

Density-Based Spatial Clustering of Applications with Noise

Comprehensive Notes with Equations, Diagrams, and Implementation



Table of Contents

Introduction to DBSCAN

Core Concepts and Definitions

The DBSCAN Algorithm

Mathematical Formulation

Parameters and Their Effects

Implementation and Code Examples

Time and Space Complexity

Advantages and Disadvantages

Comparison with Other Algorithms

Real-world Applications

Variants and Extensions



Introduction to DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a powerful density-based clustering algorithm introduced by Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu in 1996. Unlike centroid-based algorithms like K-means, DBSCAN can discover clusters of arbitrary shapes and automatically determine the number of clusters.



Key Features of DBSCAN:

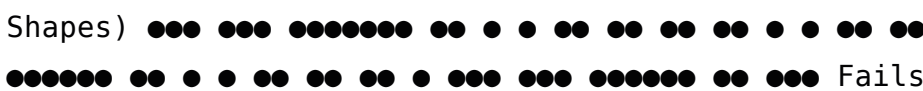
- **Density-Based:** Groups points that are closely packed
- **Noise Handling:** Identifies and handles outliers effectively
- **Arbitrary Shapes:** Can find clusters of any shape
- **Automatic:** No need to specify the number of clusters
- **Robust:** Relatively insensitive to parameter choices



Visual Comparison: K-means vs DBSCAN

Diagram

K-means (Spherical Clusters) DBSCAN (Arbitrary Shapes)



Fails with non-spherical Successfully identifies clusters and noise complex shapes and noise



Core Concepts and Definitions

Point Classifications



Core Point

Has at least **MinPts** neighbors within ϵ distance



Border Point

Within ϵ distance of a core point but has fewer than MinPts neighbors



Noise Point

Neither a core point nor a border point (outlier)

Key Definitions

ϵ (Epsilon)

The maximum distance between two points for them to be considered neighbors

MinPts

The minimum number of points required to form a dense region (including the point itself)

ϵ -Neighborhood

$$N_{\epsilon}(p) = \{q \in D \mid \text{dist}(p, q) \leq \epsilon\}$$

Density Reachable

Point q is density reachable from p if there exists a chain of core points from p to q

Visual Representation of Core Concepts

 Diagram

DBSCAN Point Classification Example ($\epsilon = 2$, MinPts = 4)

A ● — ● B ● F (Noise) | \ / | | ● | | / \ | C ● — ● D | | ● ● E


Legend: ● A, B, C, D = Core Points (≥ 4 neighbors within $\epsilon=2$) ● E = Border Point (< 4 neighbors but within ϵ of core point) ● F = Noise Point (isolated, < 4 neighbors and not within ϵ of core)

Cluster Formation: - Cluster 1: {A, B, C, D, E} (all density-connected) - Noise: {F}

Mathematical Formulation

Distance Metrics

DBSCAN can work with various distance metrics. The most common is Euclidean distance:

 Euclidean Distance: $d(p, q) = \sqrt{(\sum_i (p_i - q_i)^2)}$

Manhattan Distance: $d(p,q) = \sum_i |p_i - q_i|$

Minkowski Distance: $d(p,q) = (\sum_i |p_i - q_i|^p)^{1/p}$

Core Mathematical Definitions

ϵ -Neighborhood: $N_\epsilon(p) = \{q \in D \mid \text{dist}(p,q) \leq \epsilon\}$

Core Point Condition: $|N_\epsilon(p)| \geq \text{MinPts}$

Directly Density-Reachable: $q \in N_\epsilon(p) \wedge |N_\epsilon(p)| \geq \text{MinPts}$

Density-Connected: $\exists o \in D : p \rho^* o \wedge q \rho^* o$

Where ρ^* denotes density-reachability relation.

Cluster Definition



A cluster C in DBSCAN is defined as a maximal set of density-connected points:



$$C = \{p \in D \mid \exists q \in C : p \text{ is density-reachable from } q\}$$



The DBSCAN Algorithm



Algorithm

DBSCAN Pseudocode

PSEUDOCODE

Algorithm DBSCAN($D, \epsilon, \text{MinPts}$) Input: Dataset D , radius ϵ , minimum points MinPts Output: Set of clusters C

1. Initialize: $C = 0$ (cluster counter), Mark all points as UNVISITED
2. For each point p in D :
 - a) If p is VISITED, continue to next point
 - b) Mark p as VISITED
 - c) $\text{NeighborPts} = \text{regionQuery}(p, \epsilon)$
 - d) If $|\text{NeighborPts}| < \text{MinPts}$: - Mark p as NOISE
 - e) Else: - $C = C + 1$ (increment cluster counter) - **expandCluster**($p, \text{NeighborPts}, C, \epsilon, \text{MinPts}$)

Function expandCluster($p, \text{NeighborPts}, C, \epsilon, \text{MinPts}$)

1. Add p to cluster C
2. For each point p' in NeighborPts :
 - a)

```
If p' is UNVISITED: - Mark p' as VISITED -
NeighborPts' = regionQuery(p', ε) - If
|NeighborPts'| ≥ MinPts: * NeighborPts = NeighborPts
u NeighborPts' b) If p' is not member of any
cluster: - Add p' to cluster C Function
regionQuery(p, ε) 1. Return all points q where
dist(p,q) ≤ ε
```

Step-by-Step Algorithm Execution

1

Initialization

Mark all points as unvisited and initialize cluster counter to 0

2

Point Selection

Select an unvisited point and mark it as visited

3

Neighborhood Query

Find all points within ϵ distance (ϵ -neighborhood)

4

Core Point Check

If neighborhood has \geq MinPts points, it's a core point

5

Cluster Expansion