Data Mining: Clustering/Unsupervised Learning Density-Based Clustering. DBSCAN Algorithm

Density-Based Clustering. Preliminaries

Density-based clustering algorithms is a family of algorithms that determine density-based clusters in the data. A formal definition of a density-based cluster is supplied below.

ε-neighborhood. Let $D = \{d_1, \ldots, d_n\}$ be a set of data points, and let $dist()$ be a distance function for points in D[1]

Given a number ε, an ε-neighborhood point $d \in D$ is defined as:

$$N_\varepsilon(d) = \{d_i \in D | d_i \neq d, dist(d, d_i) \leq \varepsilon\}$$

Core points. Given an integer $minpts > 0$, a point $d \in D$ is a core point in D if

$$|N_\varepsilon(d)| \geq minpts,$$

that is, if the ε-neighborhood of d contains $minpts$ or more points.

Border (boundary) points. A point $d \in D$ is a border (boundary) point if

$$|N_\varepsilon(d)| < minpts,$$

but

$$(\exists d' \in D)(d \in N_\varepsilon(d'),$$

i.e., if the ε-neighborhood of d contains fewer than $minpts$ points, *but* d itself **is in** a ε-neighborhood of some other point $d' \in D$.

[1] A similar definition will also work for a similarity function.

Noise points. A point $d \in D$ is a noise point if it is neither core point nor boundary point in D.

Density-reachability. Given the density radius ε and the minimum density $minpts$, a point $d' \in D$ is directly density-reachable from point $d \in D$ if $d' \in N_\varepsilon(d)$.

d' is density-reachable from d if there exists a chain of points $d = d_1, d_2, \ldots, d_k = d'$, such that $d_i \in N_\varepsilon(d_{i-1})$.

Note: Density-reachability is an assymetric relationship (a boundary point x may be density-reachable from a core point y, but not the other way around).

Density connectivity. Two points $d \in D$ and $d' \in D$ are density connected, if there exists a core point $f \in D$, such that both d and d' are density-reachable from f.

Density-based cluster. A density cluster $D' \subset D$ is any maximal set of points that are density-connected to each other.

DBSCAN

DBSCAN is a key algorithm for discovery of density-based clusters. DBSCAN takes as input a dataset D, a distance function $dist()$ that is defined on all pairs of points from D[2] and two parameters:

- ε: the radius of the ε-neighborhood in which DBSCAN will search for data points;
- $minpts$: the smallest number of points in a ε-neighborhood of a point, for it to be declared a core point.

The pseudocode for DBSCAN is shown in Figure 1.

The algorithm works as follows:

- **Core point discovery.** First, DBSCAN scans through the entire dataset d and determines based on ε and $minpts$ parameters, the list of core points.
- **Cluster construction.** Each cluster is constructed as follows. The algoirithm pulls a *yet-to-be visited* core point, and recursively computes all density connected points to it. It then proceeds to search for the next unvisited/unlabeled core point until it runs out of core points to expand.
- **Output.** At the end, the algorithm returns the breakdown of points into clusters, as well as the lists of core, boundary and noise points.

[2]Usually, DBSCAN uses Eucledian distance, but it can also use other distance functions. Also, a version of DBSCAN that uses similarity measures rather than distance measures, can be obtained from the pseudocode shown in these notes in a straightforward way.

```
Algorithm DBSCAN(D,dist(), ε, minpts)
begin
  Core := ∅;
  for each d_i ∈ D do   // find core points
     Compute N_ε(d);
     cluster(d_i) := ∅;  // initialize cluster assignment for the point
     if |N_ε(d_i)| ≥ minpts then Core := Core ∪ {d_i};
  end for

  CurrentCluster := 0;   // initialize current cluster label

  for each d ∈ Core do
     if cluster(d) = ∅ then
        CurrentCluster := CurrentCluster + 1;  //start a new cluster
        cluster(d) := CurrentCluster   // assign first point to the cluster
            DensityConnected(D, d, Core, CurrentCluster);    // find all density connected
points
     endif
  end for



  ClusterList := ∅

  for k := 1 to CurrentCluster do    //assemble clusters
     Cluster[k] = {d ∈ D|cluster(d) = k};
     ClusterList := ClusterList ∪ Cluster[k];
  end for

  Noise := {d ∈ D|cluster(d) = ∅}
  Border := D − (Noise ∪ Core)
  return ClusterList, Core, Border, Noise


end

function DensityConnected(D, point, Core, clusterId)
begin
  for each d ∈ N_ε(point) do   // add all neighbors to cluster
     cluster(d) := clusterId;
     if d ∈ Core then DensityConnected(D, d, Core, clusterId);
//recursivly do it for each core point discovered
    endfor
end
```

Figure 1: Pseudocode for DBSCAN algorigthm