DBSCAN

Luu Minh Sao Khue

December 17, 2025

1. What is DBSCAN Clustering?

1.1 Motivation and problem setting

Many clustering algorithms, such as K-Means, assume that clusters can be represented by a single center and that clusters are roughly spherical. However, real-world data often violate these assumptions. Clusters may have arbitrary shapes, varying sizes, and may be contaminated by noise or outliers.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) addresses these challenges by defining clusters as **dense regions of points** separated by regions of low density. Rather than specifying the number of clusters in advance, DBSCAN identifies clusters based on local neighborhood density and explicitly labels sparse points as noise.

1.2 Key ideas of DBSCAN

DBSCAN is built on three core ideas:

- clusters correspond to regions of high point density,
- points in low-density regions are treated as noise,
- connectivity between dense regions determines cluster structure.

As a result, DBSCAN can discover clusters of arbitrary shape and is robust to outliers.

2. Notation and Data Representation

Let:

- n denote the number of data points,
- d denote the dimensionality of each data point.

Each data point is represented as

$$x_i \in \mathbb{R}^d, \quad i = 1, \ldots, n.$$

DBSCAN does not maintain centroids. Instead, it relies on local neighborhoods defined by a distance threshold.

3. Distance Measure and Neighborhoods

3.1 Distance function

DBSCAN requires a distance function to measure similarity between points. In most applications, the Euclidean distance is used:

$$\|x_i - x_j\|_2 = \sqrt{\sum_{k=1}^{d} (x_{ik} - x_{jk})^2}.$$

Other distance metrics may be used depending on the data representation.

3.2 ε-neighborhood

Given a distance threshold $\varepsilon > 0$, the **ε-neighborhood** of a point x_i is defined as

$$\mathcal{N}_\varepsilon(x_i) = \{x_j \in \mathbb{R}^d : \|x_i - x_j\|_2 \leq \varepsilon\}.$$

This set contains all points whose distance from x_i does not exceed ε.

4. Density Concepts in DBSCAN

DBSCAN classifies points based on the number of neighbors in their ε-neighborhood.

4.1 Core points

A point x_i is called a **core point** if

$$|\mathcal{N}_\varepsilon(x_i)| \geq \text{MinPts},$$

where MinPts is a user-specified minimum number of neighbors.

Core points lie in dense regions and form the interior of clusters.

4.2 Border points

A point x_i is a **border point** if

$$|\mathcal{N}_\varepsilon(x_i)| < \text{MinPts}$$

but x_i lies within the ε-neighborhood of a core point.

Border points are part of a cluster but do not themselves satisfy the density requirement.

4.3 Noise points

A point that is neither a core point nor a border point is labeled as **noise** (or an outlier). Noise points do not belong to any cluster.

5. Density Connectivity

5.1 Direct density reachability

A point x_j is **directly density-reachable** from a point x_i if:

- x_i is a core point,
- $x_j \in \mathcal{N}_\varepsilon(x_i)$.

This relation is not symmetric.

5.2 Density reachability

A point x_j is **density-reachable** from x_i if there exists a sequence of points

$$x_i = p_1, p_2, \ldots, p_m = x_j$$

such that each p_{k+1} is directly density-reachable from p_k.

5.3 Density connectivity

Two points x_i and x_j are **density-connected** if there exists a point x_k such that both x_i and x_j are density-reachable from x_k.

Clusters in DBSCAN are defined as maximal sets of density-connected points.

6. The DBSCAN Algorithm

6.1 Input parameters

DBSCAN requires two parameters:

- ε: neighborhood radius,
- MinPts: minimum number of points required to form a dense region.

6.2 Algorithm description

Input: data points $\{x_i\}_{i=1}^n$, parameters ε and MinPts.

1. Mark all points as unvisited.
2. For each unvisited point x_i:
   (a) Mark x_i as visited.
   (b) Compute $\mathcal{N}_\varepsilon(x_i)$.
   (c) If $|\mathcal{N}_\varepsilon(x_i)| <$ MinPts, label x_i as noise.
   (d) Otherwise:
      i. create a new cluster,
      ii. add all points in $\mathcal{N}_\varepsilon(x_i)$ to the cluster,
      iii. recursively expand the cluster by visiting all density- reachable points.
3. Output cluster labels and noise points.

6.3 Stopping condition

The algorithm terminates when all points have been visited exactly once.

7. Evaluation of DBSCAN Clustering

7.1 Number of clusters and noise

DBSCAN automatically determines the number of clusters. The result consists of:

- a set of clusters,
- a set of points labeled as noise.

The fraction of noise points provides insight into data sparsity and parameter selection.

7.2 Internal evaluation metrics

Standard clustering metrics can be applied to DBSCAN results:

- silhouette score (computed only on non-noise points),
- Davies–Bouldin index,
- Calinski–Harabasz index.

These metrics evaluate compactness and separation but should be interpreted with care in the presence of noise.

8. Computational Complexity

Without spatial indexing, DBSCAN requires computing pairwise distances, leading to a time complexity of

$$O(n^2).$$

When spatial data structures such as KD-trees or ball trees are used, the average complexity improves to approximately

$$O(n \log n),$$

depending on data dimensionality.

9. Assumptions and Limitations

DBSCAN performs well when clusters are dense and well separated by sparse regions. However, it has limitations:

- sensitivity to parameter selection,
- difficulty handling clusters with varying densities,
- reduced effectiveness in high-dimensional spaces due to distance concentration.

10. Comparison Between K-Means and DBSCAN

K-Means and DBSCAN are both unsupervised clustering algorithms, but they are based on different principles. K-Means is a *centroid-based* method that minimizes within-cluster variance, while DBSCAN is a *density-based* method that identifies clusters as connected regions of high point density. As a result, the two algorithms are suitable for different data characteristics and analysis goals.

Table 1: Comparison of K-Means and DBSCAN

Aspect	**K-Means**	**DBSCAN**
Clustering principle	Centroid-based (minimizes squared distance to centroids)	Density-based (groups dense regions)
Number of clusters	Must be specified in advance (K)	Determined automatically
Cluster shape	Compact, spherical clusters	Arbitrary shapes
Cluster density	Assumes similar densities	Assumes locally dense regions
Noise and outliers	All points are assigned to clusters	Explicitly labels noise points
Sensitivity to outliers	High (centroids can be distorted)	Low (outliers excluded)
Distance usage	Global distance to centroid	Local neighborhood distance
Scalability	Highly scalable to large datasets	More expensive without spatial indexing
Interpretability	Centroids provide clear summaries	No explicit cluster representatives
Typical use cases	Large datasets with well-separated, compact clusters	Data with noise and irregular cluster shapes

In practice, K-Means is preferred when the number of clusters is known, clusters are compact, and computational efficiency is important. DBSCAN is preferred when the number of clusters is unknown, clusters may have arbitrary shapes, and identifying noise or outliers is essential.