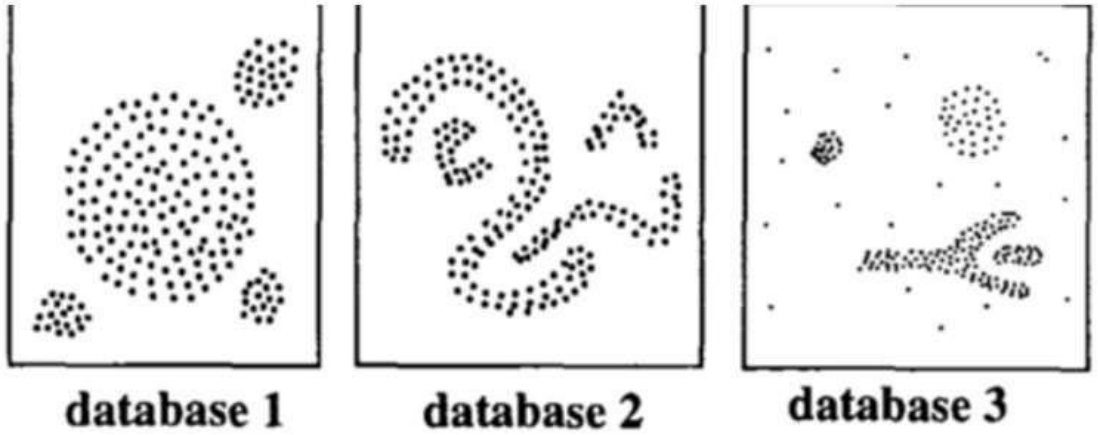
DBSCAN - A Clustering Algorithm

Review

Clustering is to group objects into meaningful subclasses. But there are some difficulties:

- Not to have information about the data to be clustered.
- The separation of clusters in ambiguous/arbitrary shapes.
- Large amount of data.



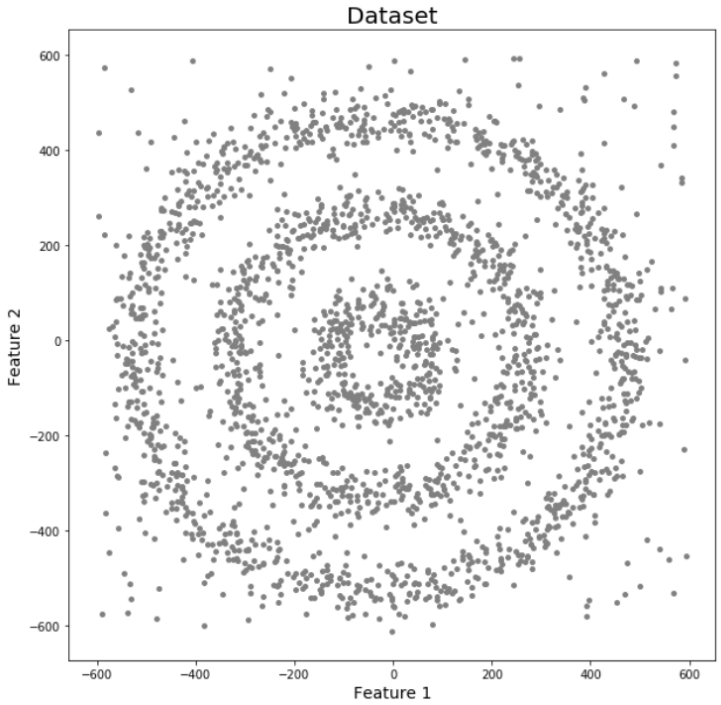
DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

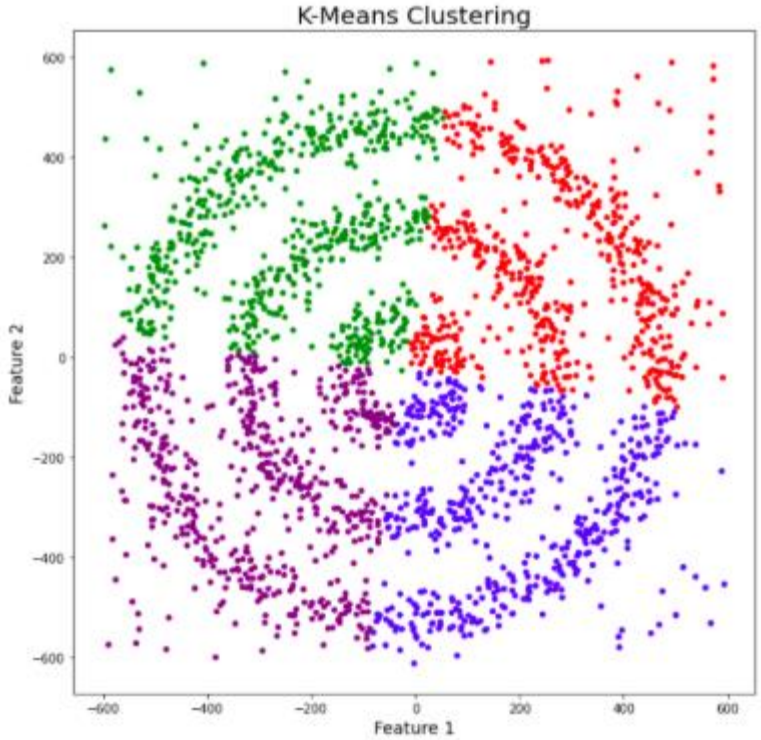
It groups ‘densely grouped’ data points into a single cluster. It can identify clusters in large spatial datasets by looking at the local density of the data points.

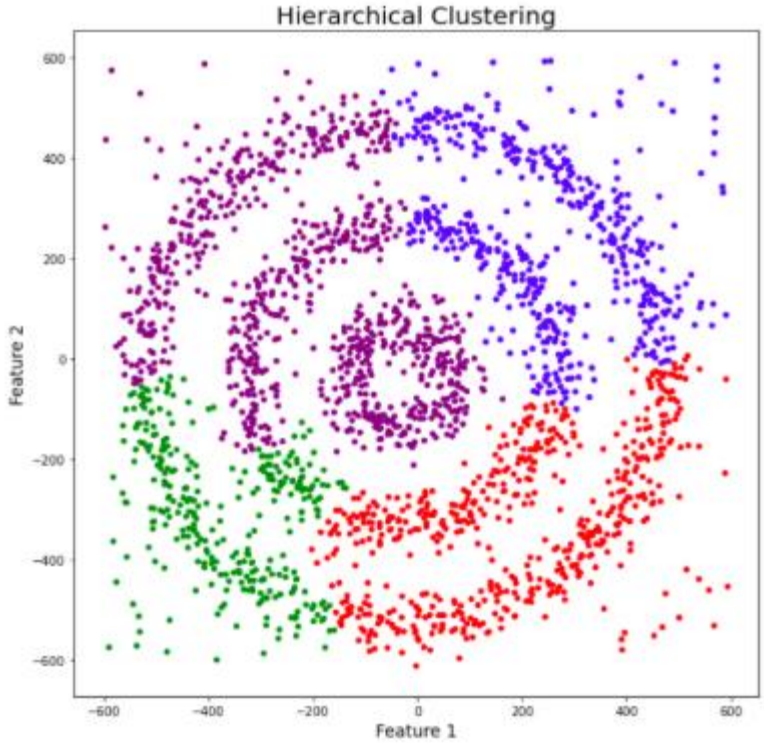
The most exciting feature of DBSCAN clustering is that it is robust to outliers.

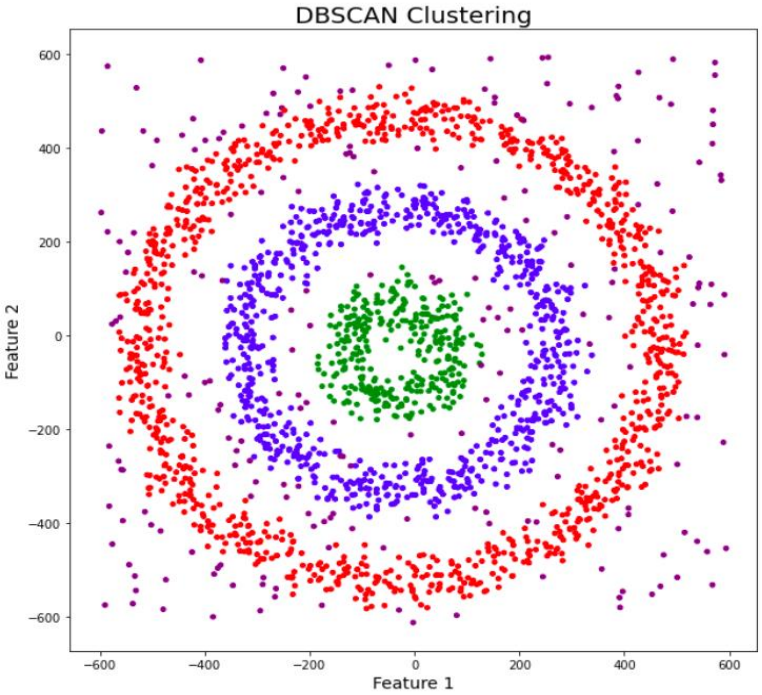
It also does not require the number of clusters to be told beforehand, unlike K-Means, where we have to specify the number of centroids.

K-Means and Hierarchical Clustering both fail in creating clusters of arbitrary shapes.
They are not able to form clusters based on varying densities. That's why we need DBSCAN clustering.




K-Means Clustering
Feature 2
Feature 1
600
400
200
0
−200
−400
−600
−600
−400
−200
0
200
400
600


Hierarchical Clustering
Feature 2
Feature 1
600
400
200
0
−200
−400
−600
−600
−400
−200
0
200
400
600


DBSCAN Clustering
Feature 2
Feature 1
600
400
200
0
−200
−400
−600
−600
−400
−200
0
200
400
600

Density-Based Clustering Methods

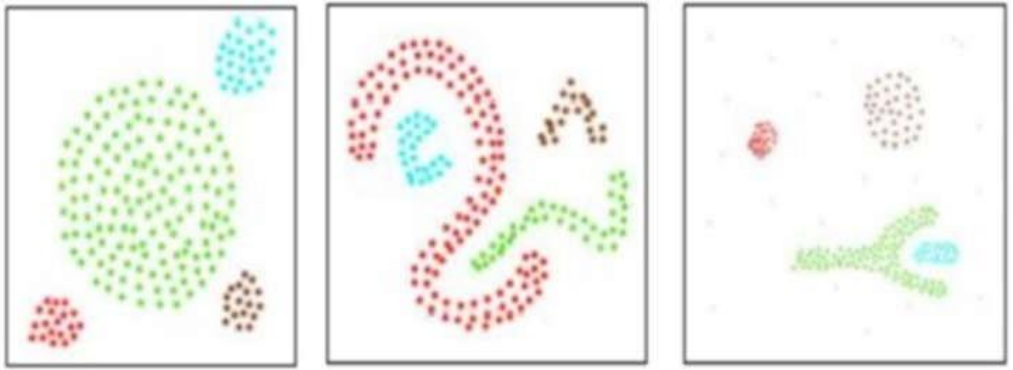
- Clustering based on density (local cluster criterion), such as density-connected points or based on an explicitly constructed density function
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters
- Several interesting studies:
  - DBSCAN: Ester, et al. (KDD'96)

DBSCAN

DBSCAN- Density-Based Spatial Clustering of Applications with Noise.

Clustering is done according to the density of the data. Therefore it is independent of shape and size. So, dbscan is also successful in arbitrary- shaped, large databases and is not affected by the noisy data.

Unlike many clustering algorithms, each point does not have to belong to a cluster.



The classification of the DBSCAN algorithm.

Algorithm marks the lonely points in low density regions and group the points located close together. Two main parameters;

- Ɛ (Epsilon=Eps): largest radius of neighborhood around a point.
- MinPts (minimum points,density): minimum number of points in the neighborhood with radius Ɛ.

Methods such as the distance from Euclidean or Manhattan or other measurement approaches can be used for density measurement.

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius r (Eps)
  - A point is a core point if it has more than a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
  - A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point.

In DBSCAN, the points are labeled in 3 different types:

- Core Point: is a data point that contains greater than or equal to minPts within radius Ɛ.
- Border Point: number of neighbors is less than minPts, but it belongs to the Ɛ- neighborhood of some core point z.
- Noise Point: neither a core nor a border point (outlier).

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.

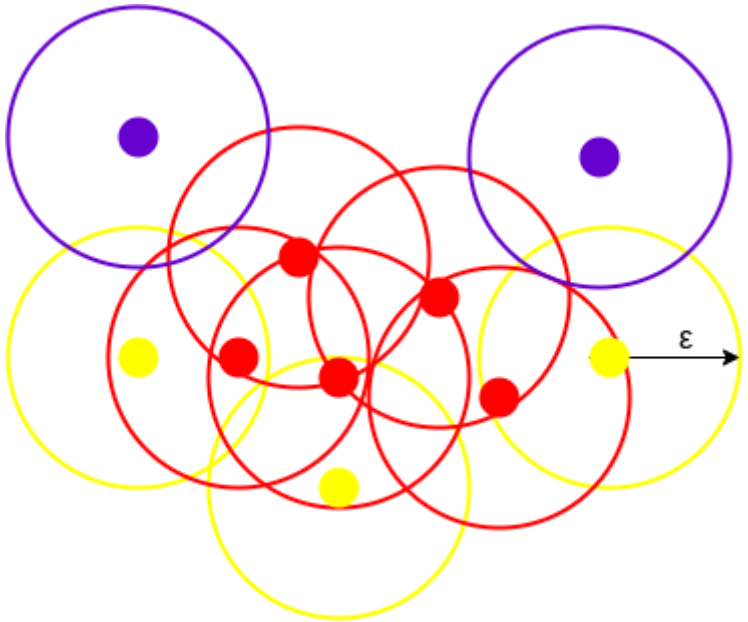
3.Find recursively all its density connected points and assign them to the same cluster as the core point.

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.

The below figure shows us a cluster created by DBCAN with *minPoints = 3*. Here, we draw a circle of equal radius *epsilon* around every data point. These two parameters help in creating spatial clusters.



All the data points with at least 3 points in the circle including itself are considered as **Core** points represented by red color.

All the data points with less than 3 but greater than 1 point in the circle including itself are considered as **Border** points.

They are represented by yellow color. Finally, data points with no point other than itself present inside the circle are considered as **Noise** represented by the purple color.

•DBSCAN starts by randomly selecting an unvisited data point and exploring its epsilon neighborhood.

•If the neighborhood contains more than min_samples points, a new cluster is created, and the process continues recursively to expand the cluster.

•If the neighborhood does not contain enough points to form a cluster, the point is labeled as noise.

•The algorithm proceeds by iteratively visiting unvisited points and expanding the clusters until all data points have been visited.

•The output of DBSCAN is a set of clusters, each represented by a set of data points.

•Data points that are not assigned to any cluster are considered noise points.

•Hence, DBSCAN has two key parameters: epsilon (ε) and min_samples. These parameters control the density threshold for identifying core points and the minimum number of points required to form a cluster.

•Choosing appropriate values for these parameters is important and depends on the specific dataset and desired clustering outcome.

Complexity DBSCAN

- Time Complexity: $O(n^2)$—for each point it has to be determined if it is a core point, can be reduced to O(n*log(n)) in lower dimensional spaces by using efficient data structures (n is the number of objects to be clustered);
- Space Complexity: O(n).

Summary DBSCAN

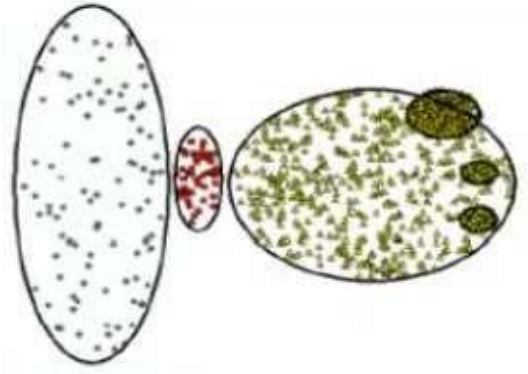
- Good: can detect arbitrary shapes, not very sensitive to noise, supports outlier detection, complexity is kind of okay, beside K-means the second most used clustering algorithm.
- Bad: does not work well in high-dimensional datasets, parameter selection is tricky, has problems of identifying clusters of varying densities (→SSN algorithm), density estimation is kind of simplistic (→does not create a real density function, but rather a graph of density-connected points)

Advantages

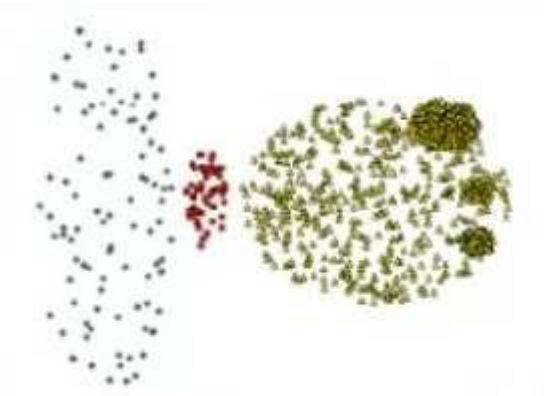
- Can handle clusters different shapes and sizes.
- Resistant to noise

Disadvantages

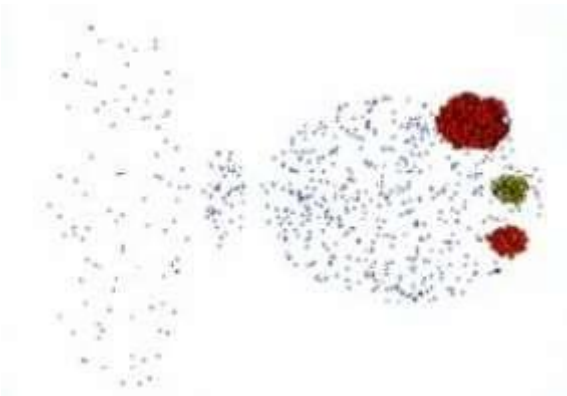
- sensitive in parameter selection.



Original Points



minPts:4, Eps: 9,75



minPts:4, Eps:9,92

Python Package for DBSCAN

```
from sklearn.cluster import DBSCAN

clustering = DBSCAN(eps=12.5, min_samples=4).fit(X_train)
```