DBSCAN

Paul Antony Job Jacob

9 August 2018

DBSCAN

DBSCAN stands for *Density-Based Spatial Clustering of Applications with Noise*

It is a density-based clustering algorithm which groups together points in a given set that are closely packed together, it also marks outlier's points that lie alone in low-density regions as Noise.

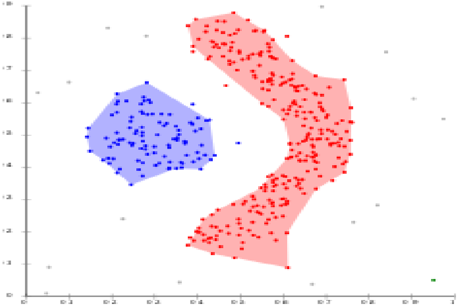

Figure: DBSCAN Clustering

DBSCAN Working

DBSCAN takes two parameters ϵ and MinPtn

ϵ is a distance parameter that defines the radius to search for nearby neighbours and MinPtn minimum number of points in the region required to form a cluster

An n dimensional sphere of radius ϵ is made around a point and points in the sphere are clustered if number of points is greater than MinPtn

There are three types of points in DBSCAN:

- **Core point** – a point that has at least a minimum number of other points (minPts) within its ϵ radius.
- **Border point** – a point is within the ϵ radius of a core point BUT has less than the minimum number of other points (minPts) within its own ϵ radius
- **Noise point** – a point that is neither a core point nor a border point

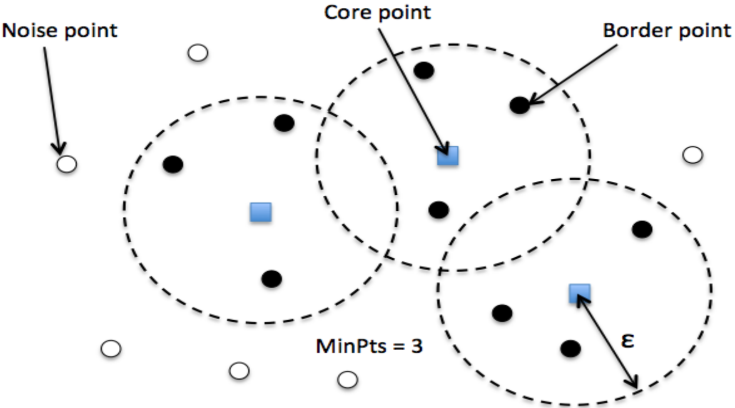

Figure: Cluster Points

Advantages

- DBSCAN does not require one to specify the number of clusters in the data
- It can find arbitrarily shaped cluster.Find cluster completely surrounded by different clusters
- It requires just two parameters and is mostly insensitive to the ordering of the points in the database
- It is robust towards outlier detection (noise)

Disadvantage

- DBSCAN is not entirely deterministic: border points that can be reached by more than one clusters can be part of either cluster depending on the order the data is processed
- If data and scale is unknown choosing a meaningful distance threshold ϵ can be difficult
- Sensitive to clustering parameter MinPoints and ϵ
- DBSCAN cannot cluster data sets well with large differences in densities, since the minPts-ϵ combination cannot then be chosen appropriately for all clusters

Algorithm

INPUT: N objects to be clustered and global parameter eps and MinPts
OUTPUT: Clusters of objects
Algorithm

1. Initialize all object label as *not visited*
2. Repeat till all points are visited
   1. Arbitrary select a point P
   2. Retrive all neighbour points of P using function regionQuery(P)
   3. If number of neighbour points is greater than MinPts then P in a core point. Its label is changed to cluster name and cluster is grown using function growCluster(P) , else P is labeled as noise

Algorithm

Algorithm growCluster(P)

1. push P to SearchQueue (SearchQueue is an normal Queue data structure)
2. while SearchQueue is not empty
   1. P = SearchQueue.pop
   2. find neighbour points of P using function regionQuery(P)
   3. check each point in neighbour is a core point, change its label to cluster name and push it into SearchQueue.
   4. If it's not a noise point change its label to cluster name

Algorithm

Algorithm regionQuery(P)

1. calculates distance between P and each points Pi in data using function EuclideanDistance(P,Pi). If distance is less than eps then the point Pi is added to list neighbour-points
2. return neighbour-points

Algorithm EuclideanDistance(P,Pi)

1. distance between points P and Pi is calculated using the Euclidean Distance equation

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + \ldots\ldots..}$$

END