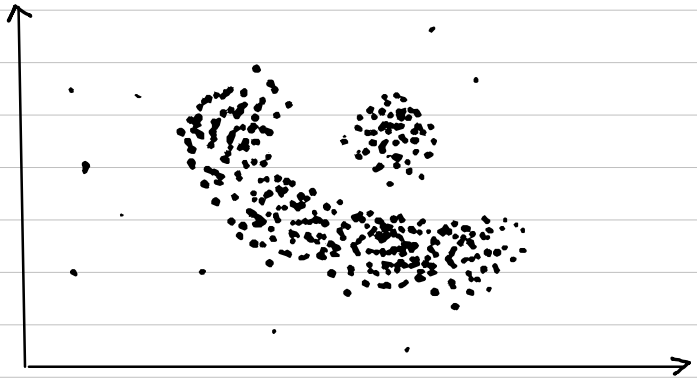
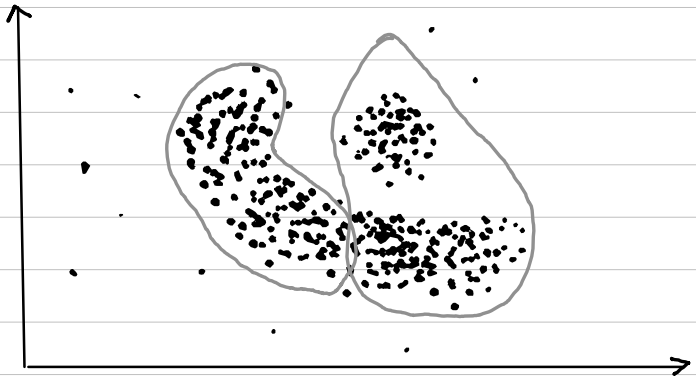
DBSCAN

Consider a dataset with Nested clusters



K-Means fails to give Nested clusters



DBSCAN is used to identify Nested clusters in high dimentions.

DBSCAN

Density Based spatial Clustering of Applications with Noise

clusters based on density unlike k-means which is based on distance

No direct control over the Number of clusters, like k means, but the clusters can be adjusted using the parameters

Works well for Noisy datasets

A point broadcasts a signal. Whoever in the radius listens to that signal comes in the cluster and broadcasts the same signal to others recursively.

Useful for Market research, pattern recognition crystallography of X-ray, satellite Imaging Recommendation systems, genomics & consumer segmentation

Can handle nested, non circular clusters unlike k-Means.

Intuition Behind DBSCAN

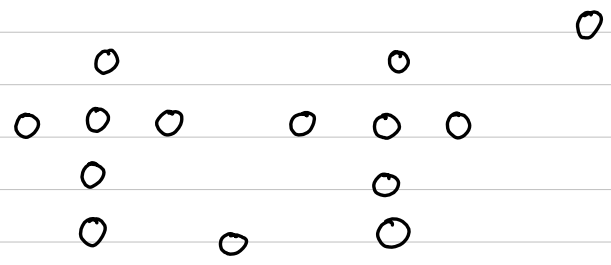
DBSCAN mimic how we form clusters by eye.

When we look at clusters, we identify clusters by the densities of points

Clusters are high density regions while outliers are low density regions.

How DBSCAN works

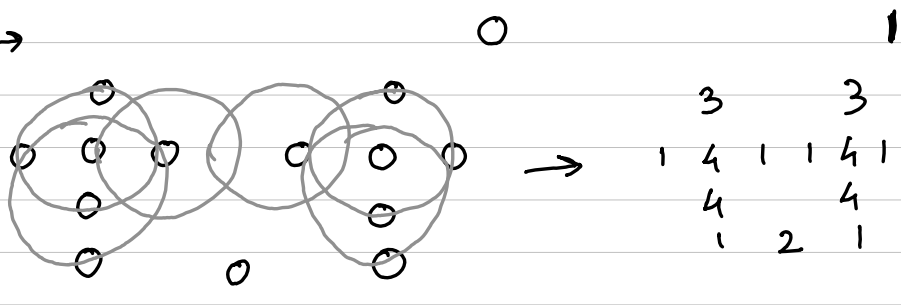
Consider a group of points



For each point, we have to check how many points are in their neighborhood

The neighborhood is defined by a radius parameter

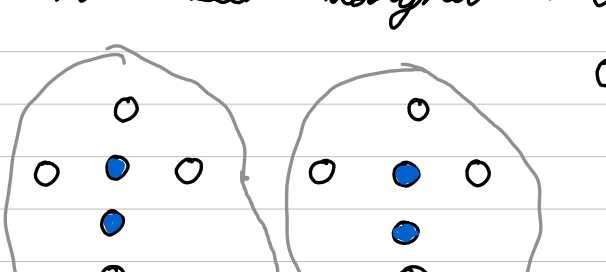
eg →



Then decide a threshold value parameter for minimum number of neighbours

eg 4.

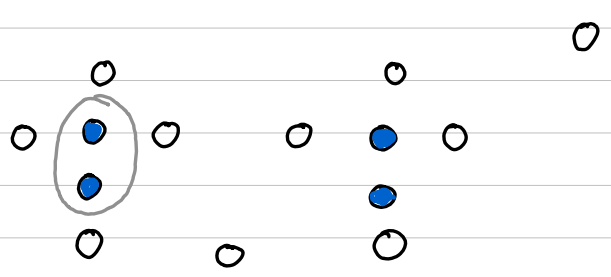
The points which have 4 neighbours in their radius are core points



Begin with a core point

For other core points in its neighbourhood, add them to their cluster.

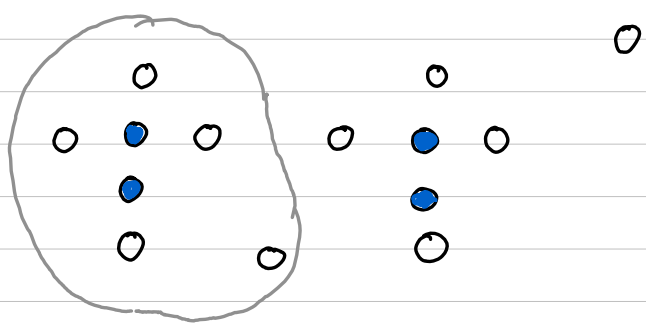
Recursively continue for all the core points.



Since Pt 2 is in neighbourhood of Pt 1 add it in cluster

Since Pt 2 has no other neighbours, end

Next step is to add non core neighbours of core points to cluster.



Now choose next core point that has not been assigned a cluster & proceed

The points that dont have any clusters are outliers (Non core points)

Parameters of DBSCAN

(ε epsillion)

① eps (Radius distance) → Defines Neighborhood around the point

If $d(P_1, P_2) < eps$ then P_1, P_2 are neighbours
This distance is in higher dimentions

If eps value is too small then a large part of data will be outlier

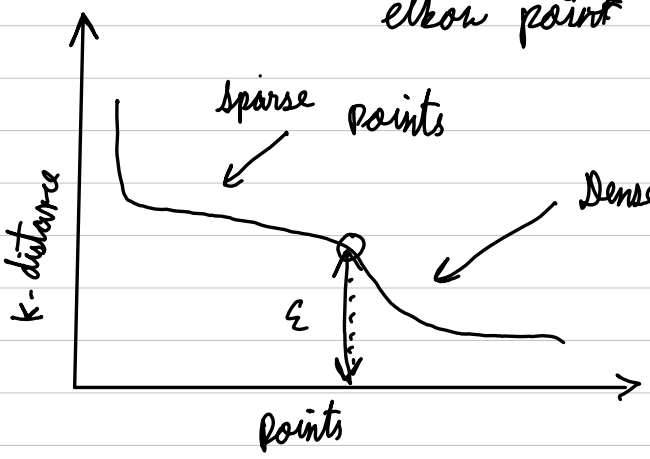
If eps value is too large then the clusters will merge and majority of datapoints will be in the same cluster

Choosing the value of eps → k-distance

eps = distance of point to k^{th} nearest neighbour.

k can be chosen as minpts or minpts -1

Calculate k distance for all points, arrange the values descendingly & check elbow point



Dont confuse this with cross validation.

② Min Pts (Threshold) → Minimum number of neighbours within eps radius.

The larger the dataset, the larger value of MinPts, otherwise everything will be a core point

General rule Minpts ≥ D+1 (D: dimentionality)
Minpts = 2D
least value of Minpts must be 3

low value of Minpts makes the algorithm more sensative to noise

High value of Minpts leads to few clusters & more points classified as noise. Small clusters may be missed

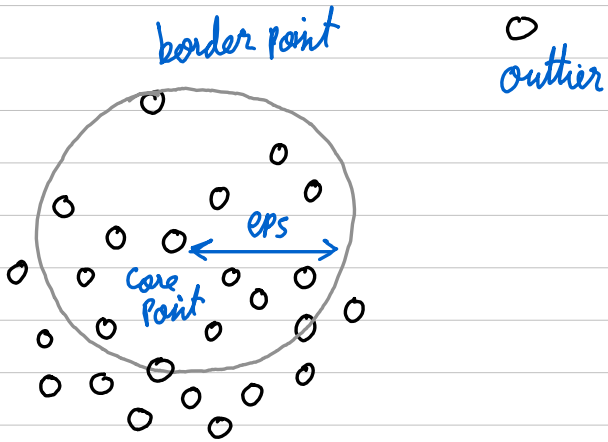
If dataset is noisier, choose high Minpts.

Types of Points

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

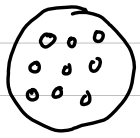
2. Border point : A point which has fewer than Minpts within eps but it is in the neighbourhood of a core point

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



Advantages →

Noise handling, unaffected by outliers



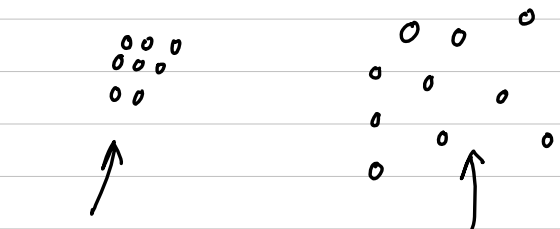
This outlier is left alone by DBSCAN. It does not pull the cluster like k-Means.

Various shaped clusters depending upon the density

Only 2 tunable parameters

Disadvantages →

Does not work well with clusters of varying densities



Needs small ε

Needs large ε

High Memory cost for large datasets

Doesnt work well with high dimentionality due to curse of dimentionality

Border points may be ambigously classified.