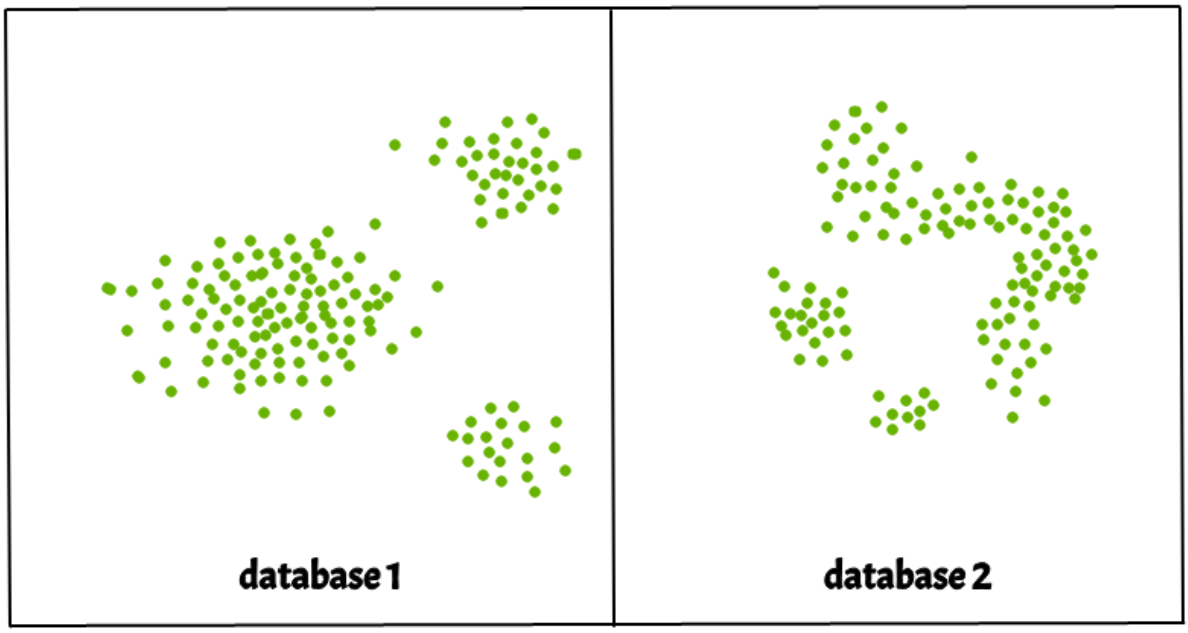
Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Clusters are dense regions in the data space, separated by regions of the lower density of points. The ***DBSCAN algorithm*** is based on this intuitive notion of "clusters" and "noise". The key idea is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.

NB. DBSCAN uses distance between nearest point

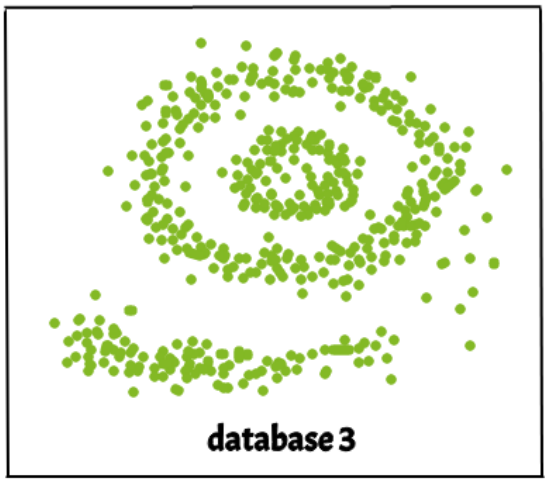


Why DBSCAN?

Partitioning methods (K-means, PAM clustering) and hierarchical clustering work for finding spherical-shaped clusters or convex clusters. In other words, they are suitable only for compact and well-separated clusters. Moreover, they are also severely affected by the presence of noise and outliers in the data.

Real-life data may contain irregularities, like:

1. Clusters can be of arbitrary shape such as those shown in the figure below.
2. Data may contain noise.

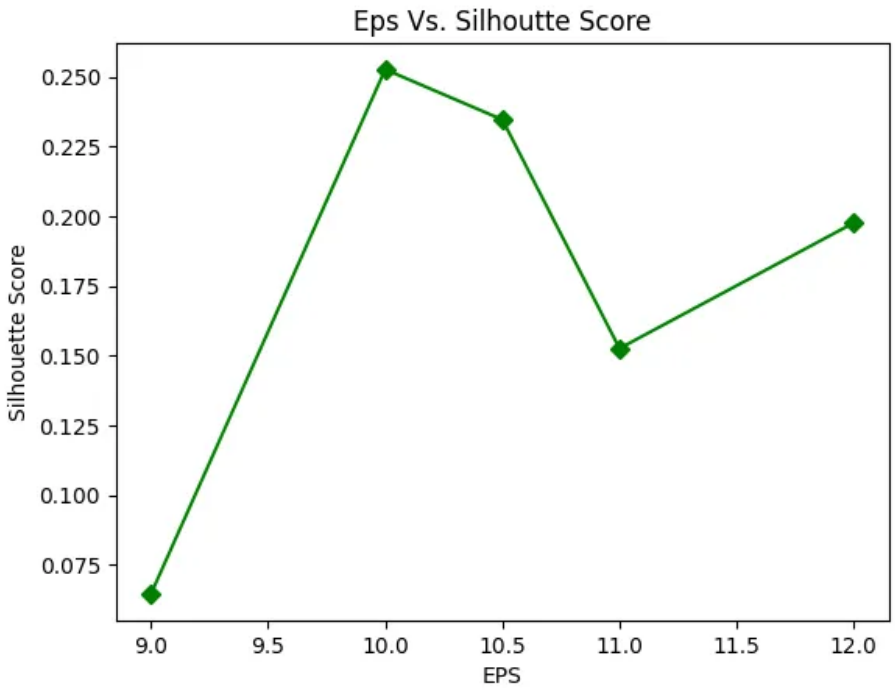


The figure above shows a data set containing non-convex shape clusters and outliers. Given such data, the k-means algorithm has difficulties in identifying these clusters with arbitrary shapes.

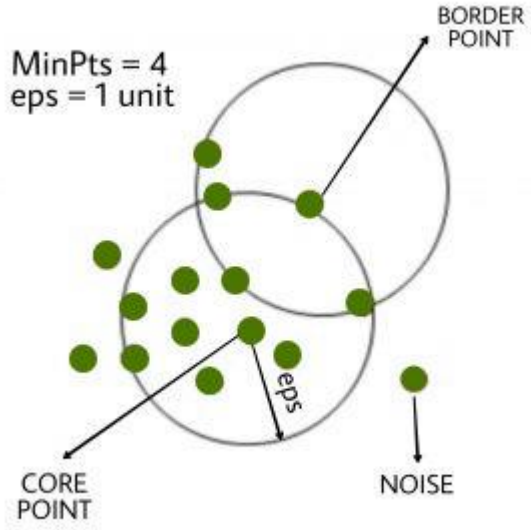
Parameters Required For DBSCAN Algorithm

1. **eps**: It defines the neighborhood around a data point i.e. if the distance between two points is lower or equal to 'eps' then they are considered neighbors. If the eps value is chosen too small then a large part of the data will be considered as an outlier. If it is chosen very large then the clusters will merge and the majority of the data points will be in the same clusters. One way to find the eps value is based on the ***k-distance graph***. In other words, the **eps** parameter determines the radius around each data point within which a sufficient number of other data points must reside for that point to be considered a core point, and thus included in a cluster.
2. **MinPts**: Minimum number of neighbors (data points) within eps radius. The larger the dataset, the larger value of MinPts must be chosen. As a general rule, the minimum MinPts can be derived from the number of dimensions D in the dataset as, MinPts >= D+1. The minimum value of MinPts must be chosen at least 3. The **min_samples** parameter specifies the minimum number of data points that must be present within the epsilon radius for a point to be considered a core point.

You can plot the Silhouette score for the models trained on various combinations of parameters and select those parameters which give the highest silhouette score.



In this algorithm, we have 3 types of data points.
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 neighborhood of a core point.
Noise or outlier: A point which is not a core point or border point.



Steps Used In DBSCAN Algorithm

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 exists a point *c* which has a sufficient number of points in its neighbors and both points *a* and *b* are within the *eps distance*. This is a chaining process. So, if *b* is a neighbor of *c*, *c* is a neighbor of *d*, and *d* is a neighbor of *e*, which in turn is neighbor of *a* implying that *b* is a neighbor of *a*.
4. Iterate through the remaining unvisited points in the dataset. Those points that do not belong to any cluster are noise.

DBSCAN Clustering Algorithm Solved Example – 1

- Apply the DBSCAN algorithm to the given data points and
- Create the clusters with
- minPts = 4 and
- epsilon (ε) = 1.9.

Data Points:

P1: (3, 7)	P2: (4, 6)
P3: (5, 5)	P4: (6, 4)
P5: (7, 3)	P6: (6, 2)
P7: (7, 2)	P8: (8, 4)
P9: (3, 3)	P10: (2, 6)
P11: (3, 5)	P12: (2, 4)

- Use Eucladian distance and calculate the distance between each points.

$$Distance(A(x_1, y_1), B(x_2, y_2)) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

P1: (3, 7)
P2: (4, 6)
P3: (5, 5)
P4: (6, 4)
P5: (7, 3)
P6: (6, 2)
P7: (7, 2)
P8: (8, 4)
P9: (3, 3)
P10: (2, 6)
P11: (3, 5)
P12: (2, 4)

minPts = 4 and epsilon (ε) = 1.9

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
P1	0											
P2	1.41	0										
P3	2.83	1.41	0									
P4	4.24	2.83	1.41	0								
P5	5.66	4.24	2.83	1.41	0							
P6	5.83	4.47	3.16	2.00	1.41	0						
P7	6.40	5.00	3.61	2.24	1.00	1.00	0					
P8	5.83	4.47	3.16	2.00	1.41	2.83	2.24	0				
P9	4.00	3.16	2.83	3.16	4.00	3.16	4.12	5.10	0			
P10	1.41	2.00	3.16	4.47	5.83	5.66	6.40	6.32	3.16	0		
P11	2.00	1.41	2.00	3.16	4.47	4.24	5.00	5.10	2.00	1.41	0	
P12	3.16	2.83	3.16	4.00	5.10	4.47	5.39	6.00	1.41	2.00	1.41	0

minPts = 4 and epsilon (ε) = 1.9

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
P1	0											
P2	1.41	0										
P3	2.83	1.41	0									
P4	4.24	2.83	1.41	0								
P5	5.66	4.24	2.83	1.41	0							
P6	5.83	4.47	3.16	2.00	1.41	0						
P7	6.40	5.00	3.61	2.24	1.00	1.00	0					
P8	5.83	4.47	3.16	2.00	1.41	2.83	2.24	0				
P9	4.00	3.16	2.83	3.16	4.00	3.16	4.12	5.10	0			
P10	1.41	2.00	3.16	4.47	5.83	5.66	6.40	6.32	3.16	0		
P11	2.00	1.41	2.00	3.16	4.47	4.24	5.00	5.10	2.00	1.41	0	
P12	3.16	2.83	3.16	4.00	5.10	4.47	5.39	6.00	1.41	2.00	1.41	0

P1: P2, P10
P2: P1, P3, P11
P3: P2, P4
P4: P3, P5
P5: P4, P6, P7, P8
P6: P5, P7
P7: P5, P6
P8: P5
P9: P12
P10: P1, P11
P11: P2, P10, P12
P12: P9, P11

P1: P2, P10

P2: P1, P3, P11

P3: P2, P4

P4: P3, P5

P5: P4, P6, P7, P8

P6: P5, P7

P7: P5, P6

P8: P5

P9: P12

P10: P1, P11

P11: P2, P10, P12

P12: P9, P11

minPts = 4 and epsilon (ε) = 1.9

Point	Status	
P1	Noise	Border
P2	Core	
P3	Noise	Border
P4	Noise	Border
P5	Core	
P6	Noise	Border
P7	Noise	Border
P8	Noise	Border
P9	Noise	
P10	Noise	Border
P11	Core	
P12	Noise	Border

P1: P2, P10

P2: P1, P3, P11

P3: P2, P4

P4: P3, P5

P5: P4, P6, P7, P8

P6: P5, P7

P7: P5, P6

P8: P5

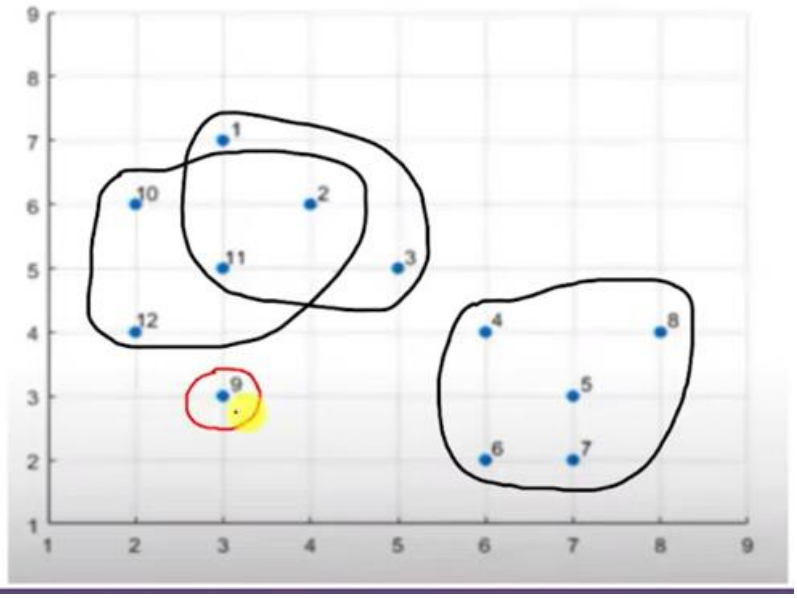
P9: P12

P10: P1, P11

P11: P2, P10, P12

P12: P9, P11

minPts = 4 and epsilon (ε) = 1.9



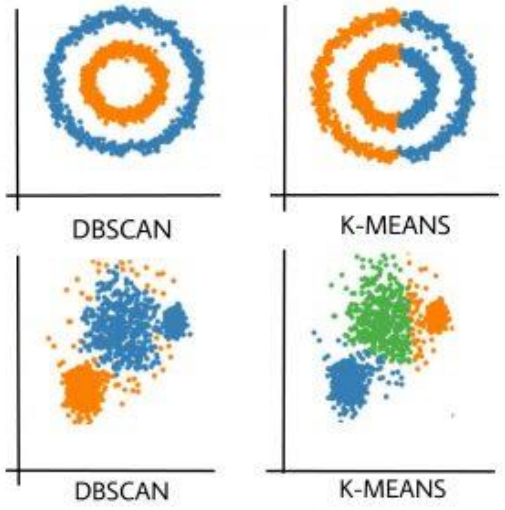
Here, P9 is noise or outlier.

When Should We Use DBSCAN Over K-Means In Clustering Analysis?

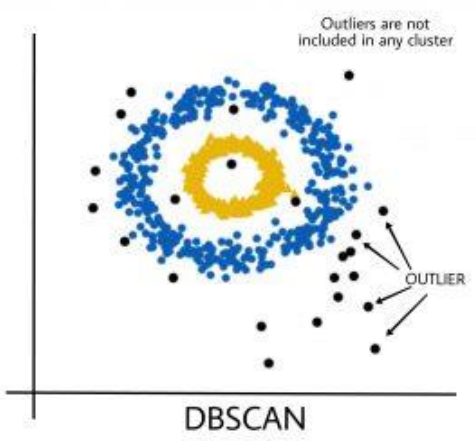
DBSCAN(Density-Based Spatial Clustering of Applications with Noise) and K-Means are both clustering algorithms that group together data that have the same characteristic. However, They work on different principles and are suitable for different types of data. We prefer to use DBSCAN when the data is not spherical in shape or the number of classes is not known beforehand.

Difference Between DBSCAN and K-Means

DBSCAN	K-Means
In DBSCAN we need not specify the number of clusters.	K-Means is very sensitive to the number of clusters so it need to specified
Clusters formed in DBSCAN can be of any arbitrary shape.	Clusters formed in K-Means are spherical or convex in shape
DBSCAN can work well with datasets having noise and outliers	K-Means does not work well with outliers data. Outliers can skew the clusters in K-Means to a very large extent.
In DBSCAN two parameters are required for training the Model	In K-Means only one parameter is required is for training the model



Clusters formed in K-means and DBSCAN



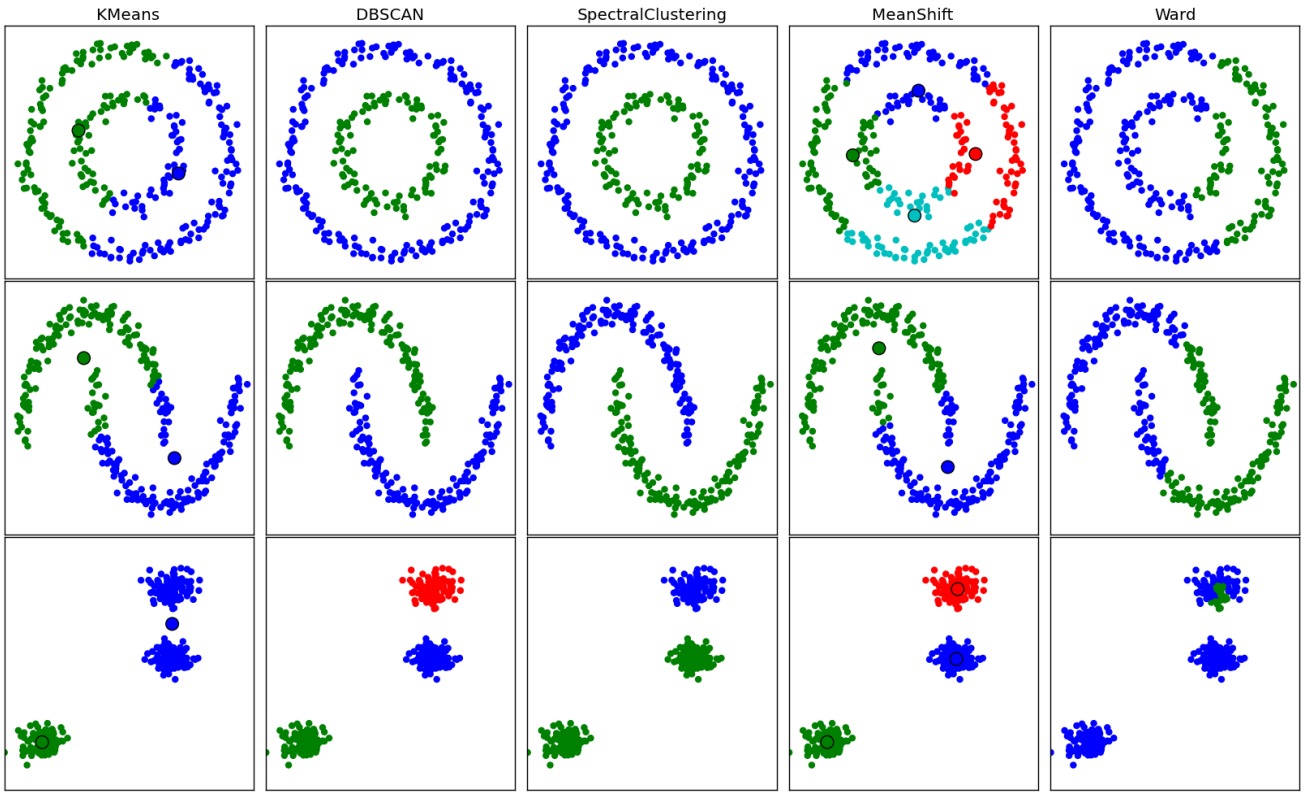
Outlier influence on DBSCAN

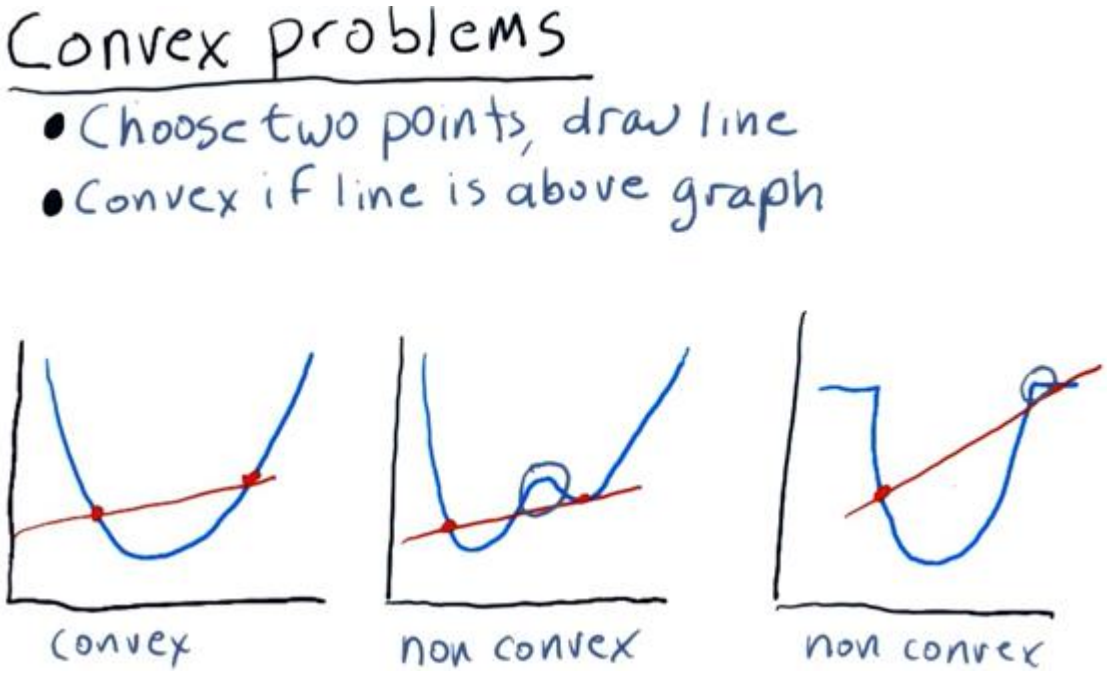
Pros of DBSCAN:
-DBSCAN can discover clusters of arbitrary shape, unlike k-means.
-It is robust to noise, as it can identify points that do not belong to any cluster as outliers.
-It does not require the number of clusters to be specified in advance.

Cons OF DBSCAN:
-It is sensitive to the choice of the Eps and MinPts parameters.
-It does not work well with clusters of varying densities.
-It has a high computational cost when the number of data points is large.
-It is not guaranteed to find all clusters in the data.

Applications

- Suppose we have an e-commerce and we want to improve our sales by recommending relevant products to our customers. We don't know exactly what our customers are looking for but based on a data set we can predict and recommend a relevant product to a specific customer. We can apply the DBSCAN to our data set (based on the e-commerce database) and find clusters based on the products that the users have bought. Using this clusters we can find similarities between customers, for example, if customer A has bought a pen, a book and one pair scissors, while customer B purchased a book and one pair of scissors, then you could recommend a pen to customer B.
- Before the rise of deep learning based advanced methodologies, researchers used DBSCAN in order to segregate genes from a genes dataset that had the chance of mediating cancer.
- Scientists have used DBSCAN in order to detect the stops in the trajectory data generated from mobile GPS devices. Stops represent the most meaningful and most important part of a trajectory.


KMeans
DBSCAN
SpectralClustering
MeanShift
Ward


Convex problems
• Choose two points, draw line
• Convex if line is above graph
convex
non convex
non convex

Reference

1. https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/
2. https://www.youtube.com/watch?v=-p354tQsKrs&t=3s
3. https://python.plainenglish.io/how-does-the-dbscan-algorithm-work-pros-and-cons-of-dbscan-bbdd589d837a
4. https://colab.research.google.com/drive/1Y5rh-O4ECfEHEJ5S3WwFWDovv19zRzy5?usp=sharing