**03 - DBSCAN**

* We studied two clustering methods; K-Means which is a centroid-based algorithm, and Agglomerative Clustering which is a hierarchical system.
* There’s one more approach to clustering known as DBSCAN which is a density-based approach.
* DBSCAN refers to Density-based spatial clustering of applications with noise
* DBSCAN works fairly well with large data and is able to handle noise and outliers very efficiently.
* First things first, here are some key ideas that build the DBSCAN.

# **Density and Dense Region**

* DBSCAN uses a concept of density, which can be defined as;
  + at a certain point 𝑃, density at point 𝑃 is the number of points within a hypersphere centered at 𝑃 with a radius of 𝑒𝑝𝑠𝑖𝑙𝑜𝑛
* Now, consider any region around the point 𝑃 within 𝑒𝑝𝑠 radius, if there are more data points than 𝑚𝑖𝑛𝑝𝑡𝑠, we call the region a **Dense** region.
* For example, let's say we have 𝑒𝑝𝑠=1 and 𝑚𝑖𝑛𝑝𝑡𝑠=10. Consider two points 𝑃1 and 𝑃2, both with a radius of 𝑒𝑝𝑠
  + Suppose there are 20 points around point 𝑃1, and only 6 points around point 𝑃2, within the radius of 𝑒𝑝𝑠, then we say the region around point 𝑃1 is dense and the region around point 𝑃2 as non-dense.

# **Min Points(**𝑚𝑖𝑛𝑝𝑡𝑠**) and Epsilon(**𝑒𝑝𝑠**)**

* 𝑚𝑖𝑛𝑝𝑡𝑠 are the minimum number of points that we need in a hypersphere around point 𝑃 with the radius of 𝑒𝑝𝑠 for considering the region as a **Dense** region.
* 𝑚𝑖𝑛𝑝𝑡𝑠 acts like a certain threshold and 𝑒𝑝𝑠 are the radius of the hypersphere

# **Core Point**

* If a point 𝑃 has points ≥𝑚𝑖𝑛𝑝𝑡𝑠 within the radius of 𝑒𝑝𝑠, then 𝑃 is a core point.
* This also implies that point 𝑃 has a dense region around it

# **Border Point**

* A point 𝑃 can be defined as a border point if:

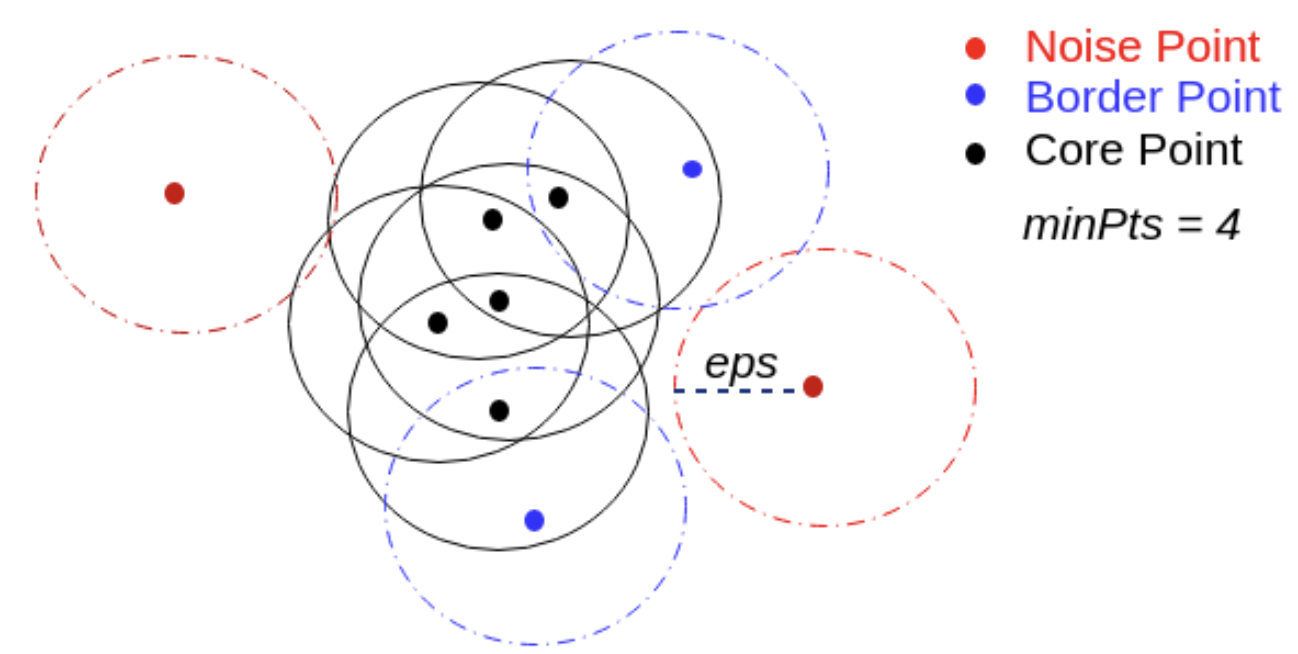
1. 𝑃 is not a core point
2. Point 𝑃 lies in the neighborhood of point 𝑄 such that point 𝑄 is a core-point

## **Neighborhood**

* A point 𝑃 is said to be in the neighborhood of point 𝑄 if distance between point 𝑃 and 𝑄 is less than 𝑒𝑝𝑠 value; i.e. 𝑑𝑖𝑠𝑡(𝑃,𝑄)≤𝑒𝑝𝑠

# **Noise Point**

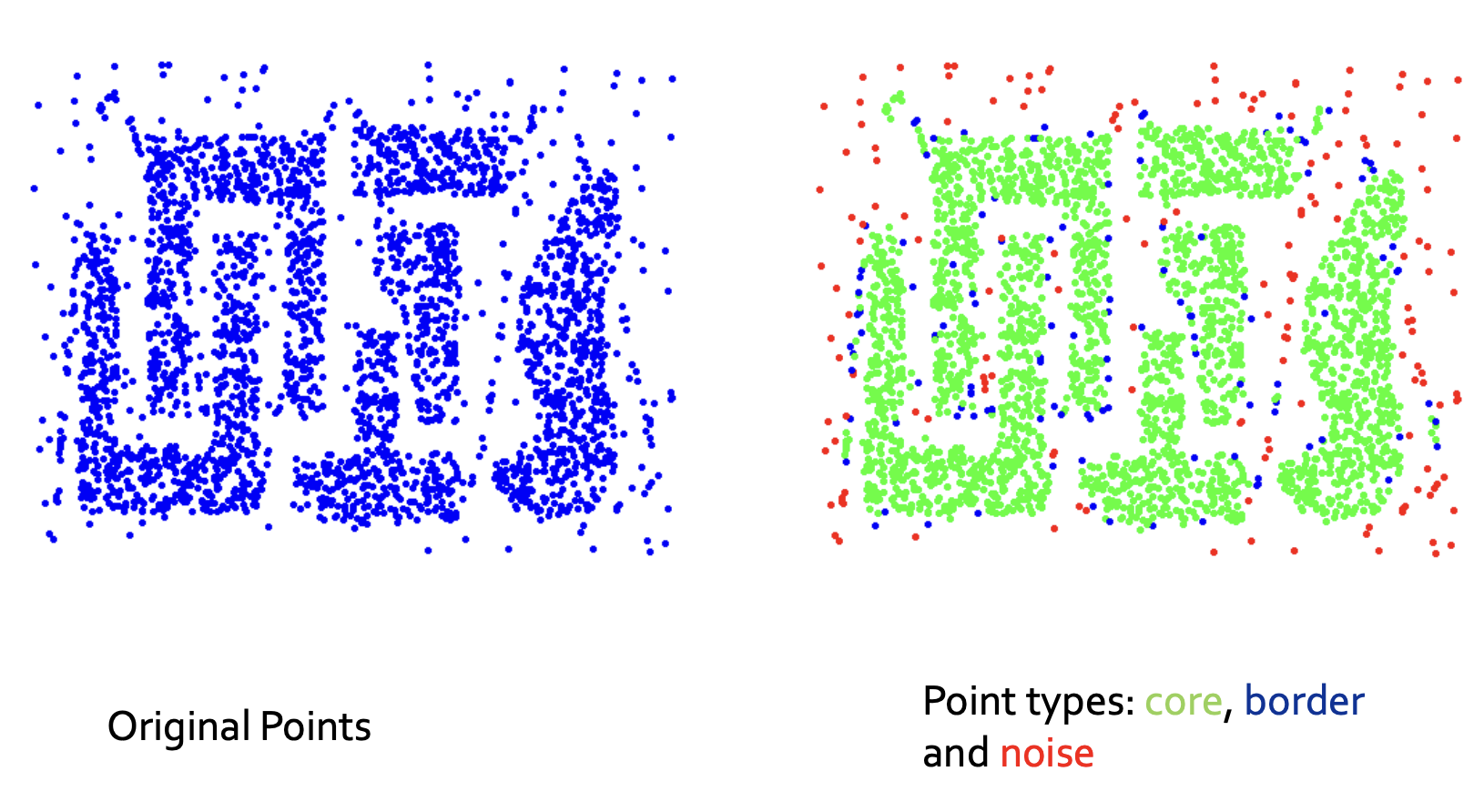
* It is a point that is neither a core point nor a border point.
* Suppose around core point 𝑃, a border point 𝑄, and a point 𝑅 which is in a non-dense region, the point 𝑅 is said to be a noise point



* One thing to understand is that, when using DBSCAN, we fix two things:

1. Min Points
2. Epsilon.

* By fixing these hyperparameters, we get core points, border points, and noise points as well



# **Density Edges and Density Connected Points**

* If points 𝑃 and 𝑄 are two core points and the distance between point 𝑃 and 𝑄 is less than or equal to 𝑒𝑝𝑠 value, then an edge between point 𝑃 and 𝑄 is known as a **density edge.**
* Points 𝑃 and 𝑄 can be said as density-connected points;
  + if both points are core points
  + if there exist other density edges connecting the points 𝑃 and 𝑄
* Imagine we have two core points, point 𝑃, and 𝑄, and there are other core points connecting point 𝑃 with point 𝑄; say 𝑃1,𝑃2,....𝑃𝑛, where the distance between each point 𝑃1,𝑃2,....𝑃𝑛 is less than 𝑒𝑝𝑠
* Then point 𝑃 and point 𝑄 are said to be density connected points.

# **DBSCAN Algorithm**

### **Step-1:**

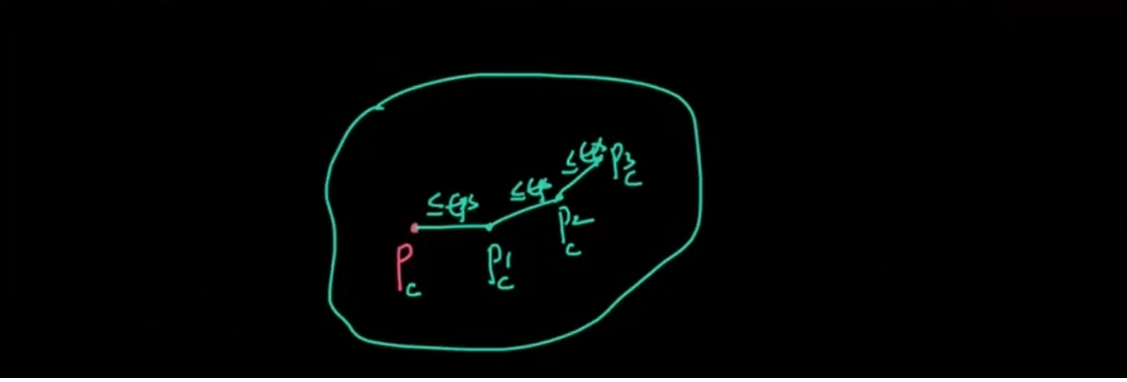
* For each point 𝑥𝑖 that belongs to the dataset 𝐷, label it as either core point, border point, or noise point.
* Time complexity of this step would be 𝑂(𝑛∗𝑙𝑜𝑔𝑁)

### **Step-2:**

* Remove all the noise points from the dataset
* Time complexity of this step would be 𝑂(𝑛)
* This is basically a noise removal step

### **Step-3:**

* For each core point 𝑃 that is not yet assigned to any clustered:
  + create a new cluster with point 𝑃
  + Add all points that are density connected to point 𝑃, to the 𝑃's cluster
* To understand this with an example, Consider a core point 𝑃 and there are three core points 𝑃1,𝑃2 and 𝑃3 which are density connected.
* Then, we group all the three points in the cluster of point 𝑃
* Time complexity of this step would be 𝑂(𝑛∗𝑙𝑜𝑔𝑁)



### **Step-4:**

* For each border point, we assign it to the nearest core points' cluster.
  + For example, if we're having a cluster having core points 𝑃1,𝑃2....𝑃9, and a border point P10 which is near the cluster.
  + We merge border point P10, into the cluster of core points 𝑃1,𝑃2....𝑃9
* Time complexity of this step would be 𝑂(𝑛)∗𝑙𝑜𝑔𝑁
* **DBSCAN Animation Links:**
* [DBSCAN Animation](https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/)
* [DBSCAN Animation Statquest](https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/016/028/original/ezgif-5-dc143adcc5_%281%29.mp4?1665381737)

# **Adjusting Min Points**

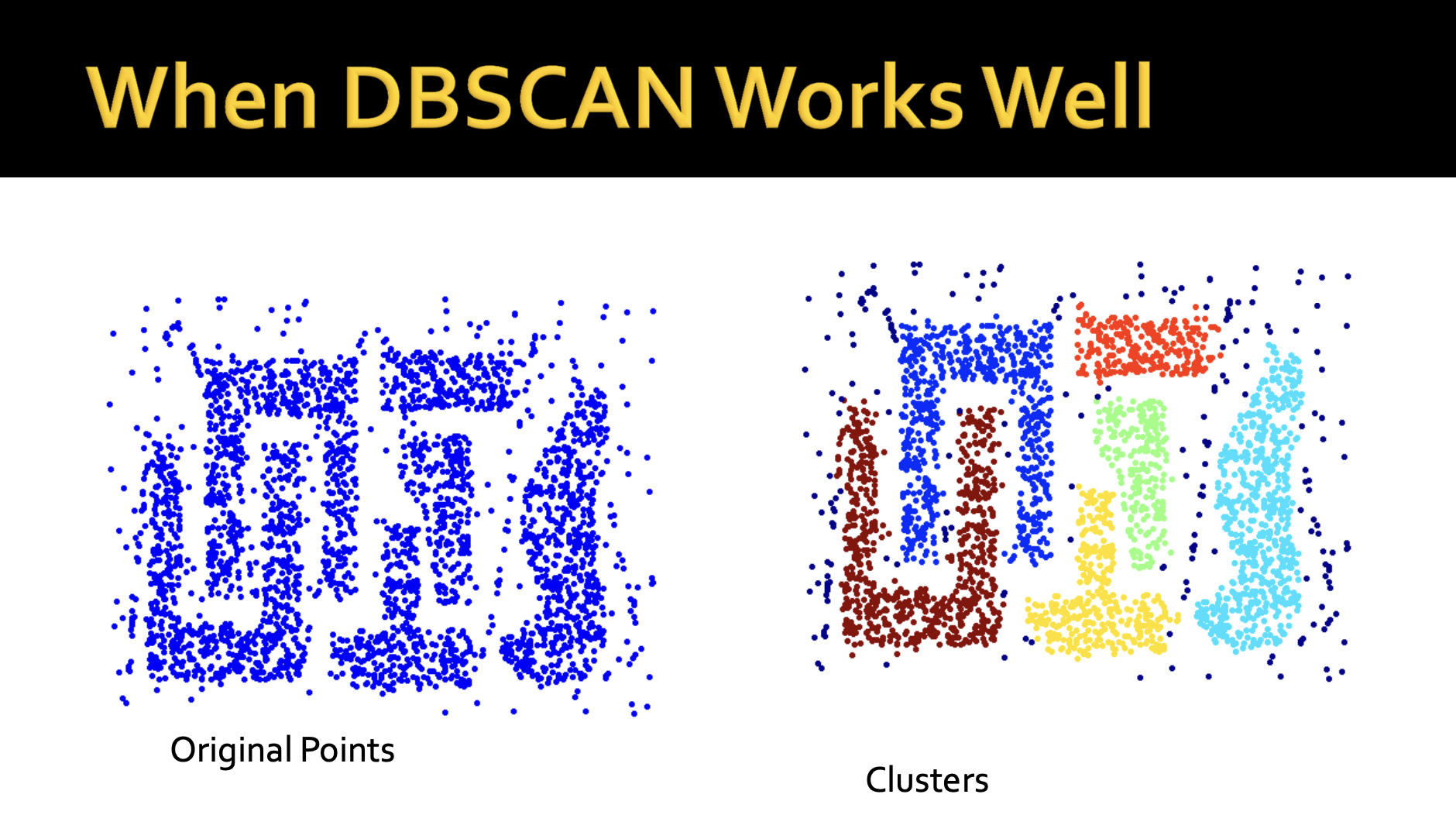
* So there are some rules of thumb that people have made over the past years, which typically works well. They are:
  + value of 𝑚𝑖𝑛𝑝𝑡𝑠 should be greater than or equal to 𝑑+1; where 𝑑 is dimensionality of the data
  + lot of libraries use the value of 𝑚𝑖𝑛𝑝𝑡s approximately equal to 2∗𝑑
* The points mentioned above are typically rules of thumb and these are used because they tend to work fairly good in most of the cases
* Given an epsilon value, if the dataset is noisy, we pick larger 𝑚𝑖𝑛𝑝𝑡𝑠

# **Adjusting Epsilon**

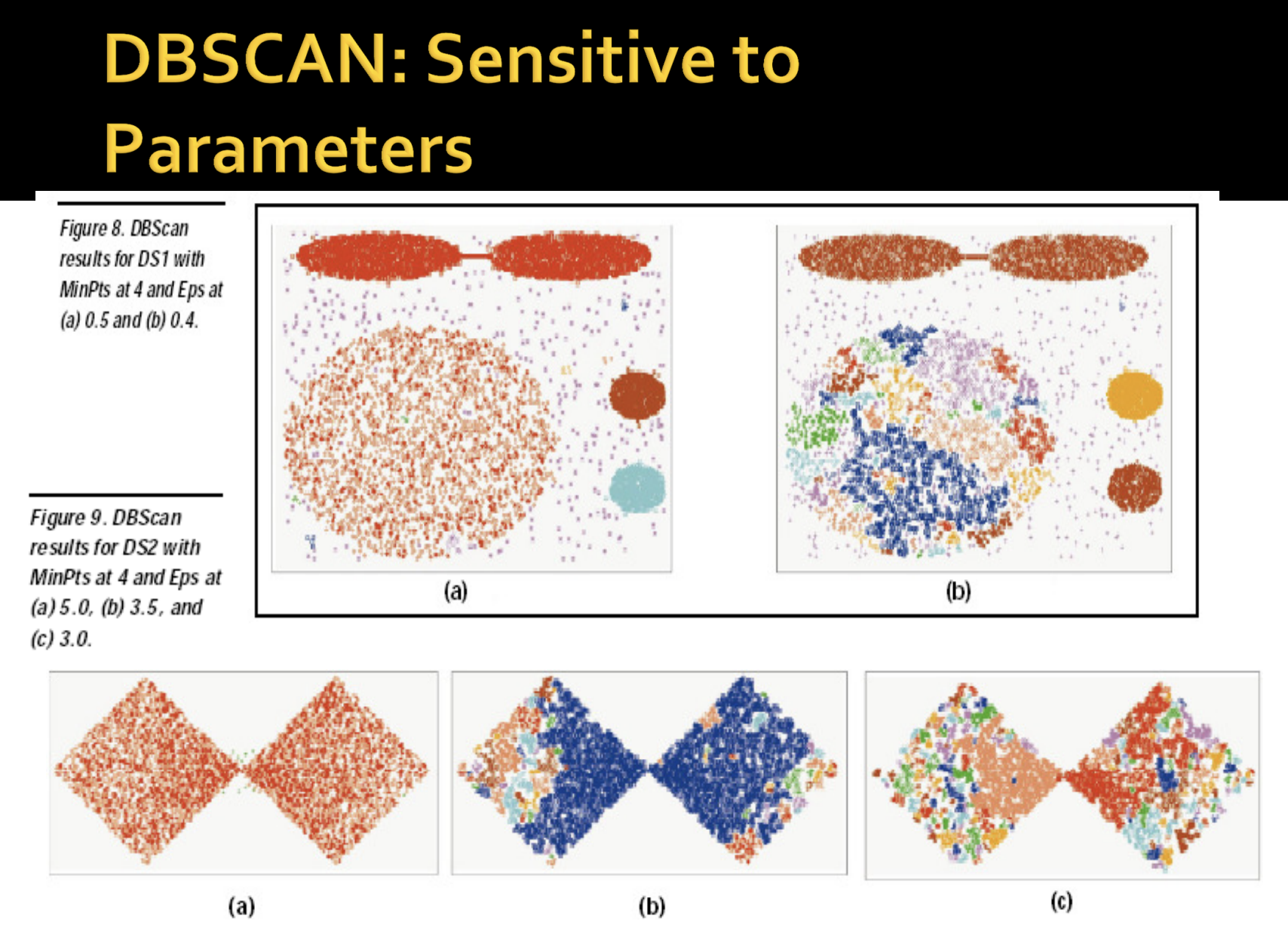
* Let's assume we've fixed the value of 𝑚𝑖𝑛𝑝𝑡𝑠 = 4.
* **Step 1:**
  + for every point 𝑥𝑖 in dataset, we compute a distance 𝑑𝑖
  + 𝑑𝑖 refers to the distance from 𝑥𝑖 to 𝑥𝑖's 4𝑡ℎ nearest neighbor (because we've set 𝑚𝑖𝑛𝑝𝑡𝑠 = 4)
* **Step 2:**
  + Sort the values of 𝑑𝑖's and plot them. You'll notice that the distance will increase graudally and then suddenlly, at a certain point, the value of distance will get boosted
  + So, the index at which the value of 𝑑𝑖 distance got boosted will be used as the value of 𝑒𝑝𝑠
  + The indices having higher values of 𝑑𝑖's will be outliers

# **Advantages of DBSCAN**

* It’s resistant to noise
* Can handle clusters of different shapes and sizes.
* It doesn’t require one to specify the number of clusters a priori.
* It requires only two parameters: MinPts and Epsilon.
* It is designed for use with databases as it’s created by the database community.



# **Limitations of DBSCAN**

* Even a small change in the hyperparameters, we can get a completely different type of clusters. So, it’s quite sensitive to the choice of hyperparameters.
* Cannot handle varying densities and data with higher dimensions.

