**DBSCAN CLUSTERING ALGORITHM**

1. **What is DBSCAN algorithm?**

The DBSCAN algorithm relies on a density-based notion of clusters. Clusters are identified by looking at the density of points. Regions with a high density of points depict the existence of clusters whereas regions with a low density of points indicate clusters of noise or clusters of outliers. This algorithm is particularly suited to deal with large datasets, with noise, and is able to identify clusters with different sizes and shapes.

1. **How DBSCAN algorithm be performed**

Two main parameters to DBSCAN are ‘ε’ and ‘minPoints’. ‘ε’ defines radius of the ‘neighborhood region’ and ‘minPoints’ defines the minimum number of points that should be contained within that neighborhood. Since it has a concept of noise, it works well even with noisy datasets.

Epsilon neighborhood (Nε) : set of all points within a distance ‘ε’.

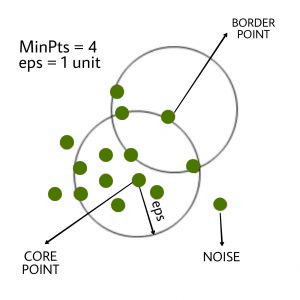
Core point : A point that has at least ‘minPoint’ (including itself) points within it’s Nε .

Direct Density Reachable (DDR) : A point q is directly density reachable from a point p if p is core point and q ∈ Nε .

Density Reachable (DR) : Two points are DR if there is a chain of DDR points that link these two points.

Border Point: Point that are DDR but not a core point.

Noise : Points that do not belong to any point’s Nε.



DBSCAN clustering with minPoints=4

DBSCAN algorithm can be abstracted in the following steps:

1. Find all the neighbor points within eps and identify the core points or visited with more than MinPts neighbors.
2. For each core point if it is not already assigned to a cluster, create a new cluster.
3. Find recursively all its density connected points and assign them to the same cluster as the core point.  
   A point*a* and *b* are said to be density connected if there exist a point *c* which has a sufficient number of points in its neighbors and both the points*a* and *b* are within the *eps distance*. This is a chaining process. So, if *b* is neighbor of *c*, *c* is neighbor of*d*, *d* is neighbor of *e*, which in turn is neighbor of *a* implies that *b* is neighbor of*a*.
4. Iterate through the remaining unvisited points in the dataset. Those points that do not belong to any cluster are noise.

This is the pseudo code:

DBSCAN(dataset, eps, MinPts){

# cluster index

C = 1

for each unvisited point p in dataset {

mark p as visited

# find neighbors

Neighbors N = find the neighboring points of p

if |N|>=MinPts:

N = N U N'

if p' is not a member of any cluster:

add p' to cluster C

}