Density Based Clustering

Dr. Wedad Hussein
wedad.hussein@cis.asu.edu.eg

Dr. Mahmoud Mounir
mahmoud.mounir@cis.asu.edu.eg

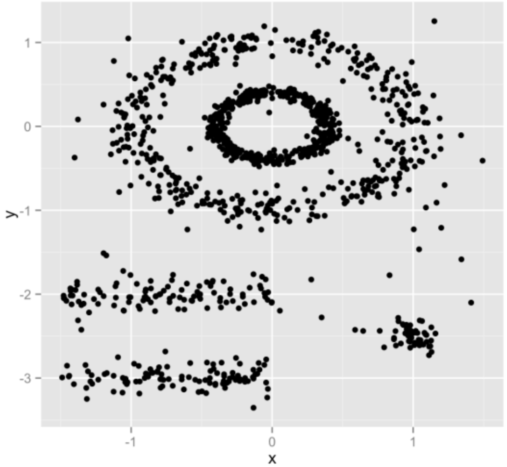
TYPES OF CLUSTERING

Clustering algorithms

- Connectivity-based Clustering
- Centroid-based Clustering
- Distribution-based Clustering
- Density-based Clustering
- Graph – based Clustering

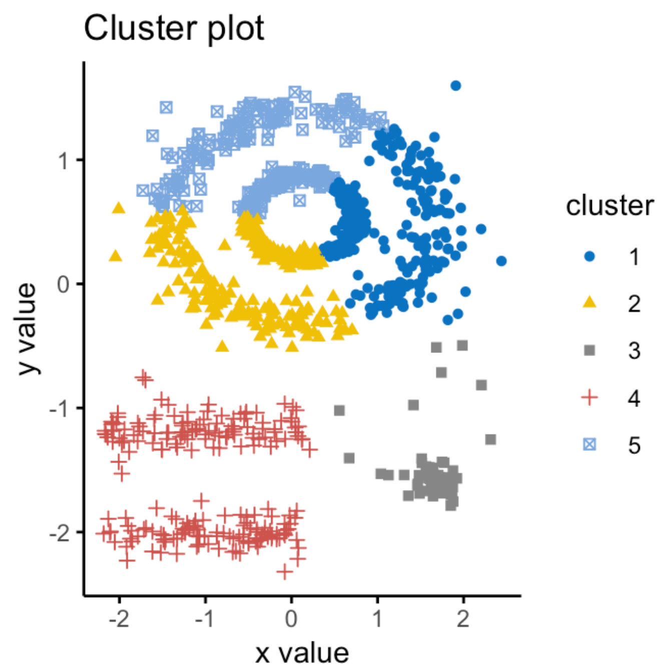
DBSCAN

- **K-Means** is suitable for finding spherical-shaped clusters or convex clusters.
  - In other words, it works well for compact and well separated clusters.
  - Moreover, it is also severely affected by the presence of noise and outliers in the data.
  - Unfortunately, **real life data may contain**:
    - **Clusters can be of arbitrary shape** (oval, linear, and “S” shape).
    - Data may contain **noise and outliers**.
- The plot contains **5 clusters** and **outlier**
- including:
  - 2 oval clusters.
  - 2 linear clusters.
  - 1 compact cluster.



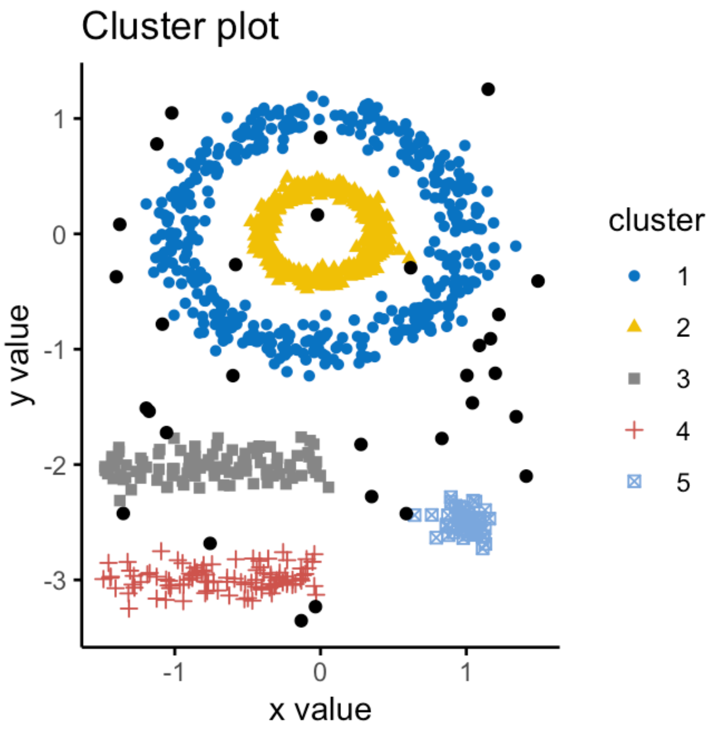
DBSCAN

- Given such data, **k-means** algorithm has difficulties for identifying theses clusters with arbitrary shapes.
- We know there are **5 clusters** in the data, but it can be seen that k-means method inaccurately identifies the 5 clusters.



DBSCAN

- It can be seen that DBSCAN **performs better** for these data sets and can identify the **correct set of clusters** compared to **k-means** algorithms.



DBSCAN

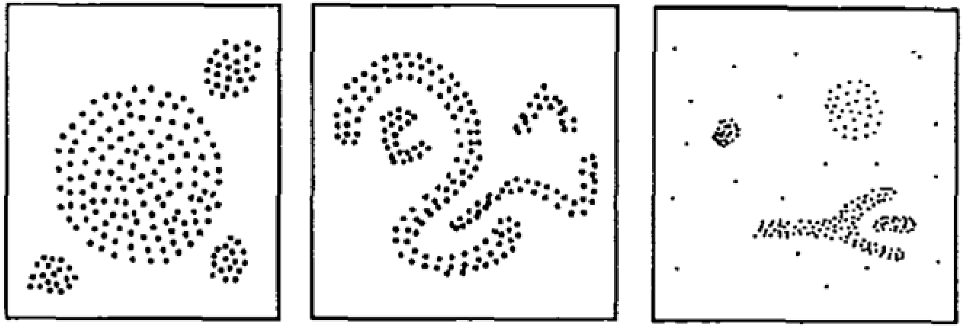
- The **DBSCAN**, a **density-based clustering algorithm**, can be used to **identify clusters of any shape** in dataset **containing noise and outliers**.

- **DBSCAN** stands for **D**ensity-**B**ased **S**patial **C**lustering and **A**pplication with **N**oise.

- **The advantage of DBSCAN:**
  - Unlike **K-means**, DBSCAN **does not require the user to specify the number of clusters** to be generated.
  - DBSCAN **can find any shape of clusters**. The cluster doesn't have to be circular.
  - DBSCAN **can identify outliers**.

DBSCAN

- The **basic idea behind the density-based clustering** approach is derived from a human intuitive clustering method.
  - For instance, by looking at the **figure** below, one can easily identify **four clusters along with several points of noise**, because of the differences in the density of points.
  - As illustrated in the figure, **clusters are dense regions** in the data space, **separated by regions of lower density** of points.
  - **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**.



Algorithm of DBSCAN

- The goal is to **identify dense regions**, which can be measured by the number of objects close to a given point.

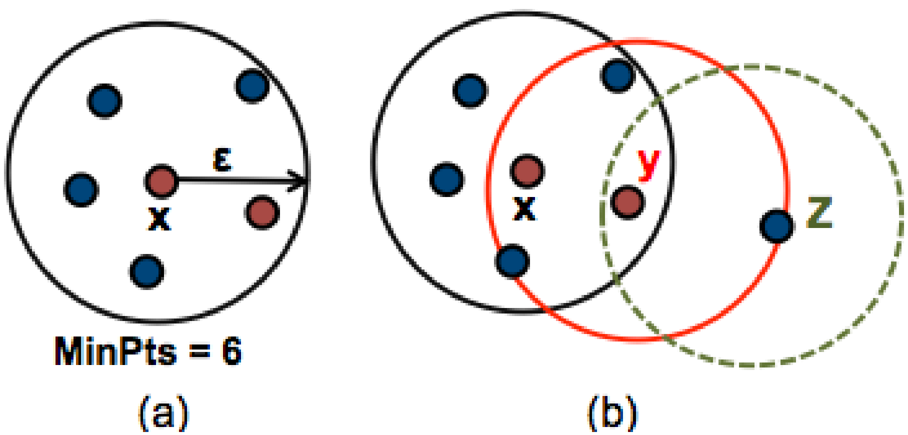
- Two important parameters are required for DBSCAN:
  - ***epsilon*** ("*eps*")
  - ***minimum points*** ("*MinPts*").
  - The parameter ***eps*** defines the **radius of neighborhood around a point x**. It's called the epsilon-neighborhood of x.
  - The parameter ***MinPts*** is the **minimum number of neighbors within "eps" radius**.

Algorithm of DBSCAN

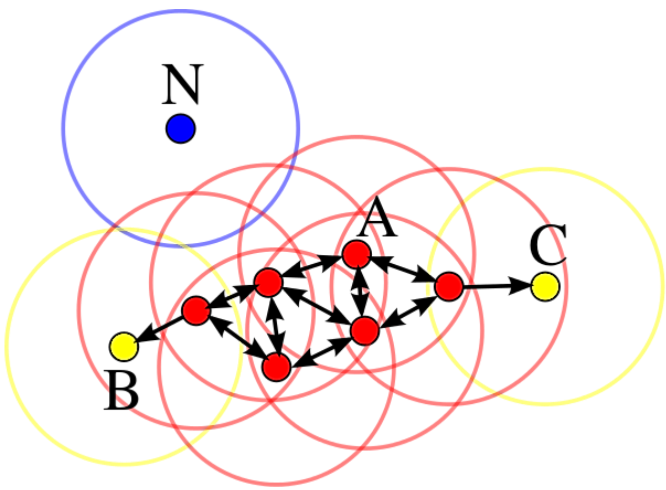
- Any point x in the dataset, with a neighbor count greater than or equal to ***MinPts***, is marked as a ***core point***.
- We say that x is ***border point***, if the number of its neighbors is less than *MinPts*, but it belongs to the epsilon-neighborhood of some core point.
- Finally, if a point is neither a core nor a border point, then it is called a ***noise point*** or an ***outlier***.

Algorithm of DBSCAN

- The figure below shows the different types of points (core, border and outlier points) using **MinPts = 6**.
  - **x is a core point** because neighbours_epsilon(x)=6,
  - **Y is a border point** because neighbours_epsilon(y)<MinPts, but **it belongs to the ϵ-neighborhood of the core point x**.
  - **z is a noise point**.



Algorithm of DBSCAN



The points are classified as follows:

- A point ***p*** is a ***core point,*** if at least **MinPts** points are within distance (*eps)* of it (including *p*). Those points are said to be ***directly reachable from p***.

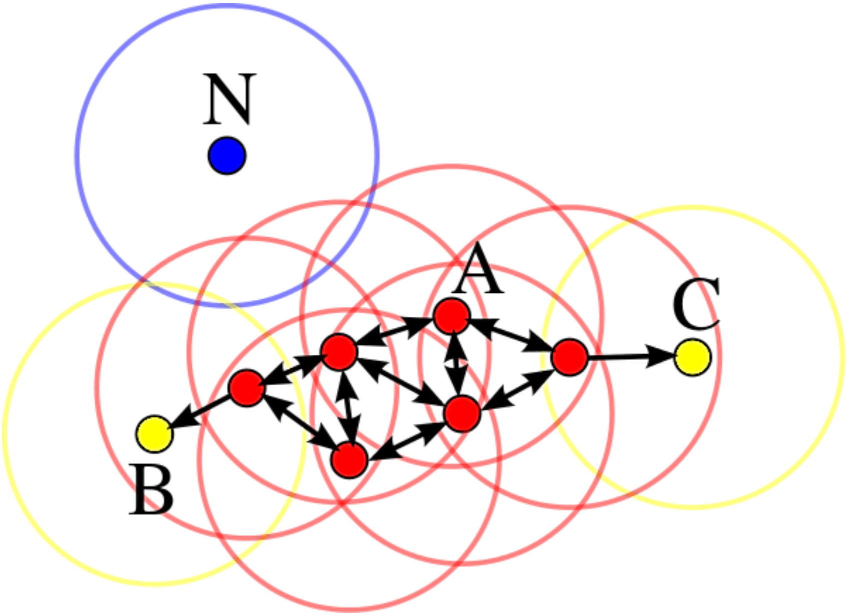
- A point ***q* is *directly reachable* from *p*** if point *q* is within distance (*eps)* from core point *p* *and* ***p must be a core point***.

- A point ***q is density reachable from p*** if there is a path $p_1, \ldots, p_n$ with $p_1 = p$ and $p_n = q$, where **each p_{i+1} is directly reachable from p_i**. (***all points on the path must be core points***, with the possible exception of *q)*.
- Two points p and q are ***density connected*** if there are a core point x, such that p and q are ***density reachable*** from x.
- All points not reachable from any other point are ***outliers*** or ***noise*** *points*.

Algorithm of DBSCAN

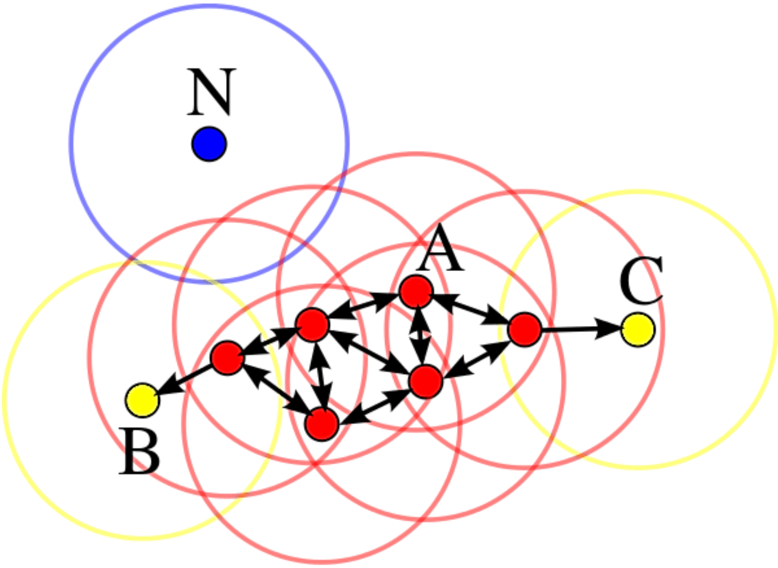
MinPts = 4.

- **Red points** are **core points.**
- **Points B and C** are not core points but are **reachable from A** (via other core points) and thus belong to the cluster as well.
- **Point N** is a **noise point** that is neither a core point nor directly-reachable.



Algorithm of DBSCAN

- A **density-based cluster** is defined as a group of density connected points.
- Now if ***A is a core point***, then it forms a ***cluster*** together with **all points (core or non-core) that are reachable from it**.



Algorithm of DBSCAN

- **The algorithm of DBSCAN works as follow:**

1. For each point x_i, **compute the distance between x_i and the other points**.
   - Finds all neighbor points within distance ***eps*** of the starting point (x_i).
   - Each point, with a neighbor count greater than or equal to ***MinPts***, is marked as ***core point*** or ***visited***.
2. For each ***core point***, if it's not already assigned to a cluster, create a new cluster. Find recursively all its density connected points and assign them to the same cluster as the core point.
3. Iterate through the remaining unvisited points in the data set.

Those points that do not belong to any cluster are treated as outliers or noise.

DBSCAN Example

Given 8 data points:

A1 = (2, 10), A2 = (2, 5), A3 = (8, 4), A4 = (5, 8) , A5 = (7, 5), A6 = (6 , 4) , A7 = (1, 2), A8 = (4, 9).

Apply the DBSCAN algorithm to find the final clusters and identify outlier points in the given data points.

1. *(Use epsilon (eps) = 2 and Minpts =2 and the Euclidean distance as a distance measure)*
2. *What if eps =* $\sqrt{10}$*.*
3. *Draw a 10 X 10 grid to illustrate your answer and the discovered clusters along with the outliers with each epsilon in 1 and 2.*

DBSCAN Example *(eps = 2 , Minpts = 2)*

Step 1: Construct distance matrix

	A1	A2	A3	A4	A5	A6	A7	A8
A1	0	$\sqrt{25}$	$\sqrt{36}$	$\sqrt{13}$	$\sqrt{50}$	$\sqrt{52}$	$\sqrt{65}$	$\sqrt{5}$
A2		0	$\sqrt{37}$	$\sqrt{18}$	$\sqrt{25}$	$\sqrt{17}$	$\sqrt{10}$	$\sqrt{20}$
A3			0	$\sqrt{25}$	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{53}$	$\sqrt{41}$
A4				0	$\sqrt{13}$	$\sqrt{17}$	$\sqrt{52}$	$\sqrt{2}$
A5					0	$\sqrt{2}$	$\sqrt{45}$	$\sqrt{25}$
A6						0	$\sqrt{29}$	$\sqrt{29}$
A7							0	$\sqrt{58}$
A8								0

A1 = (2, 10)
A2 = (2, 5)
A3 = (8, 4)
A4 = (5, 8)
A5 = (7, 5)
A6 = (6 , 4)
A7 = (1, 2)
A8 = (4, 9)

DBSCAN Example *(eps = 2 , Minpts = 2)*

Step 2: Find the Epsilon neighborhood of each data point

eps = 2 , Minpts = 2

N (A1) = {} ✗

N (A2) = {} ✗

N (A3) = {A5, A6} ✓

N (A4) = {A8} ✓

N (A5) = {A3, A6} ✓

N (A6) = {A3, A5} ✓

N (A7) = {} ✗

N (A8) = {A4} ✓

	A1	A2	A3	A4	A5	A6	A7	A8
A1	0	$\sqrt{25}$	$\sqrt{36}$	$\sqrt{13}$	$\sqrt{50}$	$\sqrt{52}$	$\sqrt{65}$	$\sqrt{5}$
A2		0	$\sqrt{37}$	$\sqrt{18}$	$\sqrt{25}$	$\sqrt{17}$	$\sqrt{10}$	$\sqrt{20}$
A3			0	$\sqrt{25}$	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{53}$	$\sqrt{41}$
A4				0	$\sqrt{13}$	$\sqrt{17}$	$\sqrt{52}$	$\sqrt{2}$
A5					0	$\sqrt{2}$	$\sqrt{45}$	$\sqrt{25}$
A6						0	$\sqrt{29}$	$\sqrt{29}$
A7							0	$\sqrt{58}$
A8								0

DBSCAN Example *(eps = 2 , Minpts = 2)*

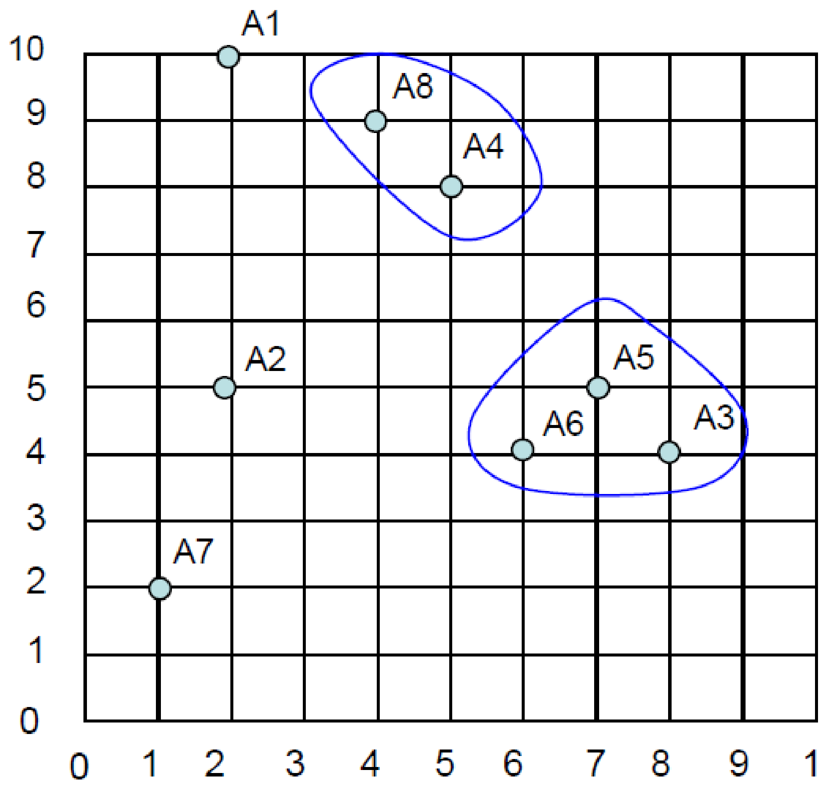
Step 3: Identify the final clusters and outliers

Cluster (1) = {A3, A5, A6}

Cluster (2) = {A4, A8}

Outliers

A1, A2, A7



Epsilon = 2

DBSCAN Example *(eps = $\sqrt{10}$, Minpts = 2)*

Step 1: Construct distance matrix

	A1	A2	A3	A4	A5	A6	A7	A8
A1	0	$\sqrt{25}$	$\sqrt{36}$	$\sqrt{13}$	$\sqrt{50}$	$\sqrt{52}$	$\sqrt{65}$	$\sqrt{5}$
A2		0	$\sqrt{37}$	$\sqrt{18}$	$\sqrt{25}$	$\sqrt{17}$	$\sqrt{10}$	$\sqrt{20}$
A3			0	$\sqrt{25}$	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{53}$	$\sqrt{41}$
A4				0	$\sqrt{13}$	$\sqrt{17}$	$\sqrt{52}$	$\sqrt{2}$
A5					0	$\sqrt{2}$	$\sqrt{45}$	$\sqrt{25}$
A6						0	$\sqrt{29}$	$\sqrt{29}$
A7							0	$\sqrt{58}$
A8								0

A1 = (2, 10)
A2 = (2, 5)
A3 = (8, 4)
A4 = (5, 8)
A5 = (7, 5)
A6 = (6 , 4)
A7 = (1, 2)
A8 = (4, 9)

DBSCAN Example *(eps = $\sqrt{10}$, Minpts = 2)*

Step 2: Find the Epsilon neighborhood of each data point

eps = $\sqrt{10}$, Minpts = 2

N (A1) = {A8}

N (A2) = {A7}

N (A3) = {A5, A6}

N (A4) = {A8}

N (A5) = {A3, A6}

N (A6) = {A3, A5}

N (A7) = {A2}

N (A8) = {A1,A4}

	A1	A2	A3	A4	A5	A6	A7	A8
A1	0	$\sqrt{25}$	$\sqrt{36}$	$\sqrt{13}$	$\sqrt{50}$	$\sqrt{52}$	$\sqrt{65}$	$\sqrt{5}$
A2		0	$\sqrt{37}$	$\sqrt{18}$	$\sqrt{25}$	$\sqrt{17}$	$\sqrt{10}$	$\sqrt{20}$
A3			0	$\sqrt{25}$	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{53}$	$\sqrt{41}$
A4				0	$\sqrt{13}$	$\sqrt{17}$	$\sqrt{52}$	$\sqrt{2}$
A5					0	$\sqrt{2}$	$\sqrt{45}$	$\sqrt{25}$
A6						0	$\sqrt{29}$	$\sqrt{29}$
A7							0	$\sqrt{58}$
A8								0

DBSCAN Example *(eps = $\sqrt{10}$, Minpts = 2)*

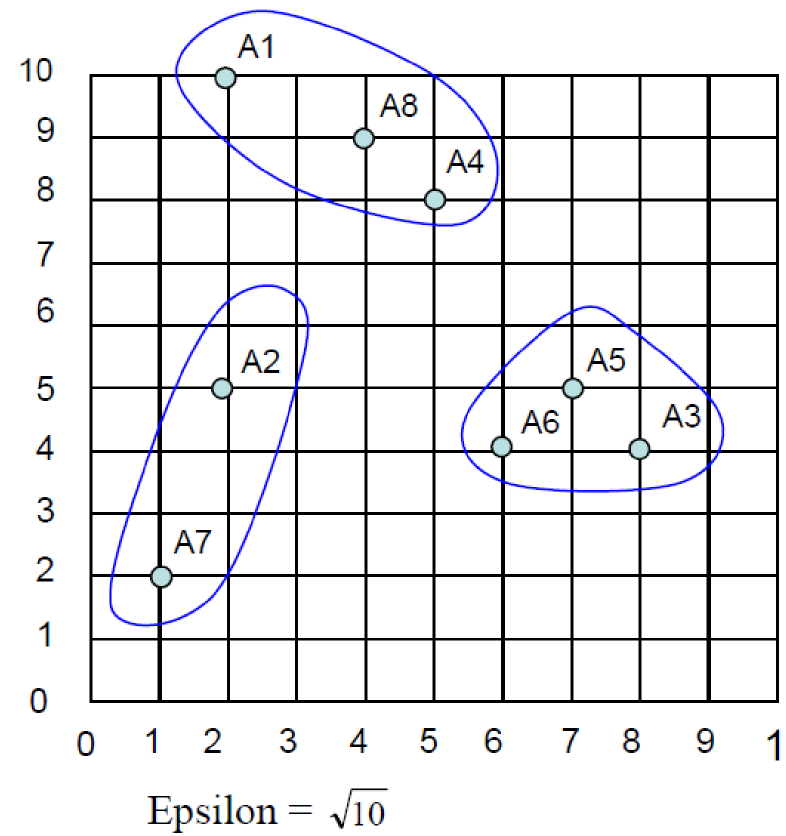
Step 3: Identify the final clusters and outliers

Cluster (1) = {A1, A4, A8}

Cluster (2) = {A3, A5, A6}

Cluster (3) = {A2, A7}

No Outliers



Parameter Estimation of DBSCAN

- DBSCAN algorithm requires the user to identify the optimal values for ***eps*** *and* ***MinPts****.*
  - ***MinPts***: As a general rule, a minimum *minPts* can be derived from the number of dimensions *D* in the data set, as ***MinPts* ≥ *D* + 1**.
    - Larger values are usually better for data sets with noise and will yield more significant clusters.
    - ***The minimum value for MinPts must be 3***, but it may be necessary to choose larger values for very large data.
  - ***eps****:*
    - if it is too small, a large part of the data will not be clustered; It will be considered outliers.
    - On the other hand if it is too high, clusters will merge and the majority of objects will be in the same cluster.
    - In general, small values of ***eps*** are preferable