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

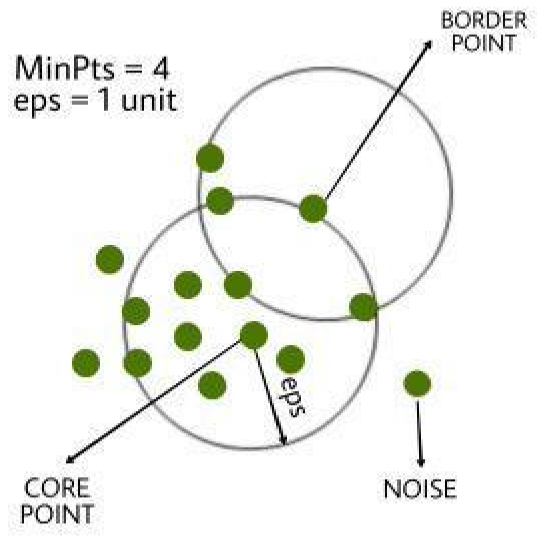
Density-Based Clustering refers to unsupervised learning methods that identify distinctive groups/clusters in the data, based on the idea that a cluster in data space is a contiguous region of high point density, separated from other such clusters by contiguous regions of low point density. It can discover clusters of different shapes and sizes from a large amount of data, which is containing noise and outliers.

There are three types of points after the DBSCAN clustering is complete

Core Point: A point is a core point if it has more than MinPts points within eps.

Border Point: A point which has fewer than MinPts within eps but it is in the neighbourhood of a core point.

Noise or outlier: A point which is not a core point or border point.



The DBSCAN algorithm uses the below parameters

Minimum number of points (minPts) - This is the number of points that we want in the neighbourhood of our point in focus (within the circle). This is the minimum number of points (a threshold) clustered together for a region to be considered dense. As a rule of thumb, a minimum minPts can be derived from the number of dimensions D in the data set, as minPts ≥ D + 1. The low value minPts = 1 does not make sense, as then every point on its own will already be a cluster. With minPts ≤ 2, the result will be the same as of hierarchical clustering with the single link metric, with the dendrogram cut at height ε.
Therefore, minPts must be chosen at least 3. However, larger values are usually better for

data sets with noise and will yield more significant clusters. As a rule of thumb, minPts = 2·dim can be used.

Epsilon (ε) - This is the radius of the circle that we must draw around our pointing focus. To calculate the value of Eps, we shall calculate the distance between each data point to its closest neighbour using the Nearest Neighbours. After that, we sort them and finally plot them. From the plot, we identify the maximum value at the curvature of the graph. This value is our Eps. If ε is chosen much too small, a large part of the data will not be clustered, whereas for a too high value of ε clusters will merge and the majority of objects will be in the same cluster. In general, small values of ε are preferable and as a rule of thumb only a small fraction of points should be within this distance of each other.

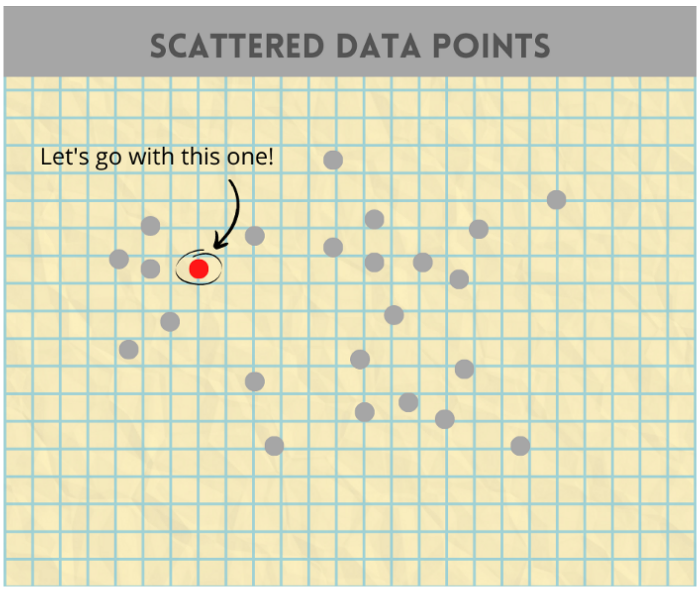
How it works

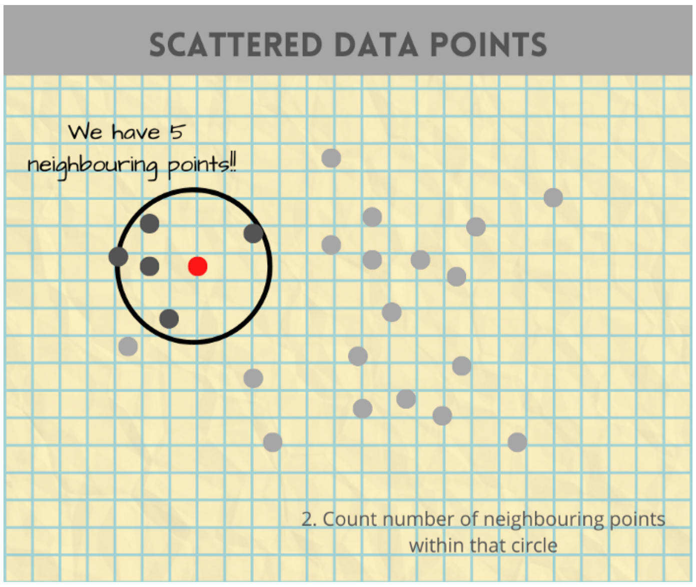
The algorithm proceeds by arbitrarily picking up a point in the dataset (until all points have been visited) and checks for following

- If the point is a core point, then considers all its neighbours within the pre-defined radius to be part of one cluster and repeat the steps for all the neighbours points

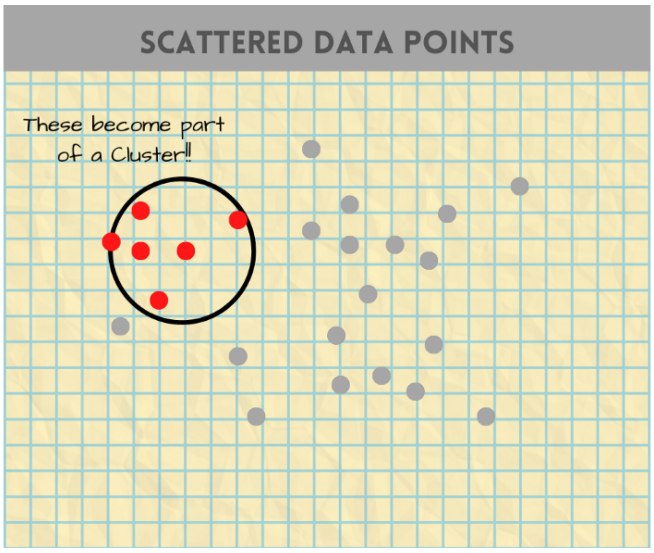
- If the point is a border point, even though it becomes part of the cluster of the core point that lies within the circle of pre-defined radius, we will not check its neighbourhood to continue the cluster and consider any of the unvisited points yet, repeat the steps and create a new cluster.

- if the point is a noise point, ignore the point and consider any of the unvisited points yet, repeat the steps and create a new cluster.

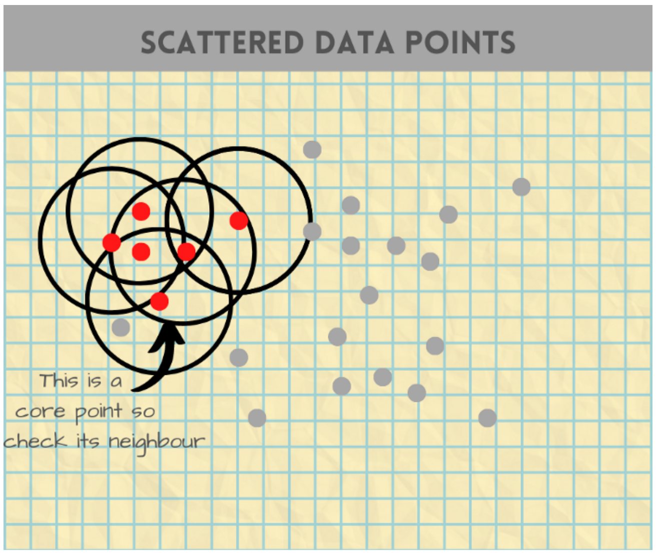



SCATTERED DATA POINTS
We have 5 neighbouring points!!
2. Count number of neighbouring points within that circle

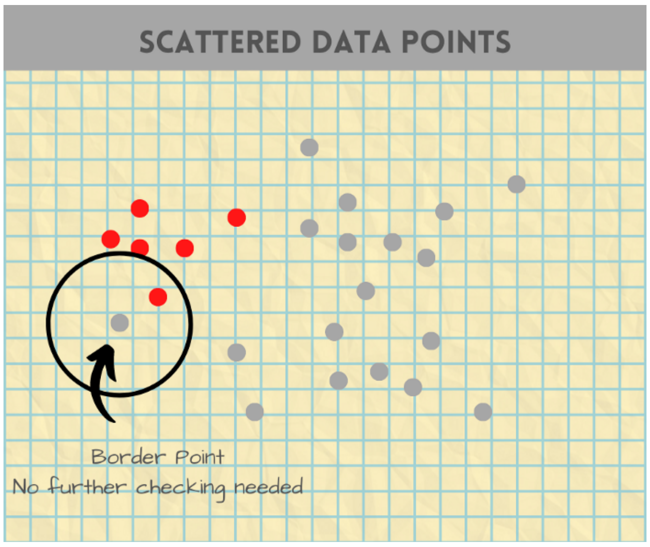
Assuming minPts = 3, this point is a core point since it has more neighbours (5) than the number we decided (3). These all points are said to be part of a single cluster and all neighbours of the core point will be visited to decide if they are core, border or noise points.



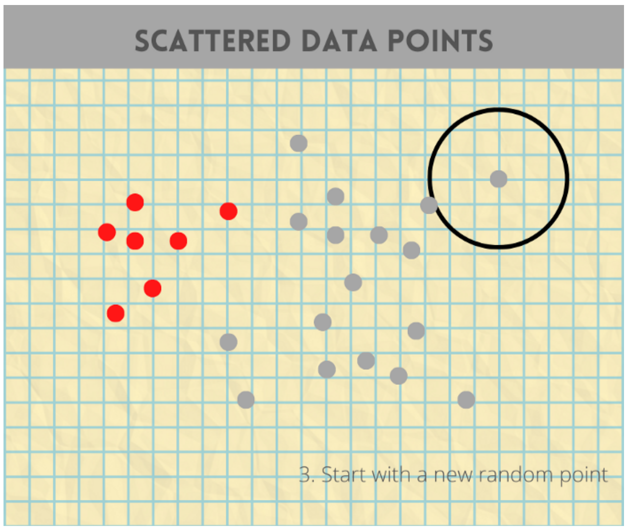
Repeating the same steps for all the neighbours of the core point and deciding whether they are core/border/noise points. Since they are core points, we keep making them part of red cluster and repeating steps for these core points too.



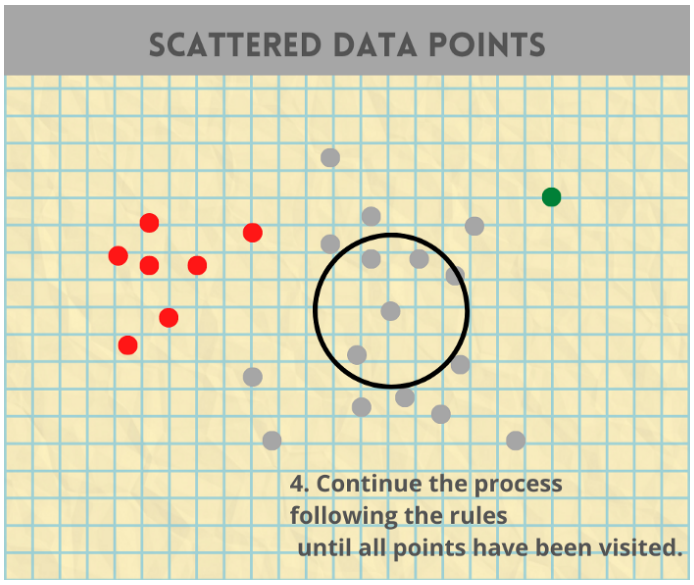
So we check this new point to continue the cluster.



Seems like this is a border point, even though it becomes part of the red cluster we will not check its neighbourhood to continue the cluster. Well, this means we start the algorithm all over with new points, creating a new cluster.



This point seems like a dead end on its own; it doesn't have any neighbours. This point is what we label as Noise.


SCATTERED DATA POINTS
4. Continue the process following the rules until all points have been visited.

Following the algorithm, we apply the set rules to all unvisited points until we have them all in a cluster.

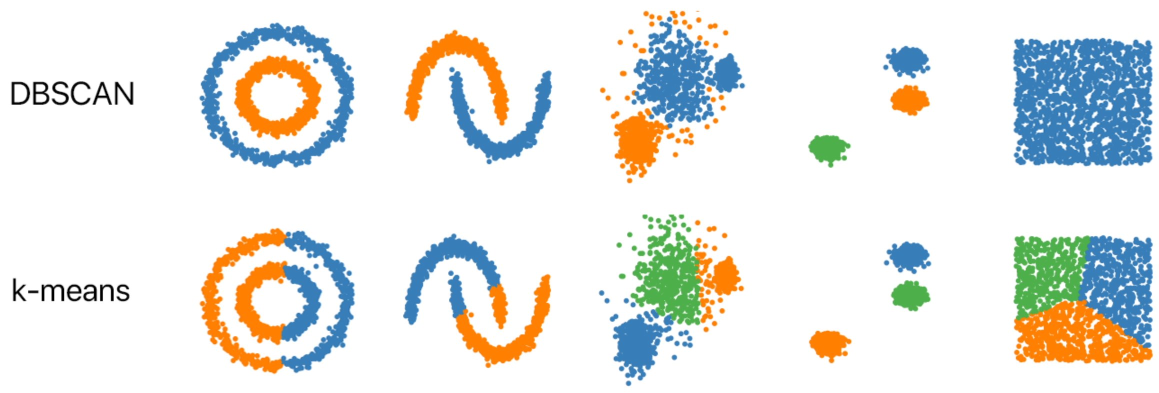
Just like Kmeans, DB Scan clustering algorithm can be evaluated using silhouette score

Comparison

- It is not sensitive to outliers. In other algorithms every observation becomes a part of some cluster eventually, even if the observations are scattered far away in the vector space i.e. even if it is an outlier and a slight change in data points might affect the clustering outcome. In DBSCAN, Outliers don't become part of any cluster and thus every observation doesn't need to be a part of some cluster eventually.

- It doesn't require users to specify the number of clusters.

- It also produces more reasonable results than k-means across a variety of different distributions.



- It would be a big concern to use DBSCAN if the data has a very large variation in densities across clusters because you can only use one pair of parameters (eps and MinPts) on one dataset.

- It could be super hard to define eps without the domain knowledge of the data.