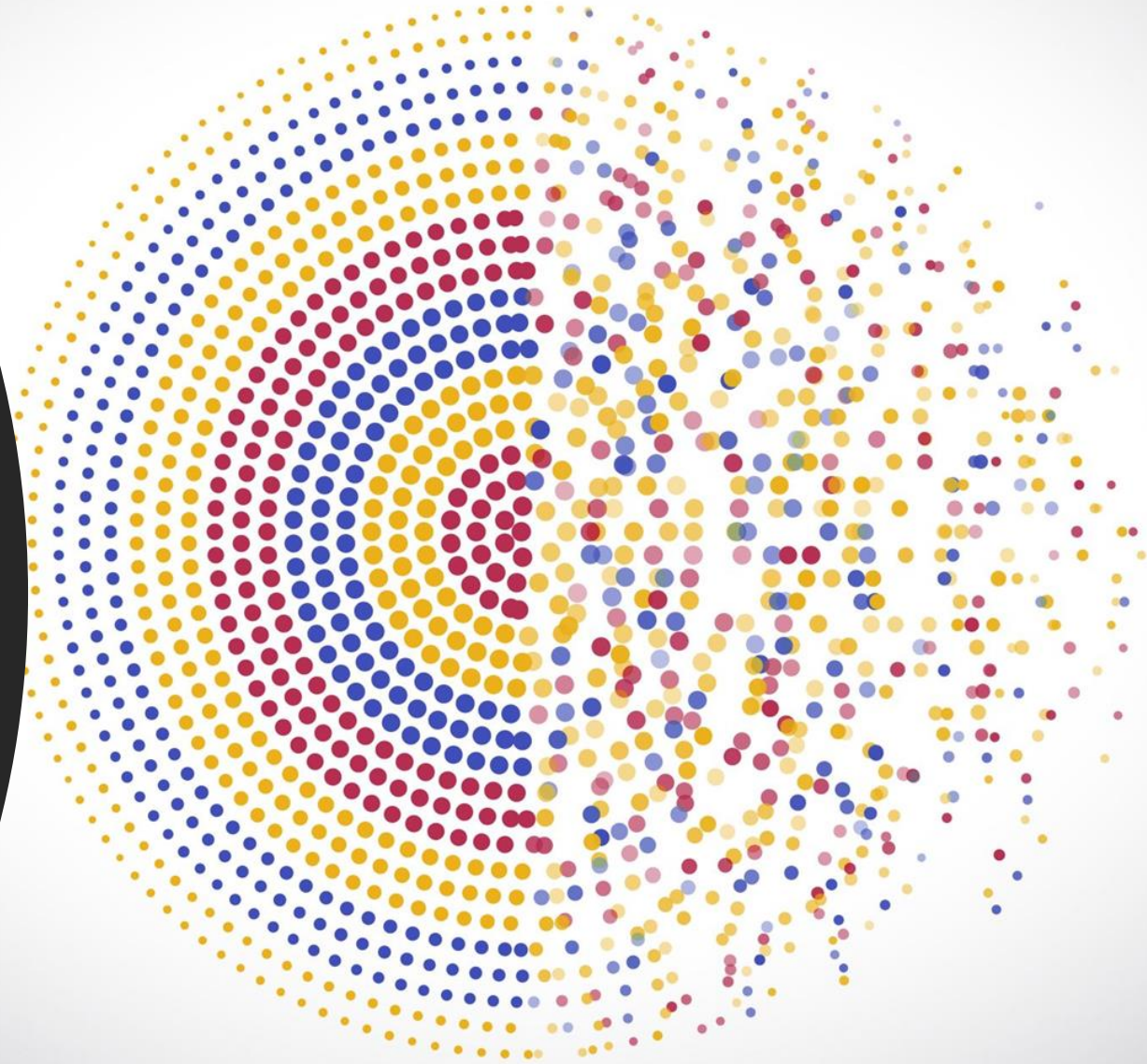
Unsupervised Learning

Dr. Mohamed AlHajri



Content

- *k*-means clustering
- Density-based spatial clustering of applications with noise (DBSCAN)
- Soft-clustering

k-means clustering

1. Specify the number of clusters k.
2. Randomly select the representative $(z_1, \cdots, z_k)$.
3. Iterate until there is no change in the cost

3.1 Given $(z_1, \cdots, z_k)$, go over all x_i and for each x_i assign to it the closest z_j, more precisely:

$$\min_{j=1,\cdots,k} \left|\left|x_i - z_j\right|\right|_2^2$$

and the total cost function is:

$$cost(z_1, \cdots, z_k) = \sum_{i=1}^{n} \min_{j=1,\cdots,k} \left|\left|x_i - z_j\right|\right|_2^2$$

$$C_j = \{i | z_j is\ the\ closest\ to\ x_i\}$$

3.2 Given partitions (clusters) $(C_1, \cdots, C_k)$ find the best representative

$$cost(C_1, \cdots, C_k) = \min_{z_1,\cdots,z_k} \sum_{j=1}^{k} \sum_{i \in C_j} \left|\left|x_i - z_j\right|\right|_2^2$$

k-means clustering

3.2 Given partitions (clusters) $(C_1, \cdots, C_k)$ find the best representative

$$cost(C_1, \cdots, C_k) = \min_{z_1, \cdots, z_k} \sum_{j=1}^{k} \sum_{i \in C_j} \left|\left| x_i - z_j \right|\right|_2^2$$

- The step 3.2 is the general way to find the minimum, but in this case and since we are assuming we have a squared Euclidean distance we can find it much easier and with a lower computational complexity.

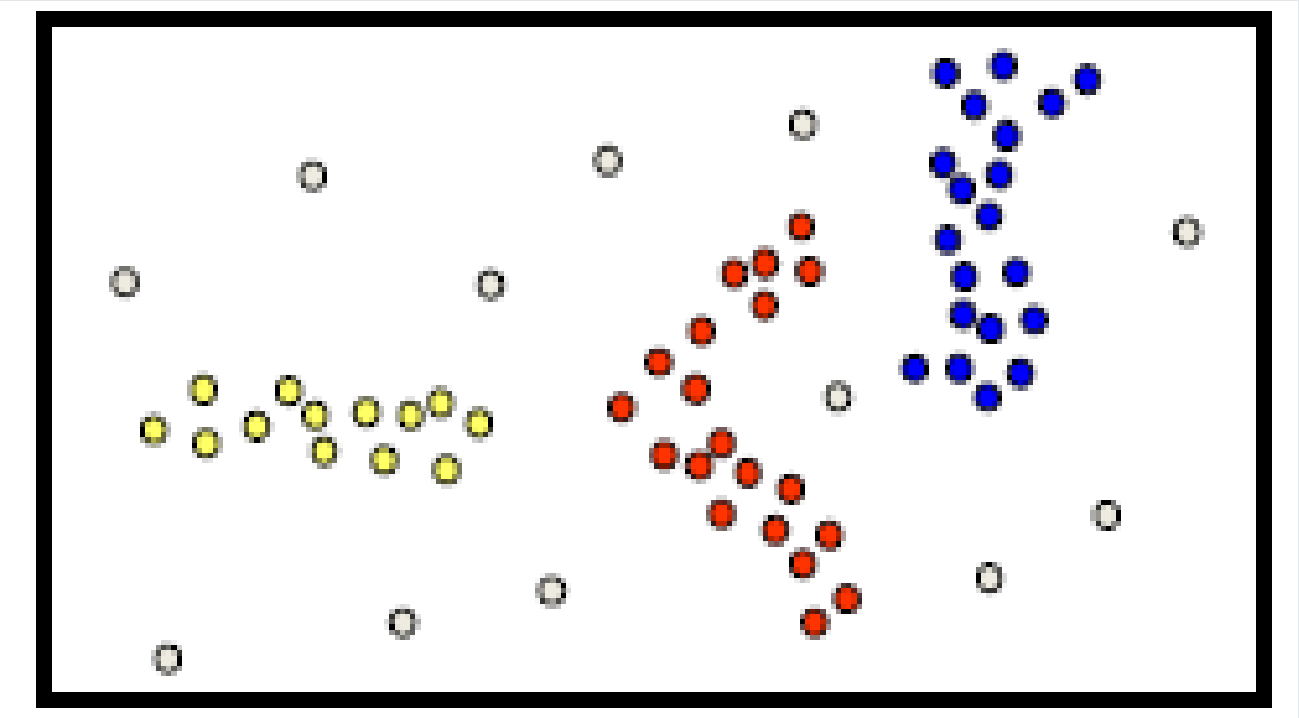
$$\frac{\partial}{\partial z_j} \sum_{i \in C_j} ||x_i - z_j||_2^2 = 2 \sum_{i \in C_j} x_i - z_j = 0$$

$$\implies \sum_{i \in C_j} x_i = \sum_{i \in C_j} z_j$$

$$\implies z_j = \frac{\sum_{i \in C_j} x_i}{|C_j|}$$

- An important thing to keep in mind here is that what we derived is only applicable when we have a euclidean squared distance.
- In addition, the way we solve this problem is in an alternating manner and so the convergence of the k-means algorithm is local and this means every time the initialization is changed we will have a different result.

Density Based Clustering

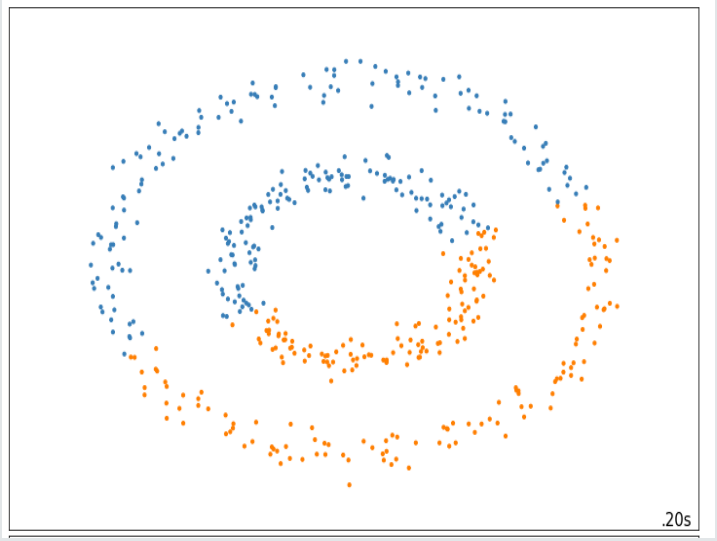


- Clusters are dense regions in the data space, separated by region of lower object density.
- A cluster is defined as a maximal set of density connected points.
- Therefore, clustering based on density (local cluster criterion), such as density-connected points
- DBSCAN

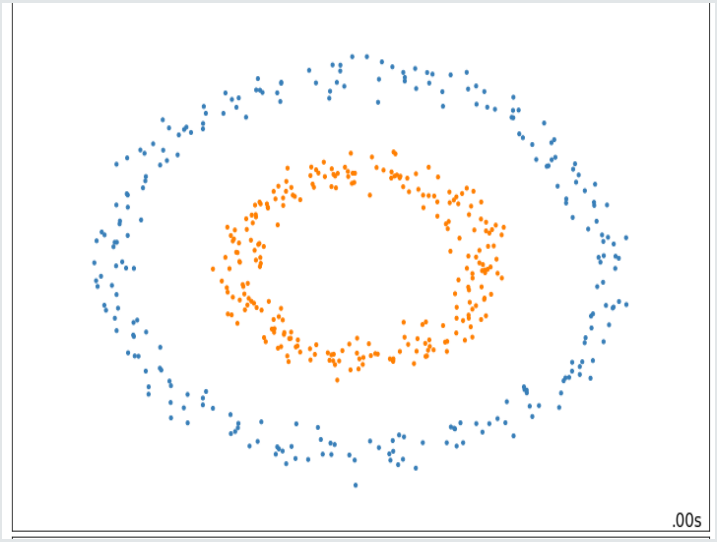
DBSCAN

- Major Features:

  1- Discover Clusters of arbitrary shape



k-nn



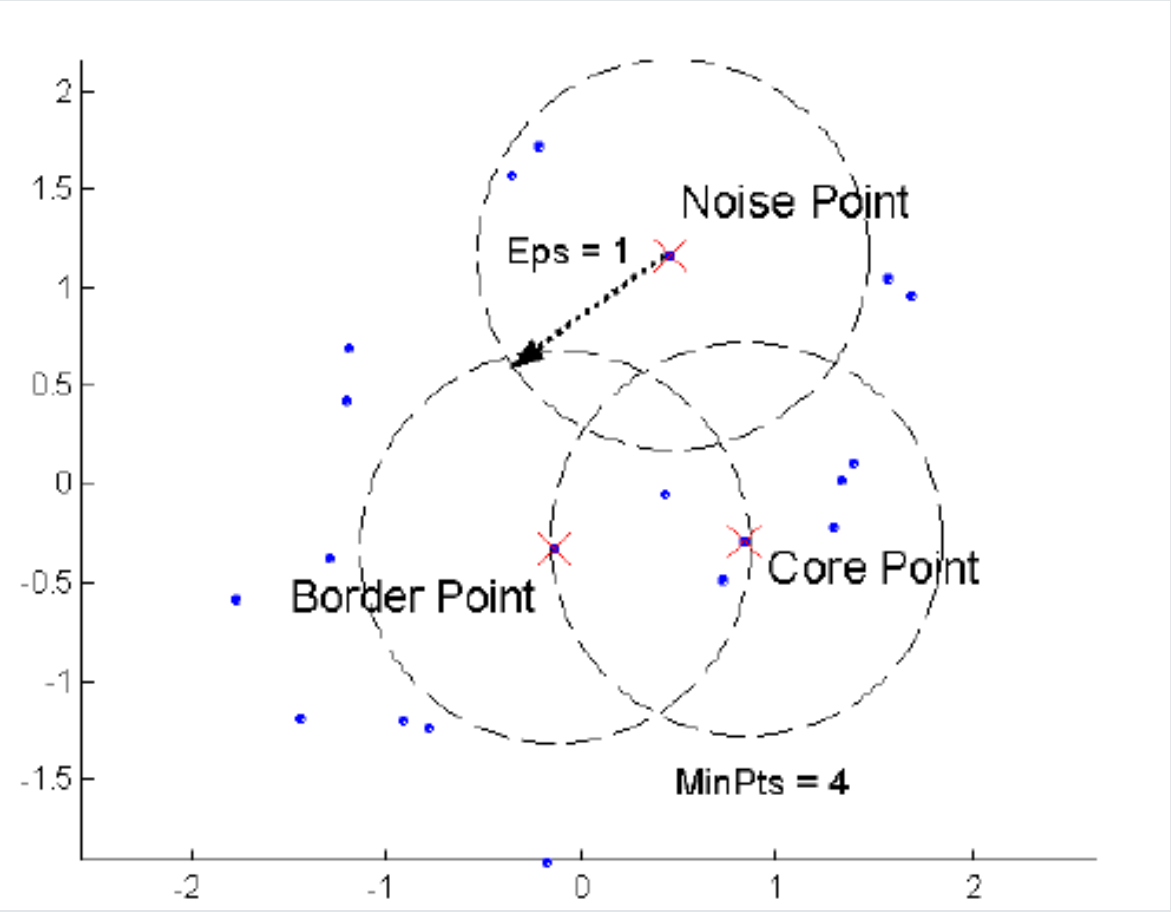
dbscan

2- Handle noise: Can handle noise by design. Points that do not belong to any dense region are classified as noise or outliers, making DBSCAN useful for datasets with noisy or irregular data.

DBSCAN

- DBSCAN is a density-based algorithm:
- Density = number of points within a specified radius *r* and *eps*
- **Core point:** It is a point that has more than a specified number of points (pts_{min}) within eps.
- **Border point:** It is a point that has fewer points that (pts_{min}) within eps, but it is in the neighborhood of a core point.
- **Noise point:** It is any point that is not a core point or border point.

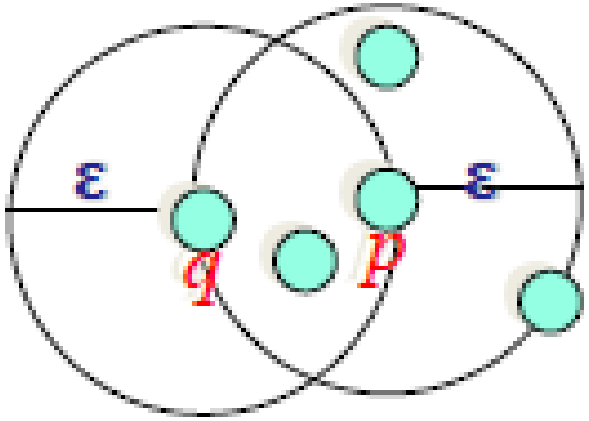
DBSCAN



https://csc.csudh.edu/btang/seminar/slides/DBSCAN.pdf

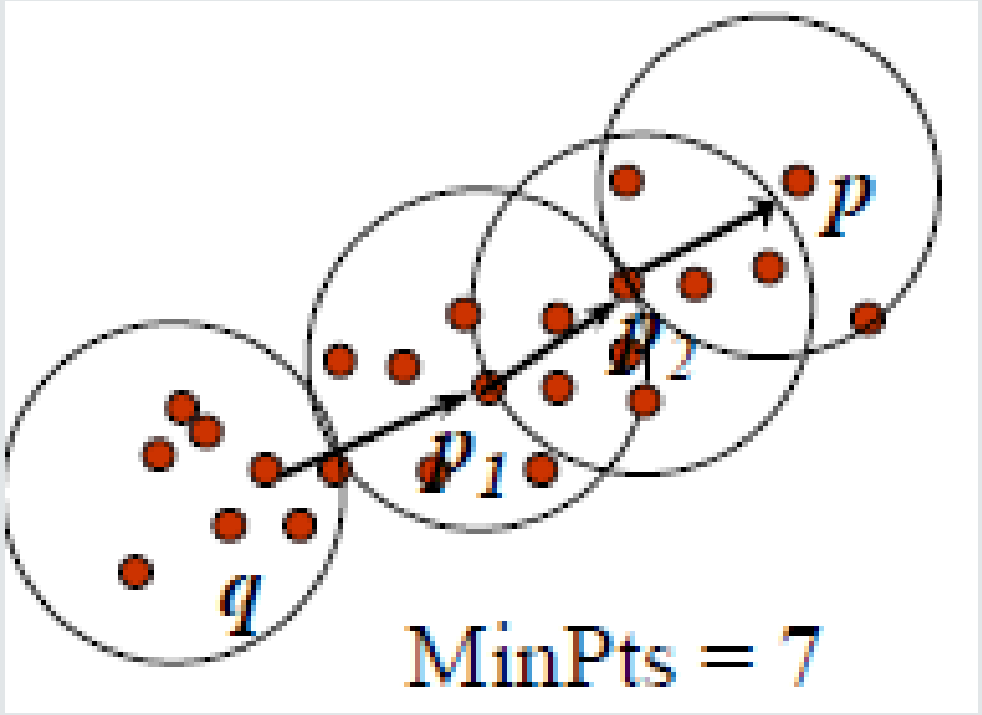
Density-reachability

- Directly density-reachable: Any object *q* is directly-reachable from object *p* if *p* is a core object and *q* is in ϵ-neighborhood.
- *q* is directly density-reachable from *p*
- *p* is not directly density-reachable from *q*
- Density-reachability is asymmetric



Density-reachability

- Density-Reachable (directly and indirectly):
  - *A point p is directly density reachable from* p_2
  - p_2 *is directly density-reachable from* p_1
  - p_1 *is directly density-reachable from* q
  - $p \leftarrow p_2 \leftarrow p_1 \leftarrow q$ *form a chain*



- p is (indirectly) density-reachable from q
- q is not density-reachable from p.

DBSCAN Algorithm

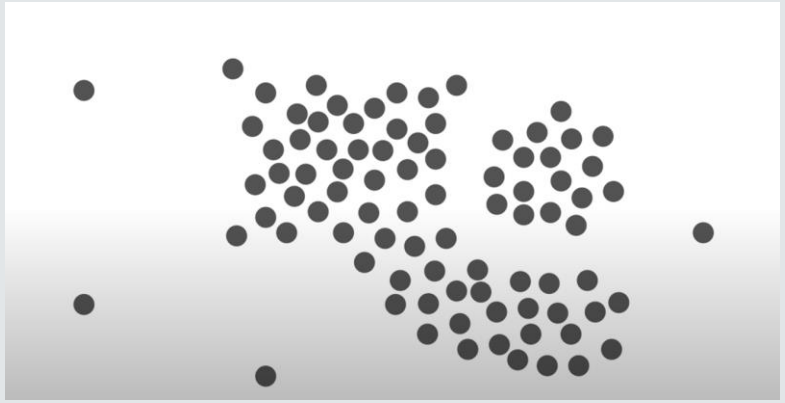
- Define the ϵ and pts_{min}
- Identify all of the points that are core, border and noise points.

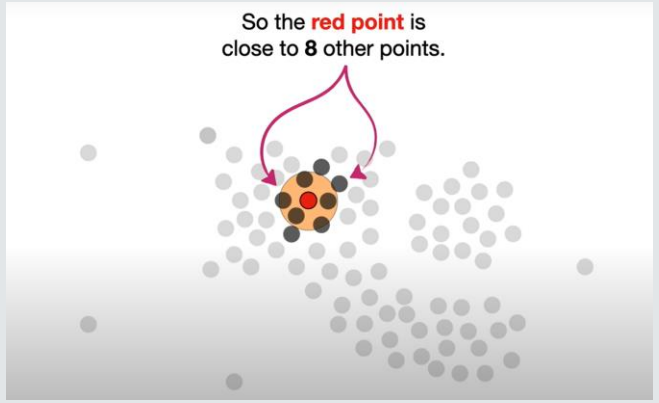
  Core Point: Number of points in the ϵ-neighborhood is $\geq pts_{min}$

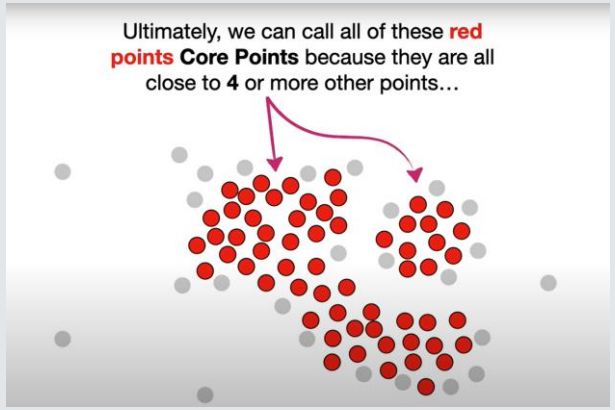
  Border Point: Number of points in the ϵ-neighborhood is $< pts_{min}$ and it is within the ϵ- neighborhood of the core point

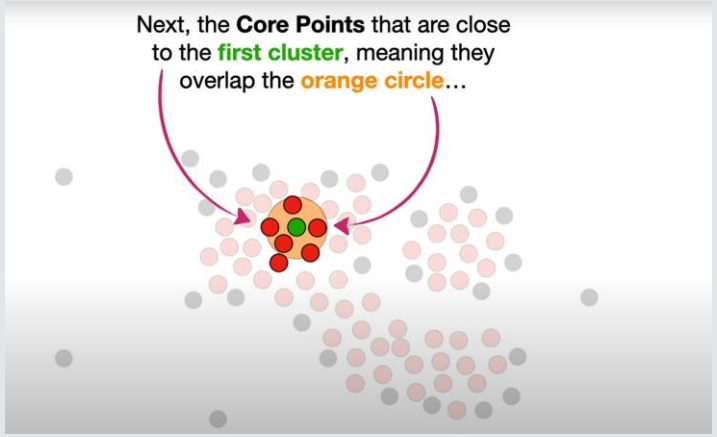
  Noise Point: Not within the ϵ - neighborhood of the core point and the number of points is $< pts_{min}$
- Choose a core point randomly and assign it to a cluster. Then all of the core points close to this core point will be assigned to the cluster. This cluster will keep expanding until all of the core points are added. Then if there are border points to the core points these will be added but it need to be highlighted that if a point is within ϵ- neighborhood of a border point and not a core point then it will not be added to the cluster. This growing mechanism allow the arbitrary shape.
- This will be repeated with other core points which will form other clusters.
- Then points which are not core/border points will be classified as noise points.

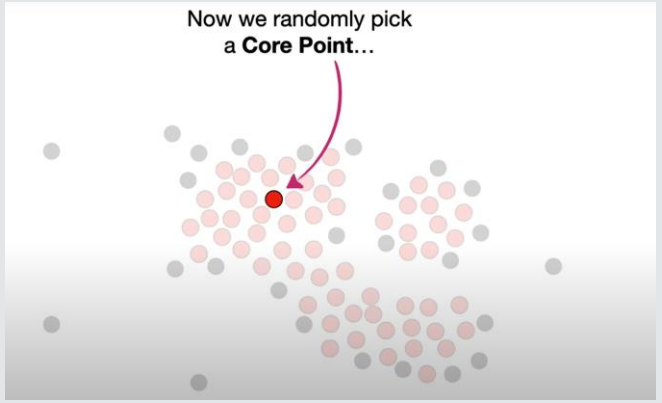
DBSCAN Algorithm

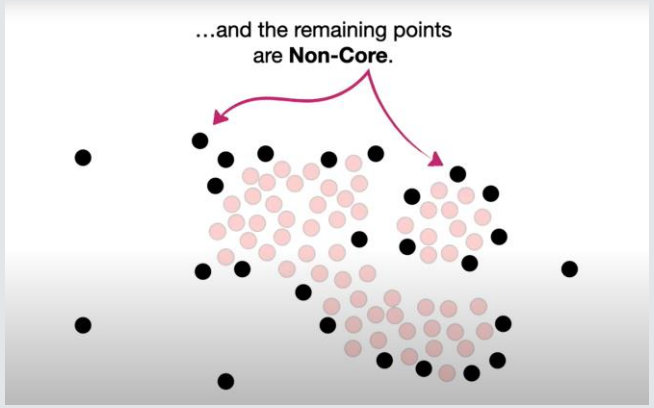






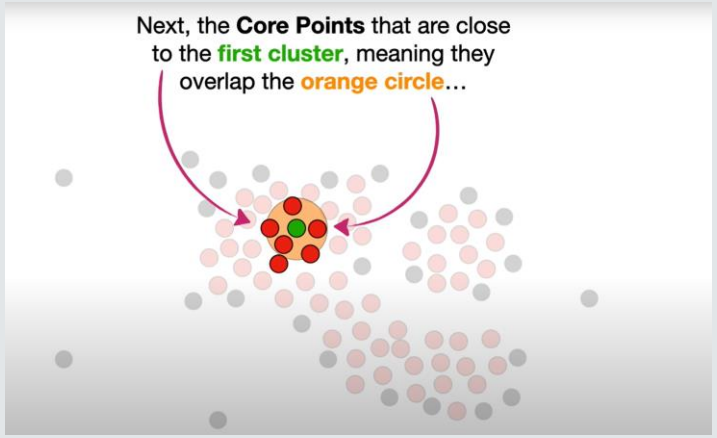


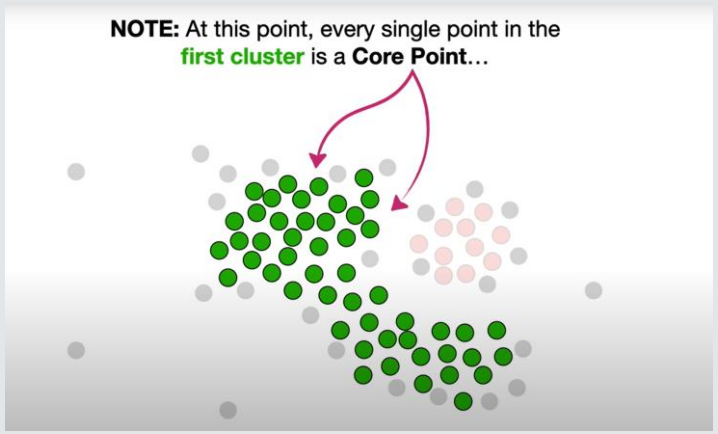


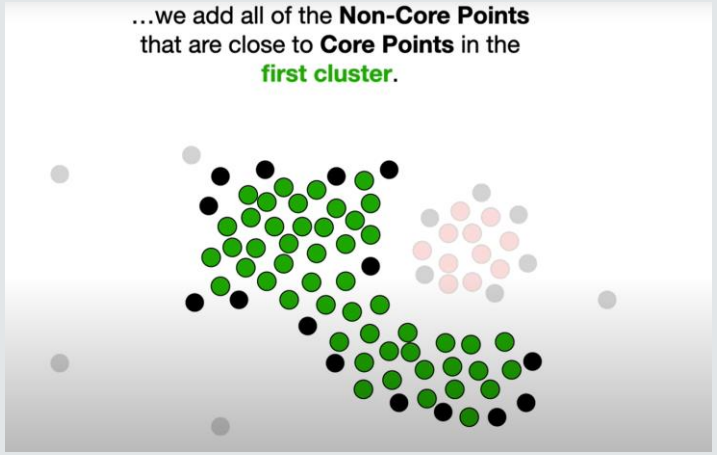


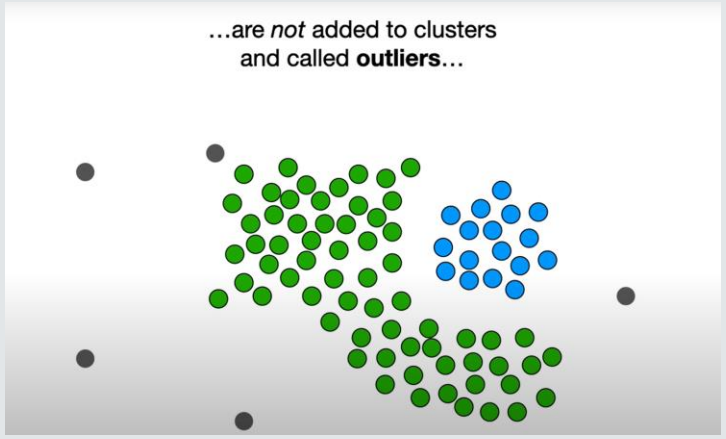
These pictures are taken from the following link https://www.youtube.com/watch?v=RDZUdRSDOok

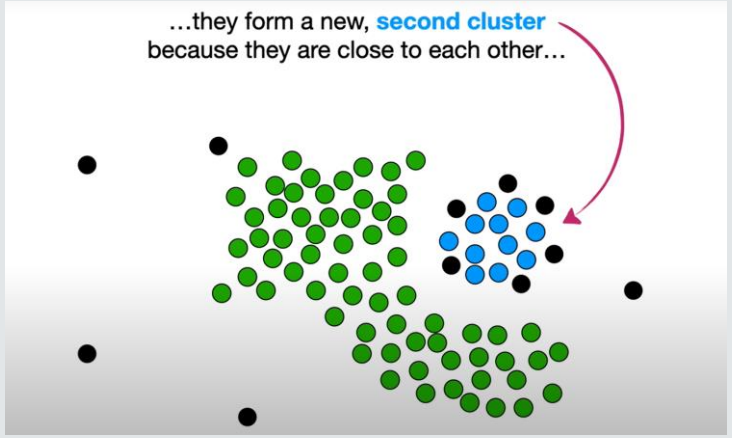
DBSCAN Algorithm

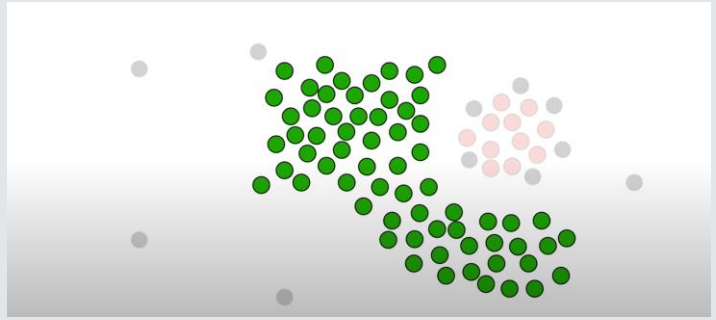












These pictures are taken from the following link https://www.youtube.com/watch?v=RDZUdRSDOok

DBSCAN Algorithm

Point	X	Y
A	1	2
B	2	2
C	2	3
D	8	7
E	8	8
F	25	80

DBSCAN Algorithm

	A	B	C	D	E	F
A	0	1	1.41	8.60	9.21	81.60
B	1	0	1	7.81	8.48	81.32
C	1.41	1	0	7.21	7.81	80.36
D	8.60	7.81	7.21	0	1	74.95
E	9.21	8.48	7.81	1	0	73.97
F	81.60	81.32	80.36	74.95	73.97	0

DBSCAN Algorithm ($\epsilon = 2, pts_{min} = 3$)

	A	B	C	D	E	F
A **(Noise)**	0	1	1.41	8.60	9.21	81.60
B **(Noise)**	1	0	1	7.81	8.48	81.32
C **(Noise)**	1.41	1	0	7.21	7.81	80.36
D **(Noise)**	8.60	7.81	7.21	0	1	74.95
E **(Noise)**	9.21	8.48	7.81	1	0	73.97
F **(Noise)**	81.60	81.32	80.36	74.95	73.97	0

DBSCAN Algorithm ($\epsilon = 2, pts_{min} = 2$)

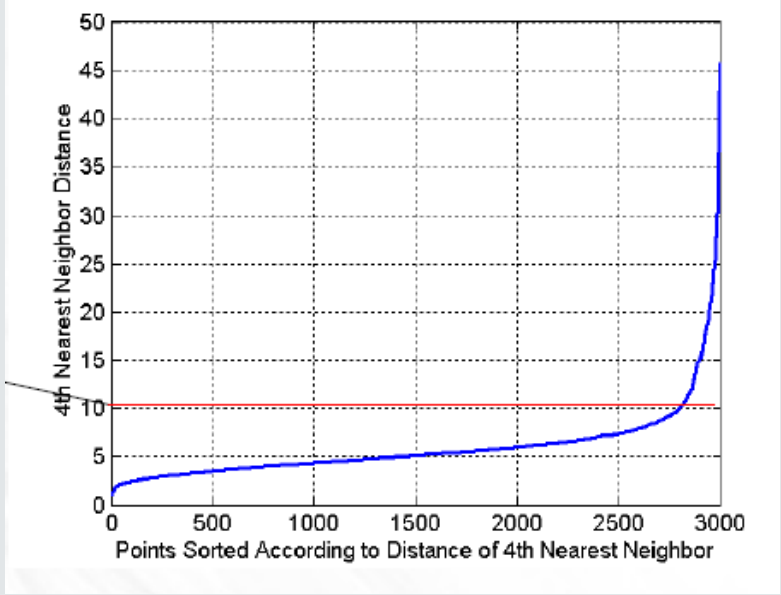
	A	B	C	D	E	F
A **(Core)**	0	1	1.41	8.60	9.21	81.60
B **(Core)**	1	0	1	7.81	8.48	81.32
C **(Core)**	1.41	1	0	7.21	7.81	80.36
D **(Noise)**	8.60	7.81	7.21	0	1	74.95
E **(Noise)**	9.21	8.48	7.81	1	0	73.97
F **(Noise)**	81.60	81.32	80.36	74.95	73.97	0

DBSCAN Algorithm ($\epsilon = 8, pts_{min} = 3$)

	A	B	C	D	E	F
A **(Border)**	0	1	1.41	8.60	9.21	81.60
B **(Core)**	1	0	1	7.81	8.48	81.32
C **(Core)**	1.41	1	0	7.21	7.81	80.36
D **(Core)**	8.60	7.81	7.21	0	1	74.95
E **(Border)**	9.21	8.48	7.81	1	0	73.97
F **(Noise)**	81.60	81.32	80.36	74.95	73.97	0

DBSCAN Algorithm

- Idea is that for points in a cluster, their kth nearest neighbors are at roughly the same distance
- Noise points have the kth nearest neighbor at farther distance
- So, plot sorted distance of every point to its kth nearest neighbor (e.g., k=4)



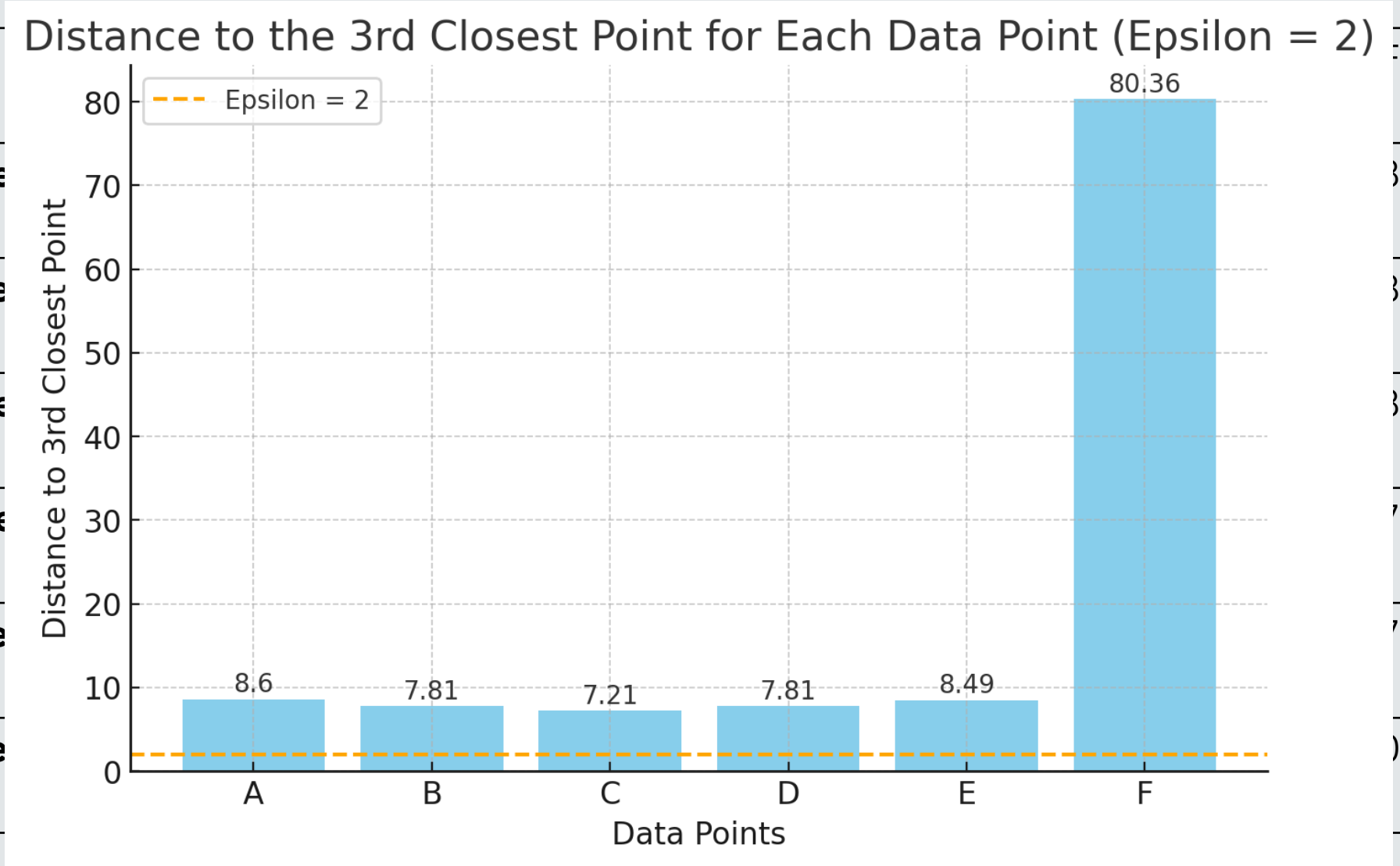
https://csc.csudh.edu/btang/seminar/slides/DBSCAN.pdf

DBSCAN Algorithm ($\epsilon = 2, pts_{min} = 3$)

	A	B	C	D	E	F
A **(Noise)**	0	1	1.41	8.60	9.21	81.60
B **(Noise)**	1	0	1	7.81	8.48	81.32
C **(Noise)**	1.41	1	0	7.21	7.81	80.36
D **(Noise)**	8.60	7.81	7.21	0	1	74.95
E **(Noise)**	9.21	8.48	7.81	1	0	73.97
F **(Noise)**	81.60	81.32	80.36	74.95	73.97	0

DBSCAN Algorithm ($\epsilon = 2, pts_{min} = 3$)

A **(Noise**	31.60
B **(Noise**	31.32
C **(Noise**	80.36
D **(Noise**	74.95
E **(Noise**	73.97
F **(Noise**	0

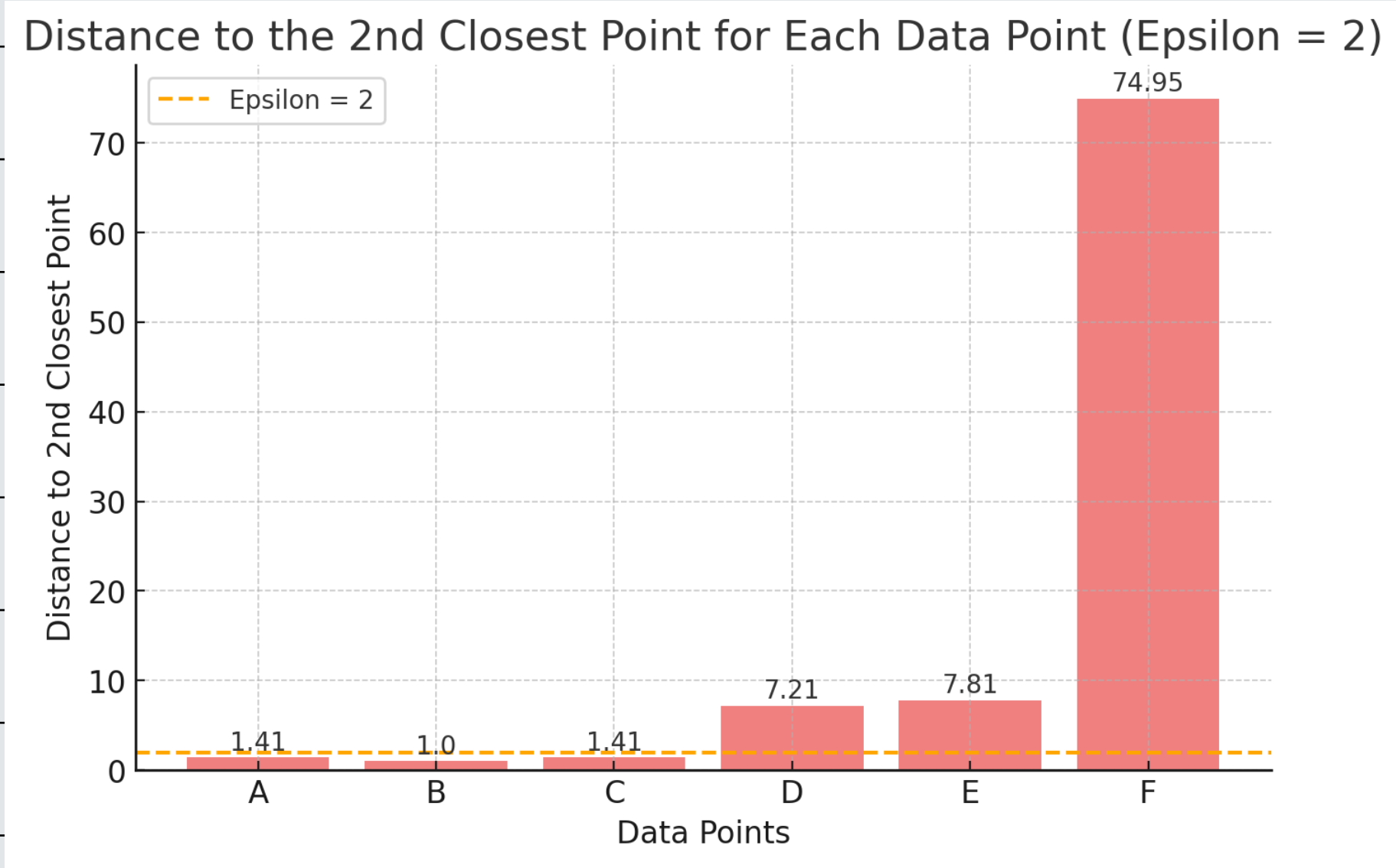


DBSCAN Algorithm ($\epsilon = 2, pts_{min} = 2$)

	A	B	C	D	E	F
A **(Core)**	0	1	1.41	8.60	9.21	81.60
B **(Core)**	1	0	1	7.81	8.48	81.32
C **(Core)**	1.41	1	0	7.21	7.81	80.36
D **(Noise)**	8.60	7.81	7.21	0	1	74.95
E **(Noise)**	9.21	8.48	7.81	1	0	73.97
F **(Noise)**	81.60	81.32	80.36	74.95	73.97	0

DBSCAN Algorithm ($\epsilon = 2, pts_{min} = 2$)

A **(Core)**
B **(Core)**
C **(Core)**
D **(Noise)**
E **(Noise)**
F **(Noise)**

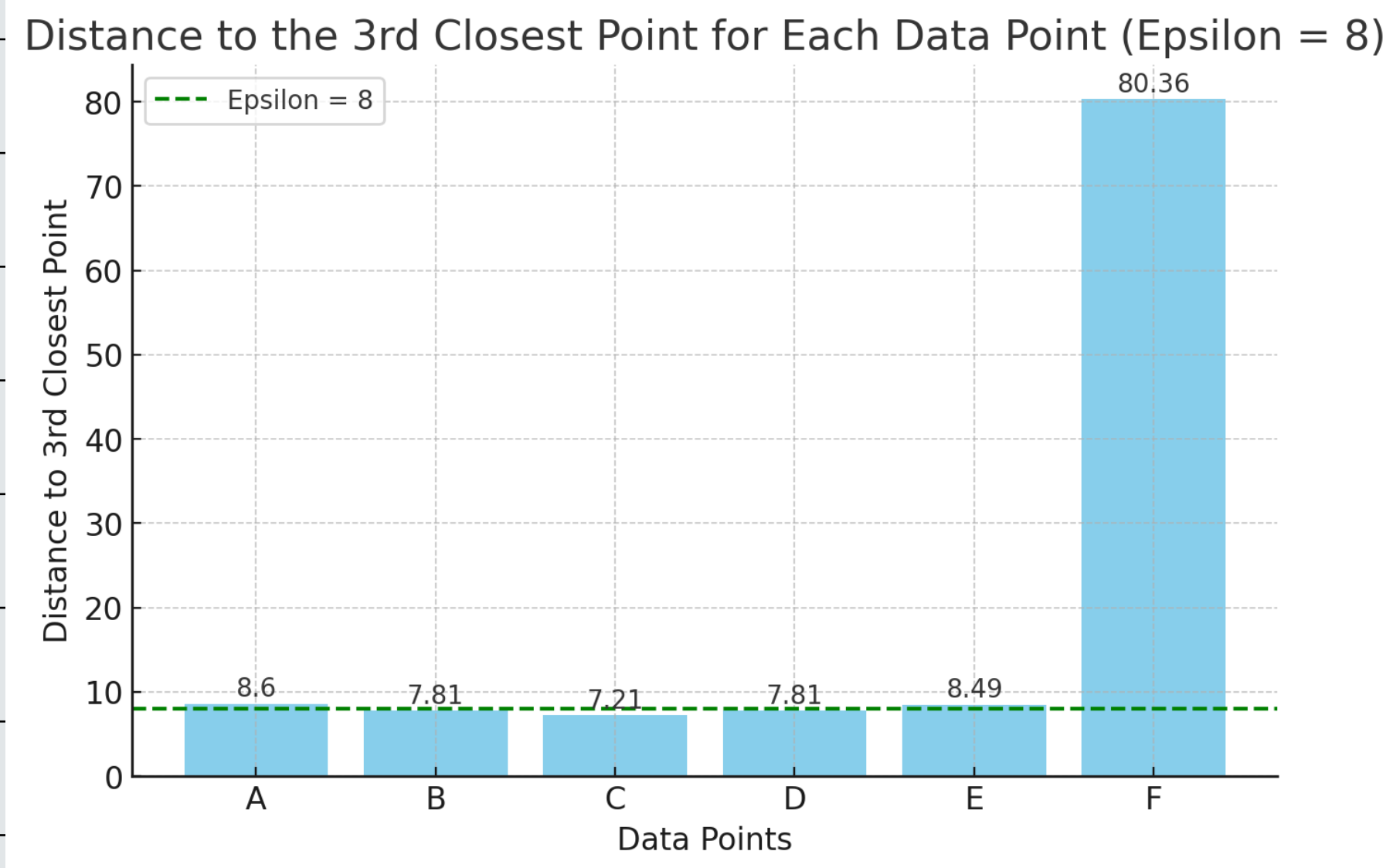


DBSCAN Algorithm ($\epsilon = 8, pts_{min} = 3$)

	A	B	C	D	E	F
A **(Border)**	0	1	1.41	8.60	9.21	81.60
B **(Core)**	1	0	1	7.81	8.48	81.32
C **(Core)**	1.41	1	0	7.21	7.81	80.36
D **(Core)**	8.60	7.81	7.21	0	1	74.95
E **(Border)**	9.21	8.48	7.81	1	0	73.97
F **(Noise)**	81.60	81.32	80.36	74.95	73.97	0

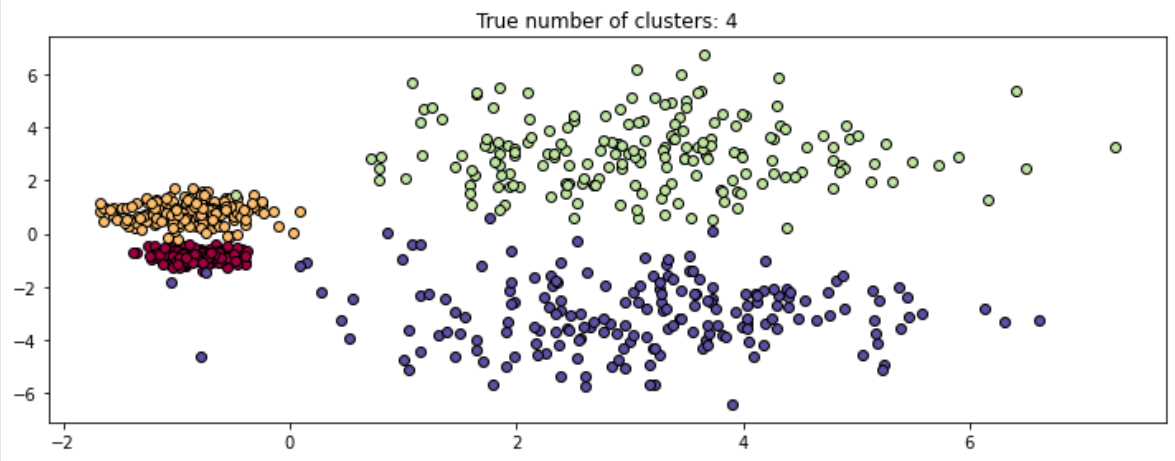
DBSCAN Algorithm ($\epsilon = 8, pts_{min} = 3$)

A **(Border)**	
B **(Core)**	
C **(Core)**	
D **(Core)**	
E **(Border)**	
F **(Noise)**	

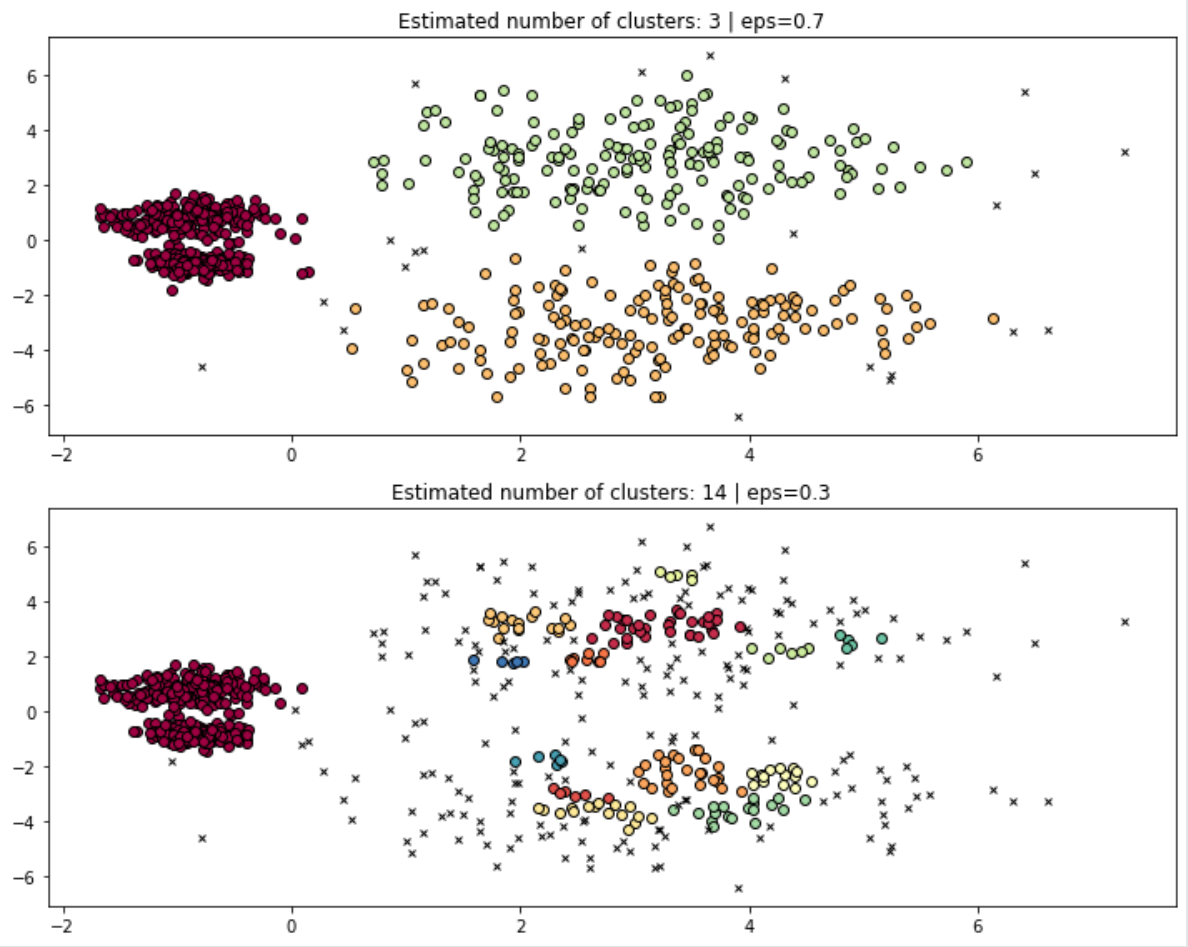


DBSCAN Algorithm

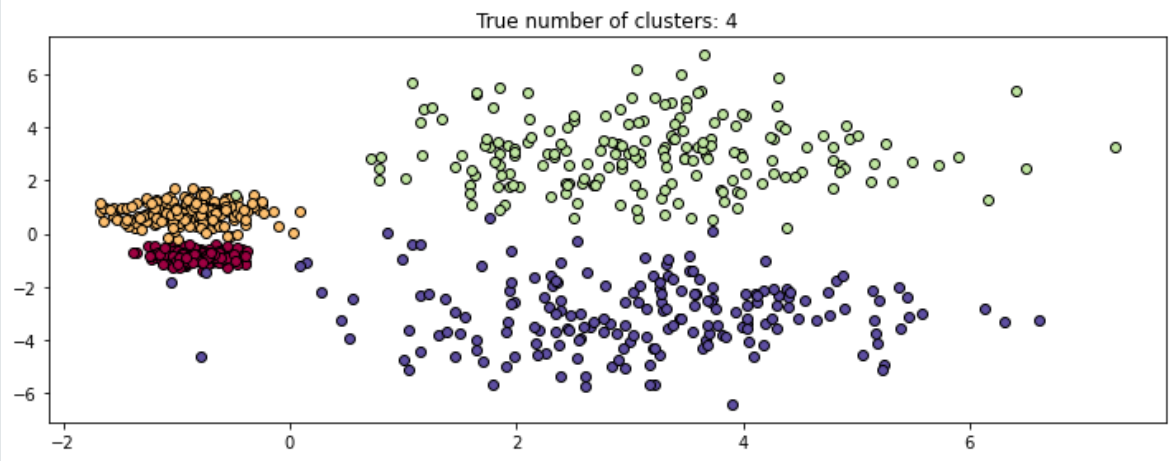
- DBSCAN is a density-based algorithm
- Discover Clusters of arbitrary shape
- Handles noise
- DBSCAN will face issues if we have varying densities and this is due to the fact that will have a fixed epsilon value for all clusters. This means DBSCAN assumes that any potential clusters are homogeneous in density.



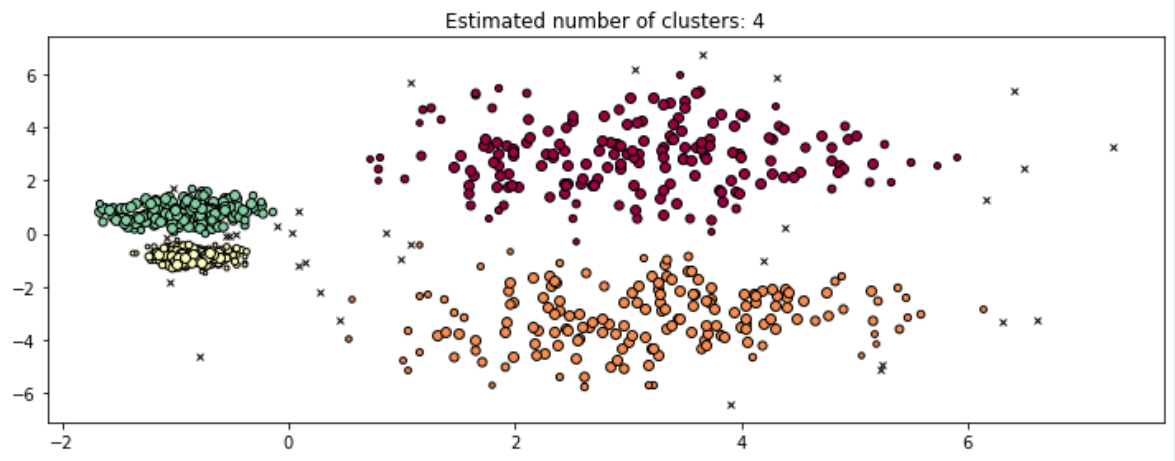
True Data



DBSCAN



True Data



HDBSCAN - multi-scale clustering, which accounts for clusters with varying density

Soft Clustering

- In the previous section, k-means , k-medoids, dbscan clustering algorithm were discussed and these algorithms are hard clustering algorithms. This means each datapoint will be associated with one cluster only. In the case of probabilistic clustering each datapoint will have a probability of how much it is associated with each cluster.
- One way to do that is to simply model the data using a probability distribution. By looking at the clusters at Fig. 4, we can model each cluster using a Gaussian distribution with a different mean and variance (or more precisely covariance). Since, we will have multiple Gaussian's to capture the different clusters we will need to find a way to combine these different Gaussian's distribution. This is exactly the Gaussian mixture model.

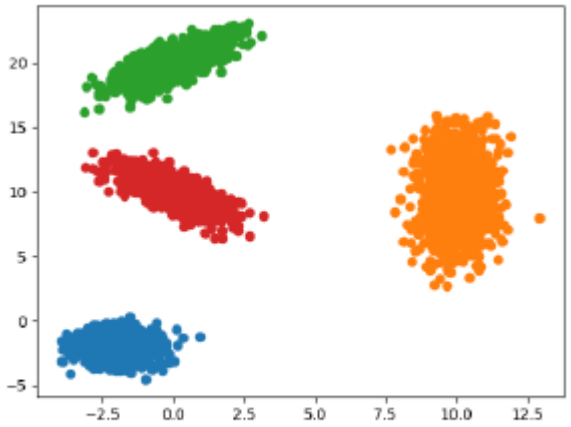


Figure 4: Clusters

Soft Clustering

- Lets assume we have k components in our mixture model.
- We will have k gaussian distributions

$$\mathcal{N}(x; \mu_j, \sigma_j^2 I) \; ; \; j = 1, \cdots, k$$

- Weighting parameter to combine these Gaussian distributions, and this weighting parameter is a multinomial distribution.

$$p_1, \cdots, p_k \; ; \; p_j \geq 0, \sum_j p_j = 1$$

- The parameters we will have in the mixture model is

$$\theta = \{p_1, \cdots, p_k, \mu_1, \cdots, \mu_k, \sigma_1^2, \cdots, \sigma_k^2\}$$

Soft Clustering

- Therefore, the likelihood (the probability that the data is generated using a specific θ) is defined as:

$$
\begin{aligned}
P(S_n|\theta) &= P(x_1, \cdots, x_n|\theta) \\
&= P(x_1|\theta)P(x_2|\theta)\cdots P(x_n|\theta)(\text{Assuming independence between data samples}) \\
&= \prod_{i=1}^{n} P(x_i|\theta) \\
&= \prod_{i=1}^{n}\sum_{j=1}^{k} p_j \mathcal{N}(x_i; \mu_j, \sigma_j^2 I)
\end{aligned}
$$

  We will be interested in maximizing the likelihood because this will result in finding the model with the parameters (θ^*) that will capture the data very precisely.

How to learn these parameters? The tricky part in this problem is that we don't known the parameters θ and also we don't know to which cluster to each point belong. Therefore, to solve this problem we will use the EM algorithm that will solve this problem.

Soft Clustering

This is a simplified version of the problem but it will help in the understanding towards the EM algorithm. If we assume that for each point we know which cluster it belongs to then this problem can be solved directly using maximum likelihood.

$$\delta(j|i) = \begin{cases} 1 & x_i \text{ is assigned to cluster} j \\ 0 & otherwise \end{cases}$$

Then in this case we will calculate the θ parameters by taking the derivative of the log likelihood. The result we will get is the following:

$$\hat{n}_j = \sum_{i=1}^{n} \delta(j|i)$$

$$\hat{p}_j = \frac{\hat{n}_j}{n}$$

$$\hat{\mu}_j = \frac{1}{\hat{n}_j} \sum_{i=1}^{n} \delta(j|i) x^{(i)}$$

$$\hat{\sigma}_j^2 = \frac{1}{\hat{n}_j} \sum_{i=1}^{n} \delta(j|i) ||x^{(i)} - \mu_j||_2^2$$

Soft Clustering

1. Randomly initialize θ and this means $\mu_1, \cdots, \mu_k, \sigma_1^2, \cdots, \sigma_k^2, p_1, \cdots, p_k$.
2. (**E-step**)

$$\begin{aligned} p(C_j|x_i,\theta) = p(j|i) &= \frac{p(i|j)p(j)}{p(i)} \\ &= \frac{p_j\mathcal{N}(x_i;\mu_j,\sigma_j^2)}{\sum_{j=1}^{k} p_j\mathcal{N}(x_i;\mu_j,\sigma_j^2)} \\ &= \frac{p_j\mathcal{N}(x_i;\mu_j,\sigma_j^2)}{p(x_i|\theta)} \end{aligned}$$

3. (**M- Step**)

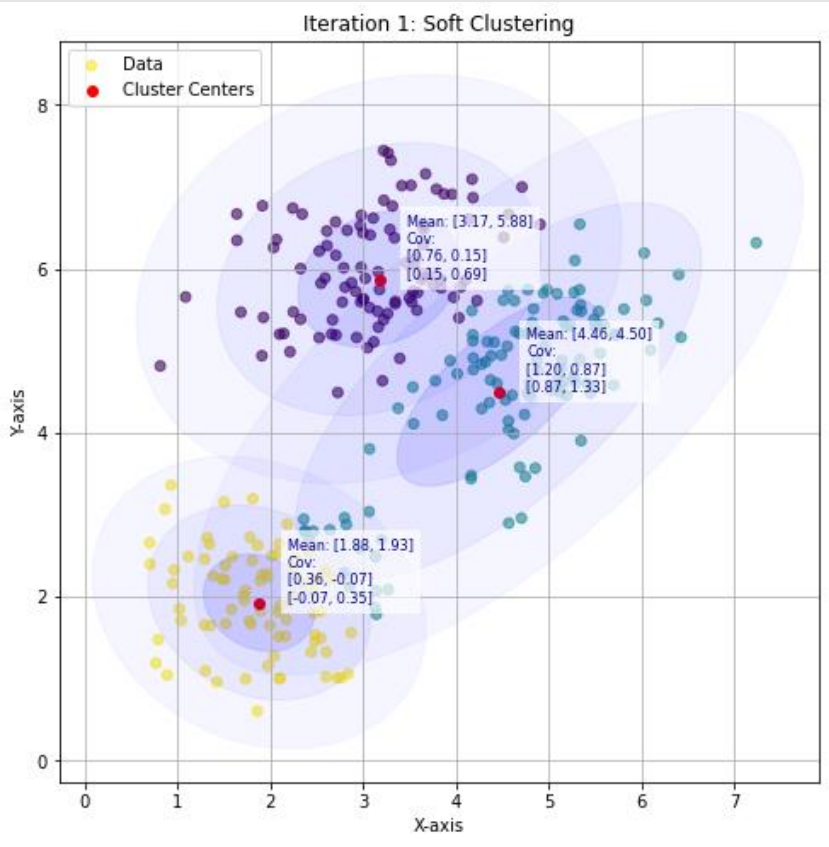
$$\begin{aligned} \hat{n}_j &= \sum_{i=1}^{n} p(j|i) \\ \hat{p}_j &= \frac{\hat{n}_j}{n} \\ \hat{\mu}_j &= \frac{1}{\hat{n}_j}\sum_{i=1}^{n} p(j|i)x^{(i)} \\ \hat{\sigma}_j^2 &= \frac{1}{\hat{n}_j}\sum_{i=1}^{n} p(j|i)||x^{(i)} - \mu_j||_2^2 \end{aligned}$$

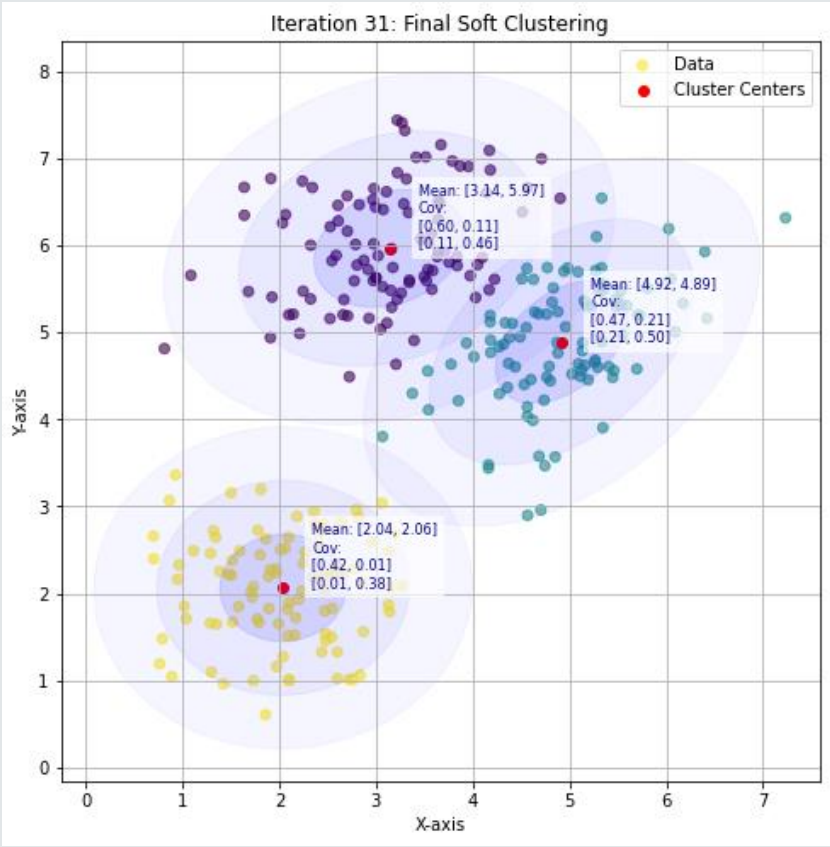
   These values are derived by simply taking the derivative of the log-likelihood with respect to each parameter.

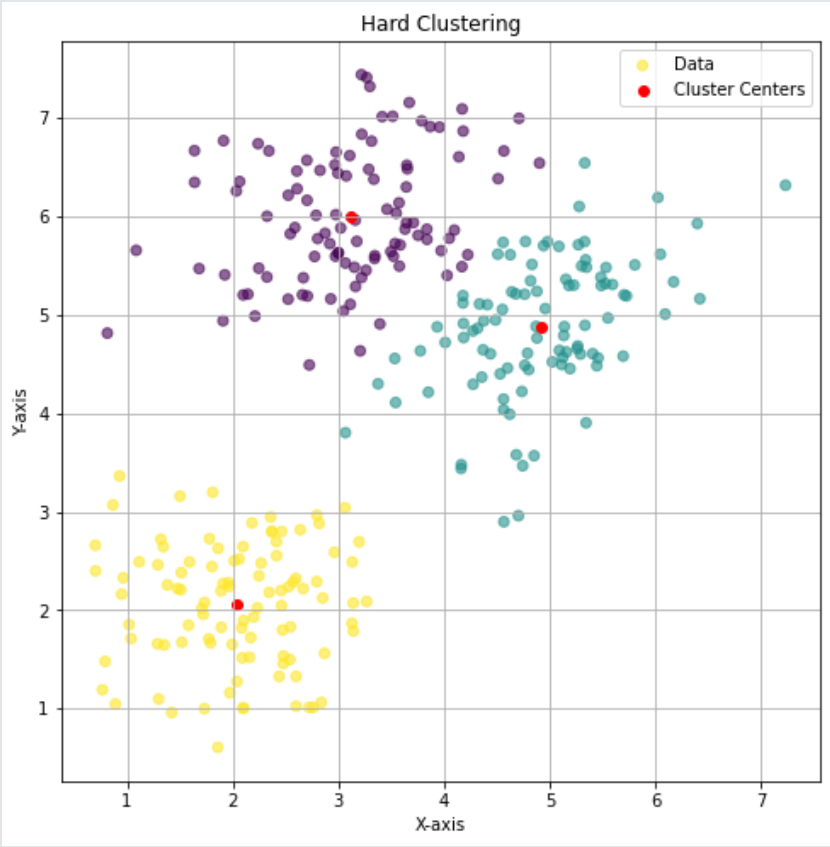
4. These two steps will be repeated until there is no change to log-likelihood ($\log(P(S_n|\theta))$).

The EM algorithm is guaranteed to converge locally. This is one of the major weakness of the EM algorithm.

Soft Clustering







Soft Clustering