

Mine Automation and Data Analytics

Prof. Radhakanta Koner

Department of Mining Engineering

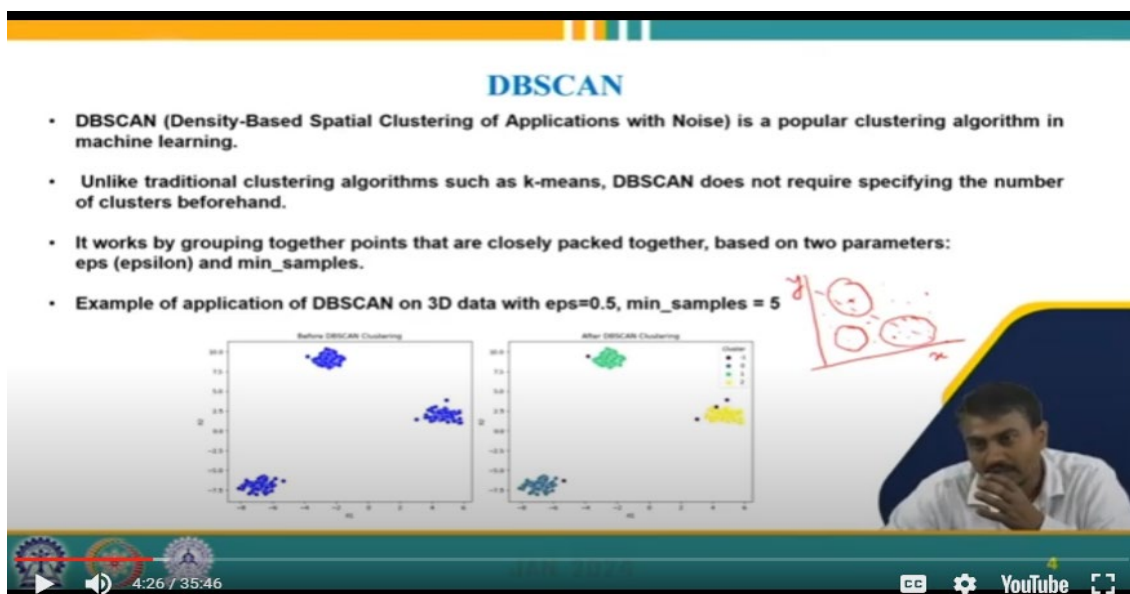
IIT (ISM) Dhanbad

Week - 11

Lecture 53: DBSCAN

Welcome back to my course, mine automation and data analytics. Today, we will discuss another clustering algorithm, the DBSCAN algorithm. So, why are we going to discuss the DBSCAN algorithm? We will discuss it in detail today. So, in this lesson, we will cover what this DBSCAN algorithm is: density-based spatial clustering of applications with noise and its algorithm, and then the fundamental assumptions for this clustering method. Then, its advantages and disadvantages, along with the examples. So, this is a DBSCAN that we are going to discuss today. Before we go into the details, I want to discuss the importance of clustering methods further.

So, for example, let us assume that the data is on the axis and data are distributed like this. For instance, so under this situation for example when these two parameters are plotted in 2D x and y, apparently using the regression method, it would not be easy to fit which line or which way it is fitting more rather than how these points are specially located closely or the affinity to each other in closeness if we measure then it is better to understand that under this category these are the points belongs to some cluster these are the points belong to some cluster some clusters. So, under this assumption, this clustering method gives us a better understanding of the correlations and associations between these points in this space. So, regression does not land on any conclusion.



DBSCAN

- **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** is a popular clustering algorithm in machine learning.
- Unlike traditional clustering algorithms such as k-means, DBSCAN does not require specifying the number of clusters beforehand.
- It works by grouping together points that are closely packed together, based on two parameters: eps (epsilon) and min_samples.
- Example of application of DBSCAN on 3D data with eps=0.5, min_samples = 5

The video shows two scatter plots side-by-side. The left plot is titled 'Before DBSCAN Clustering' and shows a single cloud of blue points. The right plot is titled 'After DBSCAN Clustering' and shows the same data points grouped into three distinct clusters: a green cluster at the top, a yellow cluster on the right, and a blue cluster at the bottom. A hand-drawn diagram to the right of the plots shows three circles representing clusters. At the bottom right of the video frame, a small inset shows a man in a white shirt looking thoughtful with his hand to his chin. The video player interface at the bottom shows a progress bar at 4:26 / 35:46 and YouTube controls.

So, this is the starting point of the clustering method. So, clustering represents some affinity closeness with neighbors, which actually indicates their interrelationship in real life. So, we want to extract that interrelationship, behavior, and pattern from the data using the clustering methods. So, we have already covered the K-means clustering method you have. In K-means clustering, we measure centroid along a centroid, and then from the centroid, we measure the distance, and along the centroid, we assign some cluster 1, cluster 2, and cluster 3. So, for a data set, it is based on the pattern of the data. The first assumption was the number of clusters we wanted to divide.

So, that assumption we started by specifying the number of clusters. Here, the clustering algorithm that we are discussing now is the DBSCAN. Here, we do not require any assumption of how many clusters to subdivide the point in 2D space. So, this DBSCAN, the complete form, is density-based spatial clustering of application with noise and does not require any assumption regarding the number of clusters. So, unlike centroid-based classification, K-means works on density measurement. For example, this is the point. So, how many points is the distance away?

So, if that comes under that, these are the core points and the neighbors again. This is the core point in 4. Several points are coming, so this belongs to some cluster. Similarly, if this density base r is the radius and n is 4, r is equal to 1, so when r is equal to 1, and n is more significant than equal to 4, that is forming a core point around that there is a narrow point so that is within that cluster. So, for each point coming as a neighbor, we put them as a member of that cluster family. So, similarly, this r and n will remain the same throughout this process of iterative iteration. So, several clusters will form, so these algorithms are beneficial when the data is very high dimensional and is handled efficiently by the DBSCAN method.

It works on the principle that the points are closely packed together in the 2D or n -dimensional space. So, here we assume the starting point is the epsilon, which is the radius I mentioned, and the mean sample is the number of points with minimum density we assign along the points. So, this is one example where epsilon is 0.5, and the minimum sample is 5, so here, DBSCAN clustering methods landed three numbers of clustering. So, here, we have not assigned any cluster number, but these algorithms help us identify that there are 3 clusters present. The essential thing is that you can observe that this is the black point. This is the outer layer. It indicates these points do not belong to any cluster because when a core point is a core point, and the radius is drawn along that core point, the number of points coming together is four less than 4.

So, maybe this point is not coming in any neighborhood with any points, so this is out there. So, this clustering method is also very efficient in removing the out layer. So, this is another advantage of this method to handle data and segregate the out layer efficiently. This particular method is very effective for a varying shape and size data set, and it can efficiently handle the

noise I have already mentioned; it can efficiently map the data space data points in the space in dimensional space and form a cluster. So, this shows moon-shaped data points.

So, this kind of moon-shaped data point is widespread in astrophysics data. So, based on this data, space astrophysicists work on that, which is the point that means this is the star I am talking about. So, which of the stars are closely located in which part of the space has more density than the stars, and which part of the space has less density than the stars present? Based on that, they formulate and work to find out the interrelationship and reason behind it, and based on that, research gives us an idea about the size and shape of our galaxies. So, this is a very efficient algorithm, and this particular data assumes epsilon equals 0.

Two, and the minimum sample is 5. So, they have accurately segregated the data points into 3 clusters using the same principle. The starting point is the core point; then, from that core point, we draw a radius of 0.2, which is the radius, and several five samples are found. So, this is a core point again. This is coming through the neighborhood again, and we start from that start and measure.

So, recursively and iteratively, this process goes on. So, the points to which we are measuring the number of satisfied samples—The core points we will discuss again with this. So, for example, if the edge core point is found after that, no epsilon is assumed, but several core points have not been found. So, these are the noncore points or border points. These are how the DBSCAN algorithm functions and advances and finally progressively lands on a cluster of clusters of data. That gives us a picture of the relationship between the data in the n-dimensional space.

So, the core concept in k means clustering. It is also a clustering method, but it works on the principle of centroid base from the centroid. The centroid is adjusted iteratively, and we found the number and the clusters. Here, we cluster by density, assuming density. So, the density of the same dimension higher densities belongs to some clusters where the low density is not within the clusters. This identifies a continuous region in high-density data points compared to the low-density sparsely pointed data that is an outlier to this particular algorithm. So, that helps us to identify and segregate with clusters, and that cluster means something, and the rest of the points are outliers that we are not assuming in our analysis.

So, this is a very efficient algorithm in segregating these two ways: the core points noncore points to a cluster, and the other is the noise. A second important concept is the core point. So, this core point is a point that has at least a specified number of neighbors. That is a minimum point. We have a sample we have already covered and within a specified distance epsilon that is r . Core points are essentially the center points of the clusters. So, there are many center points in the cluster, unlike in the centroid base k, which means clustering; there is only one centroid of a single cluster.

The border points are not the core points themselves, but they lie within the neighborhood that is the epsilon distance the r of a core point. So, these points may be a part of the cluster, but they need more neighbors to be considered core points. As I said, the DBSCAN algorithm is very efficient in removing or ignoring the noise points that are outliers. So, those points are in the pointed space. So, this cluster is an outlier or noise. It will not be considered in the analysis because, within the epsilon radius of the n point, it will not come. It does not fall within that range, so it is an outlier or noise outlier.

Overview of DBSCAN Algorithm

The DBSCAN algorithm proceeds as follows:

- 1. Initialization:** Assign all points as unvisited.
- 2. Core Point Identification:** For each point p , calculate the distance between p and all other points. If the number of points within distance ϵ is greater than or equal to minPts , mark p as a core point.
- 3. Cluster Expansion:** For each core point p , if it's not already assigned to a cluster, create a new cluster and add p to it. Then, recursively add all points that are density-reachable from p to the cluster.
- 4. Noise Point Handling:** Mark all unvisited points as noise or outliers.

So, the points that are neither core points nor border points are considered noise points or outliers. So, these are typically isolated points that do not belong to any clusters. Now, the other parameter is epsilon. So, epsilon is the distance from which we measure the number of sample points within the neighbors, which is the density. So, any point within this distance is considered a neighbor of the core point.

Minimum points are the density minimum number of data points required within the epsilon neighborhood of a point for that point to be considered a core point. Now, how does it work? The work is simple. First, it assigns all unvisited points. So, it thinks of all points first and starts with any points. For example, this is the point ok P , which calculates the distance of the P and all other points. So, if the number of points within the distance epsilon exceeds or equals the minimum point, mark P as a core point. So, this is the core point, this is a core point, this is a core point, iteratively it does.

So, based on that, if each cluster point P is not already assigned to a cluster, create a new cluster C . For example, this is a cluster C . It produces and adds that P to this cluster. So, new cluster core points are added to the cluster points, and it does recursively. So, we add all the density-reachable points from point P . So, if I assume it is P_i , another P_j is the new point we want to

find. So, we measure the distance between P_i and P_j . If it is within that epsilon epsilon, then this P_j can be added to the cluster of the P_i ok.

Similarly, if the P_j is the centroid or the core points and P_i is within that range of epsilon, then P_i will be considered the member of the P_j clusters. Then, the next step is to mark all unvisited points as noise. So, those are the points not reached. Any iterations or the last iteration does not reach this point. For example, in the n iteration, it has not reached any core point center and is extended to that. So, it is a noise outlier. So, what was the algorithmic step of this DBSCAN? So, given a data set D , this is a data set ok, and we consider the n number of data points, which has $P_1 P_2$ up to P_n , where each point is represented as a d -dimensional vector.

So, here, the DBSCAN algorithm identifies the cluster based on the density of points in the data set. So, we assume first that the epsilon is the radius of the neighborhood around each point. The minimum point is a density that is a minimum number of points within epsilon to consider a point a core point. So, in step number 1, for each point P_i , P_i lets us calculate its neighbor by finding all points that lie within a distance epsilon from P_i ok. This can be mathematically represented as $N(P_i)$, which is the number of points that is a neighbor of the P_i . So, let us do this exercise on the point P_j ; P_j is a d -dimensional space vector member.

$$N(p_i) = \{p_j \in D \mid \text{distance}(p_i, p_j) \leq \epsilon\}$$

So, let us calculate the distance between P_i and P_j . If it is less than equal to epsilon, then we will add that P_j as the neighbor of the P_i ok. So, the distance between P_i and P_j is the metric used to measure the distance between point P_i and P_j , commonly the Euclidean distance. We have already covered many distances, including Minkowski distance, Hammer distance, and so many—the core number is step number 2, the core point identification.

DBSCAN Algorithm Steps

Given a dataset D consisting of n data points $\{p_1, p_2, \dots, p_n\}$, where each point is represented as a d -dimensional vector, DBSCAN identifies clusters based on the density of points in the dataset.

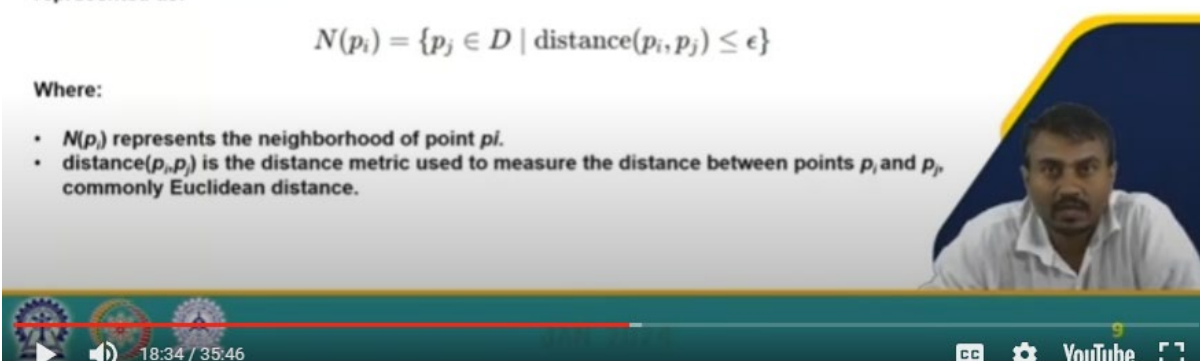
- ϵ is the radius of the neighborhood around each point.
- minPts is the minimum number of points within ϵ to consider a point a core point.

Step 1: Density Calculation
 For each point p_i , calculate its neighborhood by finding all points within a distance ϵ (epsilon) from p_i . This can be represented as:

$$N(p_i) = \{p_j \in D \mid \text{distance}(p_i, p_j) \leq \epsilon\}$$

Where:

- $N(p_i)$ represents the neighborhood of point p_i .
- $\text{distance}(p_i, p_j)$ is the distance metric used to measure the distance between points p_i and p_j , commonly Euclidean distance.



So, P_i was the starting point. So, if we found that P_i in the neighbor with epsilon radius this cardinality that is $N(P_i)$, suppose this is $N(P_i)$ is a particular value. For example, $N(P_i)$ is a value γ . It may be three, it may be four, it may be ten. Ok, it is a constant value for this method for this algorithm for the starting point, so if this P_i found that around that epsilon radius, it satisfied the cardinality of its neighbor greater than equal to these three if these three are more significant than 3 4 5 like that, then P_i is the core point ok. So, mathematically, we must satisfy that $N(P_i)$ must be greater than or equal to minimum points. Step 3, the two points P_i and P_j are density reachable if a chain of points exists connecting them such that every consecutive pair in the chain is within the neighborhood of each other. So, formally speaking, point P_j is the density reachable points from P_i if there exists a chain $P_1 P_2 P_k P_j$ such that P_1 is equal to P_i or P_1 is a core point P_k is equal to P_j P_k is directly reachable on the P_k minus one like that and so on.

$$|N(p_i)| \geq \minPts$$

Step number 4, so now we have to form the cluster. So, for each core point P_i that has not been assigned to a cluster, we have to create a new cluster that is C . So, let us add the P_i to cluster C , add P_j to cluster C , add P_k to cluster C . So, because this is the points that are lying within the same cluster. So, finally, those are the points not reached by these iterative methods.

In step number 5, the points not assigned to any cluster are considered noise points or outliers. Now, let us discuss the assumptions of this method because machine learning approaches the assumptions, and understanding the assumptions is very important. So, based on this understanding, we can efficiently use the particular machine learning algorithm for a specific case because all machine learning algorithms are not a default suitable for all kinds of data sets. So, we have to understand the data set. We have to understand the pattern of the data set based on which type of machine learning algorithm we will use and which machine learning algorithm will function well under these circumstances. So, what are the assumptions for the DBSCAN algorithm? The first assumption is the density.

So, we assume that the data are located in the n -dimensional space densely, and this densely packed data is formatted as clusters. So, the points within the clusters are densely packed together, while the points outside the cluster region are relatively sparse. So, the density packing of the data represents something about their association in the n -dimensional space. The varying density and the data density within that n -dimensional space may not be equal. So, under these conditions, the DBSCAN algorithm can handle, up to some extent, varying shape and size data. Still, other algorithms, such as the centroid-based algorithms, will not work efficiently.

So, it will struggle, especially when the clusters have irregular shapes and varying densities. So, this is the relative advantage compared to the centroid-based approach of the clustering method; the DBSCAN algorithm can function very well. In the presence of noise, this algorithm assumes the data set may contain the noise, which refers to the points that do not

belong to any clusters or have insufficient support to be considered a part of the clusters. So we can level those data as outliers quickly. Parameter sensitivity: we have started our discussion with r and m , which is epsilon and the minimum points.

So, it is sensitive to this assumption that the parameters, but tuning the parameter epsilon, minimum points, and mean points is relatively straightforward. This algorithm is relatively easy and robust enough to adjust well. The metric space we operate in is a metric space assuming the distance metric used to measure the proximity of the points to believe that they fundamentally belong to some association. So, this choice of distance metric can impact the clustering results. Selecting a metric that aligns with the data's characteristics and domain knowledge is essential.

The screenshot shows a video player with a title bar that reads "Assumptions of DBSCAN". The video content consists of three bullet points:

- **Presence of Noise:** DBSCAN assumes that the dataset may contain noise, which refers to data points that do not belong to any cluster or have insufficient support to be considered part of a cluster. The algorithm is designed to handle noise effectively by labeling such points as outliers.
- **Parameter Sensitivity:** DBSCAN's performance can be sensitive to its parameters, namely the epsilon (ϵ) neighborhood radius and the minimum number of points (MinPts) required to form a dense region. Choosing appropriate values for these parameters is crucial for the algorithm to produce meaningful clusters. However, DBSCAN is relatively robust to parameter settings compared to other clustering algorithms.
- **Metric Space:** DBSCAN operates in a metric space, assuming that the distance metric used to measure the proximity between data points is meaningful and appropriate for the dataset. The choice of distance metric can impact the clustering results, so it's essential to select a metric that aligns with the data's characteristics and domain knowledge.

Handwritten notes in red ink are visible on the right side of the slide, with "r" and "E" written vertically and "minPts" written horizontally. A small inset video of a man is visible in the bottom right corner of the slide area. The video player interface at the bottom shows a progress bar at 26:44 / 35:46, a play button, a volume icon, and YouTube controls.

For data processing, the DBSCAN algorithm assumes that the input data has been pre-processed appropriately, including handling missing values, scaling features if necessary, and removing irrelevant features or noise. So, pre-processing ensures that the clustering algorithm operates on clean and meaningful data, leading to more accurate clustering results. So, understanding all these assumptions is very important in applying the DBSCAN in practical examples. That is why, based on the data characteristics, we must understand the suitability of using the DBSCAN for that data. So, let us discuss the advantages of this algorithm.

It is robust to noise and outliers because it can efficiently segregate the noise that will not reach by any methods, such as epsilon and min points. So, those are the outliers. Ability to detect arbitrary shapes: we have already shown that it can handle the arbitrary shape size and density data compared to other clustering algorithms, and it does not require pre-specification of the number of clusters. In the first example we have covered, the three clusters are identified easily

without assuming that there are three clusters because the density based on the density epsilon is the radius and min points; we have segregated three clusters.

So, this is another advantage. Parameter tuning is relatively more accessible than other algorithms and is very efficient in handling large data sets. Okay, so this is another advantage. The disadvantage is that though min points and epsilon are straightforward to fine-tune, sensitivity is there based on the shape and size, so which parameter to select that we have to understand? So, for a wildly varying density data set, it isn't easy to apply the DBSCAN because we are assuming the densities, the four or three, and the neighbor points that will if they are not equal all along the data space.

So, it will lead to wrong conclusions. So, several clusters might be high and need to be more meaningful for that data set. Memory and computational requirements are because of all the data; the entire data set is saved in the memory when it is a large data set. We calculate the distance that is computing the distance. So, determining the neighborhood relationship requires intensive computational power for the high dimensional data. So, we have to keep that in mind. Cluster border determination has certain chances of misclassification at the border.

So, we have to understand very clearly when we apply this particular method, which is unsuitable for all types of data. As I said, all machine-handling algorithms have some pros and cons. So, to which data set which kind of data set we will apply is a critical decision of the user. So, that decision is to be taken based on understanding the algorithm very well; based on that, our performance, use, and efficiency will be very, very high. Let's go to the application side, where we can apply this method.

For example, you know of minerals specially situated in different locations, unlike coal. Coal is a very uniform deposit. So, based on the distance and other metrics, maybe your grade, the height, the spatial locations, it may be found out like this, it may be found out like this, it may be found out like this. So, this represents the potential area of suitable mineral that is found. So, when we plan for the following mining stage, we have to apply our method in such a way that efficiently, with less excavation, we can reach those points.

So, in this case, in the mine planning using the exploration data and detailed exploration data, we can use which route to take and which way to take so that less excavation is to be taken and good recovery is yielded based on this particular choice. So, for this kind of mining application, mining engineering applications, we can apply the DBSCAN algorithm to find out the best route or best path to excavate to extract these ores. So, this identification of the clusters helps ensure a smooth mining planning process. Ore body detection is similar to the example we cited based on the different data, where the ore grade is high, and the ore grade is low, and based on that particular mixing when required, we can do that kind of operation. We can mix and finally reach a uniform grade of ore to supply the final product to the third company.

Geotechnical monitoring, based on suppose we when monitoring is done, the number of data in real-time monitoring, and the number of data is high.

So, those data are plotted. For example, that data is not essential, but based on the affinity of the closeness, it gives many pictures and much understanding of the behavior of the movements. So, in those cases, this DBSCAN algorithm might be constructive to comprehend the nearness, the affinity of those points that are closely located, and that means something that when this kind of deformation, when this kind of movement occurs, a date, time, a particular situation after that or after before that, so that helps us to handle geotechnical structure in the mine better. Equipment maintenance and optimization are similar to the data we are collecting continually about the equipment behavior, and we are doing the maintenance regularly based on that specific association that gives us a clear picture of which side to give more importance to and which side does not give importance right now. Prioritizing maintenance and prioritizing the sequence of use of the machines helps use this DBSCAN algorithm. Environment monitoring is also similar to geotechnical structure monitoring. In environmental monitoring, this data can be plotted and understood which data forms clusters that give a better understanding of which part to take action on.

So, by understanding the nature of the algorithm, we can apply these methods. So, these are the references. So, let me summarize in a few sentences what we have covered today. So, we have covered a new algorithm of the clustering method, which is the DBSCAN algorithm. We have covered the algorithmic steps, assumptions, advantages and disadvantages, and applications. Thank you.