CPSC 100

Computational Thinking

Data Mining

Instructor: Firas Moosvi
Department of Computer Science
University of British Columbia

Agenda

- Clustering Techniques:
  - KMeans
  - DBSCAN
- File Systems

Learning Goals

After this week's lecture, you should be able to:

-

Course Admin

Course Admin

-

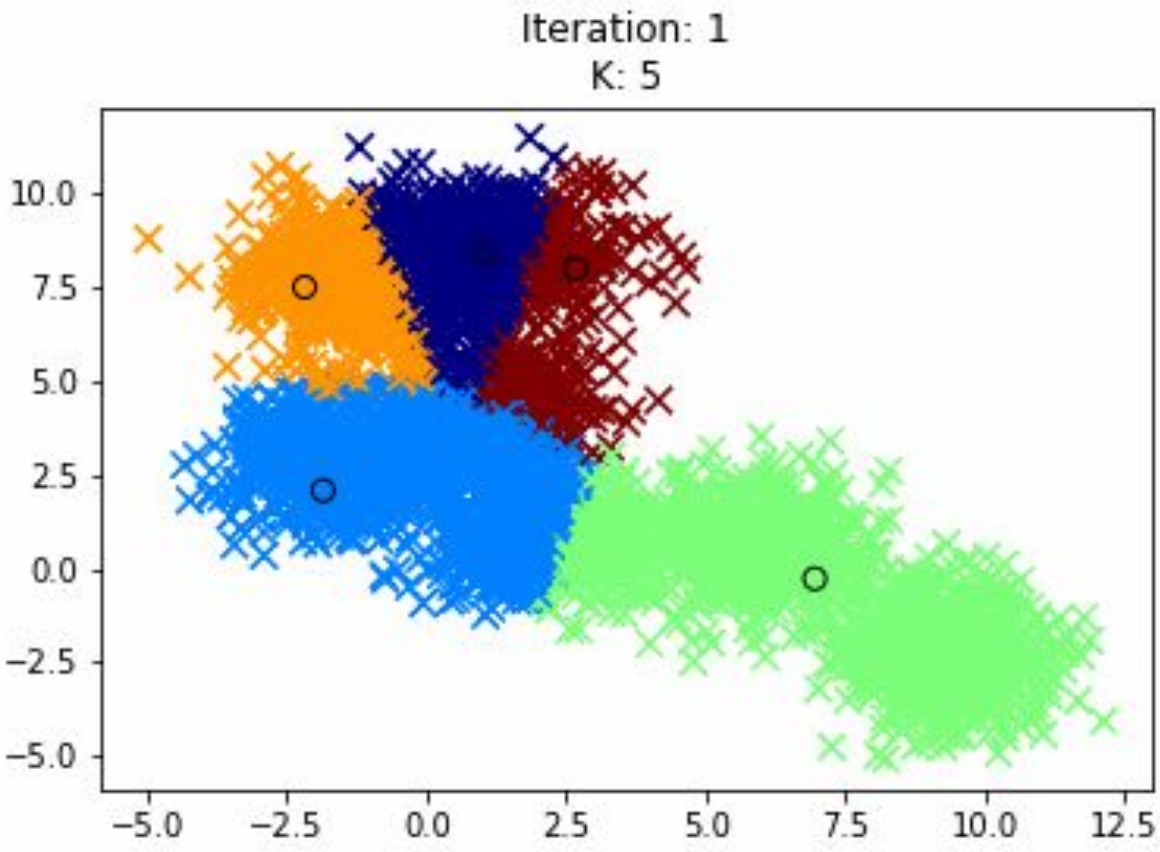
KMeans Algorithm

KMeans Algorithm

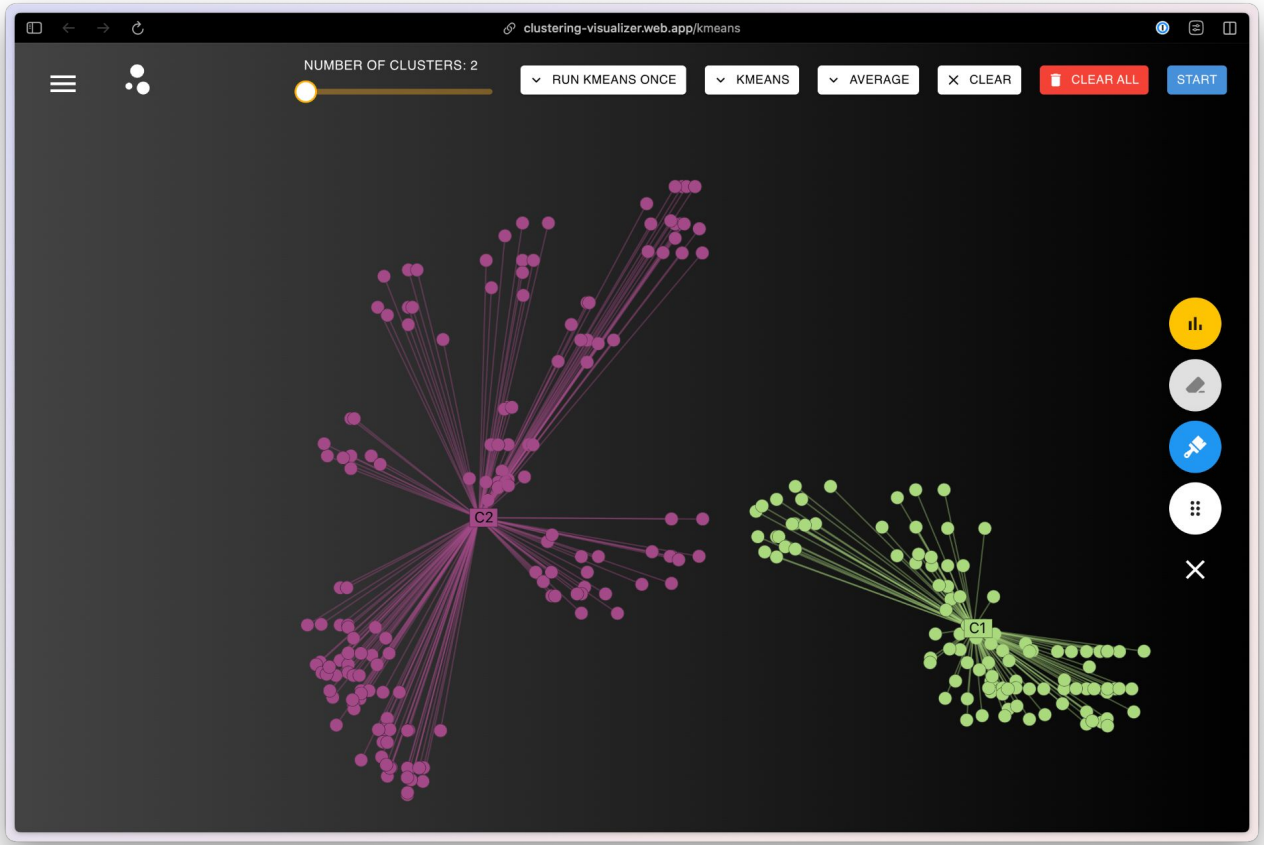
Here is the algorithm used in KMeans:

1. Select the number of clusters (K) you want to identify in your data.
2. Randomly select K data points ; these will be the starting point for the "cluster centres".
3. Calculate the Euclidean distance between each data point and the K cluster centres; each data point should now have K distances.
4. For each data point, "assign" it to the nearest cluster centre (determined by shortest Euclidean distance).
5. Once all the points are assigned to the nearest cluster centre, calculate the "mean" (average) of each cluster (in 2D, the mean of all the X-values and the mean of all the Y-values). These new locations are the **next** cluster centres.
6. Repeat steps 3-5 (i.e. calculate new Euclidean distances between each data point and the new cluster centres, re-assign them, compute the new cluster centre) until the cluster centres do not change (this means the solution converges).

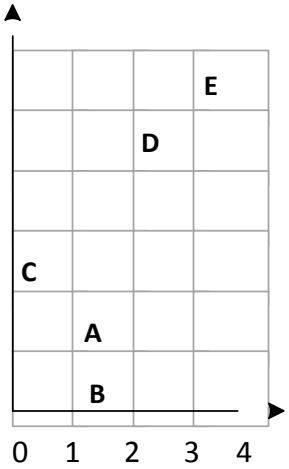
KMeans Algorithm in Action



KMeans Demo

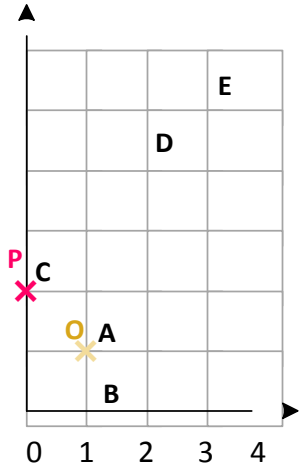

clustering-visualizer.web.app/kmeans
NUMBER OF CLUSTERS: 2
RUN KMEANS ONCE
KMEANS
AVERAGE
CLEAR
CLEAR ALL
START
C2
C1

K-means Clustering Example



Initial points – we'll label them by letters so that we can refer to them more easily through out the example

K-means Clustering Example



Step 1: choose K centroid points at random to act as the "centre" of your clusters

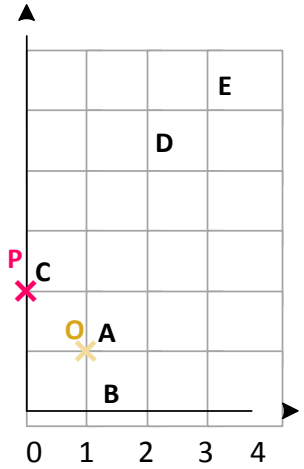
We'll let K = 2

We randomly choose the orange and pink x's to be centroids of clusters O and P respectively

Note: the centroids do not have to be points that are being clustered

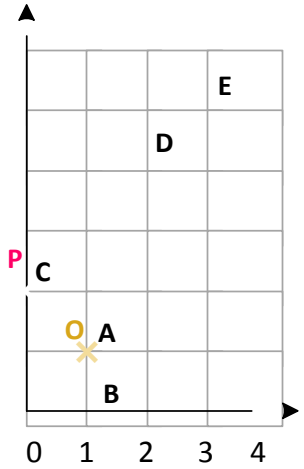
K-means Clustering Example

Step 2A: Calculate distance to centroid



Point	Distance to O	Distance to P
A		
B		
C		
D		
E		

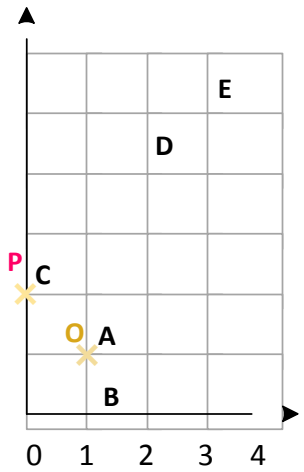
K-means Clustering Example



Step 2B: Cluster assignment: For each point, determine which of the k centroids it's closest to, and put it in the cluster of that centroid

Point	Distance to O	Distance to P
A	0	1.4
B	1	2.2
C	1.4	0
D	3.2	2.8
E	4.5	4.2

K-means Clustering Example



Step 2C: Calculate centroids new position: Average all the points inside each cluster to determine the new position of the centroid

Orange Cluster

Average x for "O" cluster = (1+1)/2 = 1

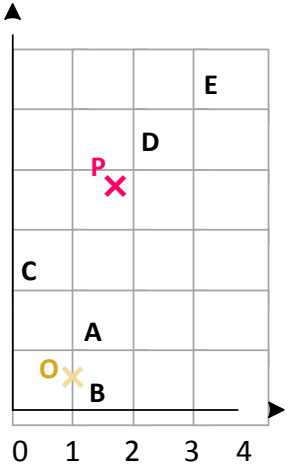
Average y for "O" cluster = (1+0)/2 = .5

Purple/Red Cluster

Average x for "P" cluster = (0+2+3)/3 = 1.7

Average y for "P" cluster = (2+4+5)/3 = 3.7

K-means Clustering Example

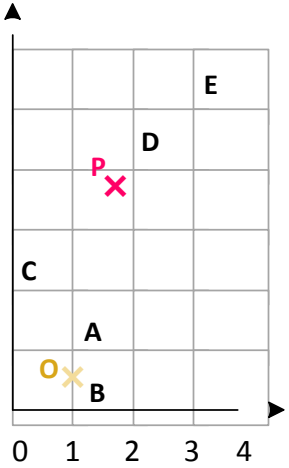


Step 2D: move centroid if position has changed

New centroid for O: (1, .5)

New centroid for P: (1.7, 3.7)

K-means Clustering Example



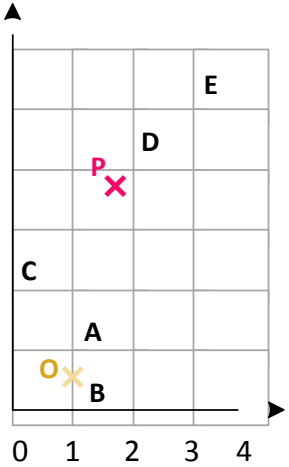
Is it stable?

Nope

So, back to the beginning! We need to calculate the distance to the new centroids

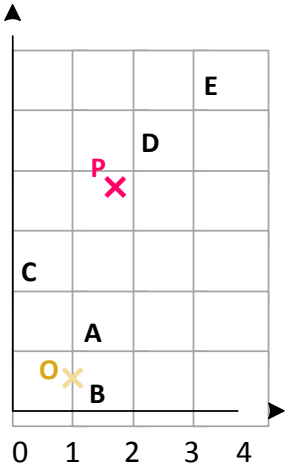
K-means Clustering Example

Step 2A: Calculate distance to centroids



Point	Distance to O	Distance to P
A		
B		
C		
D		
E		

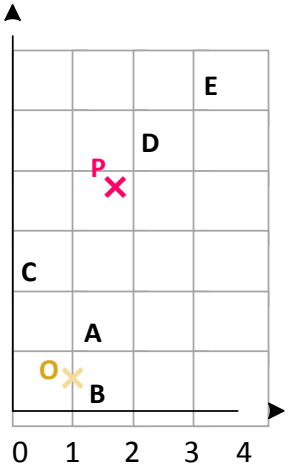
K-means Clustering Example



Step 2B: Cluster Assignment: For each point, determine which of the k centroids it's closest to. Put it in the cluster of that centroid

Point	Distance to O	Distance to P
A	0.5	2.7
B	0.5	3.7
C	1.8	2.4
D	3.6	0.5
E	4.9	1.9

K-means Clustering Example



Step 2C: Calculate Centroids' Positions: Average all the points inside each cluster to get the new position for each centroid.

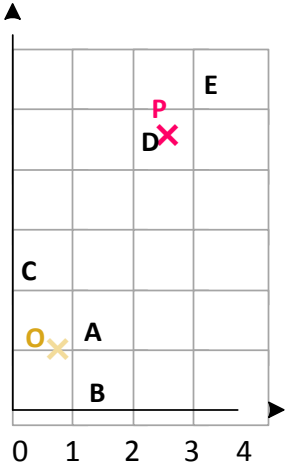
Average x for "O" cluster = (1+1+0)/3 = .7

Average y for "O" cluster = (1+0+2)/3 = 1

Average x for "P" cluster = (2+3)/2 = 2.5

Average y for "P" cluster = (4+5)/2 = 4.5

K-means Clustering Example



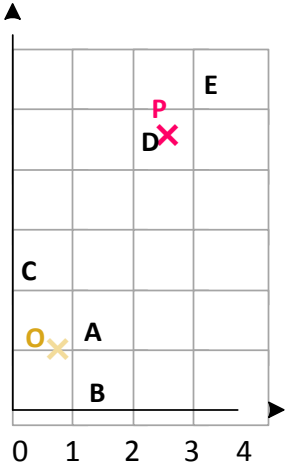
Step 2D: move centroid:

Average x for "O" cluster = (1+1+0)/3 = .7
Average y for "O" cluster = (1+0+2)/3 = 1
Average x for "P" cluster = (2+3)/2 = 2.5
Average y for "P" cluster = (4+5)/2 = 4.5

New centroid for O: (.7, 1)
New centroid for P: (2.5, 4.5)

K-means Clustering Example

What happens next?

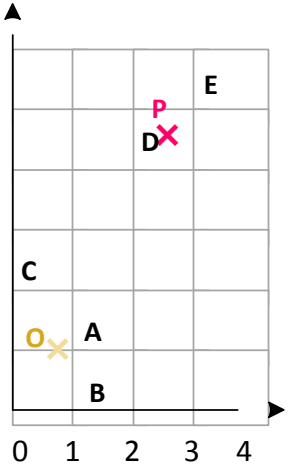


A. We re-calculate distances between points and the two centroids

B. We re-calculate the centroid positions

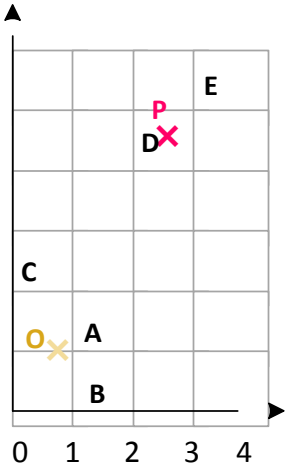
C. We stop

K-means Clustering Example



Back to the beginning (again)! We need to calculate the distance to the new centroids

K-means Clustering Example

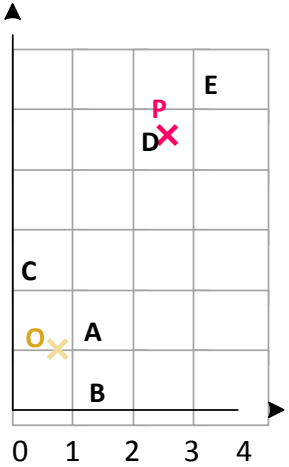


Step 2A: Calculate distance to clusters
Step 2B: Cluster assignment: For each point, determine which of the k centroids it's closest to, and put it in the cluster of that centroid

Point	Distance to O	Distance to P
A	0.3	3.81
B	1.04	4.74
C	1.22	2.4
D	3.27	0.71
E	4.61	0.71

Clustering Question K-means Clustering Example

What happens next?



A. We re-calculate distances between points and the two centroids

B. We re-calculate the centroid positions

C. We stop

Why do we need other clustering methods?

Limitations of K-means Clustering

The algorithm may give different cluster solutions depending on how the initial centroids are chosen

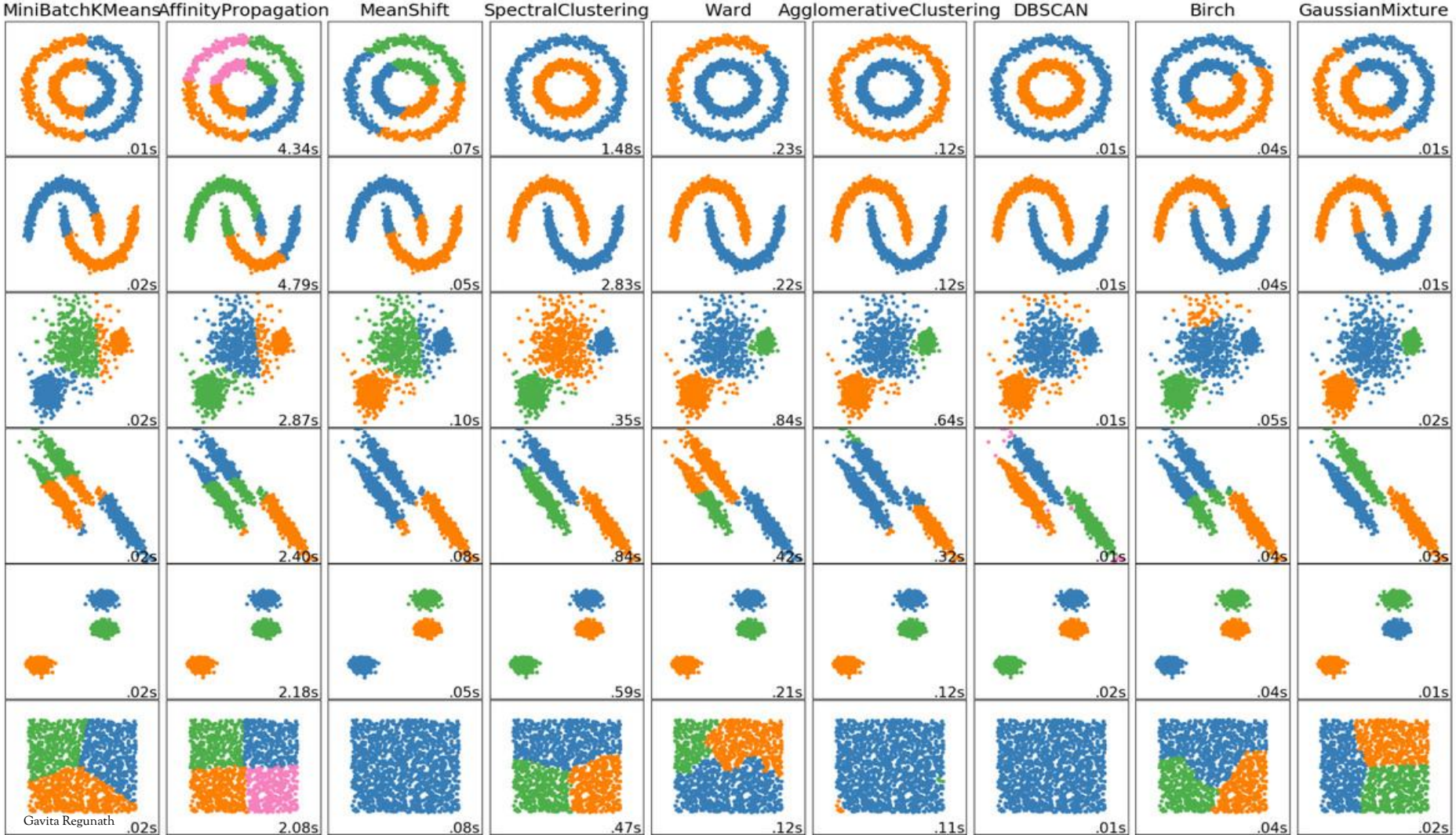
It's not always clear how to choose k *(i.e., the number of clusters)*

- If the size of the data set is small, different values of k can be chosen
- Or, a large value of k can be chosen, and then clusters can be merged to yield a hierarchical cluster structure

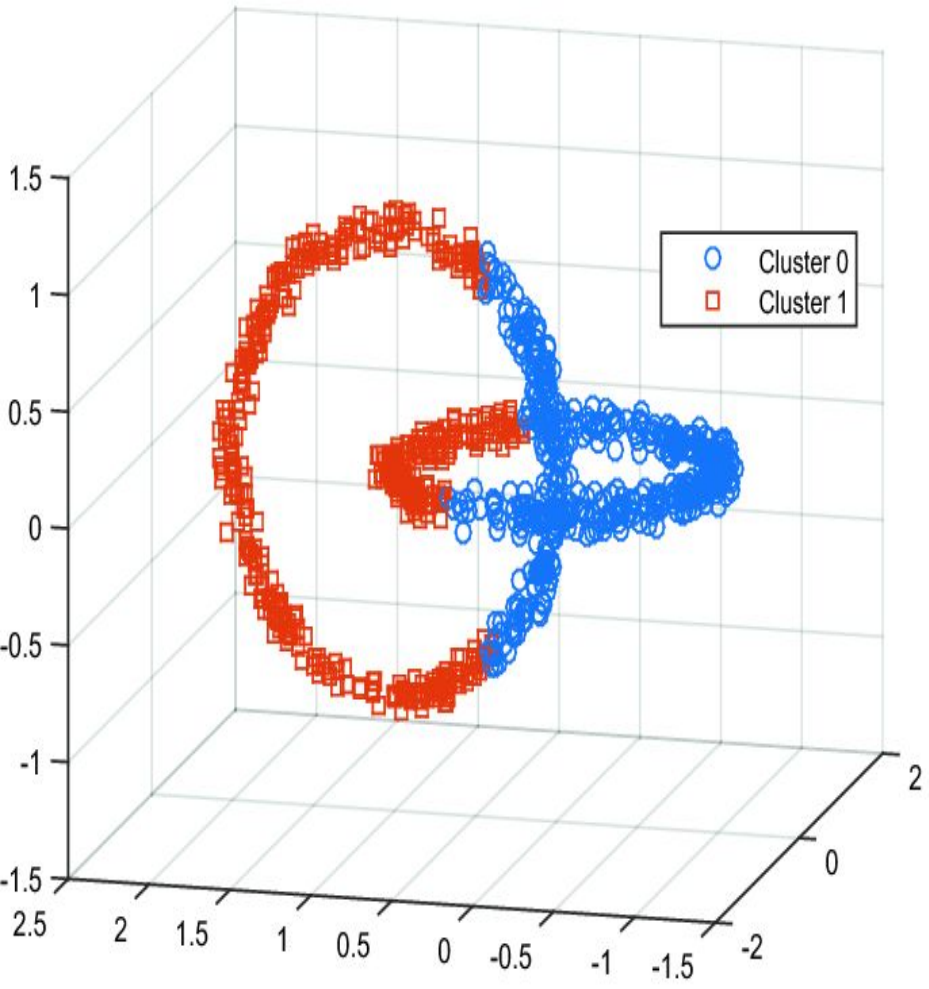
Assessing Your Understanding

We could ask you to:

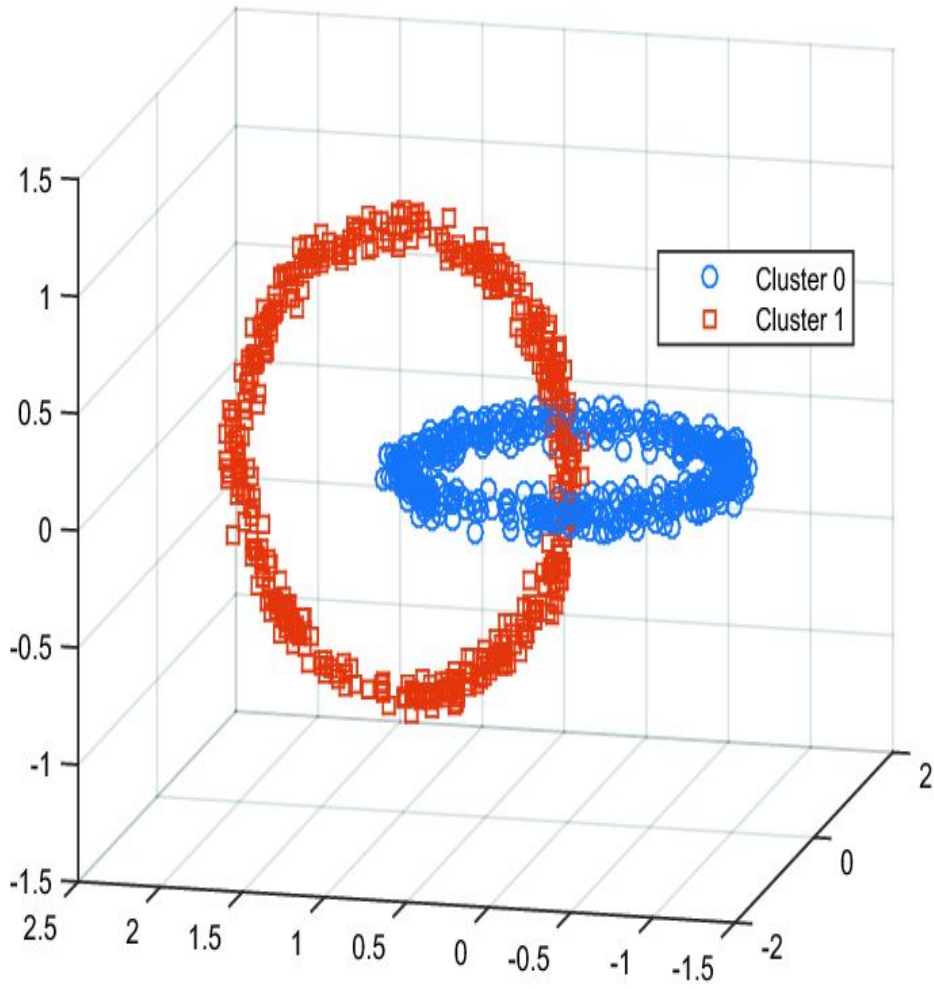
- Determine the next step that needs to be completed
- Reason about the limitations of k-means
- Describe if groups can be created using k-means
- Calculate the new position of the centroid
- Determine which cluster a point should belong to
- Cluster points given in a 2D plane



DBSCAN



(a) k-means



(b) DBSCAN

Liang et al., *Machine Learning*, 2021.

DBSCAN Algorithm

DBSCAN Clustering Algorithm

1. Choose the value of **Epsilon (ε)** and **minPoints (P)**.
2. Run through all the data points in the dataset, and do the following:

- 1. a) Draw a circle around each data point with a small radius.
- 2. b) Count the number of data points, including the central data point itself in the circle.
- 3. c) Determine whether the central data point is a **Core**, **Border**, or **Noise** point.
- 4. **Core** — This is a point that has at least minPoint points within distance ε from itself.
- 5. **Border** — This is a point that has at least one Core point at a distance ε.
- 6. **Noise** — This is a point that is neither a Core nor a Border. And it has less than minPoint points within distance ε from itself.

3. Cluster the points based on reachability and connectivity.

DBSCAN Algorithm

The algorithm works by defining clusters as dense regions separated by regions of lower density. This approach allows DBSCAN to discover clusters of arbitrary shape and identify outliers as noise.

DBSCAN revolves around three key concepts:

1. **Core Points**: These are points that have at least a minimum number of other points (MinPts) within a specified distance (ε or epsilon).
2. **Border Points**: These are points that are within the ε distance of a core point but don't have MinPts neighbors themselves.
3. **Noise Points**: These are points that are neither core points nor border points. They're not close enough to any cluster to be included.

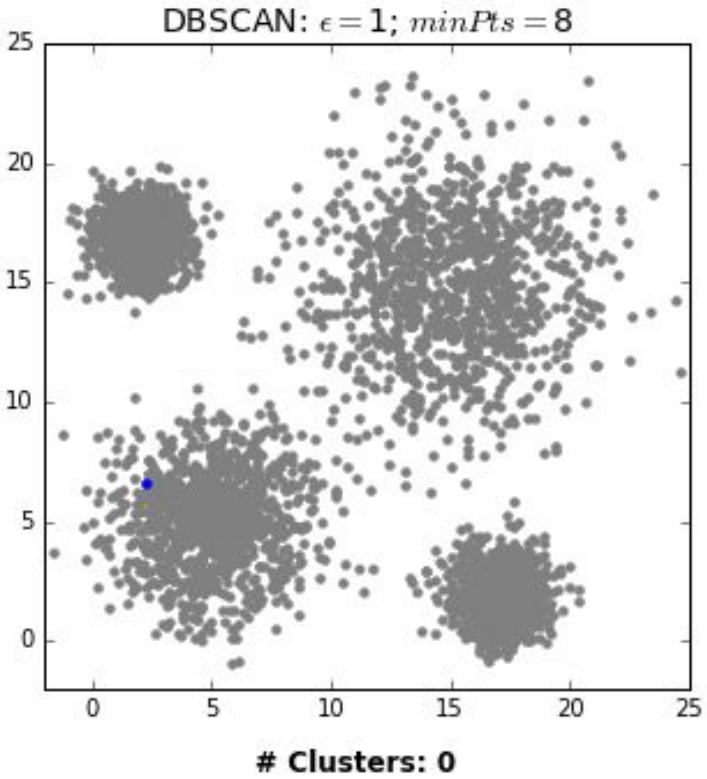
Source

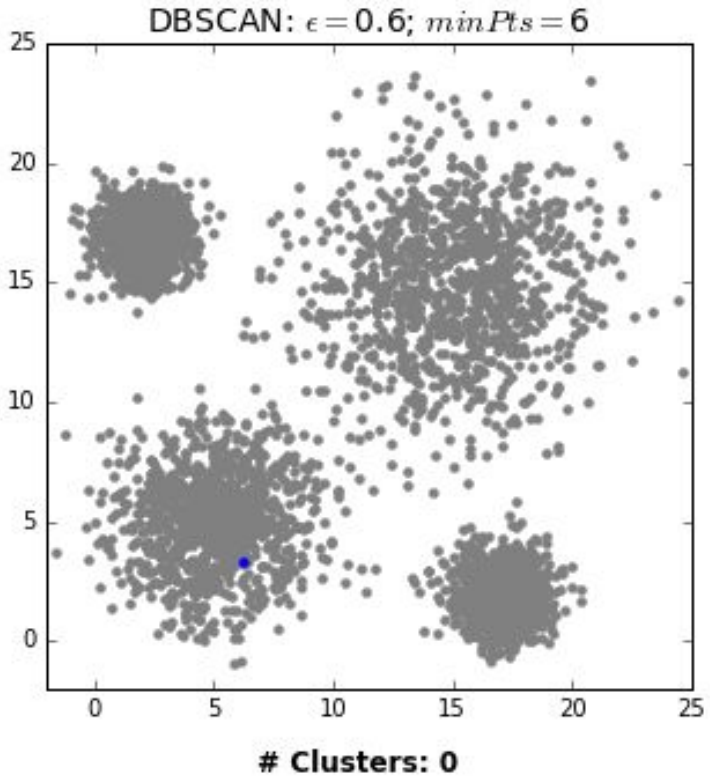
DBSCAN Algorithm

1. **Core Points:** These are points that have at least a minimum number of other points (MinPts) within a specified distance (ε or epsilon).
2. **Border Points:** These are points that are within the ε distance of a core point but don't have MinPts neighbors themselves.
3. **Noise Points:** These are points that are neither core points nor border points. They're not close enough to any cluster to be included.

Source

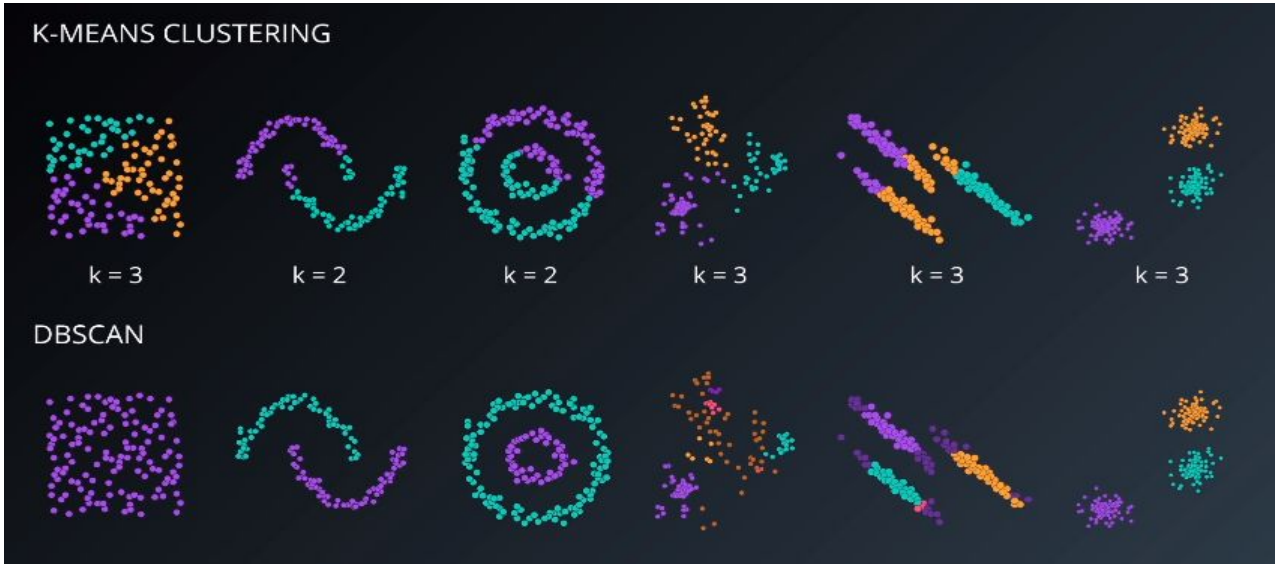
DBSCAN Algorithm in Action





DBSCAN Clustering Algorithm Advantages

- No need to specify the number of clusters
- No need to determine the centroid position
- Scans through the entire dataset once
- Robust to noise and outliers
- DBSCAN requires only input 2 parameters:
  a) Epsilon (ε) → Radius
  b) minPoints



Limitations of DBSCAN Clustering

Border points that are reachable from two clusters may be arbitrarily classified – not an issue for most datasets

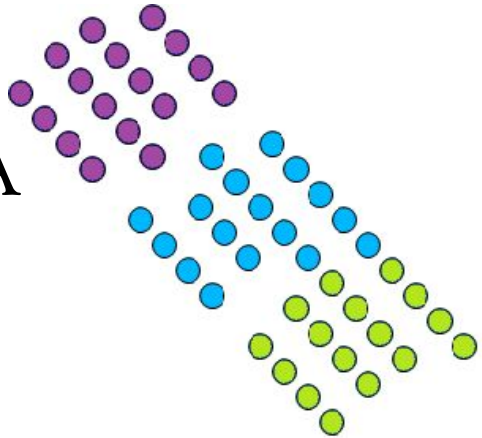
Difficult to identify clusters of varying densities

Cluster Quality

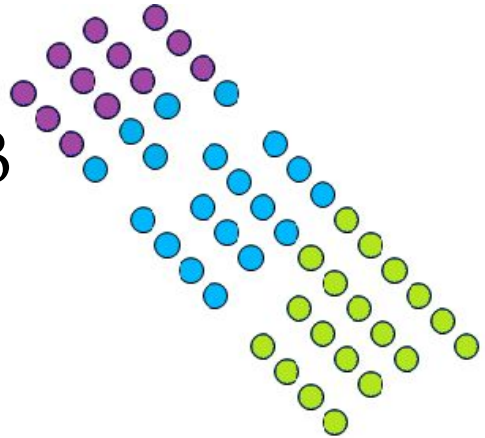
- Consider the following potential clusterings for assigning children to three different schools. Each point represents where a child lives.

  What are the relative merits of each clustering?

A



B



Cluster Quality

- Consider the following potential clusterings. What are the relative merits of each clustering?