Density-Based Clustering

Yufei Tao

Department of Computer Science and Engineering
Chinese University of Hong Kong

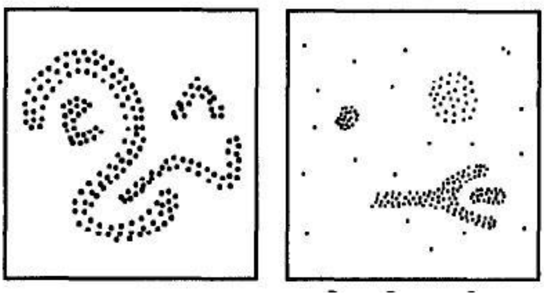
We will continue our discussion on clustering. Recall that at a high level, this problem can be stated as follows.

Let P be a set of objects to be clustered. We want to divide P into several groups—each of which is called a cluster—satisfying the following conditions:

- (Homogeneity) Objects in the same cluster should be similar to each other.
- (Heterogeneity) Objects in different clusters should be dissimilar.

In this lecture, we will consider that each object is a point in $\mathbb{R}^d$.

In some applications of reality, clusters can have arbitrary shapes:



(Excerpted from a KDD96 paper titled "A Density-based algorithm for discovering clusters in large spatial databases with noise")

Why do we care about such clusters?

- Consider each point to be a spatial location (e.g., a place where burglary has happened). A cluster involves all the locations in the same "district" (e.g., a residential area), which can have an arbitrary shape.
- In optical character recognition (OCR), we are given a picture of some letters (e.g., a photoed license plate) and want to have a computer recognize the letters automatically. In a preprocessing stage, often times we need to smooth the edges of the letters by removing the noise pixels. The remaining pixels are cut into clusters, each of which corresponds to a letter.

Today we will learn a classic algorithm called DBSCAN for discovering clusters of arbitrary shapes. This algorithm is also a representative method of density-based clustering.

DBSCAN works based on the following rationale. If a point p is in a cluster C, then intuitively at least one of the following should hold:

- there are many points around p inside C—in this case, we say that p is a core point.
- p is close to a core point of C.

As a side product, DBSCAN can also identify some points in P as outliers (i.e., noise), which are the points satisfying neither of the above conditions.

We will formalize these notions in the next few slides.

To run DBSCAN, we need to specify two parameters:

- A distance r;
- A threshold t.

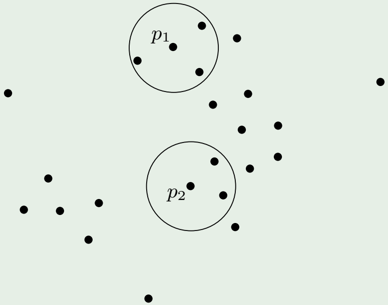
Definition 1 (Neighborhood).

Given a point p, its neighborhood is the circle centered at p with radius r.

Definition 2 (Core Point).

A point $p \in P$ is a core point if its neighborhood covers at least t points of P.

Example 3.



Suppose that r is the radius of the two circles shown, and $t = 4$. Then p_1 is a core point, but p_2 is not.

Definition 4 (Reachability).

If there is a sequence of points $p_1, ..., p_k$ $(k \geq 2)$ in P such that

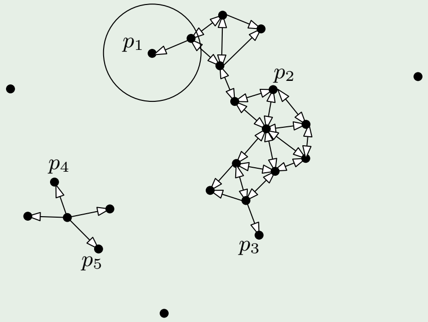
- $p_1, ..., p_{k-1}$ are all core points
- p_{i+1} is in the neighborhood of p_i for each $i \in [1, k-1]$

then we say that p_k is reachable from p_1.

Alternatively, you can look at this from the a graph perspective. Imagine adding a directed edge from each core point p to all the points in its neighborhood. Then, q is reachable from p if and only if you can find a path from p to q on the graph we have created.

Note that reachability is not symmetric.

Example 5.



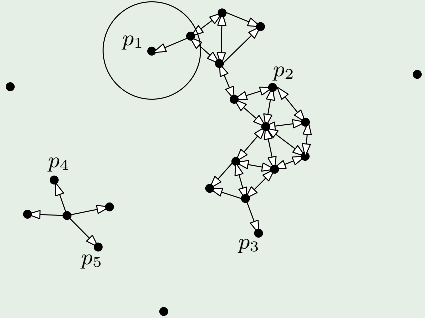
Suppose that r is the radius of the circle shown, and $t = 4$. Then:

- p_1 is reachable from p_2, but not from p_3.
- p_2 is not reachable from p_1, nor from p_3.

Definition 6 (Connected).

Two points p_1, p_2 in P are connected if there is a point $p \in P$ such that both p_1 and p_2 are reachable from p.

Example 7.



p_1 and p_3 are connected, and so are p_4 and p_5. However, p_1 and p_4 are not.

Definition 8 (Cluster).

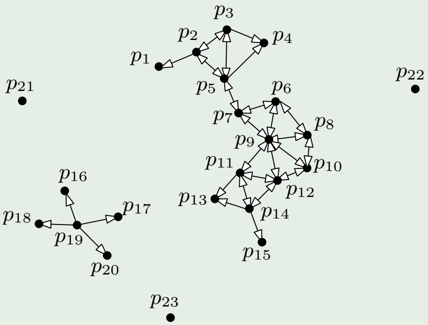
Let C be a subset of P. We say that C is a cluster if it satisfies two conditions:

- (Maximality) if a point $p \in C$ is a core point and q is reachable from p, then $q \in C$.
- (Connectivity) any two points $p, p' \in C$ (possibly $p = p'$) must be connected.

The connectivity requirement implies that each cluster must contain at least one core point.

If a point $p \in C$ is not a core point, it is called a border point.

Example 9.



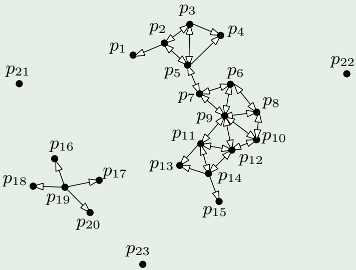
- $\{p_1\}$ is not a cluster. Neither $\{p_1, p_4\}$ nor $\{p_2\}$ is.
- $\{p_1, p_2, .., p_{15}\}$ is a cluster. But if we leave out any point from the set, then it is not a cluster.
- $\{p_1, p_2, .., p_{20}\}$ is a not a cluster.
- Border points: $p_1, p_4, p_{13}, p_{15}, p_{16}, p_{17}, p_{18}, p_{20}$.

Theorem 10 (Uniqueness).

The set $\mathcal{C}$ of clusters in P is unique.

We will prove the theorem in the next few slides.

Example 11.



$\mathcal{C} = \{\{p_1, ..., p_{15}\}, \{p_{16}, ..., p_{20}\}\}$. Note that points p_{21}, p_{22}, and p_{23} are not in any clusters; they are outliers.

- The number of clusters in $\mathcal{C}$ is not a parameter to the problem. It depends on the distribution of the data points.
- Note that the clusters in $\mathcal{C}$ are not necessarily disjoint. (Think: Why?)

To prove the uniqueness theorem, we will first prove some lemmas.

Lemma 12.

Let p be a core point, and q any point in P. If p and q are connected, then q is reachable from p.

Proof.

If q is in the neighborhood of p, then we are done. Otherwise, by definition, there exist two sequences of core points $(z_1, z_2, ..., z_a)$ and $(z'_1, z'_2, ..., z'_b)$ such that

- z_{i+1} is in the neighborhood of z_i ($i \in [1, a]$), and p is in the neighborhood of z_a;
- z'_{i+1} is in the neighborhood of z'_i ($i \in [1, b]$), and q is in the neighborhood of z'_b;
- $z_1 = z'_1$.

Hence, q is reachable from p by the following sequence: $p, z_a, z_{a-1}, ..., z_1, z'_2, ..., z'_b, q$. $\square$

To prove the uniqueness theorem, we will first prove some lemmas.

Lemma 13.

If a cluster C includes a core point p, then C consists of exactly all the points reachable from p.

Proof.

Let S be the set of points reachable from p. We will prove $C \subseteq S$ and $S \subseteq C$.

Proof of $S \subseteq C$: By the cluster definition.

Proof of $C \subseteq S$: Consider any point $q \in C$. By the cluster definition, p and q are connected. From Lemma 12, we know that q is reachable from p. $\square$

Now we are ready to prove the uniqueness theorem.

Proof of Theorem 10

Suppose that there are two different solutions $\mathcal{C}$ and $\mathcal{C}'$ to the DBSCAN problem. Then, there is a cluster $C \in \mathcal{C}$ but $C \notin \mathcal{C}'$.

As mentioned earlier, every cluster must contain at least a core point. Let p be an arbitrary core point of C. Let $C' \in \mathcal{C}'$ be an arbitrary cluster containing p. From Lemma 13, we know that $C = C'$, contradicting the fact that $C \notin \mathcal{C}'$. $\square$

Next, we give an algorithm for solving the DBSCAN problem. First, we construct the graph G as mentioned earlier in Slide 8. Formally, G is a directed graph where each vertex is a distinct point $p \in P$, and there is a directed edge from p to q if p is a core point, and q is in the neighborhood of p.

G can be easily constructed in $O(n^2)$ time.

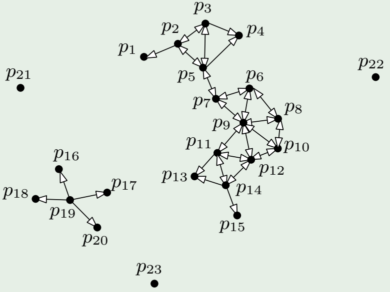
Initially, we label all the points of P as unclustered. Let V_{core} be all the core points of P. Also, initialize an empty set $\mathcal{C}$.

While V_{core} is not empty, we take a point p from V_{core}, and find the set $S(p)$ of points in P that are reachable from p. This can easily done by performing a breath first search on G from p. We add $S(p)$ as a new cluster to $\mathcal{C}$. Label all the points $S(p)$ as clustered. Remove all the core points in $S(p)$ from V_{core}. Then, repeat this procedure.

When V_{core} becomes empty, we check whether there are still points in P carrying the label unclustered. If so, output them as outliers.

The algorithm runs in $O(n^2)$ time.

Example 14.



- $V_{core} = \{p_2, p_3, p_5, ..., p_{12}, p_{14}, p_{19}\}$
- We first find $S(p_2) = \{p_1, ..., p_{15}\}$ as the first cluster. These points are labeled as "clustered". Now $V_{core} = \{p_{19}\}$.
- Then we find $S(p_{19}) = \{p_{16}, ..., p_{20}\}$ as the second cluster. These points are labeled as "clustered".
- Points p_{21}, p_{22}, p_{23} are still labeled as "unclustered". They are reported as outliers.

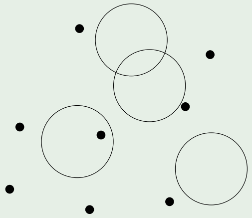
The $O(n^2)$ time complexity of our algorithm is rather expensive in practice, and essentially limits the applicability of DBSCAN to small datasets.

For a long time, the data mining community had been looking for an algorithm with complexity $O(n \log^c n)$ for constant dimensionality d (where c can be any constant). It turned out that this is impossible, unless unlikely breakthroughs could be made in theoretical computer science. In the absence of those breakthroughs, even for $d = 3$, all DBSCAN algorithms must incur $\Omega(n^{4/3})$ time in the worst case, as shown in the next few slides.

Let us now introduce the unit-spherical emptiness checking (USEC) problem:

> Let S_{pt} be a set of points, and S_{ball} be a set of balls *with the same radius*, all in data space $\mathbb{R}^d$, where the dimensionality d is a constant. The objective of USEC is to determine whether there is a point of S_{pt} that is covered by some ball in S_{ball}.

Example 15.



For the above input points and balls, the answer is *yes*.

Set $n = |S_{pt}| + |S_{ball}|$. In 3D space, the USEC problem can be solved in $O(n^{4/3} \cdot \log^{4/3} n)$ expected time. Finding a 3D USEC algorithm with running time $o(n^{4/3})$ is a big open problem in computational geometry, and is widely believed to be impossible.

Lemma 16.

For any dimensionality d, if we can solve the DBSCAN problem in $T(n)$ time, then we can solve the USEC problem in $T(n) + O(n)$ time.

This lemma indicates that $T(n)$ must be $\Omega(n^{4/3})$ unless the USEC problem can be solved in $o(n^{4/3})$ time.

Proof.

Recall that the USEC problem is defined by a set S_{pt} of points and a set S_{ball} of balls with equal radii, both in $\mathbb{R}^d$. Denote by $\mathcal{A}$ an DBSCAN algorithm in $\mathbb{R}^d$ that runs in $T(m)$ time on m points. Next, we describe an algorithm that deploys $\mathcal{A}$ as a *black box* to solve the USEC problem in $T(n) + O(n)$ time, where $n = |S_{pt}| + |S_{ball}|$.

Proof (Cont.).

Our algorithm is simple:

1. Obtain P, which is the union of S_{pt} and the set of centers of the balls in S_{ball}.
2. Set r to the identical radius of the balls in S_{ball}.
3. Run $\mathcal{A}$ to solve the DBSCAN problem on P with this r and $t = 1$.
4. If any point in S_{pt} and any center of S_{ball} belong to the same cluster, then return *yes* for the USEC problem (namely, a point in S_{pt} is covered by some ball in S_{ball}). Otherwise, return *no*.

It is fundamental to implement the above algorithm in $T(n) + O(n)$ time. Next, we prove its correctness.

Proof (Cont.).

Case 1: We return yes. We will show that in this case there is indeed a point of S_{pt} that is covered by some ball in S_{ball}.

Recall that a *yes* return means a point $p \in S_{pt}$ and the center q of some ball in S_{ball} have been placed in the same cluster, which we denote by C. By connectivity of Definition 8, there exists a point $z \in C$ such that both p and q are reachable from z.

By setting $t = 1$, we ensure that *all* the points in P are core points. In general, if a *core point* p_1 is reachable from p_2 (which by definition must be a core point), then p_2 is also reachable from p_1. This means that z is reachable from p, which—together with the fact that q is reachable from z—shows that q is reachable from p.

Proof (Cont.).

It thus follows that there is a sequence of points $p_1, p_2, ..., p_t \in P$ such that (i) $p_1 = p, p_t = q$, and (ii) $dist(p_i, p_{i+1}) \le r$ for each $i \in [1, t-1]$. Let k be the smallest $i \in [2, t]$ such that p_i is the center of a ball in S_{ball}. Note that k definitely exists because p_t is such a center. It thus follows that p_{k-1} is a point from S_{pt}, and that p_{k-1} is covered by the ball in S_{ball} centered at p_k.

Case 2: We return no. We will show that in this case no point of S_{pt} is covered by any ball in S_{ball}.

This is in fact very easy. Suppose on the contrary that a point $p \in S_{pt}$ is covered by a ball of S_{ball} centered at q. Thus, $dist(p, q) \le r$, namely, q is reachable from p. Then, by maximality of Definition 8, q must be in the cluster of p (recall that all the points of P are core points). This contradicts the fact that we returned *no*. □