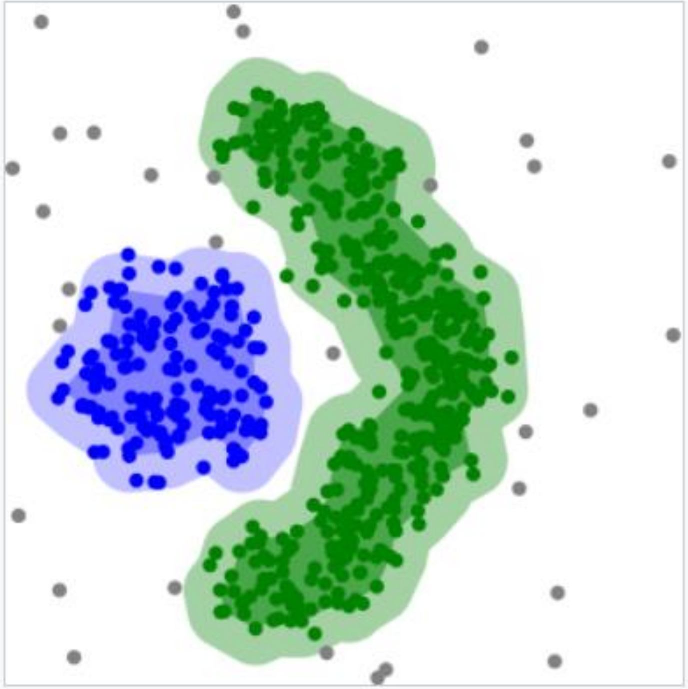
Clustering - DBSCAN

Lin Guosheng
School of Computer Science and Engineering
Nanyang Technological University

DBSCAN

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



DBSCAN can find non-linearly separable clusters. This dataset cannot be adequately clustered with k-means or Gaussian Mixture EM clustering.

DBSCAN

There are two key parameters of DBSCAN:

- epsilon_radius (epsilon or eps):
  This radius defines the epsilon-neighbourhoods. Two points are considered to be neighbours if the distance between them are less than or equal to eps.

- minPts: Minimum number of data points to define a cluster (or a core point).

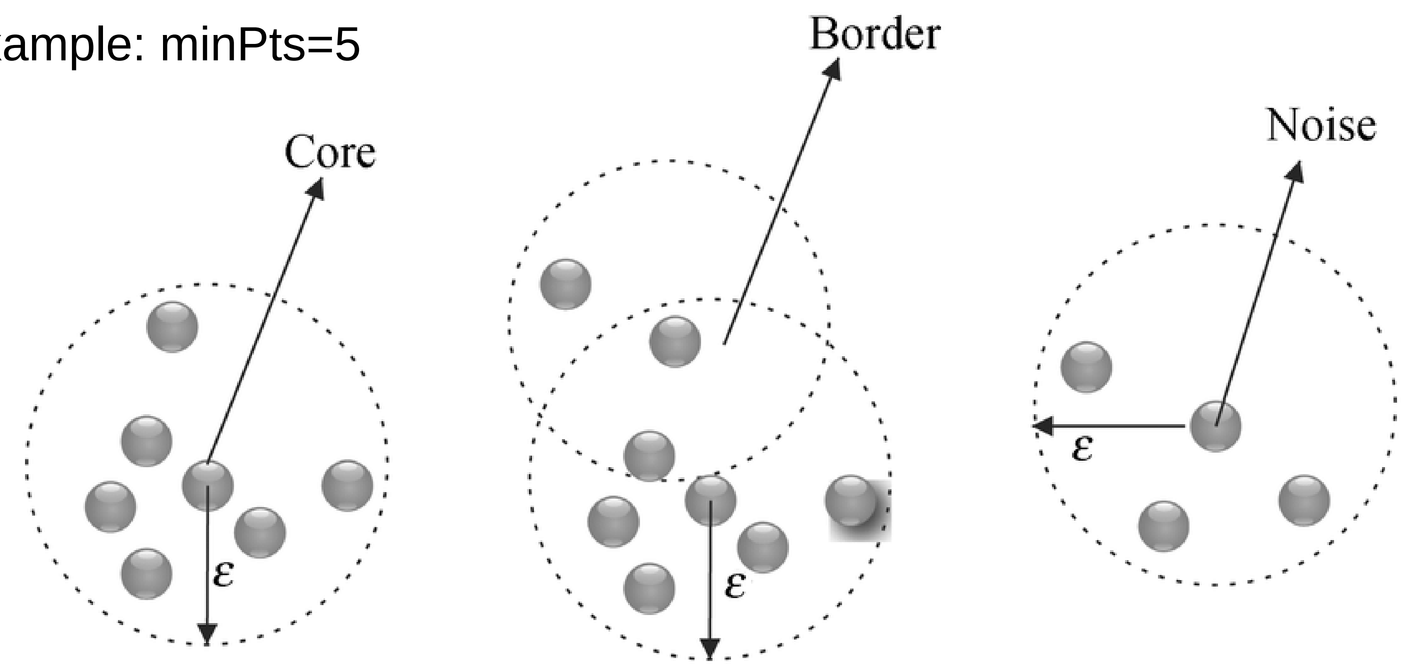
https://towardsdatascience.com/dbscan-clustering-explained-97556a2ad556

- **Core point:** A point is a core point if there are at least minPts number of points (including the point itself) in its surrounding area with radius eps (within the epsilon-neighbourhood).

- **Border point**: A point is a border point if it is not a core point and it is directly reachable from a core point.

- **Outlier** (**noise point**): A point is an outlier if it is not a core point and not reachable from any core points.

Example: minPts=5



https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html

Algorithm

- Step 1. Start a cluster.
  - An unvisited point x is selected at random. If x is a core point, construct a new cluster starting from x using Step 2. Otherwise, the point x is marked as a noise point; mark the point as visited; jump to step 1.
- Step 2.Construct a cluster (cluster expansion)
  - construct the cluster using breadth-first search
  - Mark all points in the cluster as visited.
- Stop until all points are visited.

https://en.wikipedia.org/wiki/DBSCAN

Cluster creation

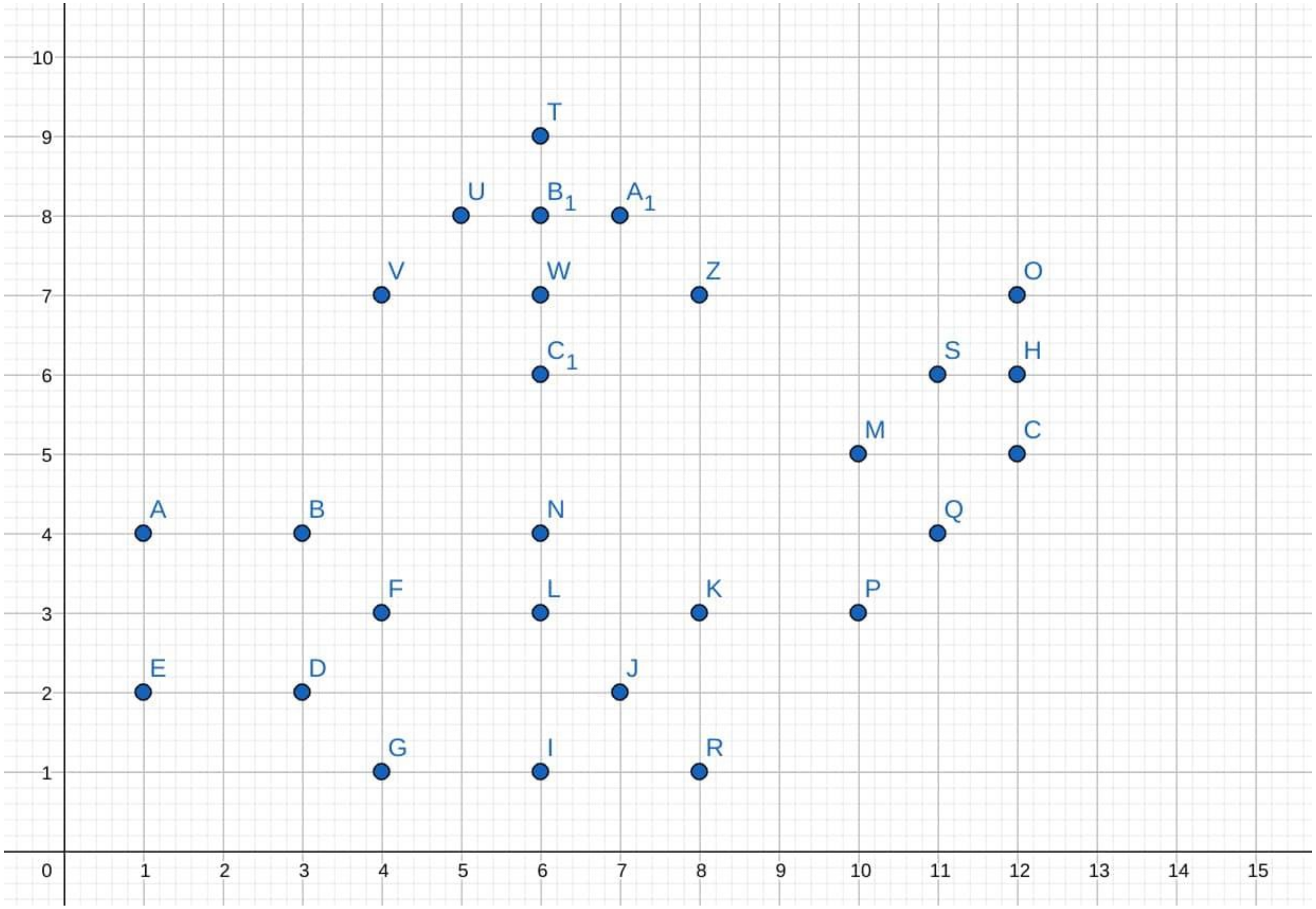
- Create the cluster as building a tree
  - 1) initialization:
    - The starting core point is the root node of the tree
    - Create an empty queue, add the root node into the queue
      - The queue is to implement breadth-first search (layer-wise traversal) of the tree. (walk through all nodes on the same level before moving on to the next level).
      - The queue is to maintain a list of TODO nodes.

Cluster creation

- Create the cluster as building a tree
  - 2) dequeue to get the target node:
    - if the target node is a core point:
      - Identify all neighbouring points (directly reachable nodes).
      - If a neighbouring node is not in the tree
        - Tree update: add it as a child node under the target node.
        - Implementation: add it to the queue for future processing
  - 3) Goto step 2) to process the next node
    - Stop expanding the tree if the queue is empty.
  - All the nodes in the tree are the members of the cluster.

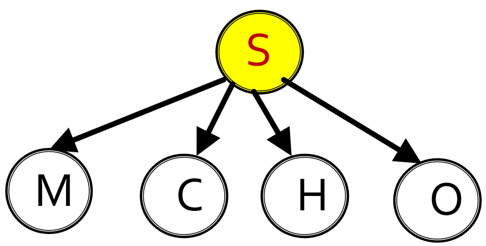
DBSCAN Example

Question: given the data points in the figure and the DBSCAN parameters: **minPts=4, radius eps=1.5**, illustrate the process of finding one cluster starting from point S. If cannot find a cluster starting from S, provide the discussion.



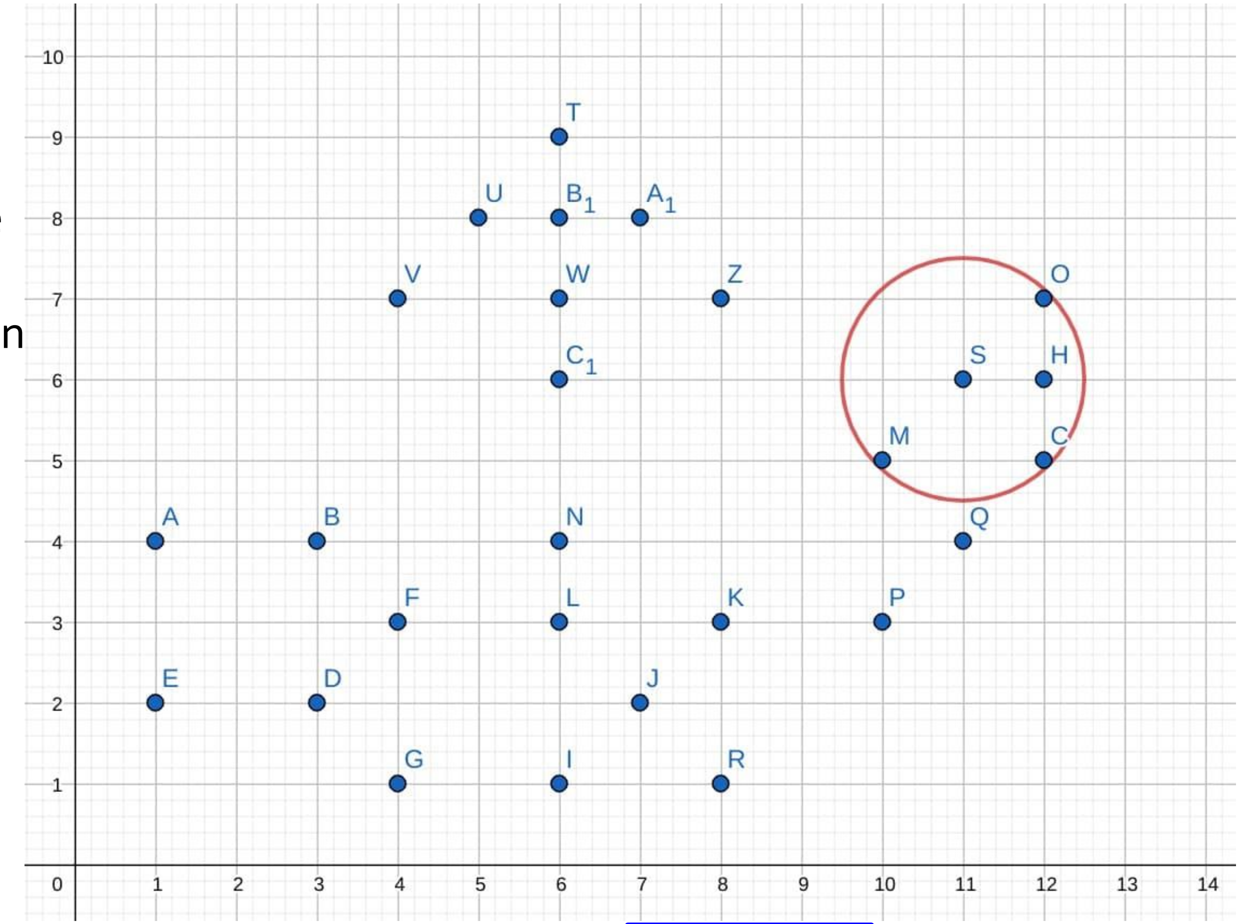
Starting from S

1. check whether the point is a core point or not
2. If it is a core point, find out all neighbors. (directly reachable nodes). Add them to the tree as child nodes if they are not in the tree.



A tree describes a cluster

Core points are highlighted



Queue: S

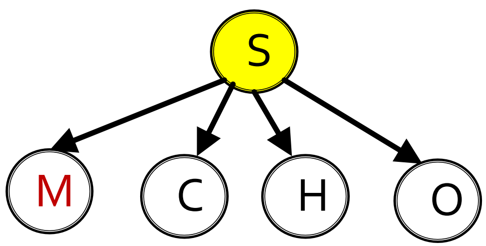
Dequeue and add new nodes

Queue: M C H O

S is a core point,
as the number of neighbors within eps_radius:
5 >= minPts (4) (including the point S itself!)

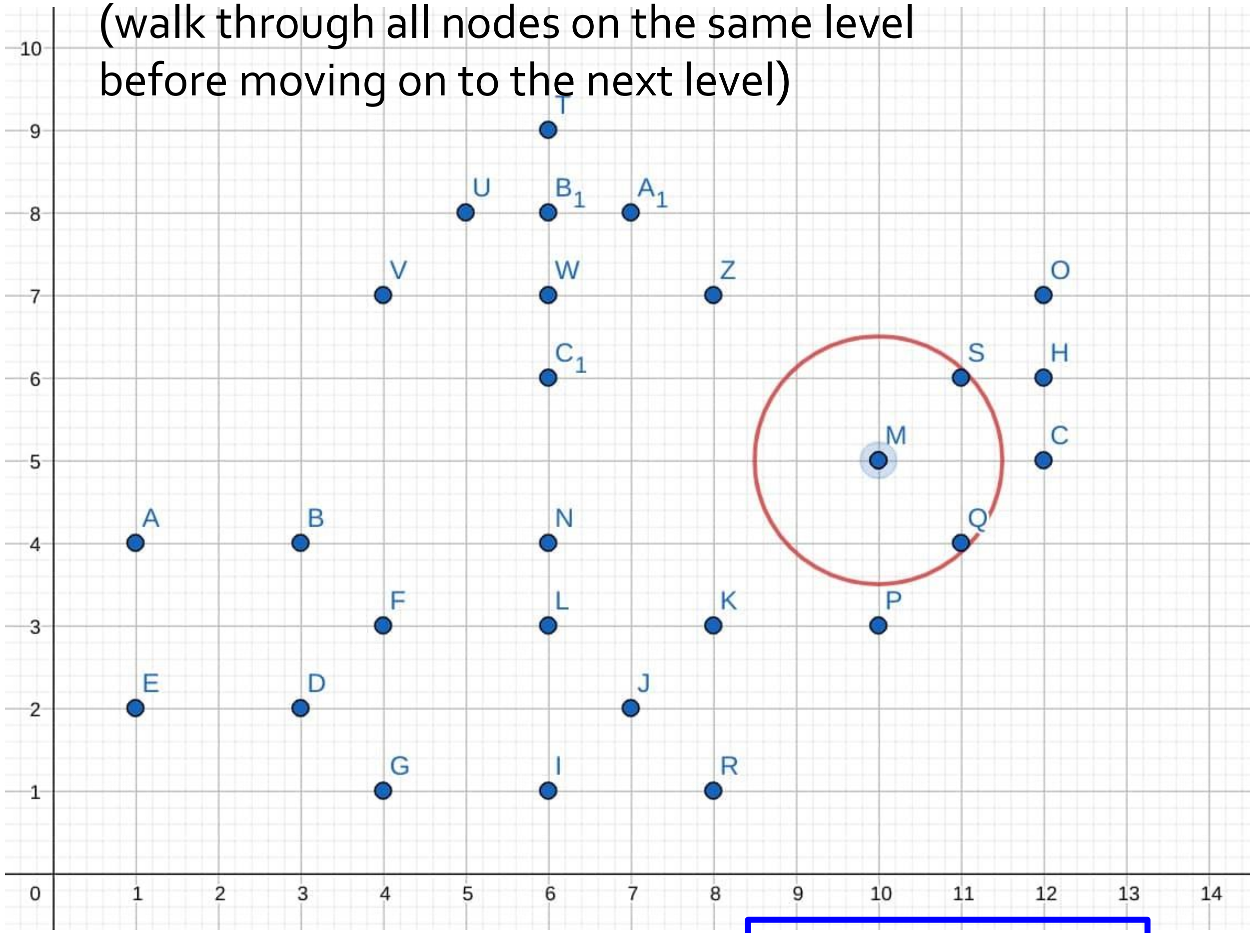
Explore M

1. Determine whether the point is a core point or not



A tree describes a cluster

Breadth-first search
(walk through all nodes on the same level before moving on to the next level)



Queue: M C H O

Dequeue

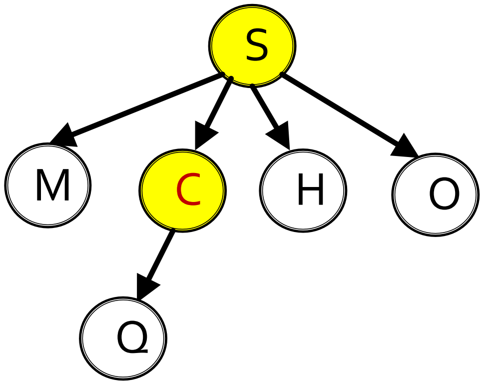
Queue: C H O

M is not a core point,
as the number of neighbors within eps_radius: 3 < minPts (4)

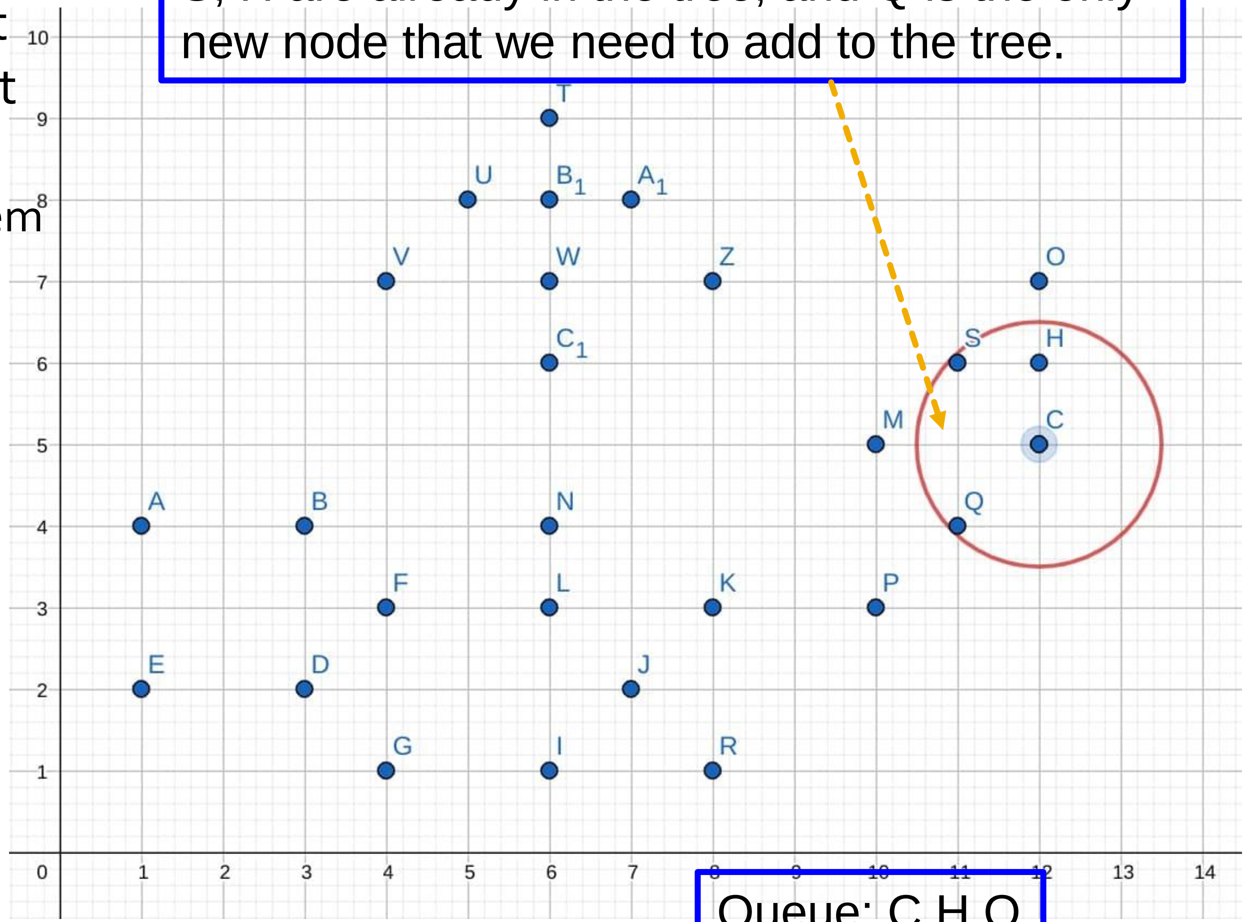
Explore C

1. Determine whether the point is a core point or not
2. If it is a core point, find out all neighbors (directly reachable nodes). Add them to the tree as child nodes if they are not in the tree.

C has 3 neighbours: S, H, Q.
S, H are already in the tree, and Q is the only new node that we need to add to the tree.



A tree describes a cluster



Queue: C H O

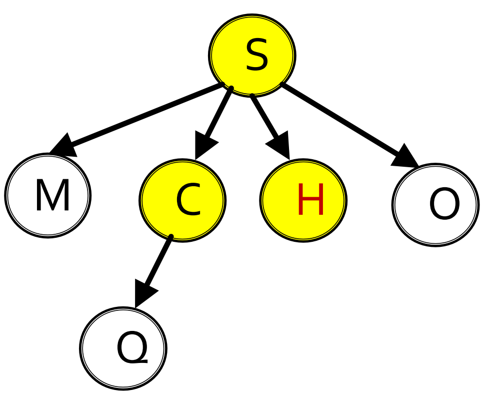
Dequeue and add new nodes

Queue: H O Q

C is a core point, as the number of neighbors within eps_radius: 4 >= minPts (4) (including the point C itself!)

Explore H

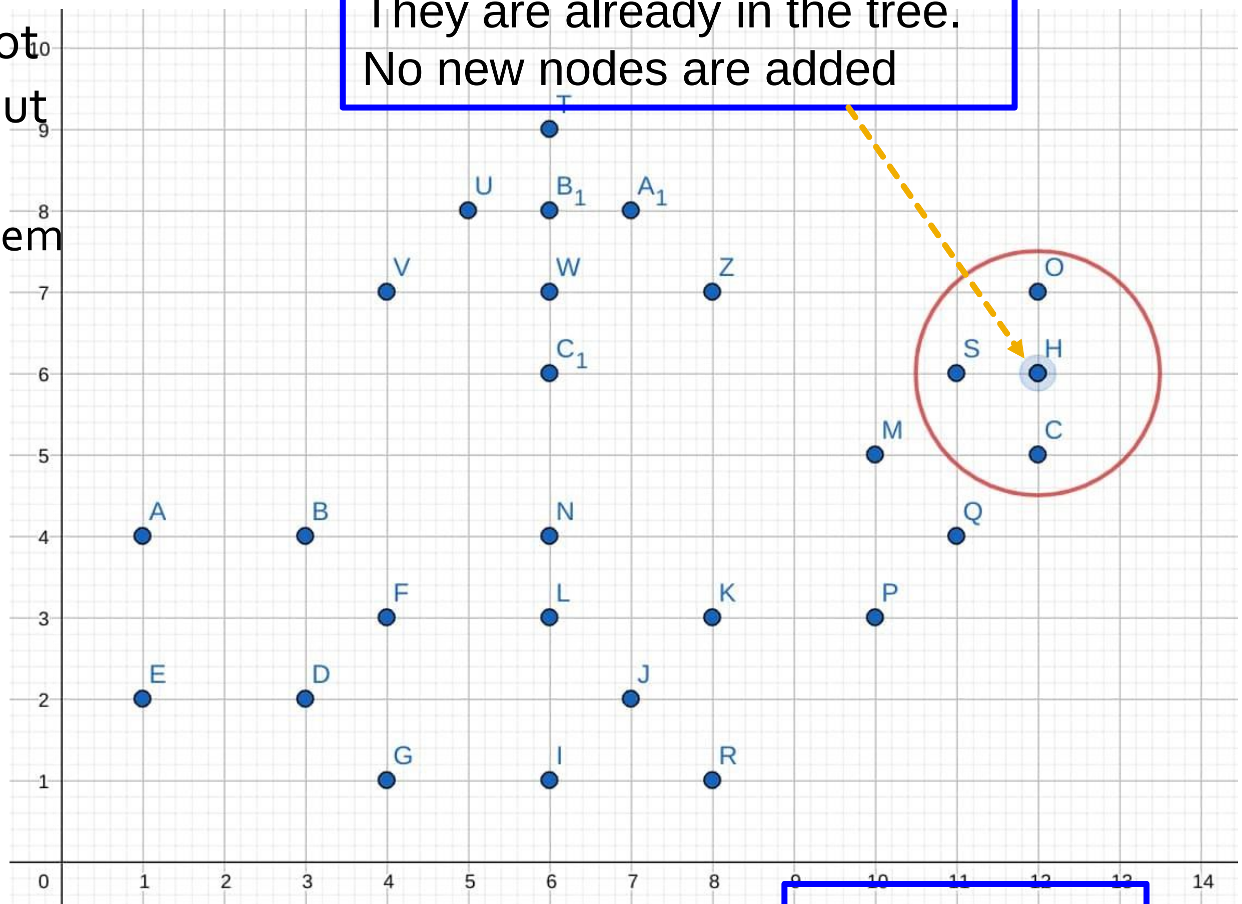
1. Determine whether the point is a core point or not
2. If it is a core point, find out all neighbors (directly reachable nodes). Add them to the tree as child nodes if they are not in the tree.



A tree describes a cluster

H is a core point,
as the number of neighbors within eps_radius: 4 >= minPts (4)

H has 3 neighbours: S, C, O.
They are already in the tree.
No new nodes are added



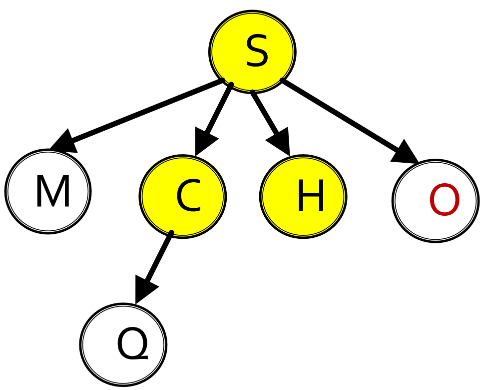
Queue: H O Q

Dequeue

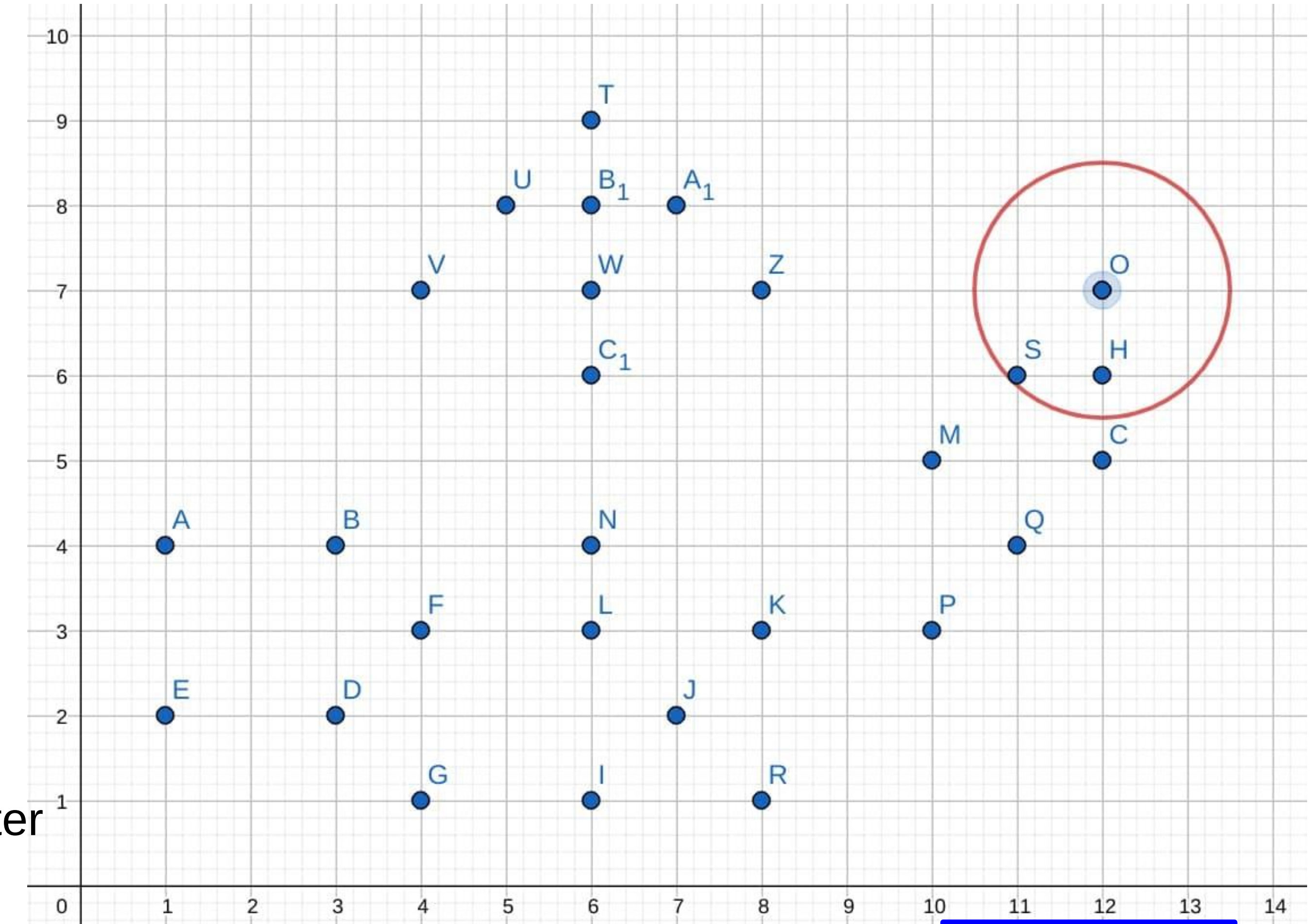
Queue: O Q

Explore O

1. Determine whether the point is a core point or not



A tree describes a cluster



Queue: O Q

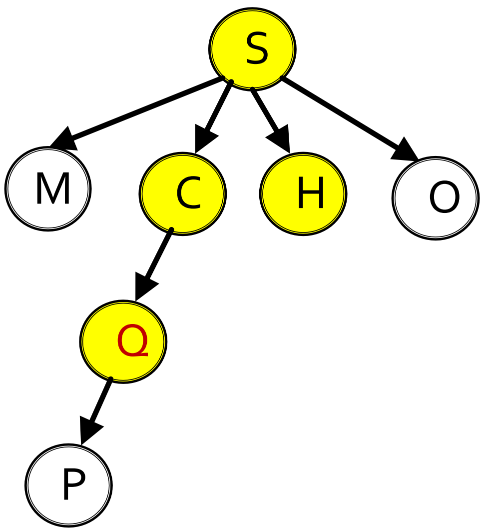
Dequeue

Queue: Q

O is not a core point,
as the number of neighbors within eps_radius: 3 < minPts (4)

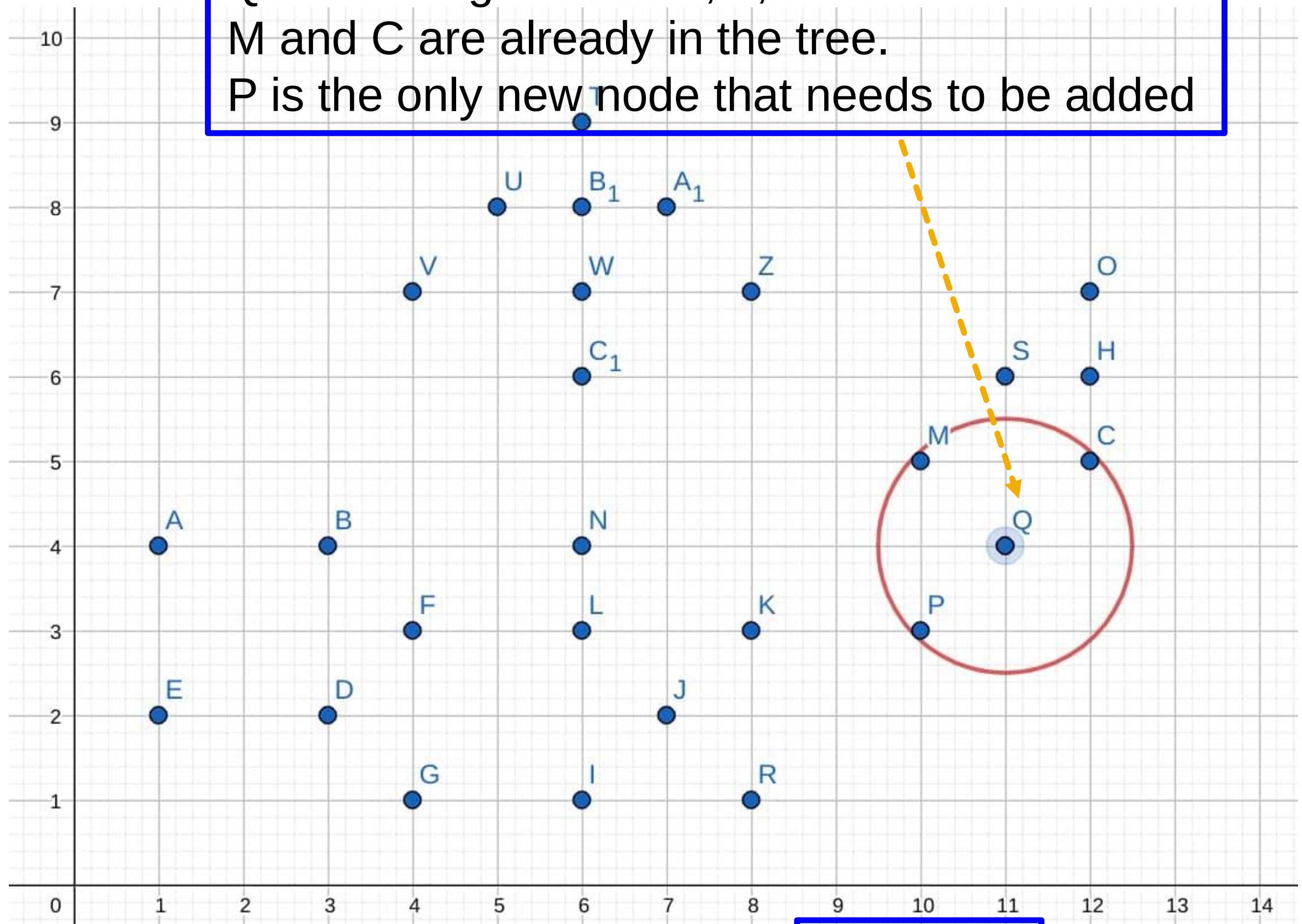
Explore Q

Proceed to the
next layer: Q



A tree describes a cluster

Q has 3 neighbours: M, P, C.
M and C are already in the tree.
P is the only new node that needs to be added



Q is a core point,
as the number of neighbors within
eps_radius: 4 >= minPts (4)

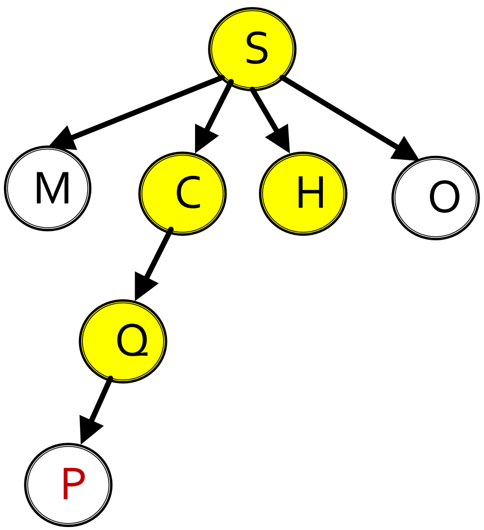
Queue: Q

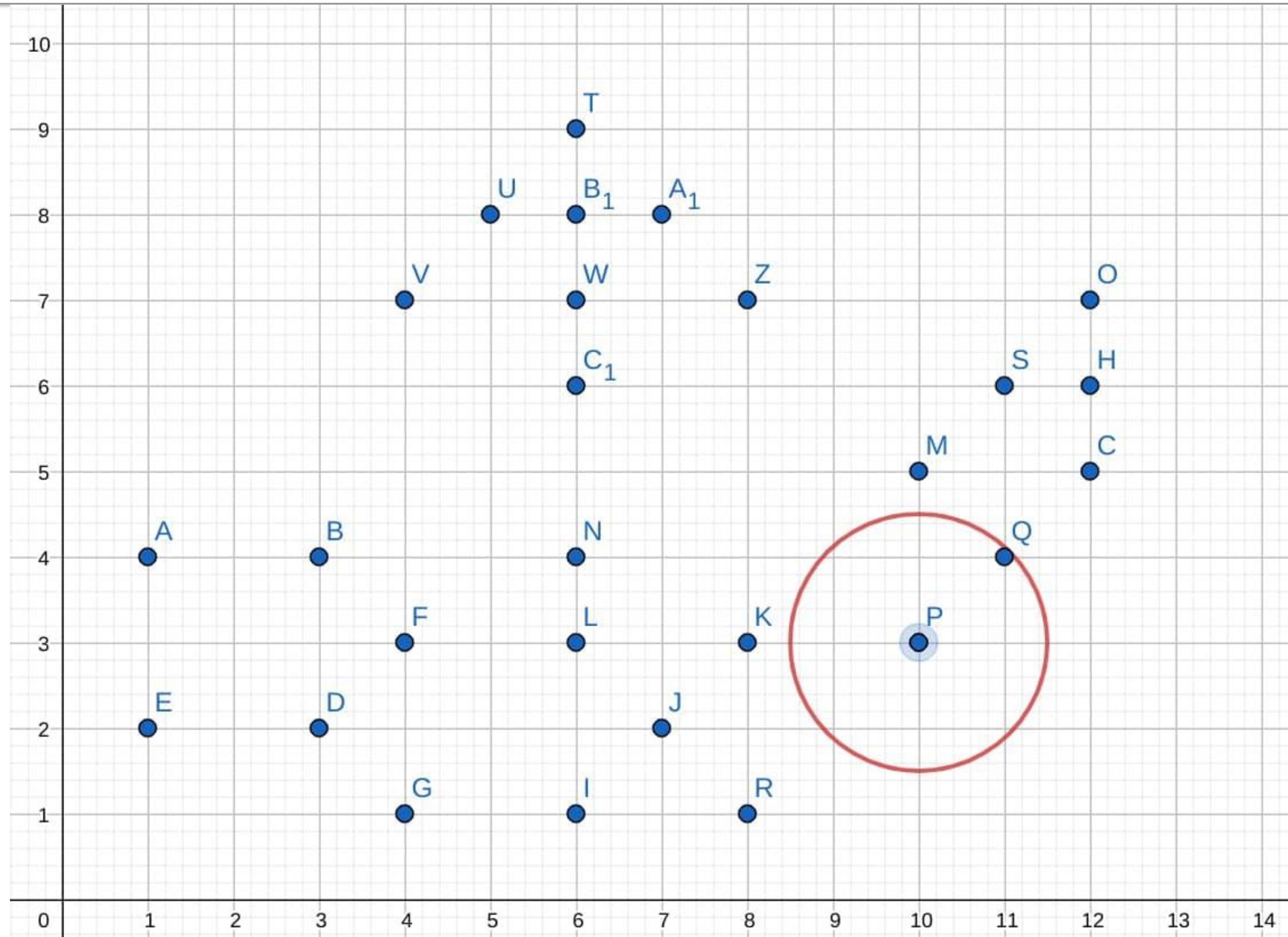
Dequeue and add new nodes

Queue: P

Explore P

Proceed to the next layer: P





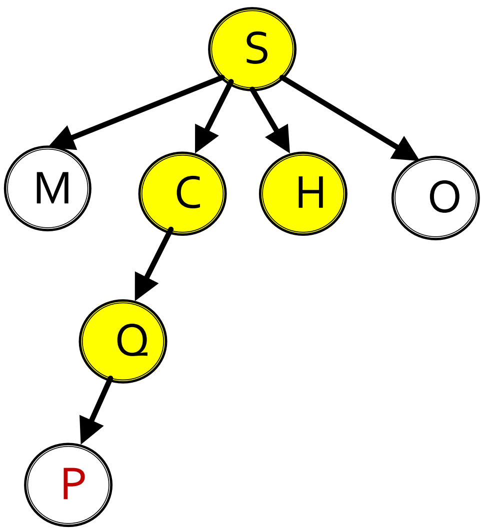
Queue: P

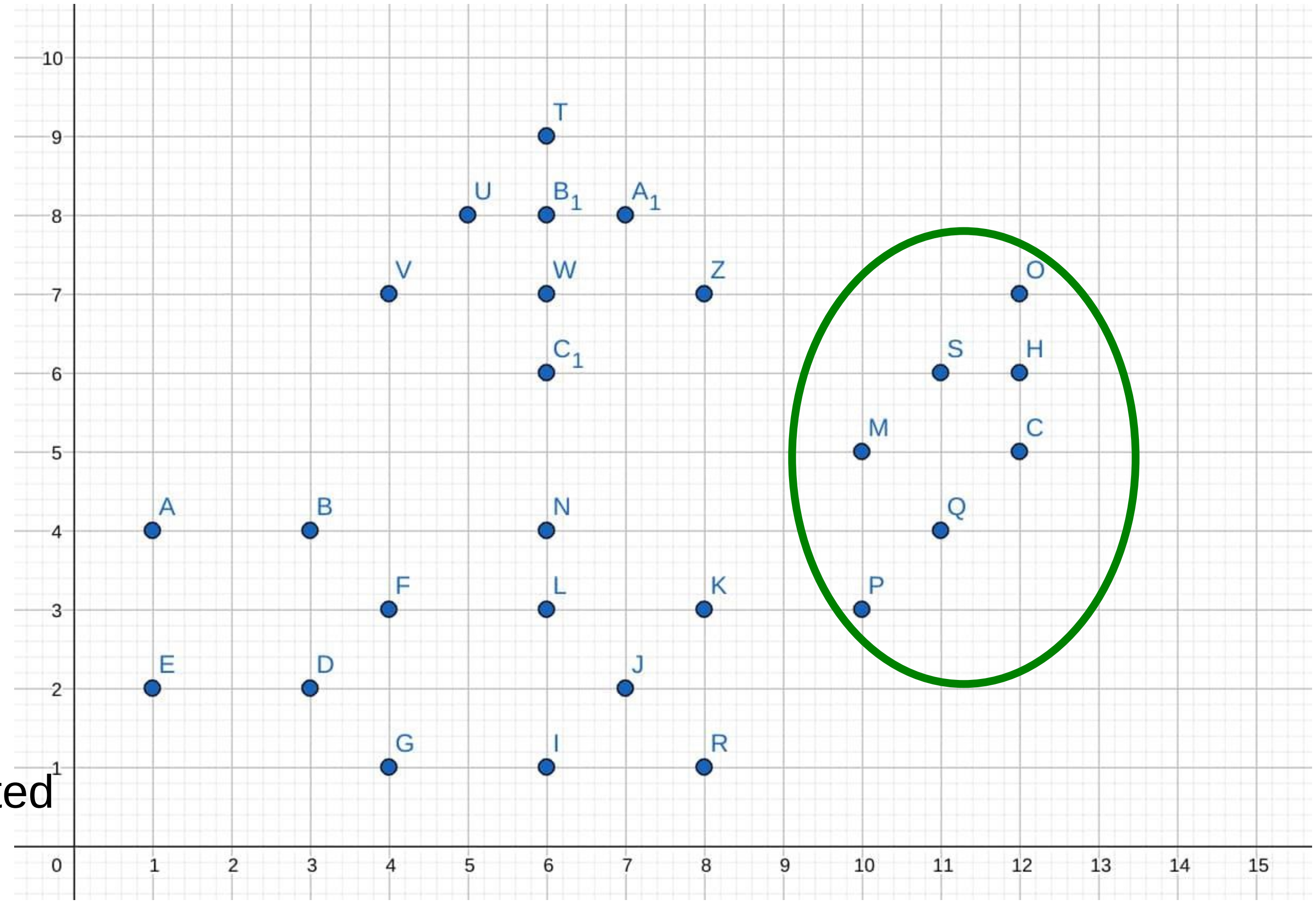
Dequeue

Queue: <empty>

P is not a core point,
as the number of neighbors within eps_radius: 2 < minPts (4)

Final result

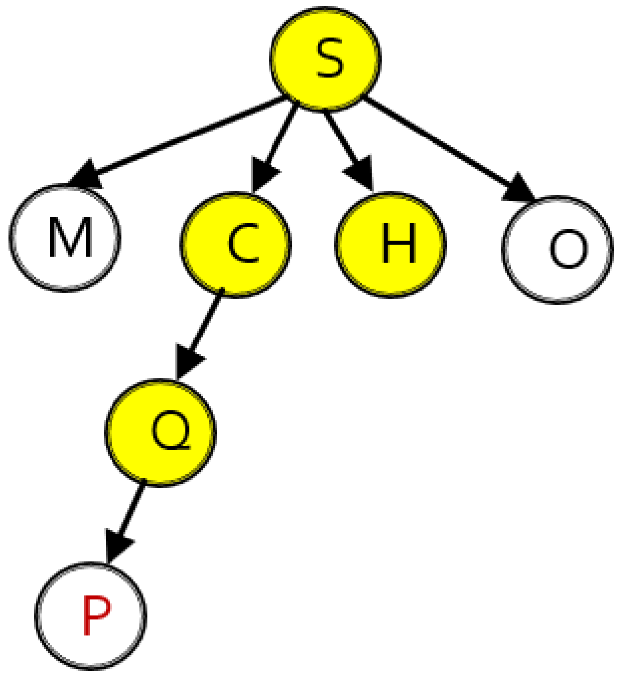




Core points are highlighted

Starting from data point S, we can construct the above cluster

Discussion:

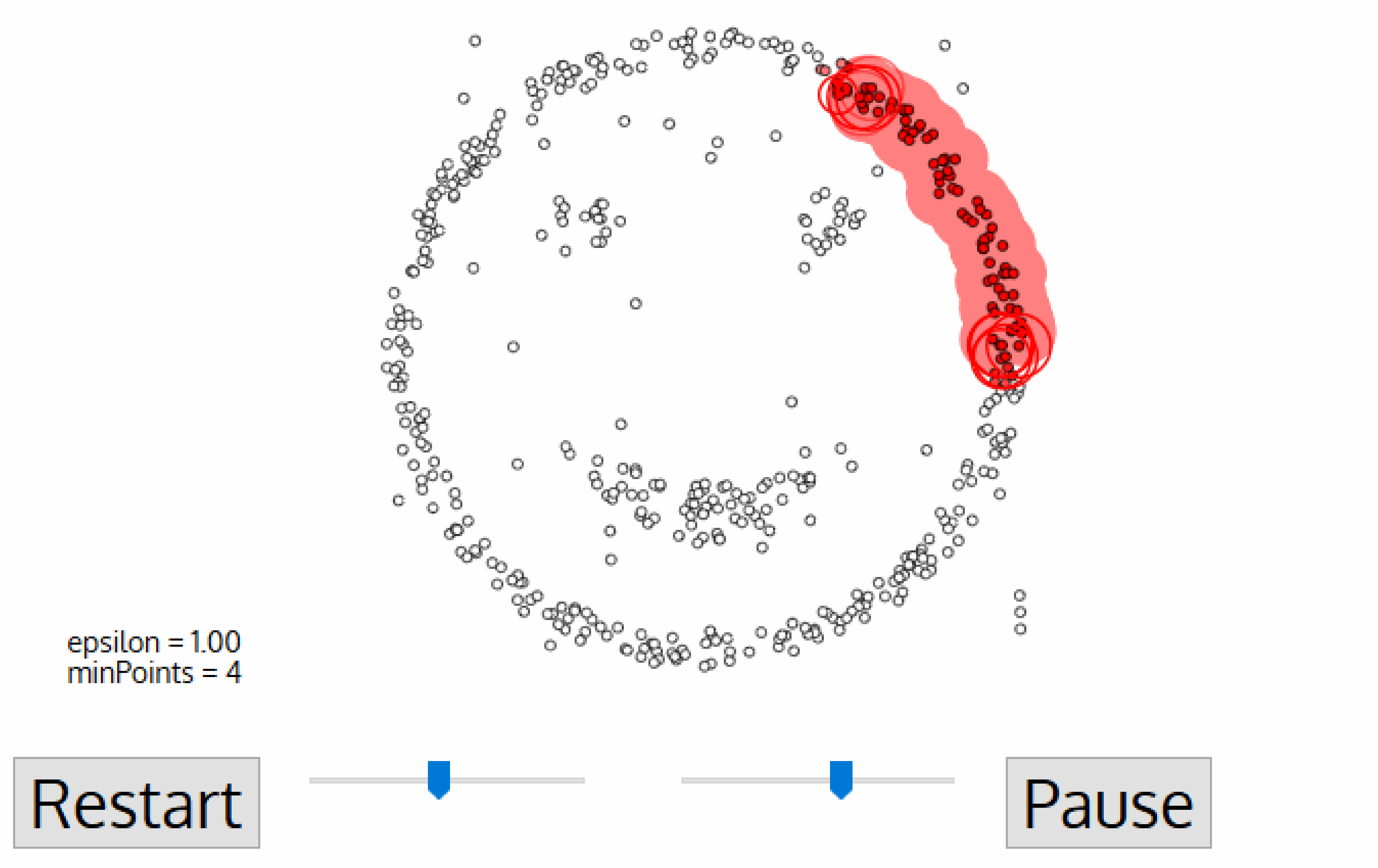


Use a tree to describe the process of cluster growth in DBSCAN
A parent node: represents a core point
A child node: a neighbour of the parent node

1. User breadth-first search (layer-wise traversal) to expand the tree:
walk through all nodes on the same level before moving on to the next level

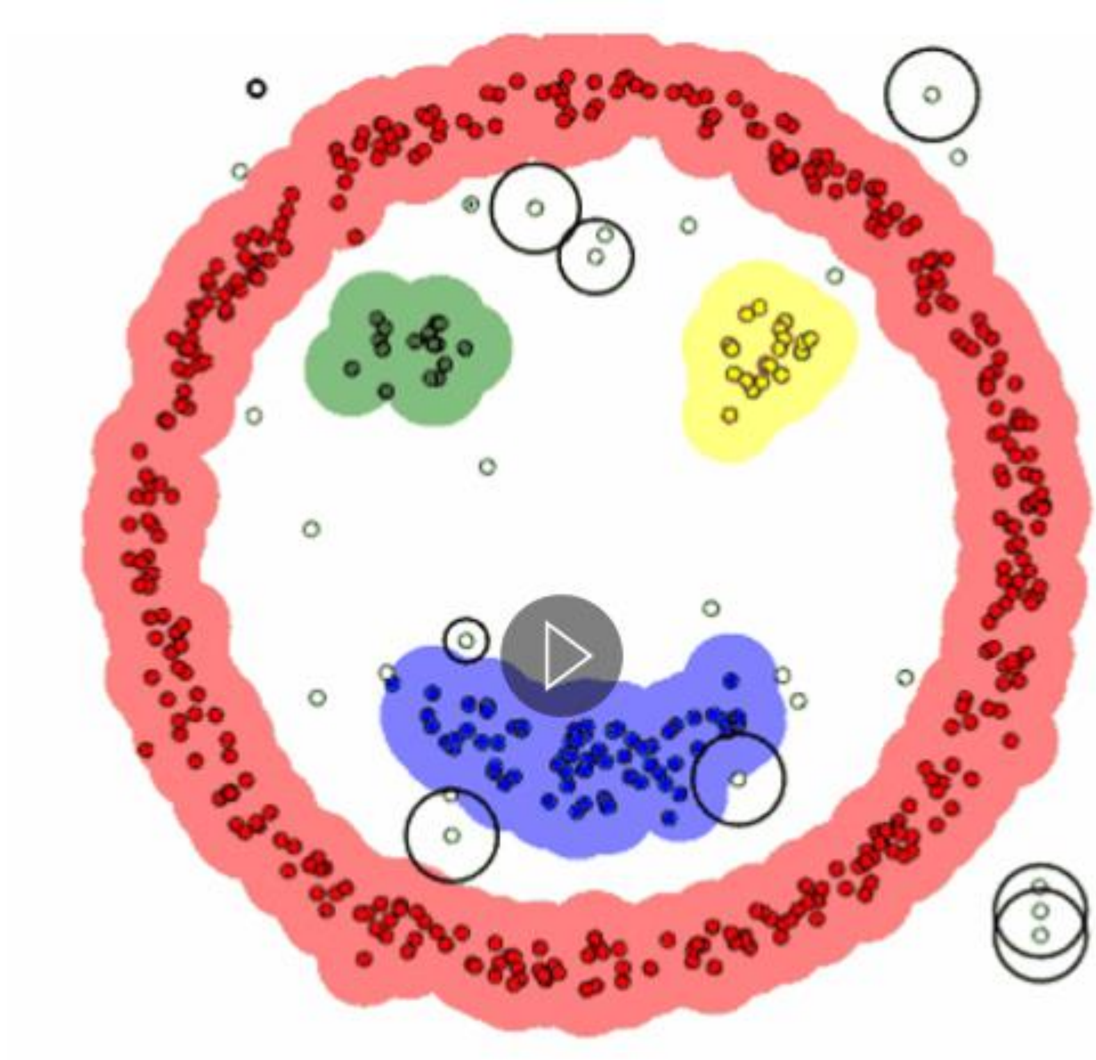
2. The Leaf node can be a border point or core point

Another example: illustration for cluster growth (animation)



https://www.digitalvidya.com/blog/the-top-5-clustering-algorithms-data-scientists-should-know/

Another example: illustration for cluster growth



Discussion:

Construct clusters in DBSCAN:

- A cluster is constructed by merging reachable core points and their border points.
  - A cluster consists of core points that are reachable from one another and all the border points of these core points.
- The requirement to form a cluster is to have at least one core point.

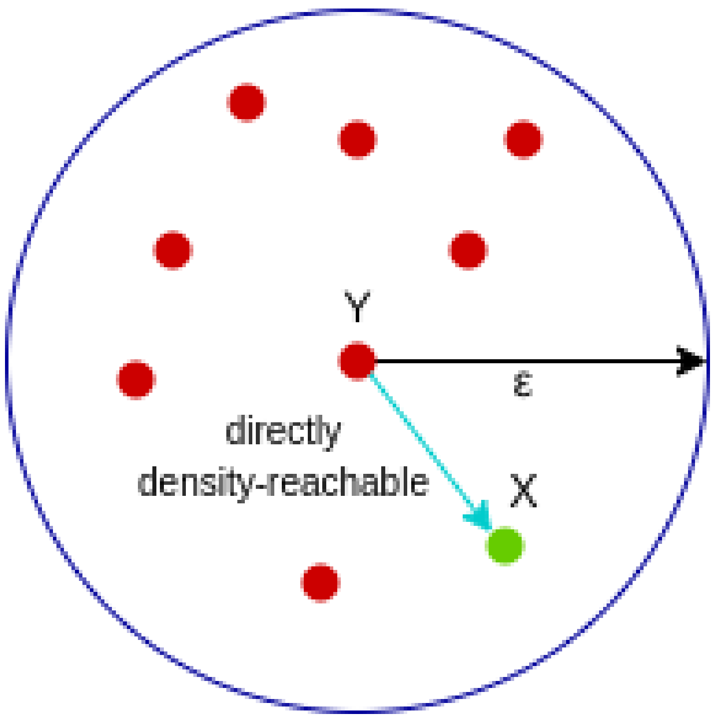
https://towardsdatascience.com/dbscan-algorithm-complete-guide-and-application-with-python-scikit-learn-d690cbae4c5d “directly reachable” is also called “directly density-reachable”

1. A point **X** is **directly density-reachable (or directly reachable)** from point **Y** w.r.t *epsilon*, *minPoints* if,

1) **X** belongs to the epsilon-neighborhood of **Y**, i.e, *dist(X, Y) <= epsilon*

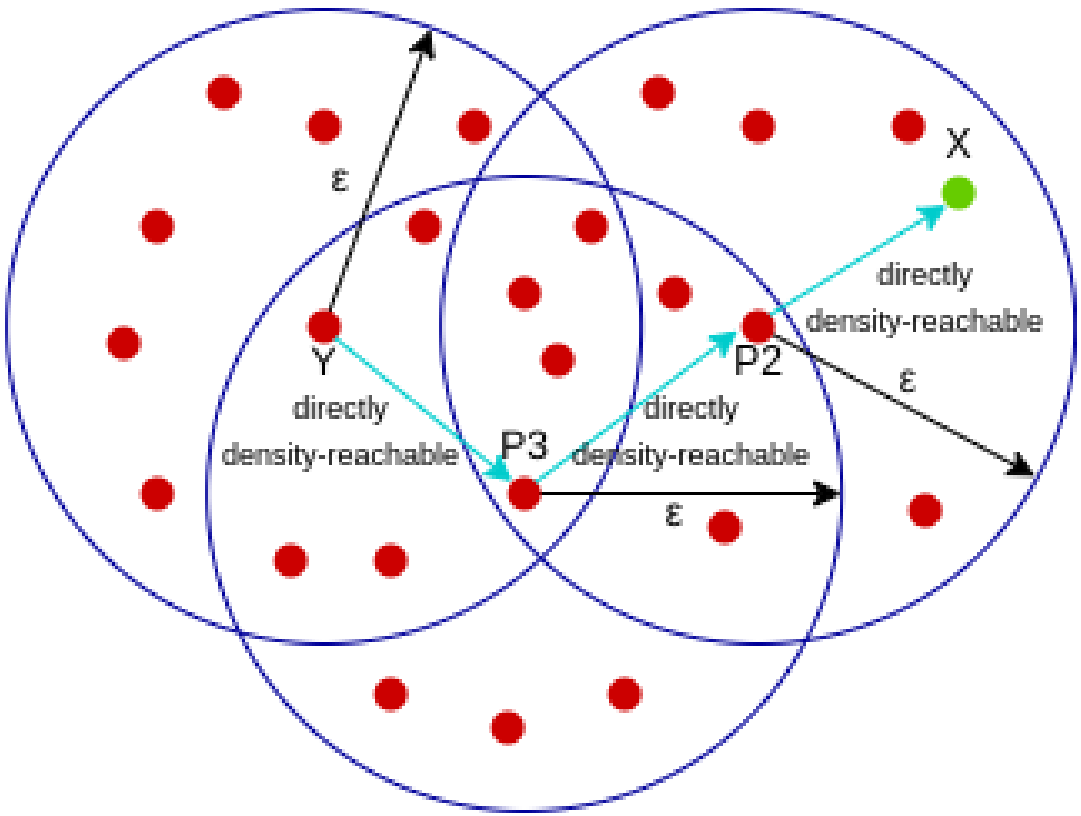
It can be a border point or core point

2) **Y** is a core point



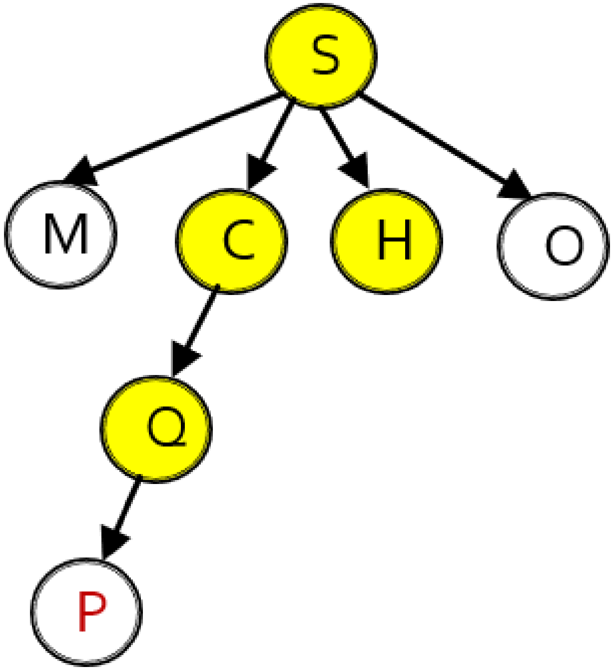
https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/

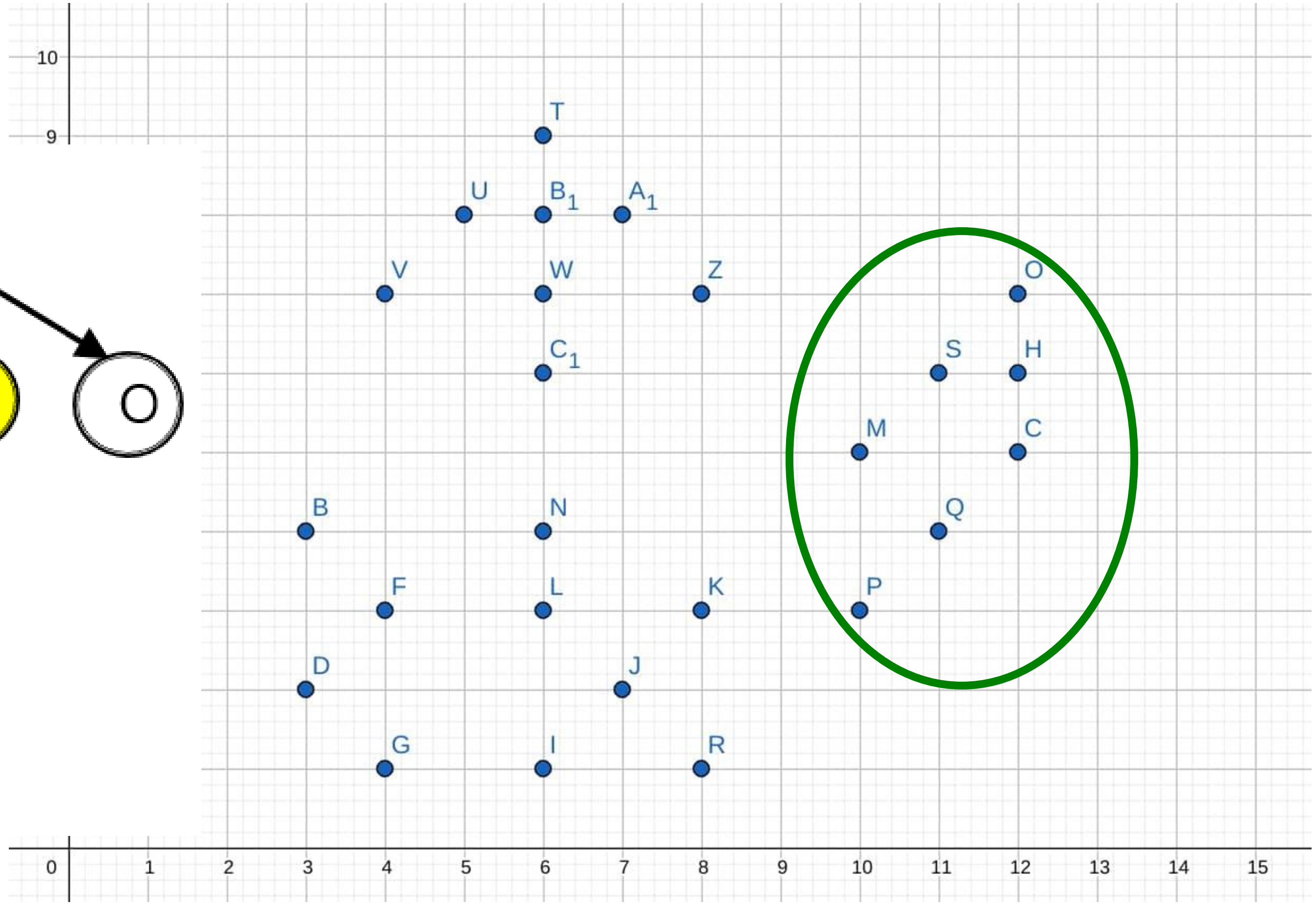
2. A point **X** is **density-reachable (or reachable)** from point **Y** w.r.t *epsilon*, *minPoints* if there is a chain of points p_1, p_2, p_3, ..., p_n and p_1=**X** and p_n=**Y** such that p_{i+1} is directly density-reachable from p_i.



If X is reachable from Y (source point) :
we can find a path connecting points x and y, where each point in the path is directly reachable from the previous one. (The path is constructed by “directly reachable” core points)

Example:





A tree to describe the process of cluster growth in DBSCAN

A parent node: represents a core point

A child node: a neighbour of the parent node

A connection from a parent node to its child indicates directly reachable

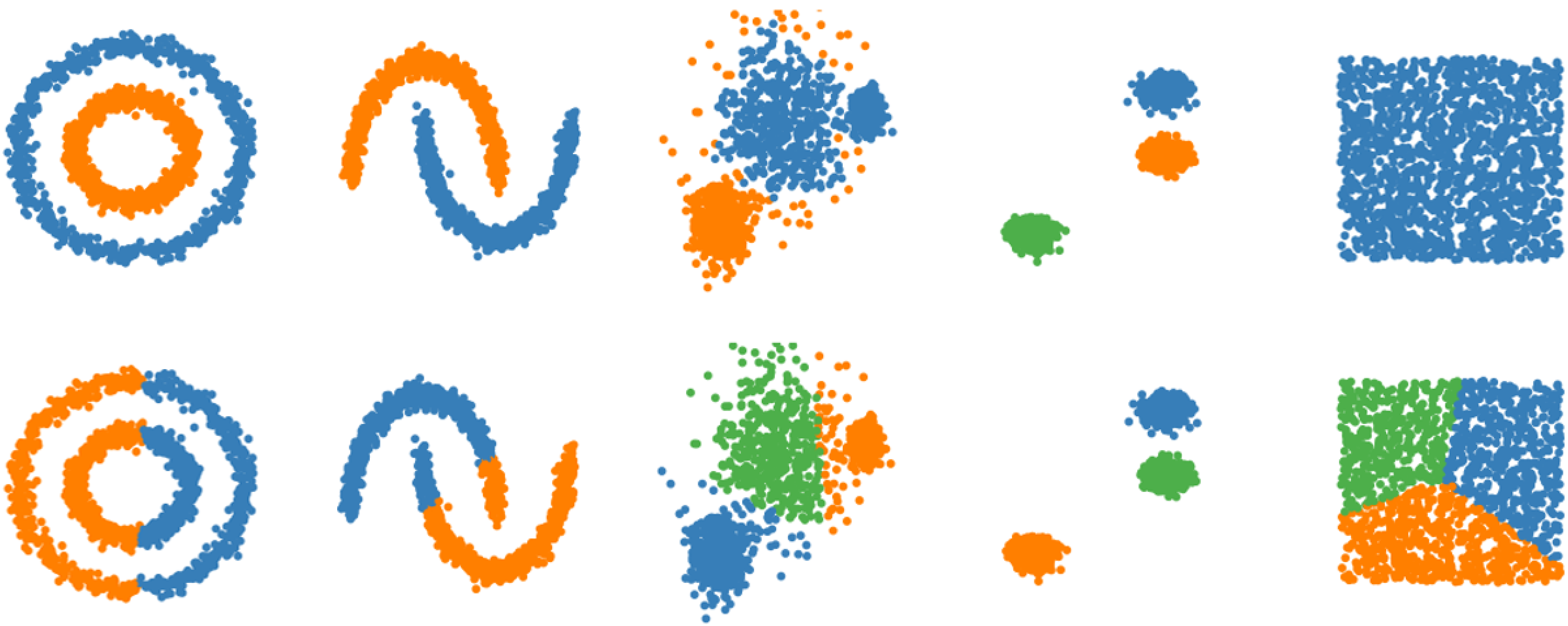
One point is reachable from any core points in the cluster (can find a path in the tree)

Discussion

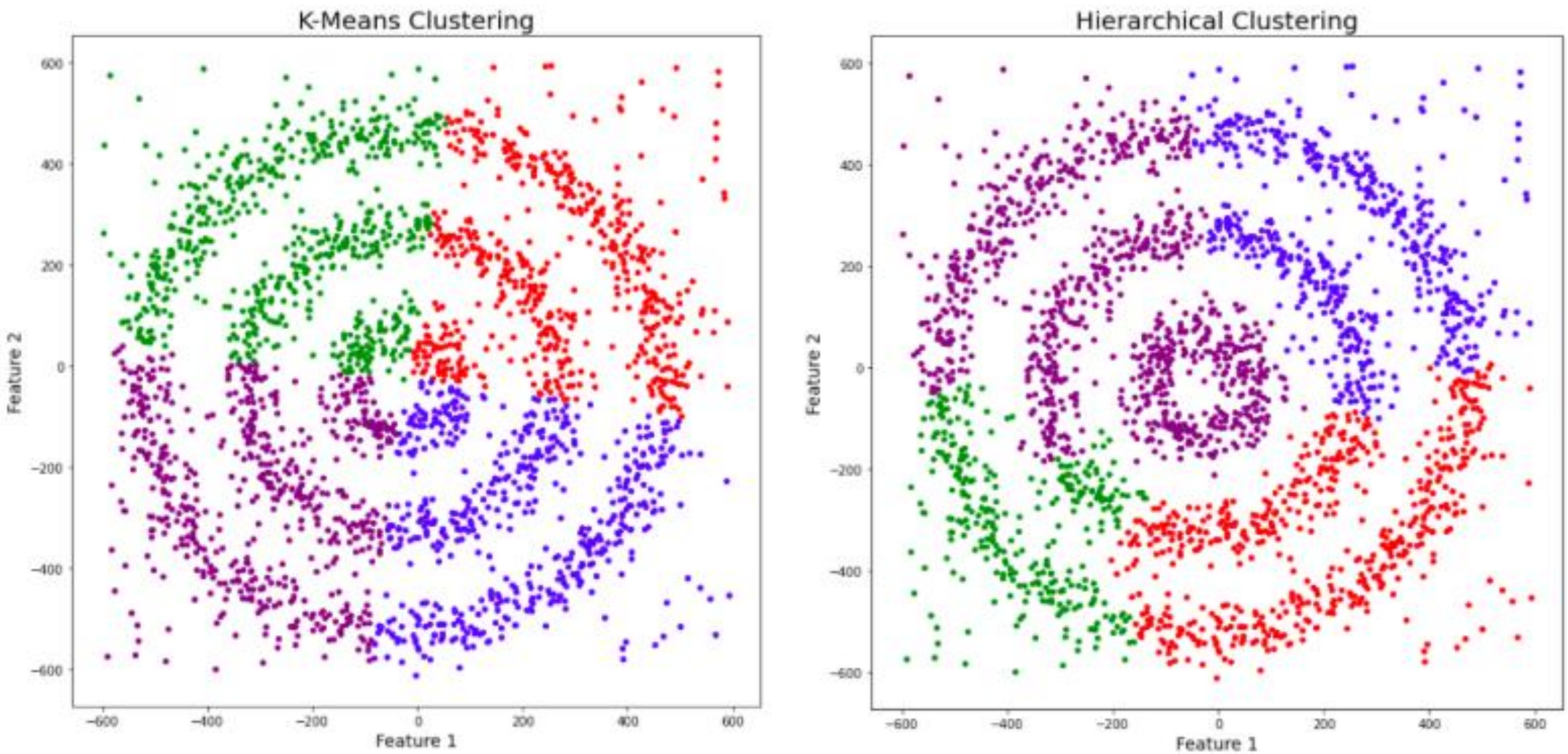
- DBSCAN
  - All core points are equally important to the determine the shape of one cluster, so it can work for clusters with arbitrary shapes
- K-Means:
  - the centroid is important.
  - the shape of the cluster is determined by only one point
  - only works well for clusters with spherical shapes
- DBSCAN is density-based clustering
  - defines clusters as dense regions separated by low-density regions.

https://github.com/NSHipster/DBSCAN

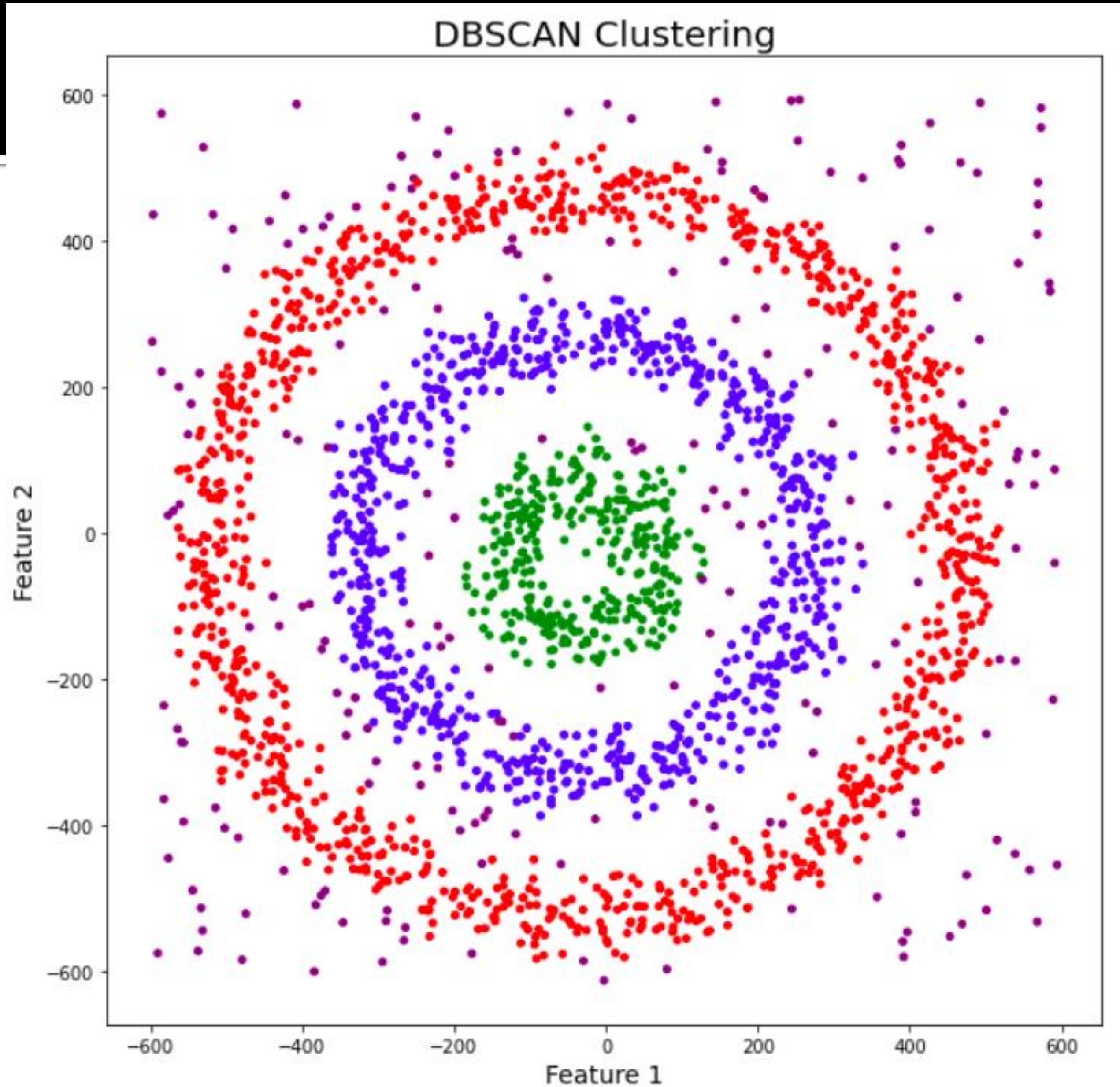
DBSCAN



K-means



K-means, Hierarchical Clustering are sensitive to noise



https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/

Discussion

Pros and Cons of DBSCAN

Pros:

- Does not require to specify number of clusters beforehand.
- Performs well with arbitrary shapes clusters.
- DBSCAN is robust to outliers and able to detect the outliers.

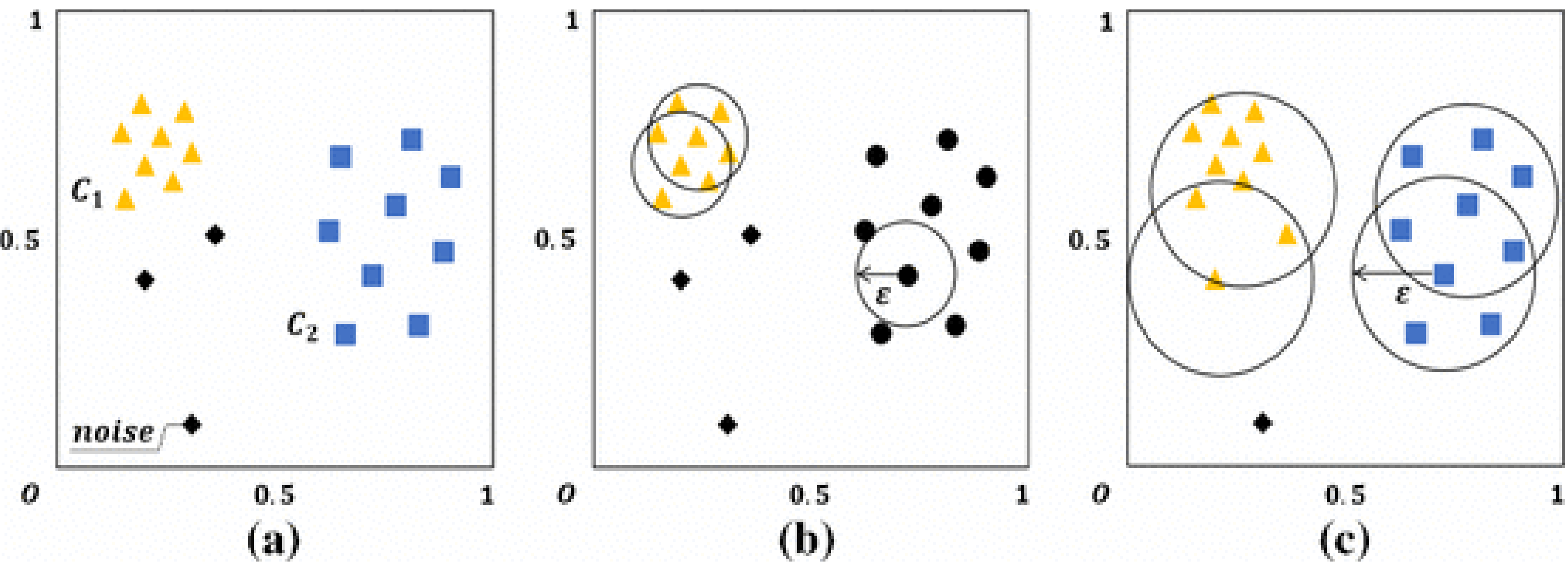
Discussion

- Cons
  - It is not very effective when you have clusters of varying densities.
    - if there are different density levels,
      it is difficult to choose a setting of the neighbourhood distance threshold (epsilon) and MinPts that can work well for all density levels.

  - If you have high dimensional data, the determining of the distance threshold Ɛ becomes a challenging task.

https://www.digitalvidya.com/blog/the-top-5-clustering-algorithms-data-scientists-should-know/

DBSCAN is not very effective for clusters with varying density



ideal result

DBSCAN with a small epsilon
-> many outliers (noise points)

DBSCAN with a large epsilon
-> many outliers are wrongly
included in the dense cluster

Figure credit: AA-DBSCAN: an approximate adaptive DBSCAN for finding clusters with varying densities

Extended discussion

Non-examinable

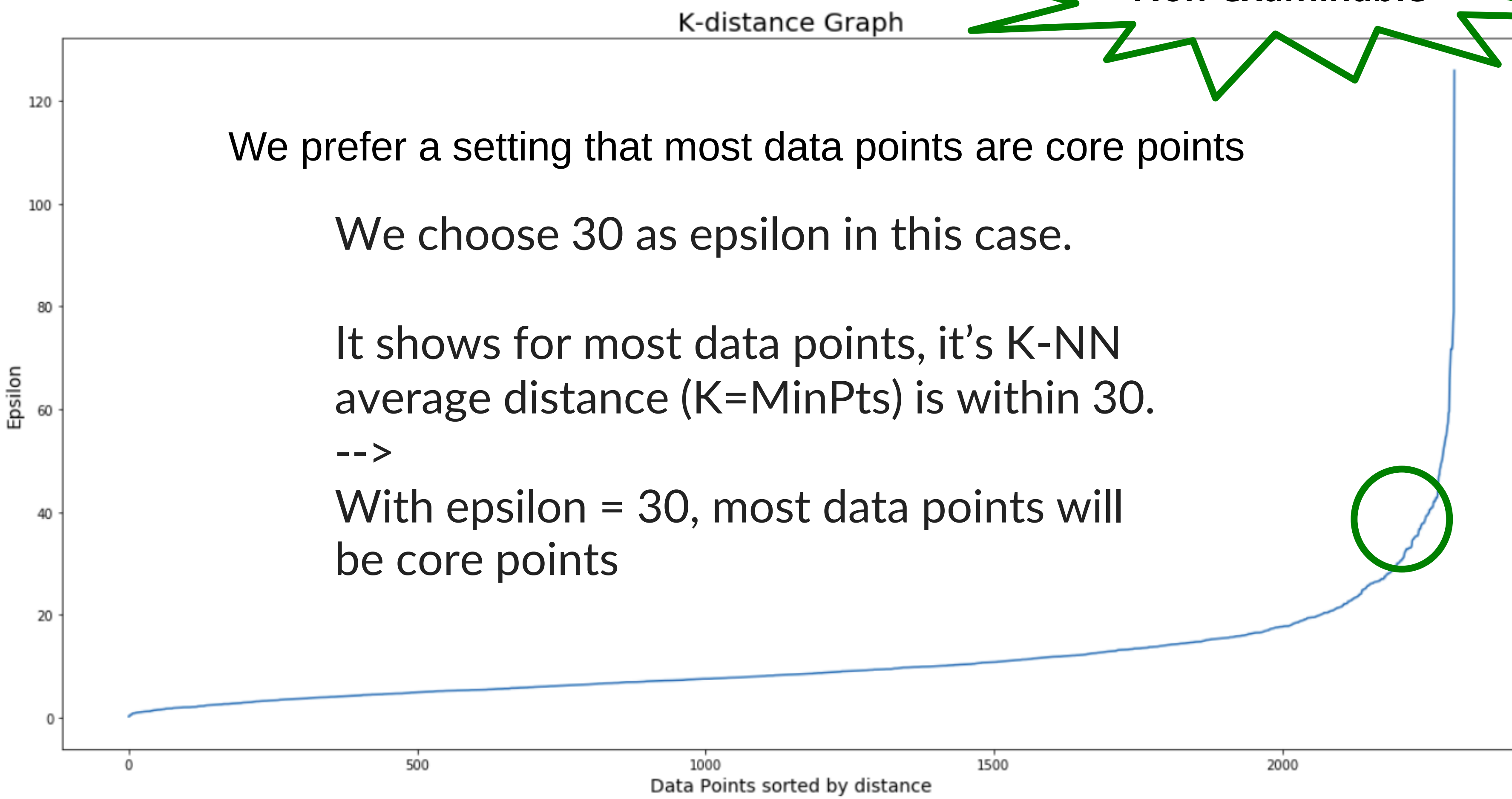
- Two parameters:
  - MinPts and epsilon
  - Select a value for MinPts and then search for epsilon

- How to choose epsilon? (Given MinPts)
  - Use K-distance graph
  - Step 1: Calculate the average distance between each point in the data set and its K nearest neighbors (set K as the MinPts value).
  - Step 2: Sort distance values by ascending value and plot the K-distance graph
  - Step 3: find the elbow point in the graph and use the corresponding distance as Epsilon

https://medium.com/@tarammullin/dbscan-parameter-estimation-ff8330e3a3bd

Y: Averaged distance of K-NN (K= MinPts)

Non-examinable



We prefer a setting that most data points are core points

We choose 30 as epsilon in this case.

It shows for most data points, it's K-NN average distance (K=MinPts) is within 30.
-->
With epsilon = 30, most data points will be core points

X: Number of data points

https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/