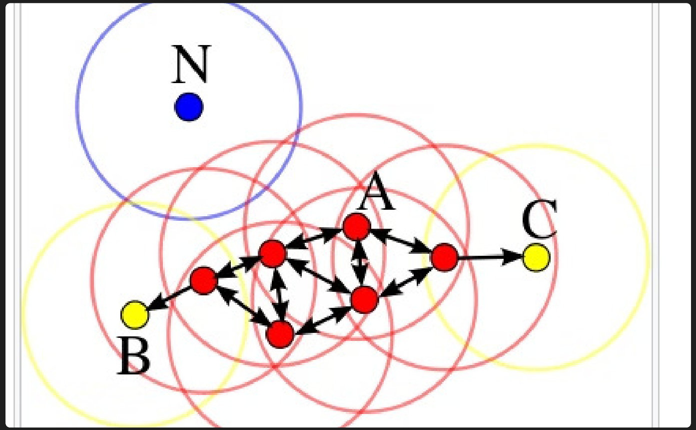
Handles Complex Shapes

Its density-based approach allows it to identify clusters of arbitrary shapes



Robust to Noise

Outliers are explicitly labeled as noise, minimizing their impact on the final clustering result.



Parameter Sensitivity

Performance is highly dependent on setting the optimal values for **ε** (radius) and **MinPts**.

Choosing Between K-Means and DBSCAN

The choice of algorithm depends entirely on the nature and structure of your data.

1 K-Means

Ideal for structured, clean data where cluster boundaries are well-defined.

- Clusters are **spherical** and balanced.
- You **know K** (number of clusters) beforehand.
- Focus is on scalability and speed.

2 DBSCAN

Essential for data with natural gaps, complex shapes, or significant noise contamination.

- Clusters have **irregular shapes** or varying densities.
- You want to **identify noise**/outliers explicitly.
- The number of clusters is unknown or variable.

Experimentation is key: Test both methods if your initial assumptions about the data structure are unclear.

Summary & Takeaway

Data Guides Choice

Cluster shape, density, and noise level are the primary factors determining which algorithm is superior.

Different Tools

K-Means optimizes for partition homogeneity; DBSCAN optimizes for density connectedness and noise detection.

Practical Application

Explore implementations in scikit-learn (Python) to apply these concepts hands-on.

Complementary Insights

Use both methods to gain a comprehensive understanding of complex datasets.

Selecting the right clustering algorithm transforms raw data into meaningful segments for actionable insights.