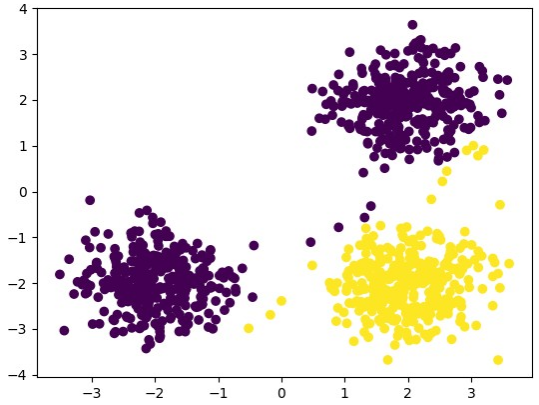
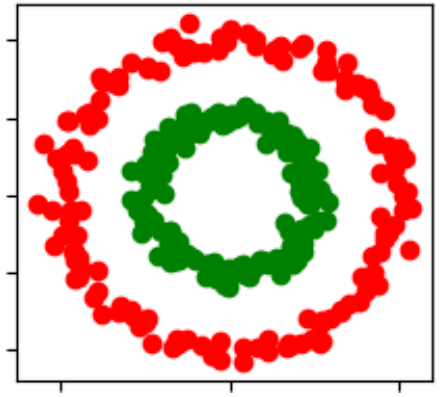
COT 4521: Introduction to Computational Geometry

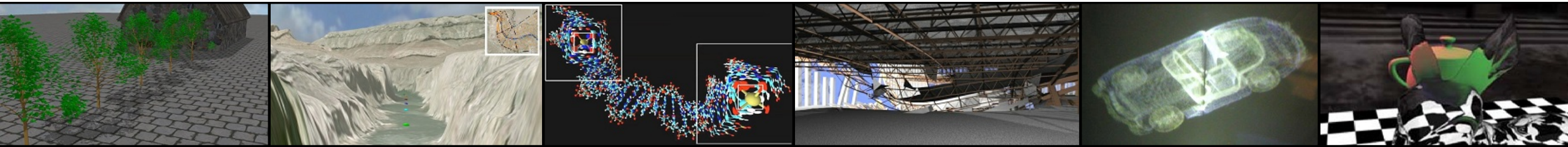
Clustering Algorithms

Paul Rosen
Assistant Professor
University of South Florida

Some slides from Mustafa Hajij





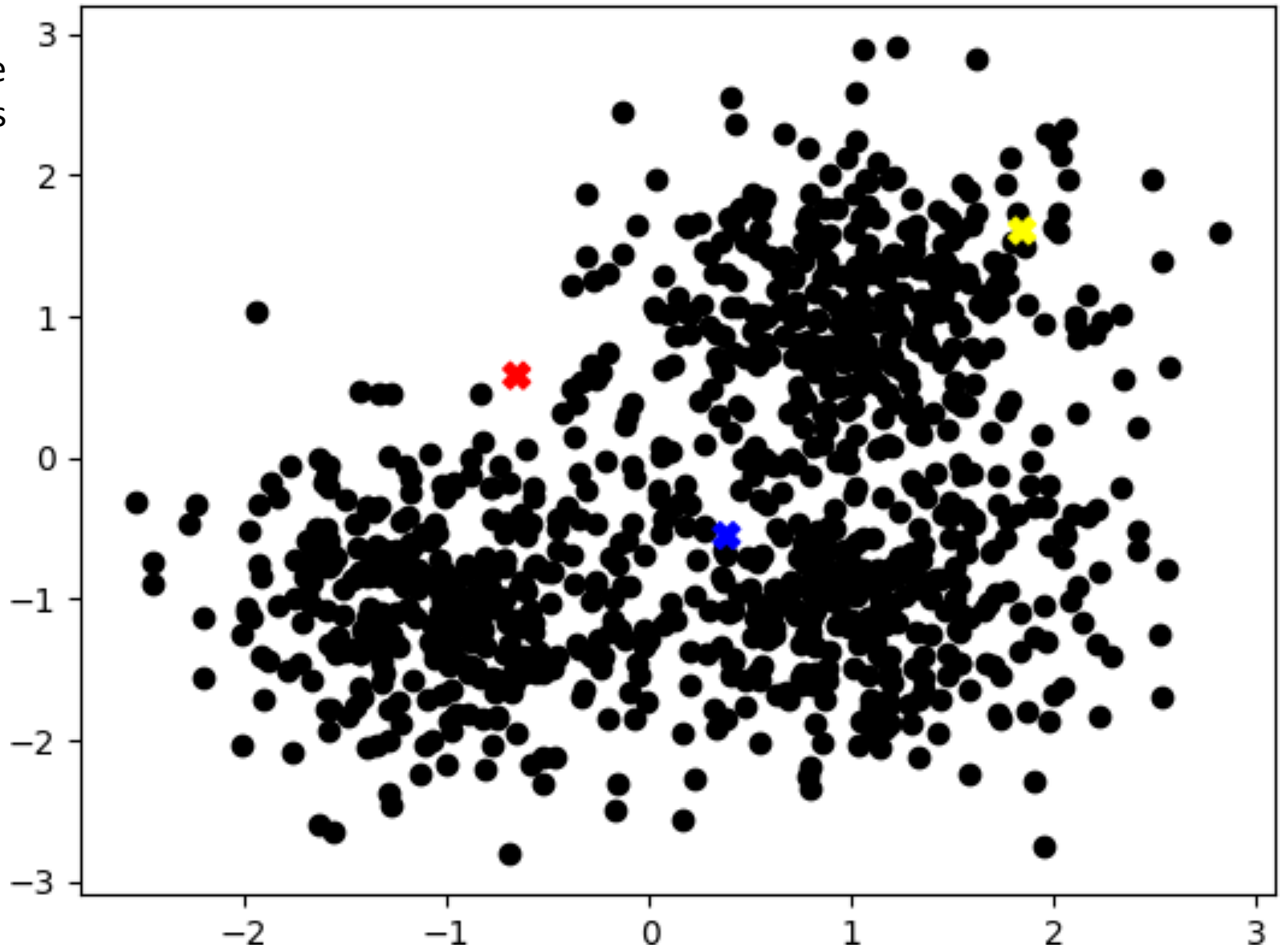


Clustering Algorithms

- A cluster is a collection of data objects
- A clustering algorithm tries to put similar objects to one another within the same cluster and dissimilar objects in other clusters
- Clustering is an unsupervised classification: The data is unlabeled
- A clustering algorithm tries to understand what kind of structure in the data: what sub-population does the data have?

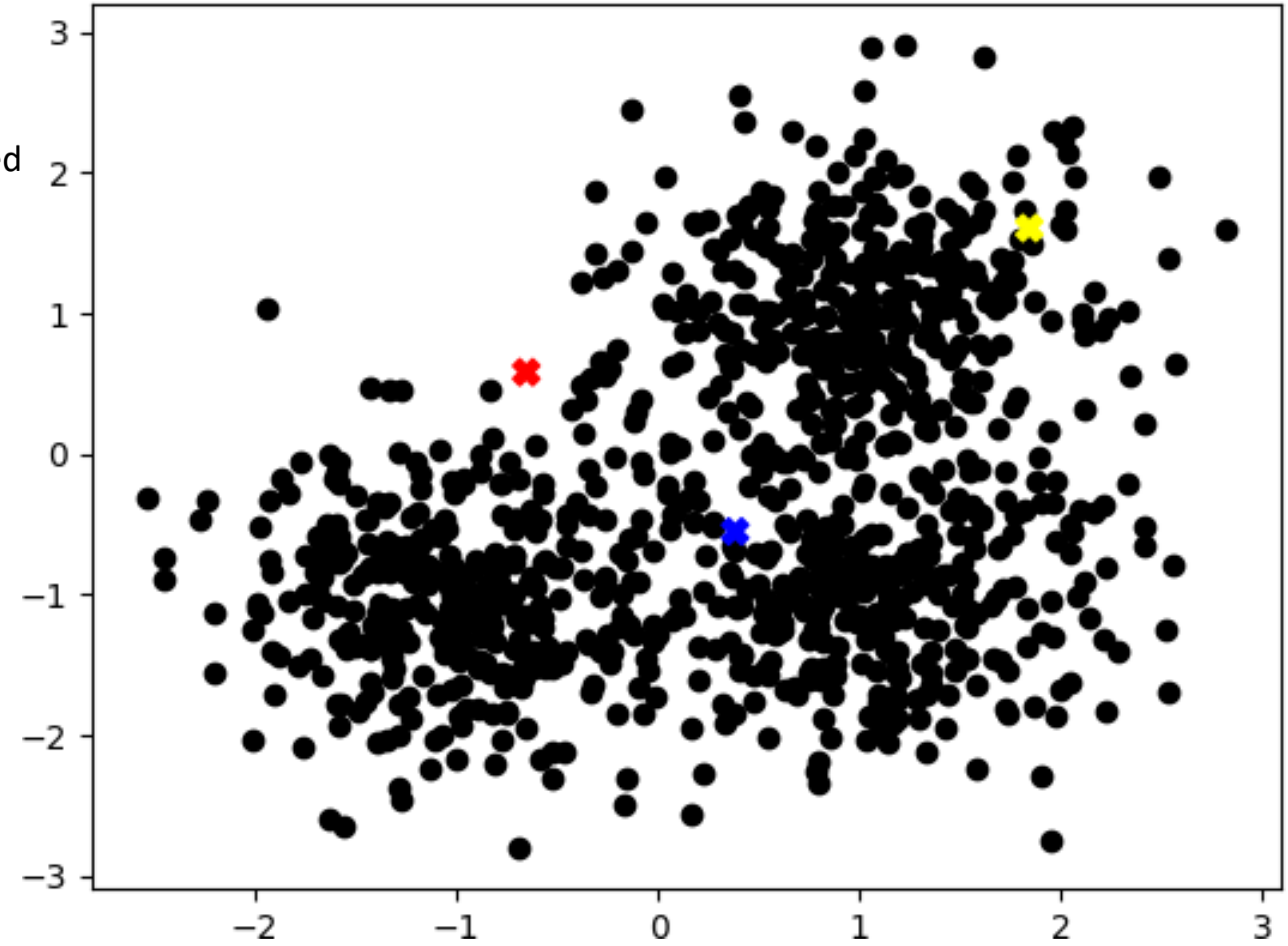
K-Means Algorithm: Example

Say that we have the following data points in the plan.



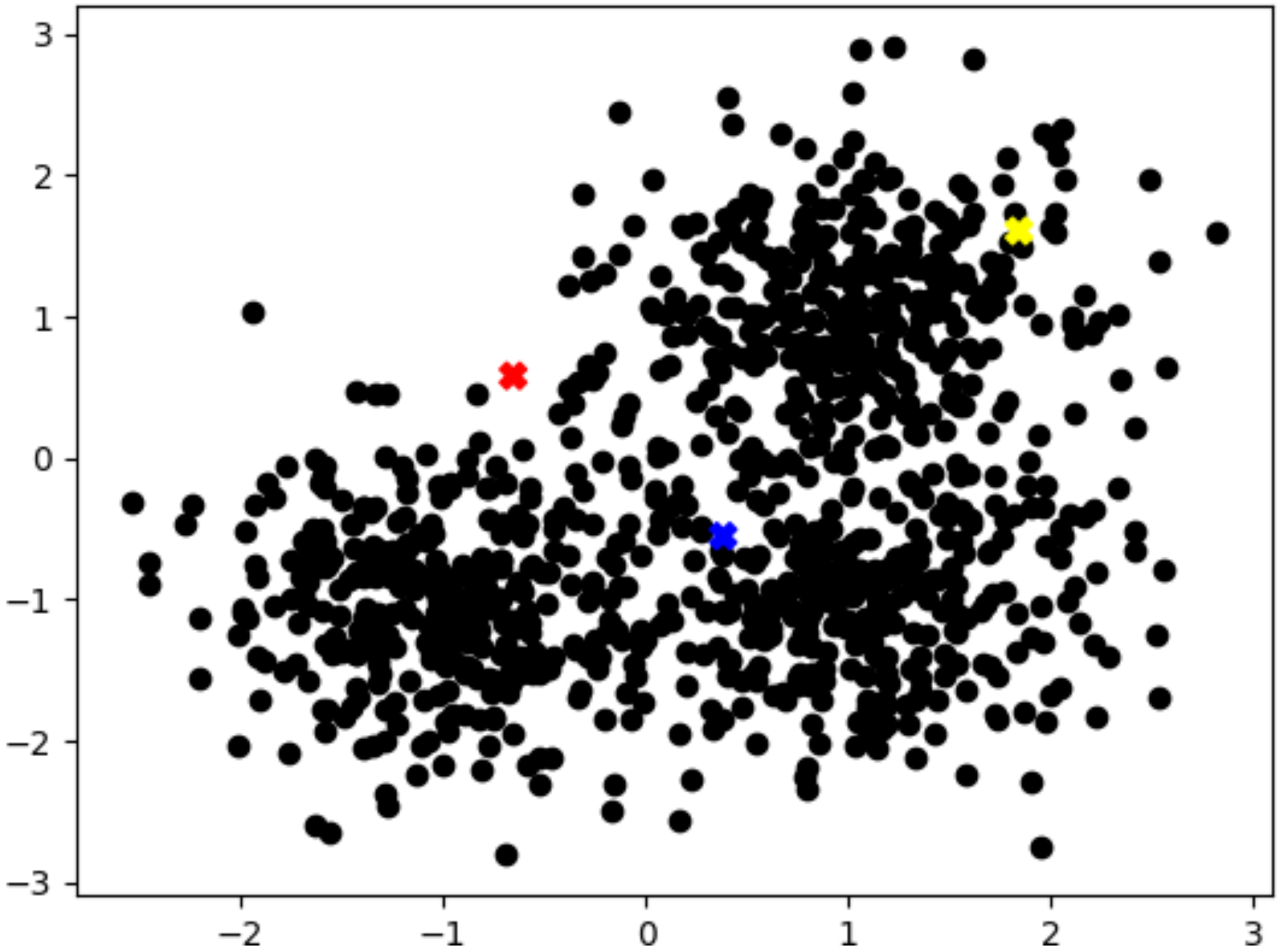
K-Means Algorithm: Example

K-means is a clustering algorithm that is best explained via an example



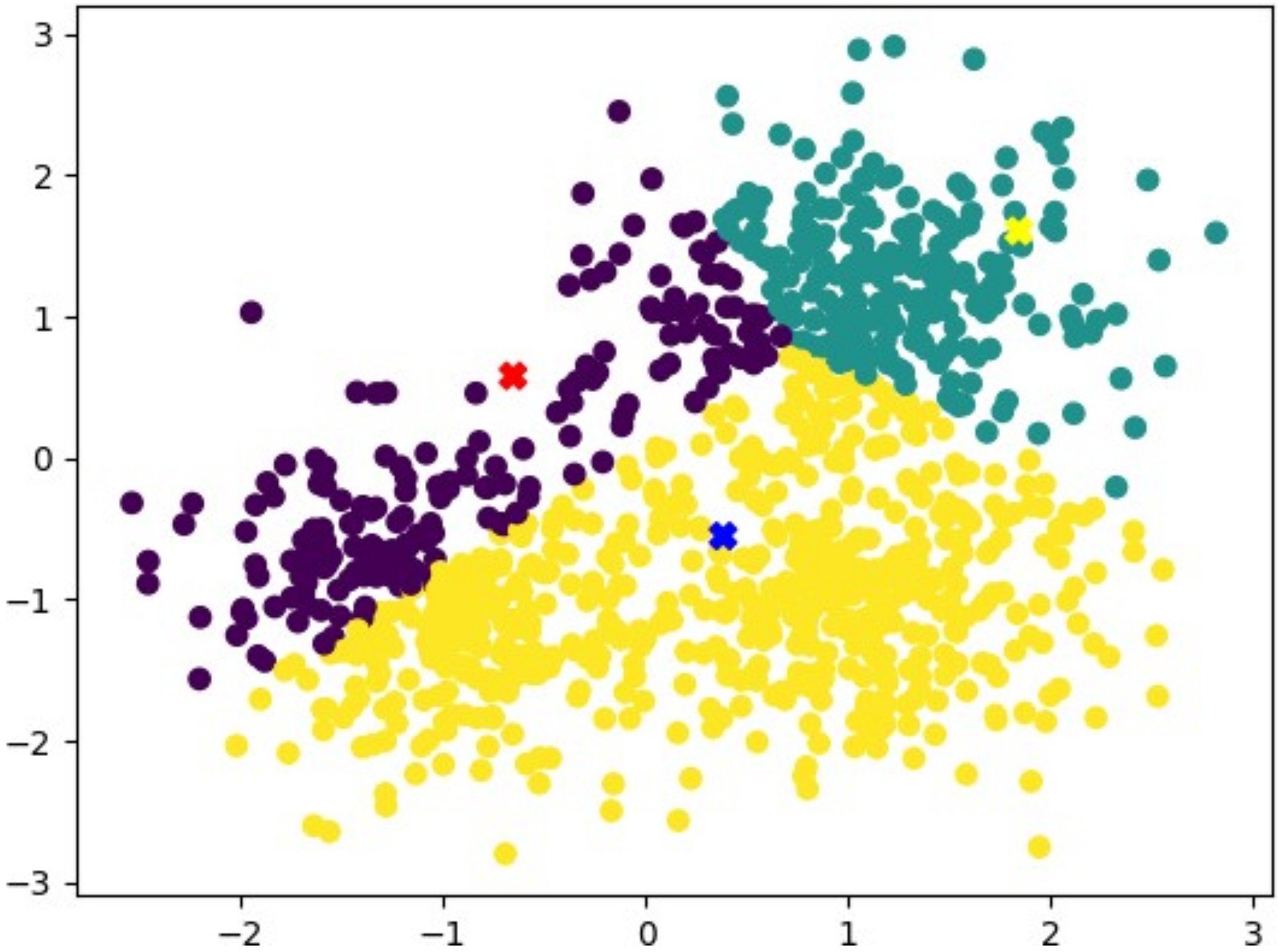
K-MEANS ALGORITHM: EXAMPLE

Choose randomly 3 centroids : $c1$, $c2$, $c3$ (the points appear in blue, red and yellow)



K-MEANS ALGORITHM: EXAMPLE

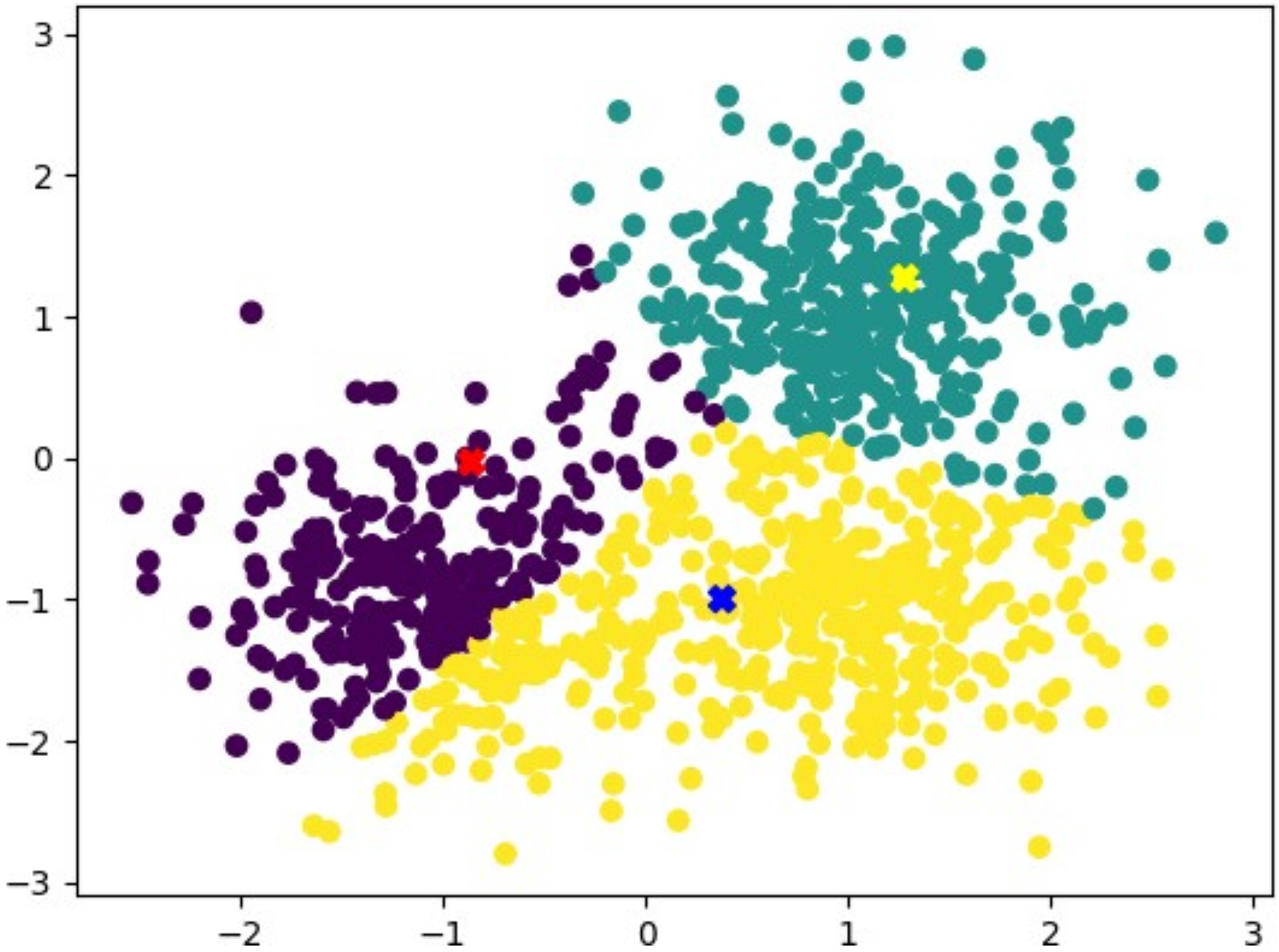
assign each point x in the set to the closest centroid



K-MEANS ALGORITHM: EXAMPLE

Update the centroids $c1$, $c2$, $c3$ as follows :

$$c_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j$$



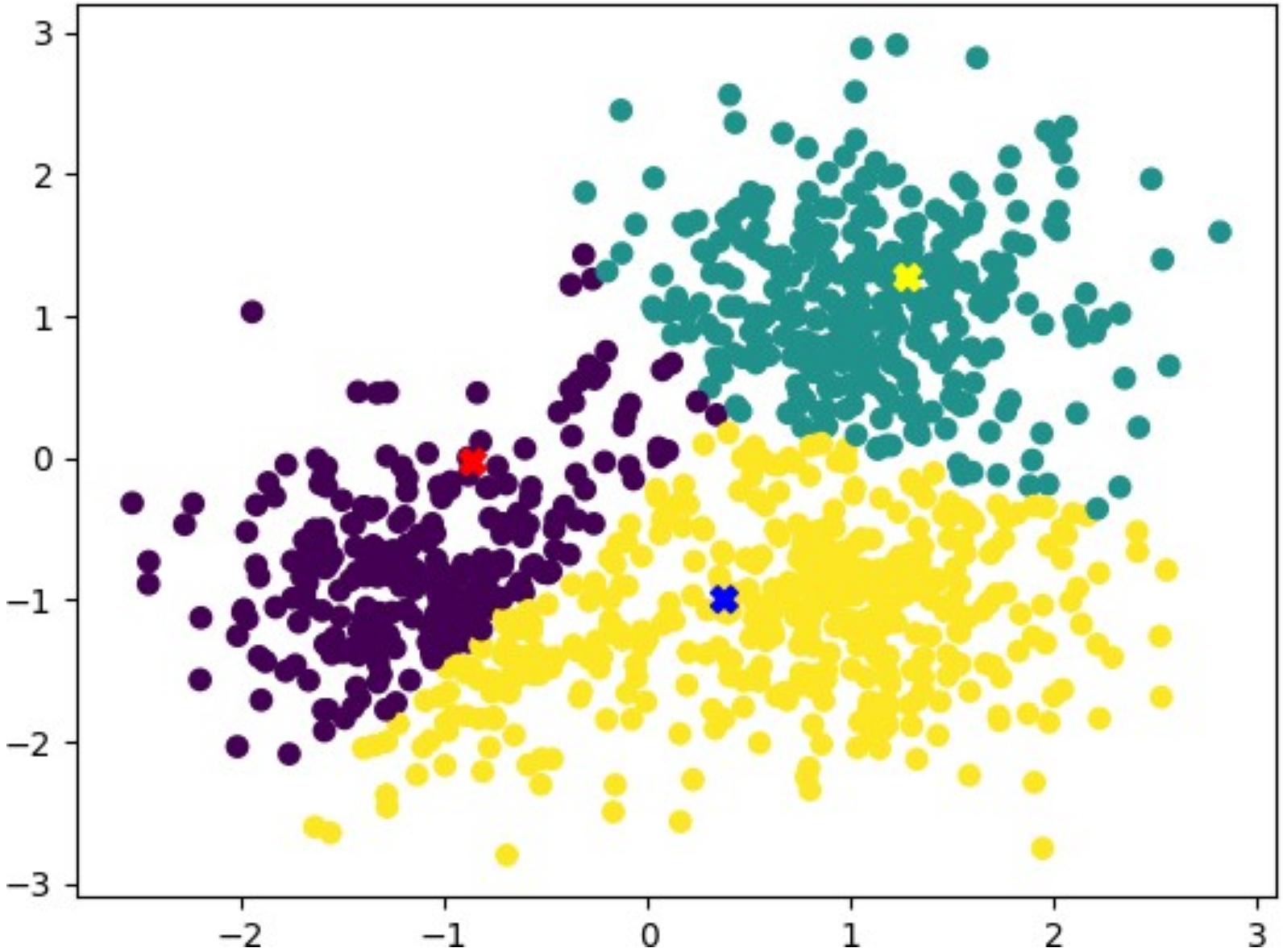
K-MEANS ALGORITHM: EXAMPLE

Update the centroids $c1$, $c2$, $c3$ as follows :

$$c_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j$$

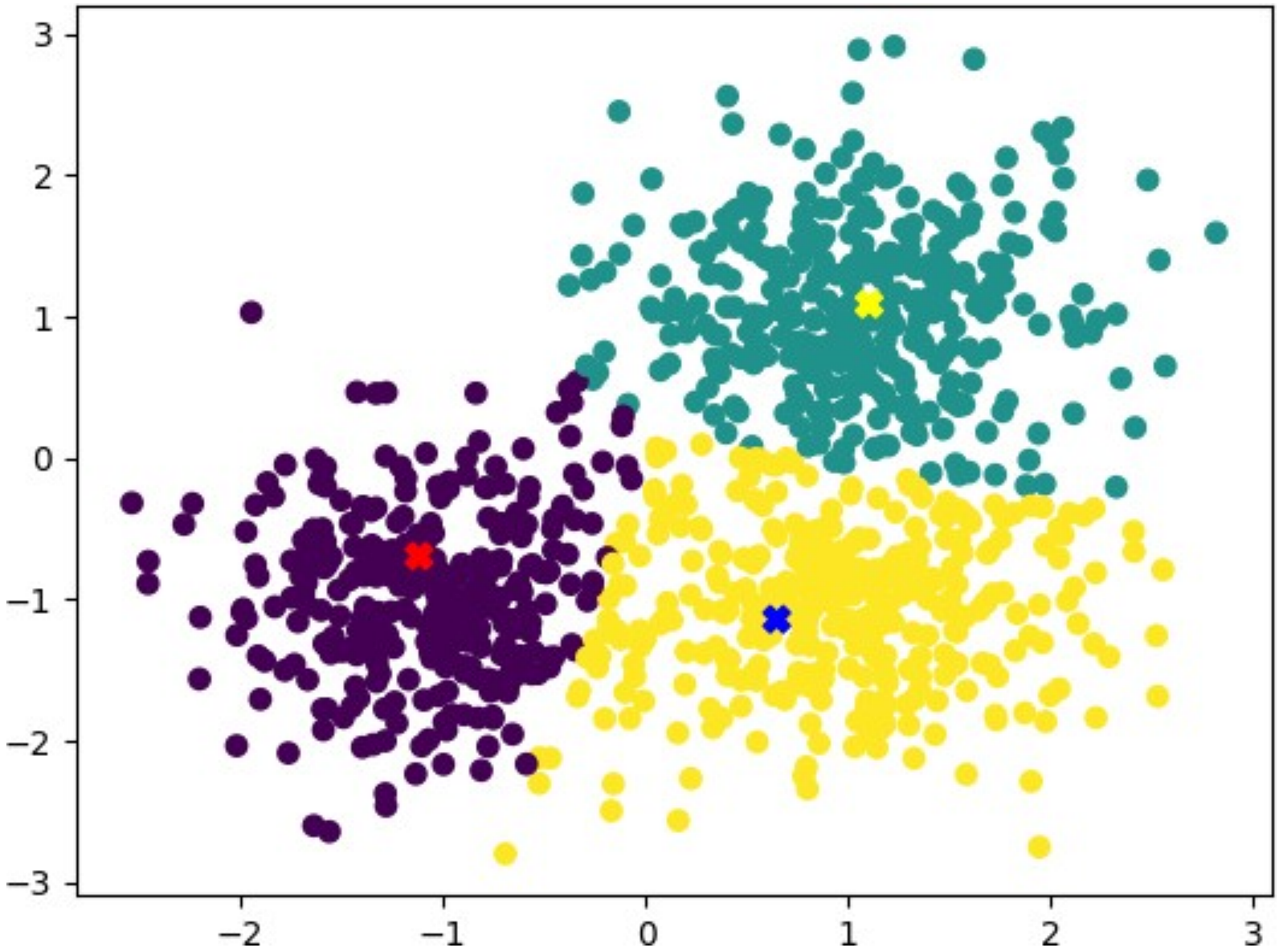
In other words, the new center is the average of the cluster members

assign each point x in the set to the closest centroid



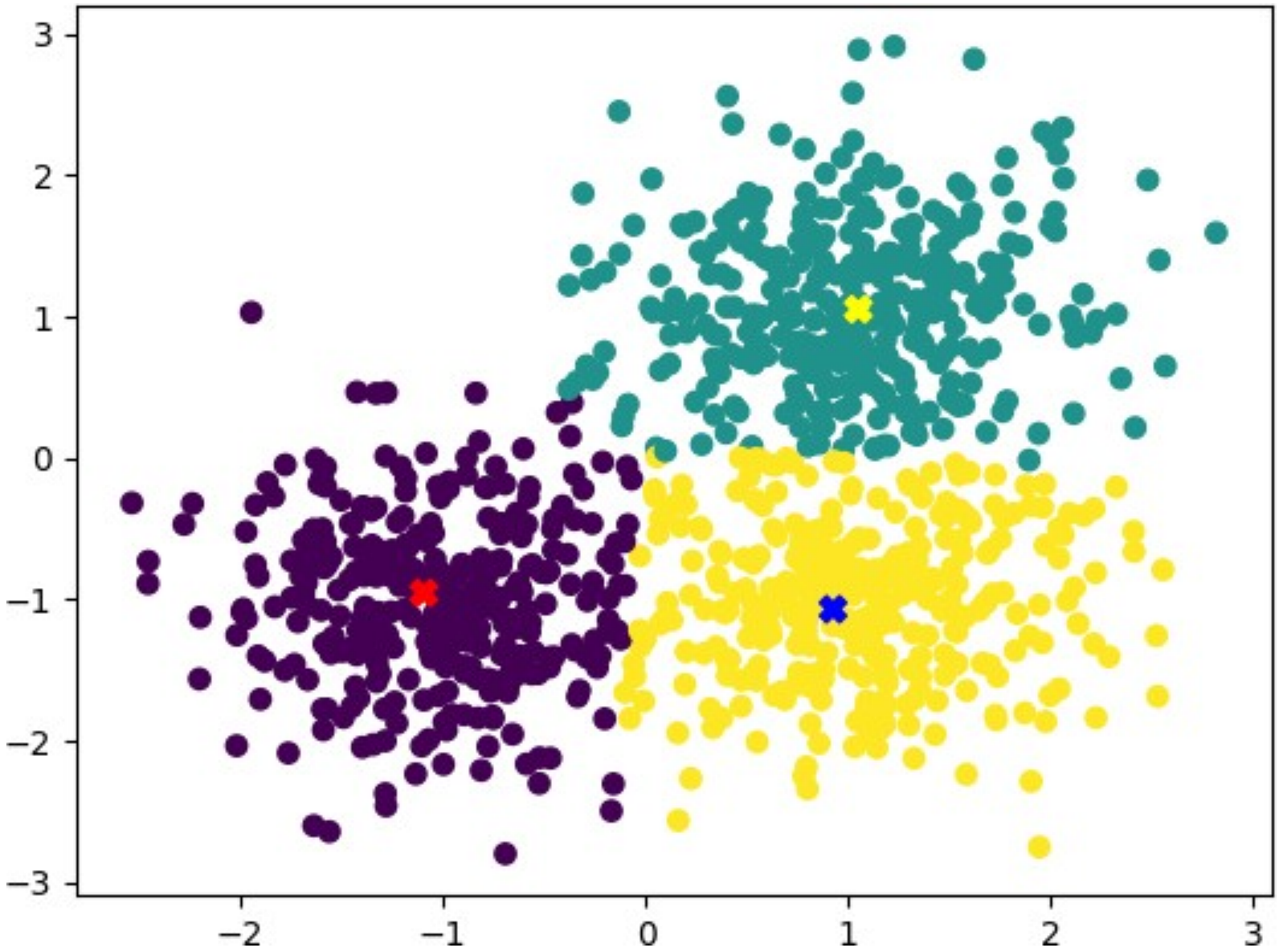
K-Means Algorithm: Example

Repeat until convergence



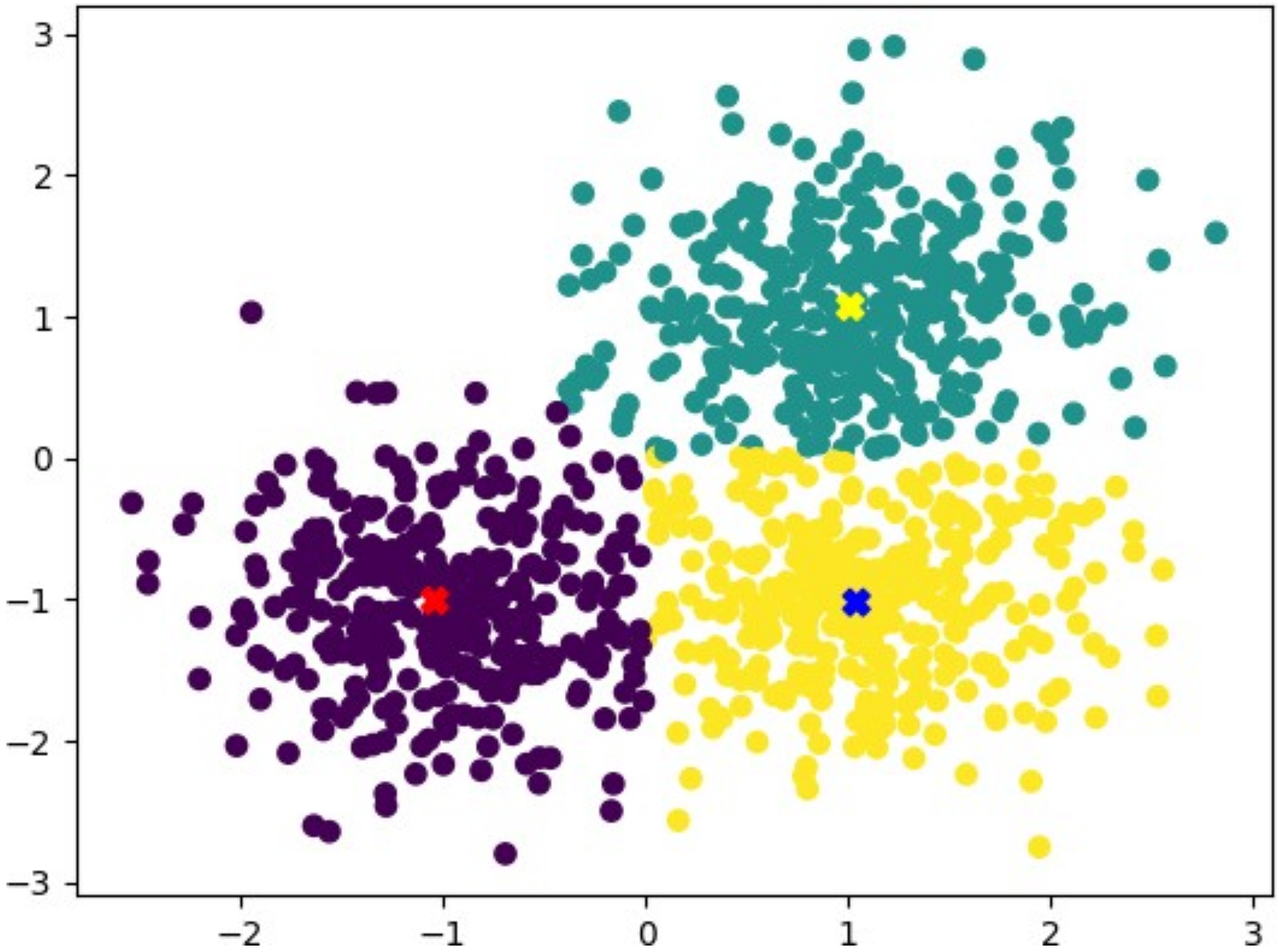
K-Means Algorithm: Example

Repeat until convergence



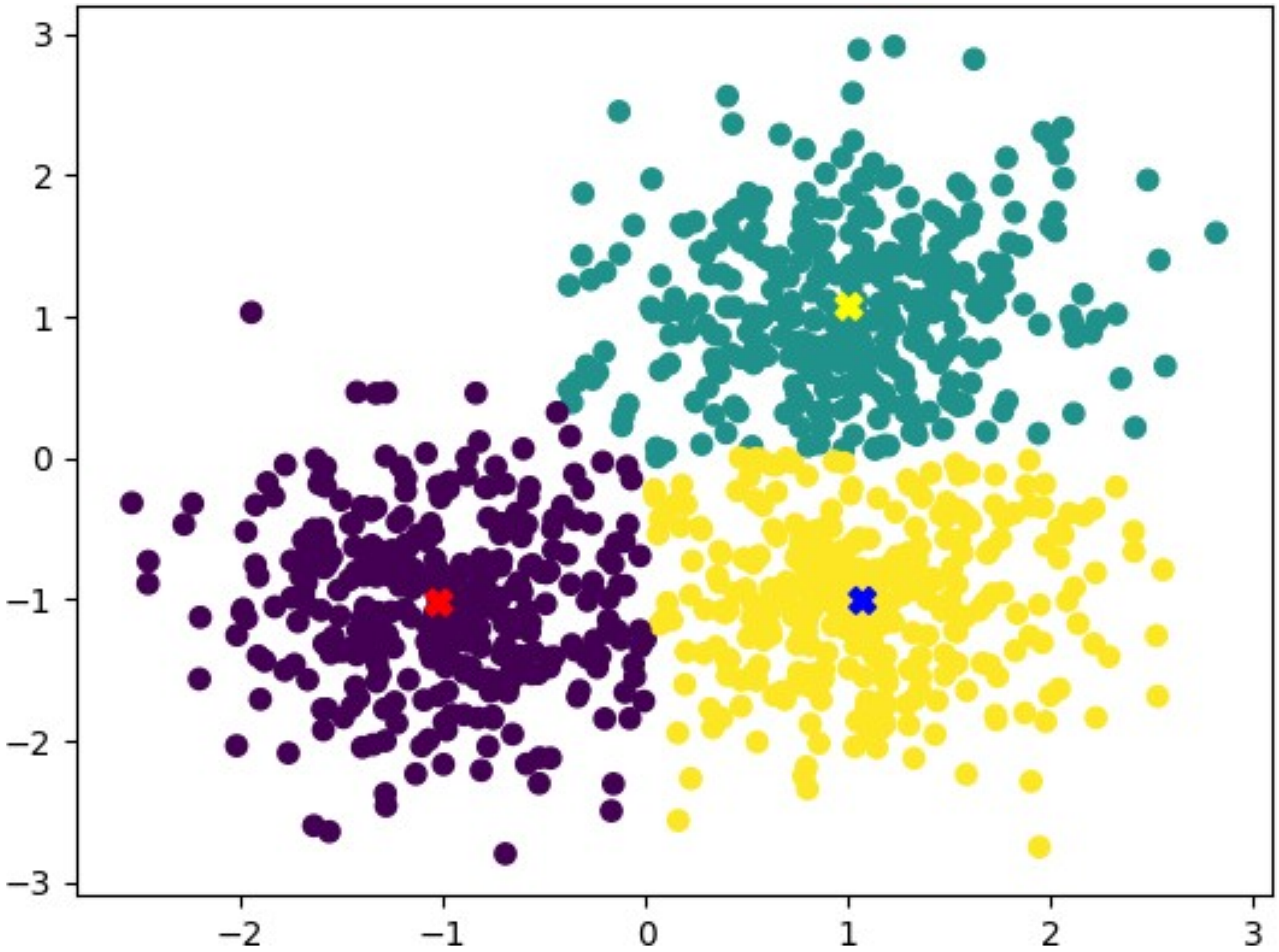
K-Means Algorithm: Example

Repeat until convergence



K-Means Algorithm: Example

Repeat until convergence



NOTATIONS

- THE n^{th} EUCLIDIAN SPACE WILL BE DENOTED BY R^n. A POINT IN R^n WILL BE DENOTED BY x. IN THIS LECTURE THE TERM DATA, OR THE TRAINING SET, X WILL MEAN A FINITE SET OF POINTS IN R^n.

- IN OTHER WORDS, $X = \{x_1, \ldots, x_m\}$ WHERE x_j $is\ a\ point\ in\ R^n$

- THE EUCLIDIAN DISTANCE BETWEEN TWO POINTS x AND y IN R^n WILL BE DENOTED BY $d(x, y)$.

K-MEANS ALGORITHM

- THE K-MEANS ALGORITHM TAKES TWO INPUTS:
  - A parameter K, which is the number of clusters one wants to find in the data.
  - The training set X of the points. $Here\ X = \{x_1, \dots, x_m\}$ where $x_j\ is\ a\ point\ in\ R^n$.

- THE ALGORITHM RETURNS A LABEL PARTITIONING X INTO K-CLUSTERS.

K-MEANS ALGORITHM

- CHOOSE RANDOMLY K CENTROIDS : $c_1, c_2, \dots, c_K$ IN R^n
- REPEAT UNTIL CONVERGENCE
  - We assign each point x in the set to the closest centroid. ← Cluster assignment step
  - Update the centroids $c_1, c_2, \dots, c_K$ as follows: ← Centroid move step

$$c_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j$$

    - Here S_i is the cluster associated with the centroid c_i

  - Convergence:
    - None of the cluster assignments change
    - Centroids do not change

K-MEANS ALGORITHM

- CHOOSE RANDOMLY K CENTROIDS : $c_1, c_2, \ldots, c_K$ IN R^n
- REPEAT UNTIL CONVERGENCE
  - We assign each point x in the set to the closest centroid. ← Cluster assignment step
  - Update the centroids $c_1, c_2, \ldots, c_K$ as follows: ← Centroid move step

$$c_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j$$

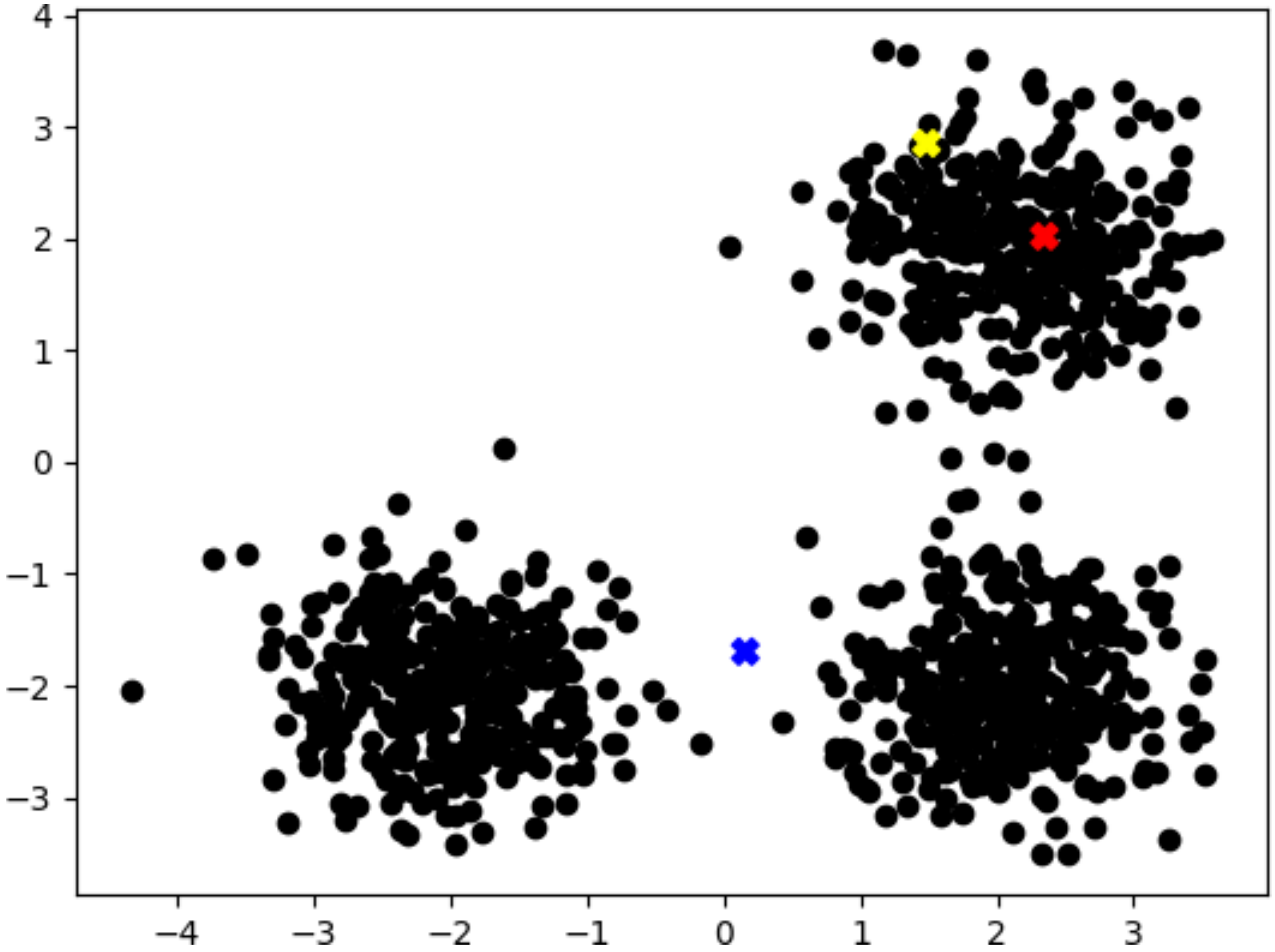
    - Here S_i is the cluster associated with the centroid c_i

This is where *mean* comes from

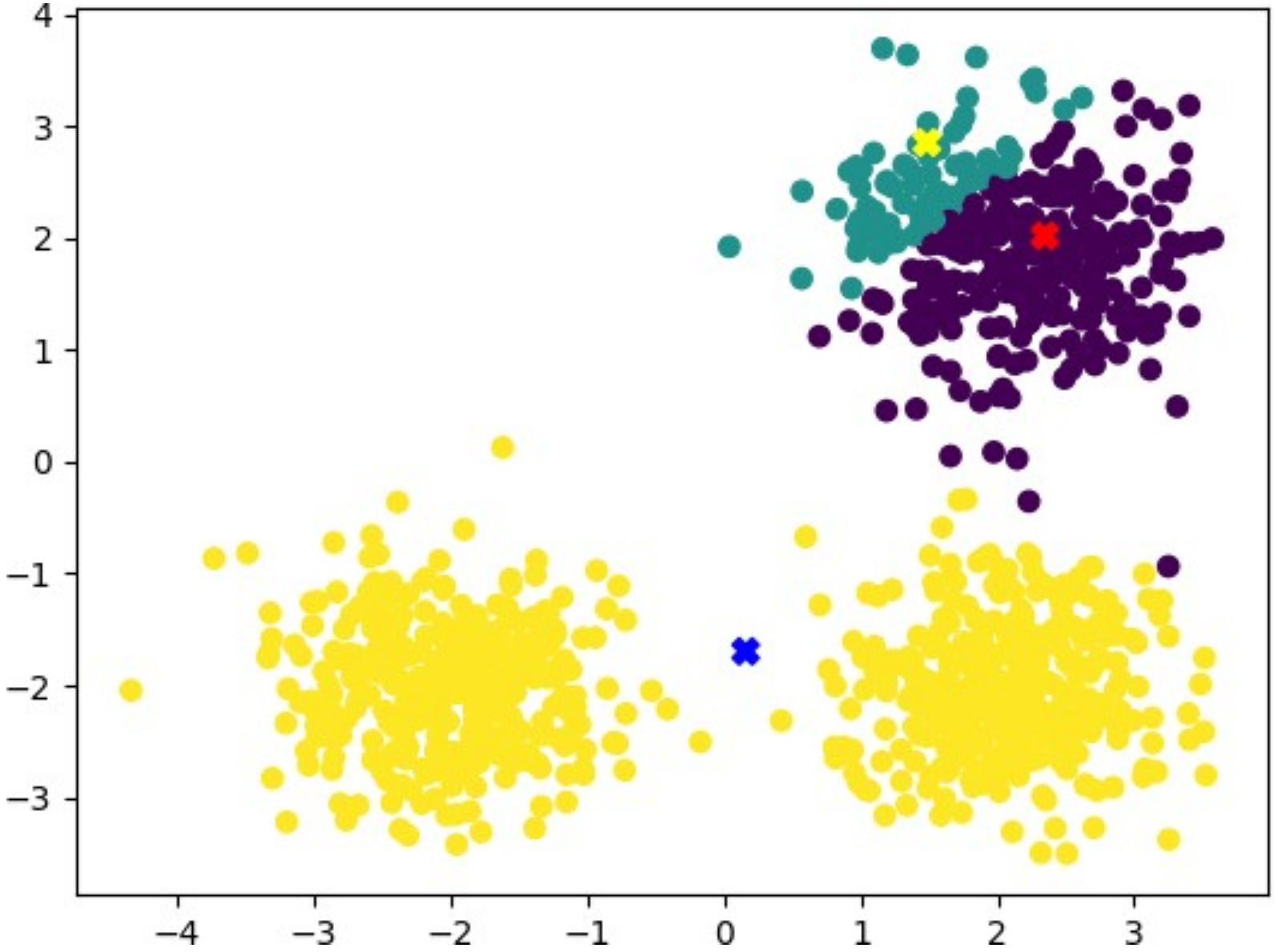
  - Convergence:
    - None of the cluster assignments change
    - Centroids do not change

K-MEANS ALGORITHM: EXAMPLE

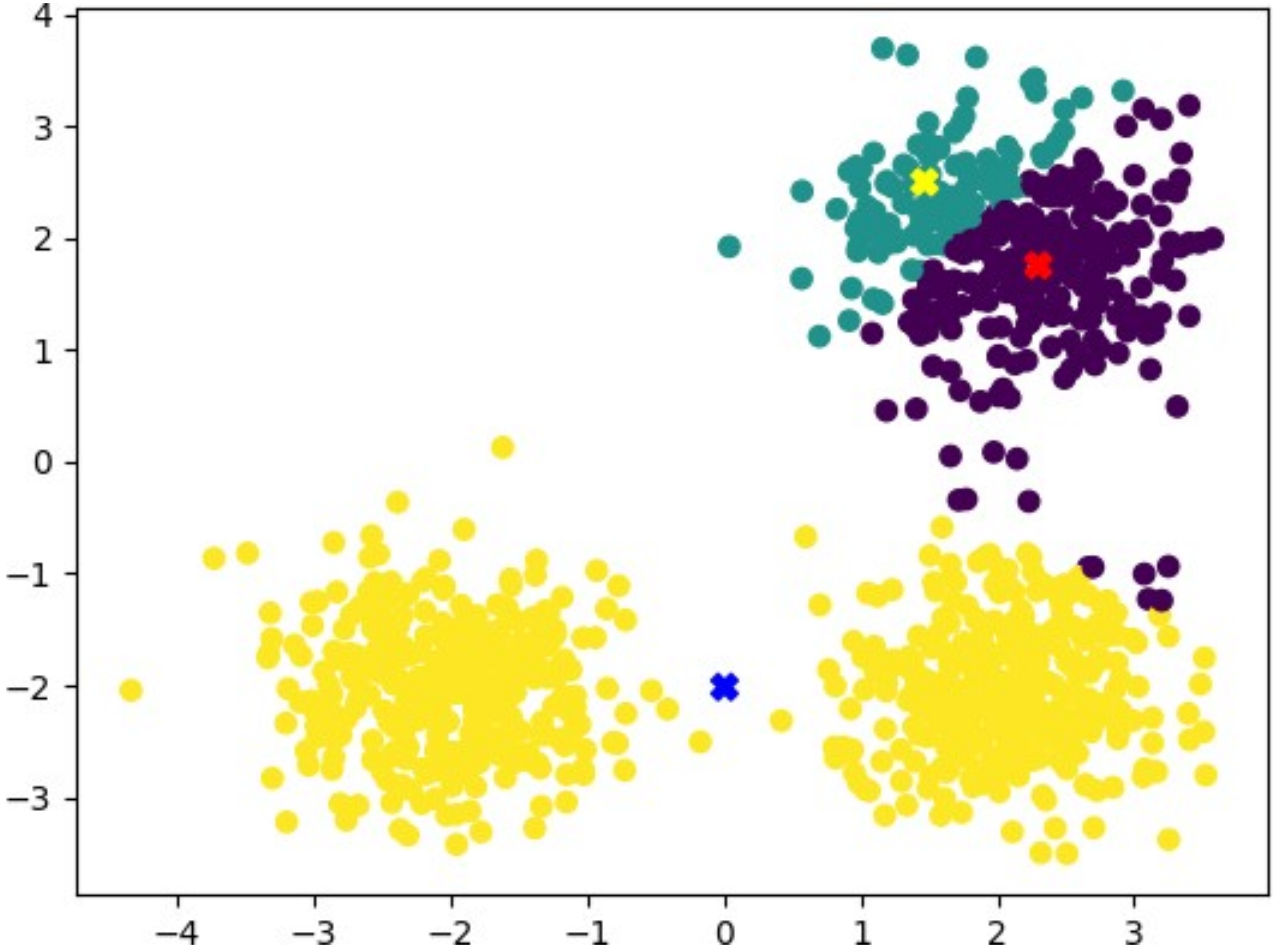
Lets consider another example



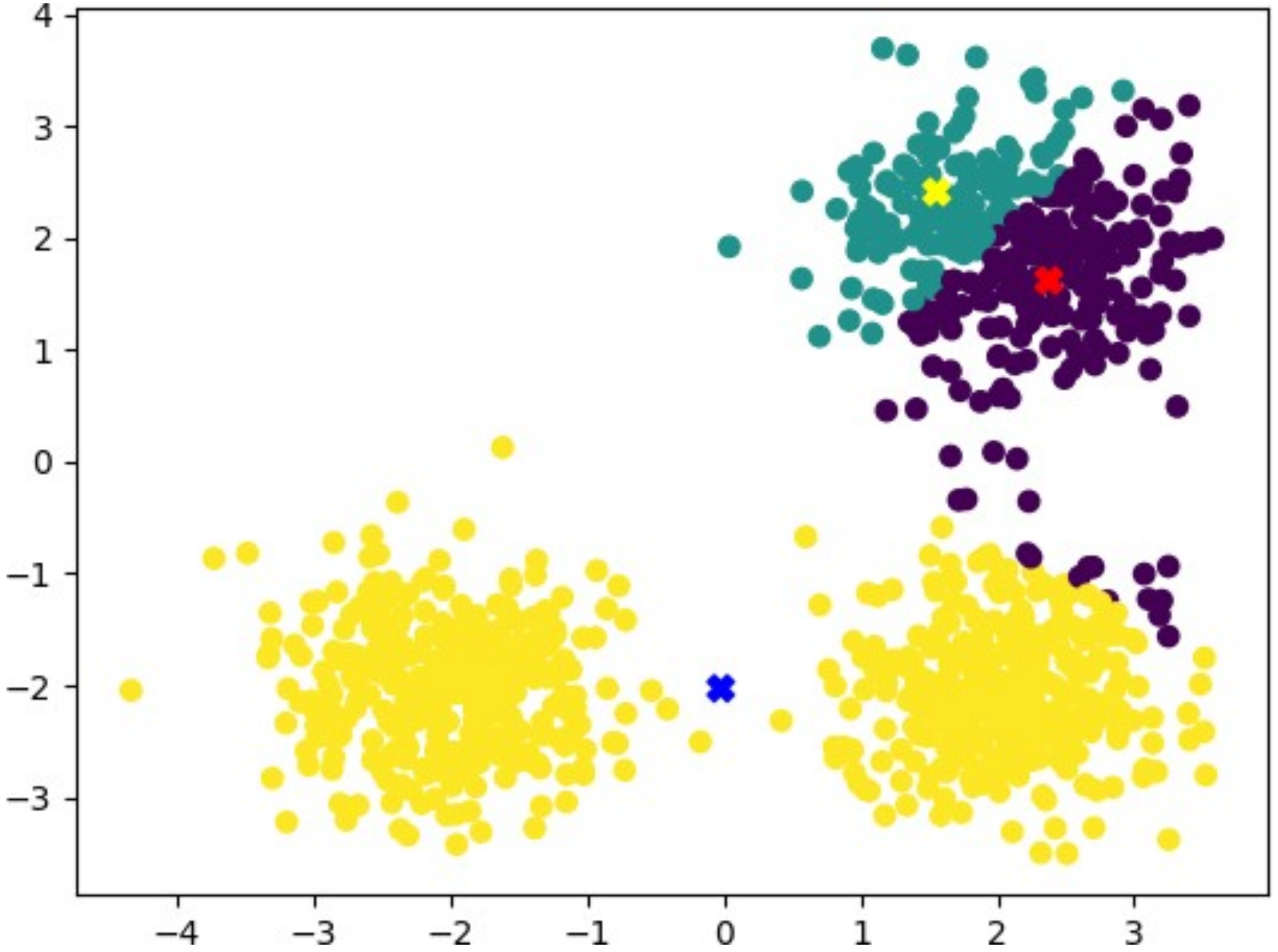
K-MEANS ALGORITHM: EXAMPLE



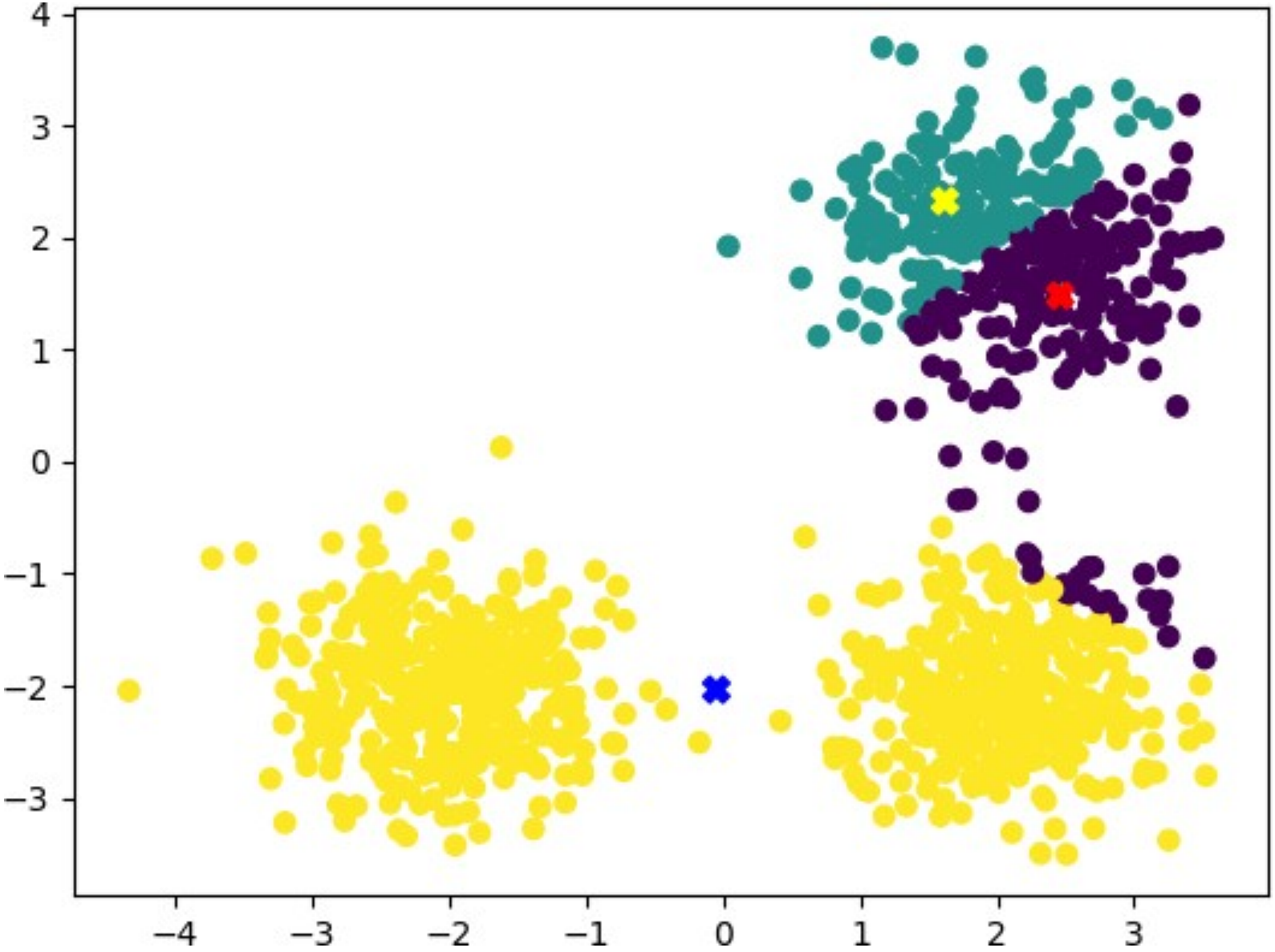
K-MEANS ALGORITHM: EXAMPLE



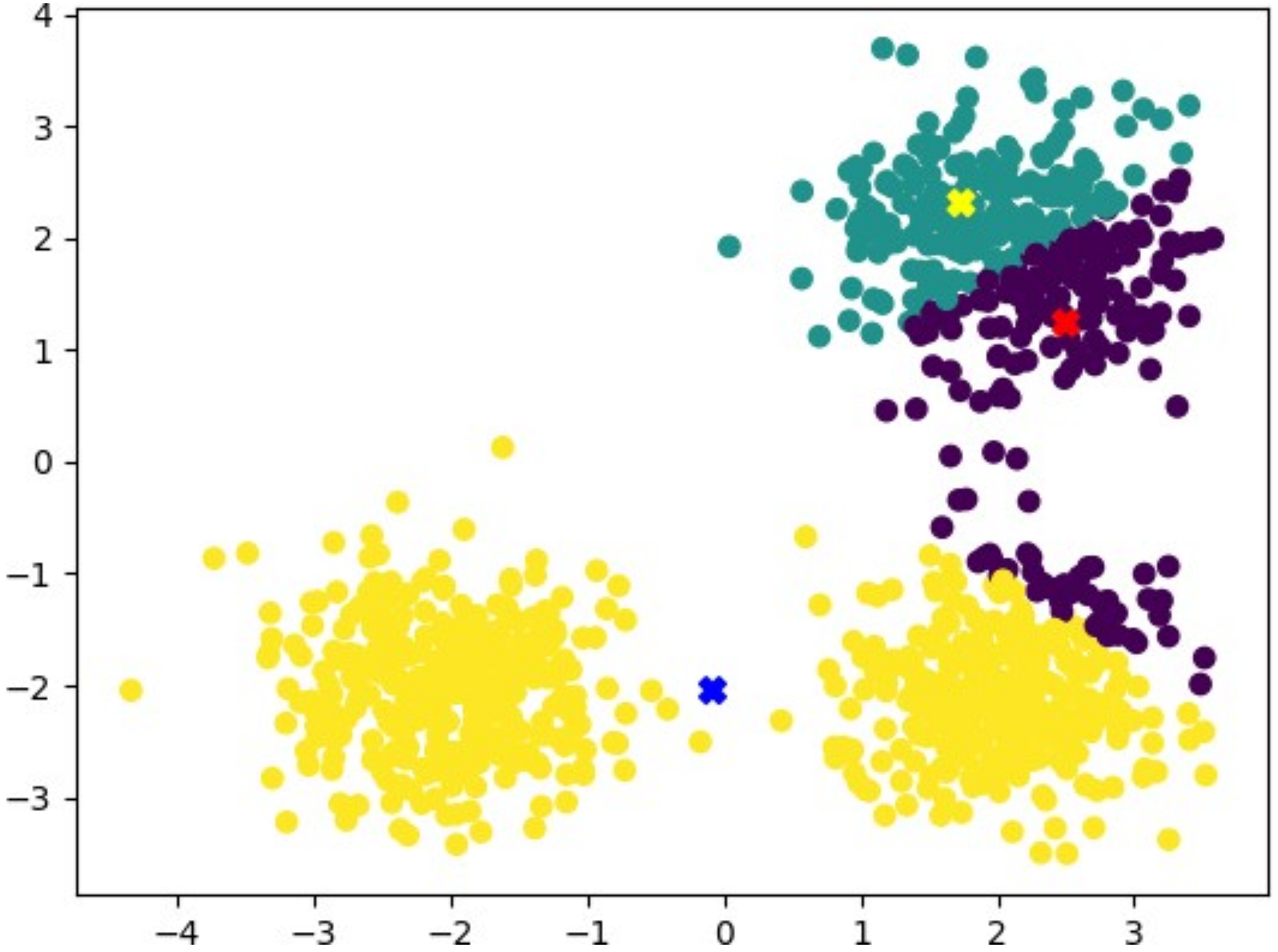
K-MEANS ALGORITHM: EXAMPLE



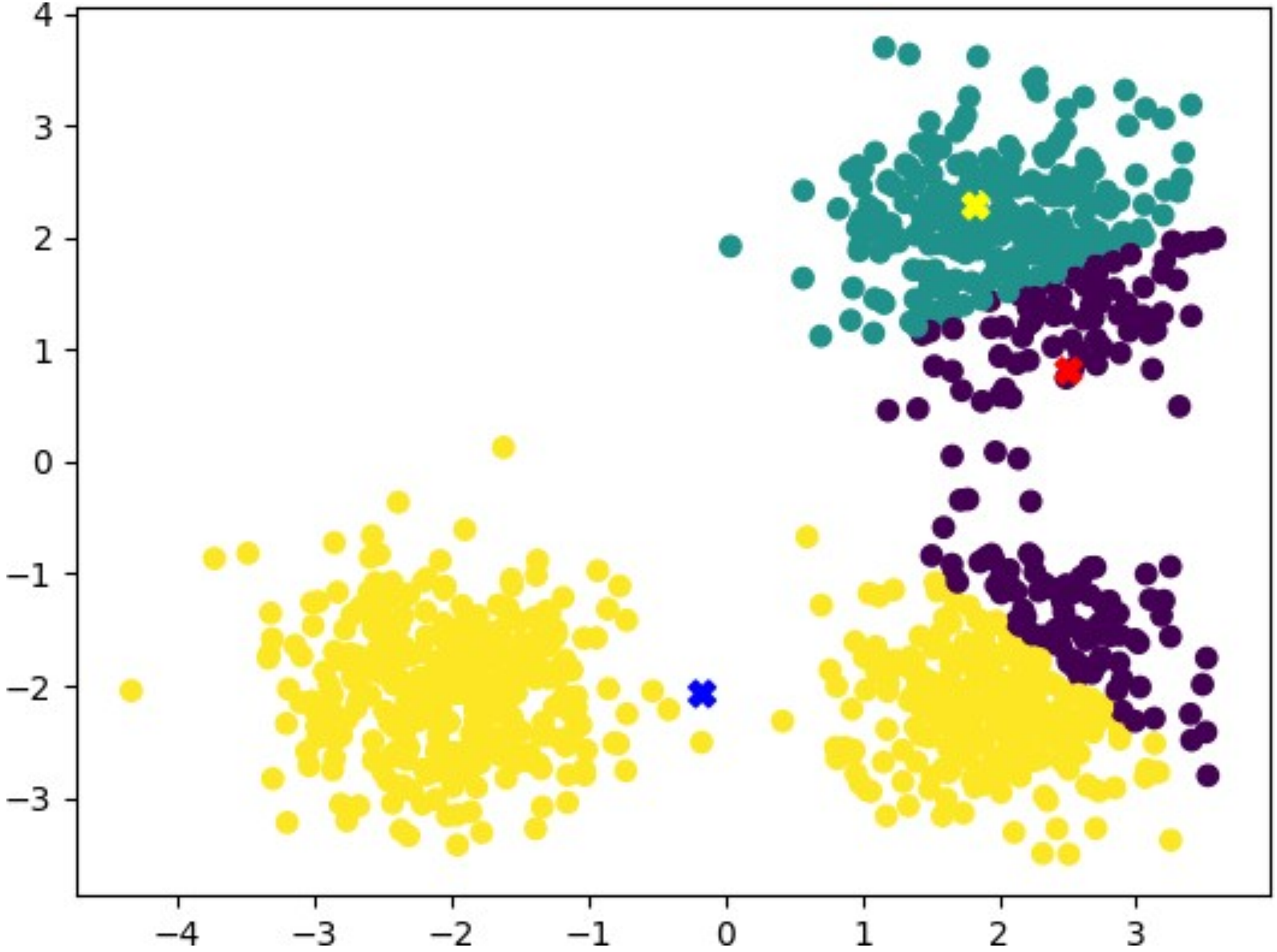
K-MEANS ALGORITHM: EXAMPLE



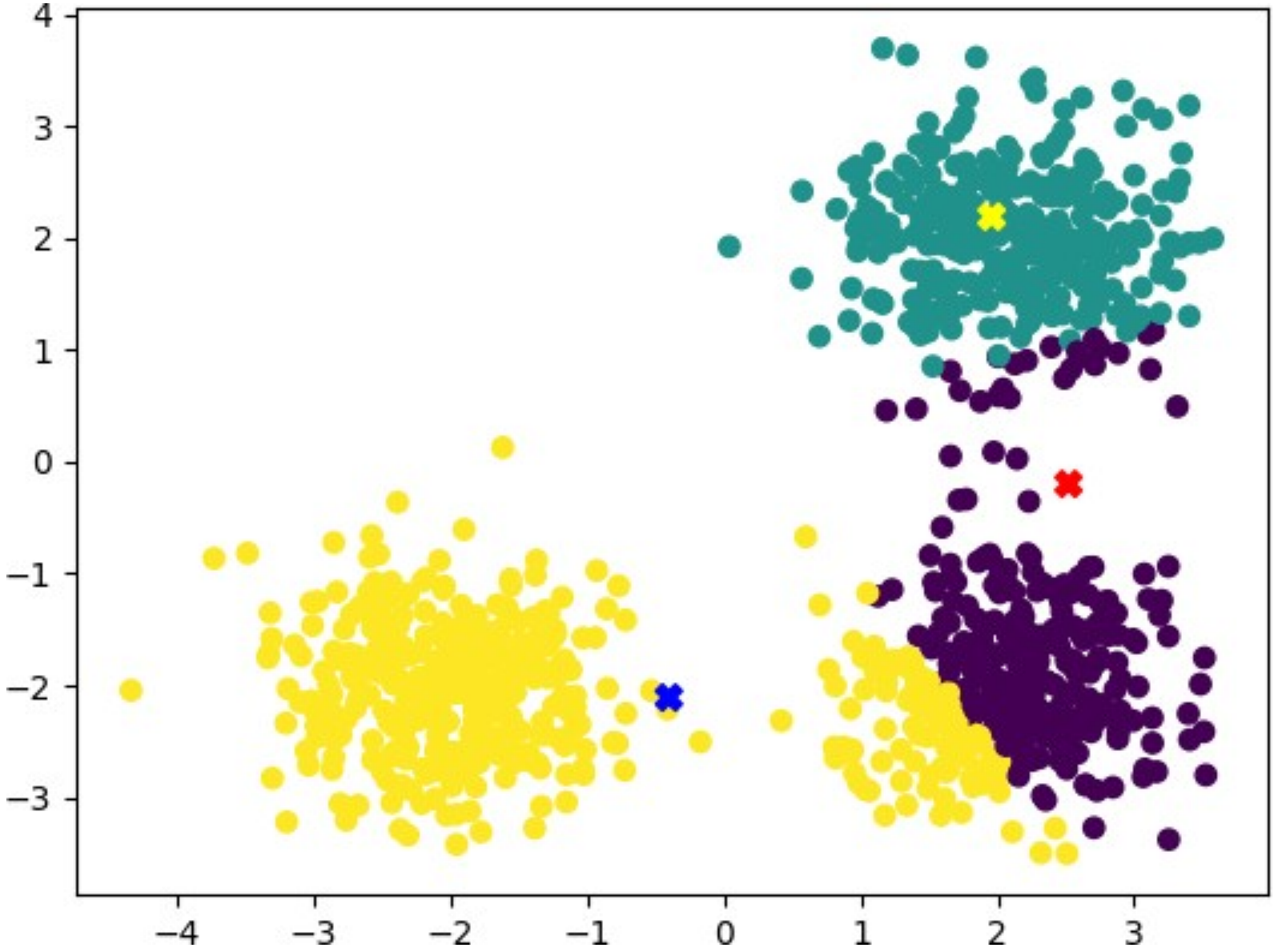
K-MEANS ALGORITHM: EXAMPLE



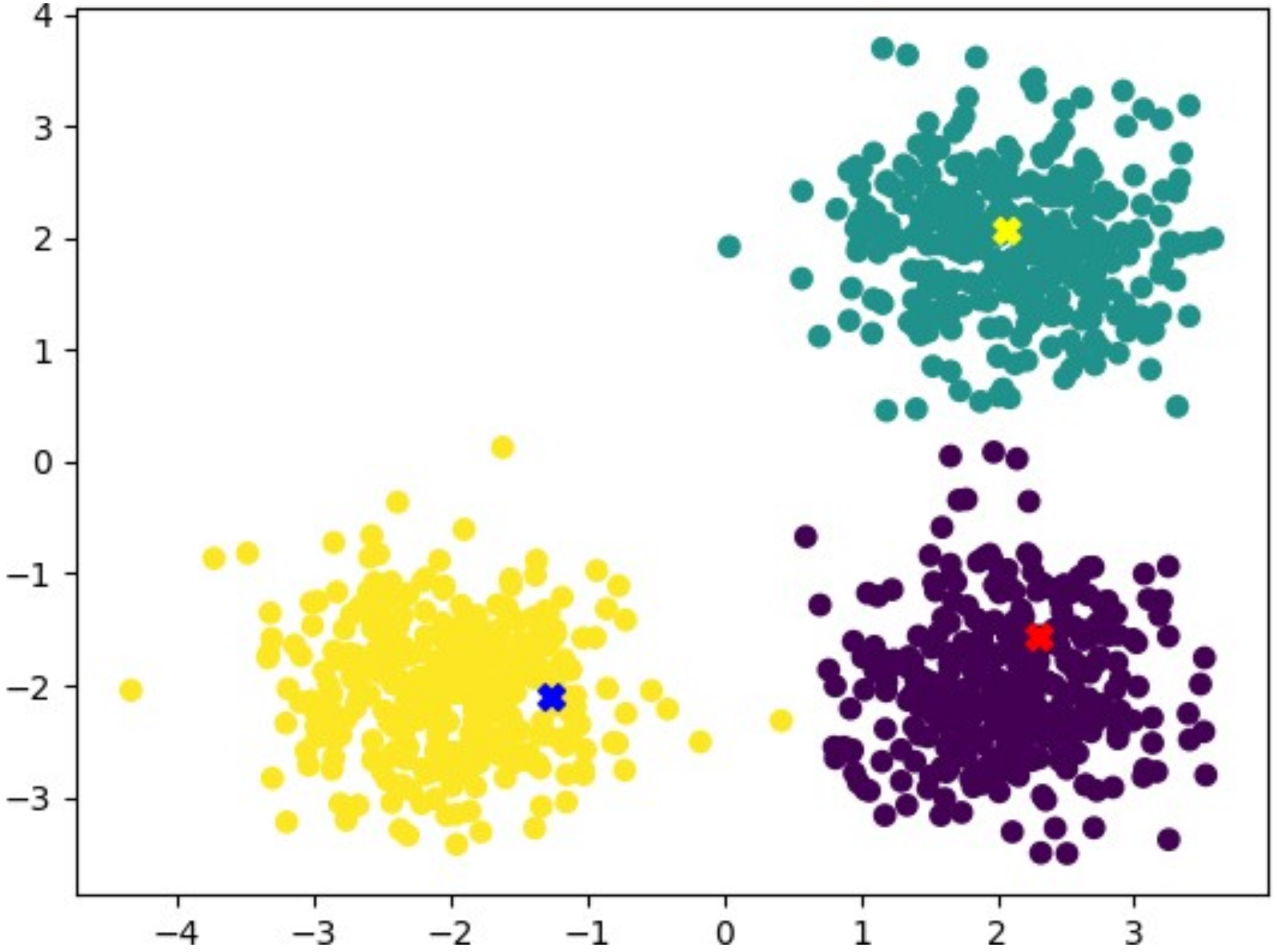
K-Means Algorithm: Example



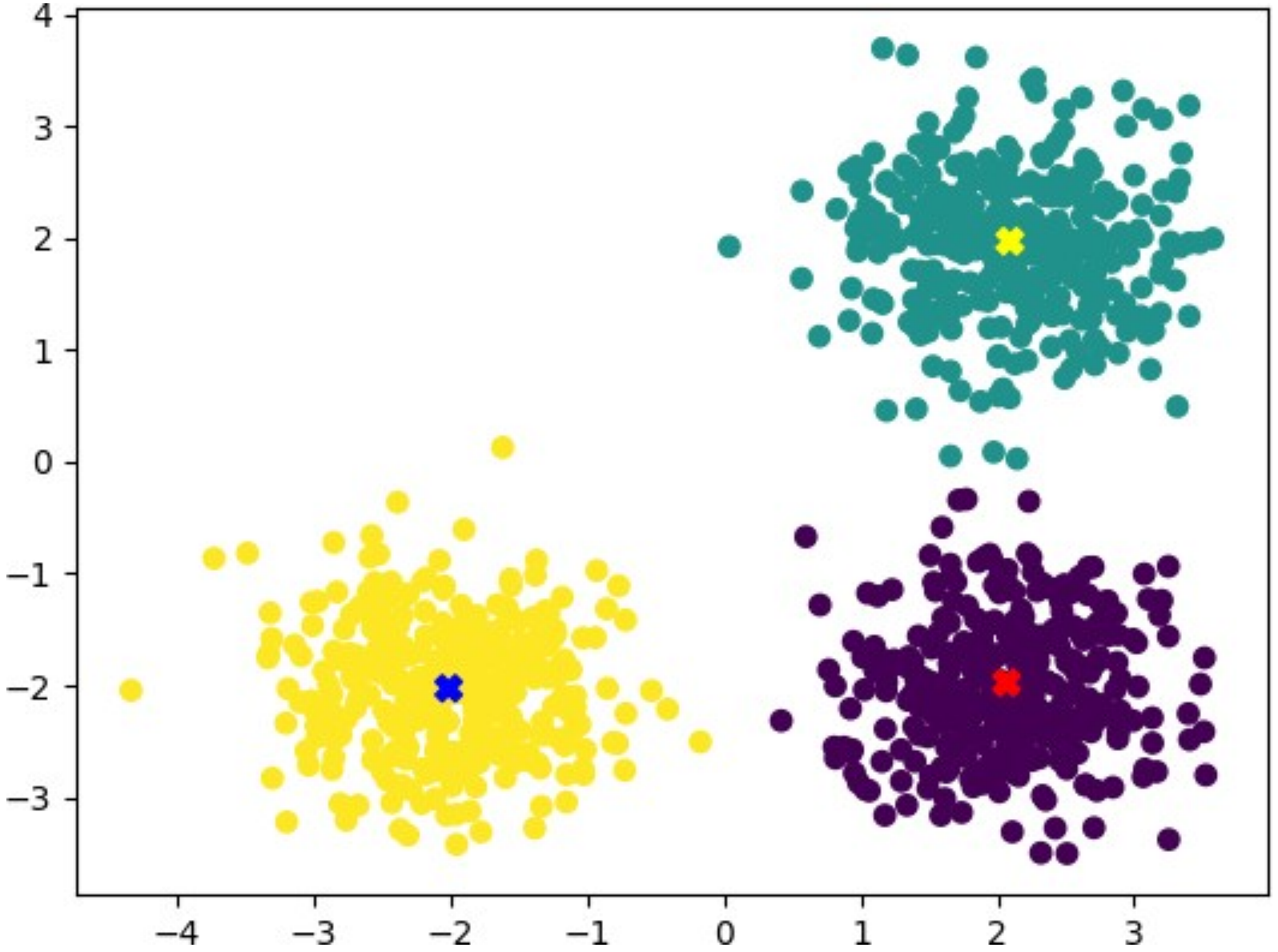
K-MEANS ALGORITHM: EXAMPLE



K-MEANS ALGORITHM: EXAMPLE



K-MEANS ALGORITHM: EXAMPLE



K-MEANS ALGORITHM: COMPLEXITY

- CHOOSE RANDOMLY K CENTROIDS : $c_1, c_2, \ldots, c_K$ IN R^n
- REPEAT UNTIL CONVERGENCE
  - We assign each point x in the set to the closest centroid.
  - Update the centroids $c_1, c_2, \ldots, c_K$ as follows:

$$c_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j$$

    - Here S_i is the cluster associated with the centroid c_i

K-Means Algorithm: Complexity

- Complexity is $O(m * K * I * n)$
  - m = number of points in the data set
  - K = number of clusters
  - I = number of iterations
  - n = number of attributes=number of features= the dimension of the space R^n

- How can we do better?

K-MEANS ALGORITHM: CONVERGENCE

- WHAT EXACTLY IS THE OPTIMIZATION FUNCTION OF THIS ALGORITHM ?
  - Cost function of K-means: $\sum_i \sum_{x \in S_i} d(c_i, x)$
- *THIS IS THE TOTAL SQUARED DISTANCE FROM THE CENTROID TO THE POINTS OF THE CLUSTER ASSOCIATED TO THE CENTROID*

K-MEANS ALGORITHM: CONVERGENCE

- SINCE WE START WITH RANDOM CENTERS EVERY TIME WE RUN THIS ALGORITHM:
  - Is it guaranteed to give the same clustering configuration?
  - Is the algorithm guaranteed to converge ?

K-Means Algorithm: convergence

- The algorithm converges (in the sense that each iteration minimizes the cost function above) but it converges to a local min.

- Which means that:
  - (1) the solution might not be the optimal solution and
  - (2) one might get different results for different initial starts.

K-MEANS ALGORITHM: CONVERGENCE

- WHAT EXACTLY IS THE OPTIMIZATION FUNCTION OF THIS ALGORITHM ?
  - Cost function of K-means: $\sum_i \sum_{x \in S_i} d(c_i, x)$
- *THIS IS THE TOTAL SQUARED DISTANCE FROM THE CENTROID TO THE POINTS OF THE CLUSTER ASSOCIATED TO THE CENTROID*

Choose randomly k centroids : $c_1, c_2, \dots, c_K$ in R^n
Repeat until convergence
    Assign each point x to the closest centroid
    Update the centroids $c_1, c_2, \dots, c_K$

Minimize the cost function with respect to the clusters

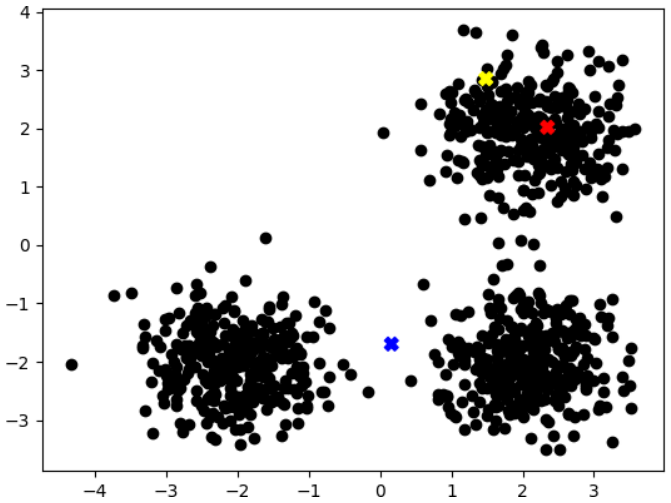
Minimize the cost function with respect to the centroids

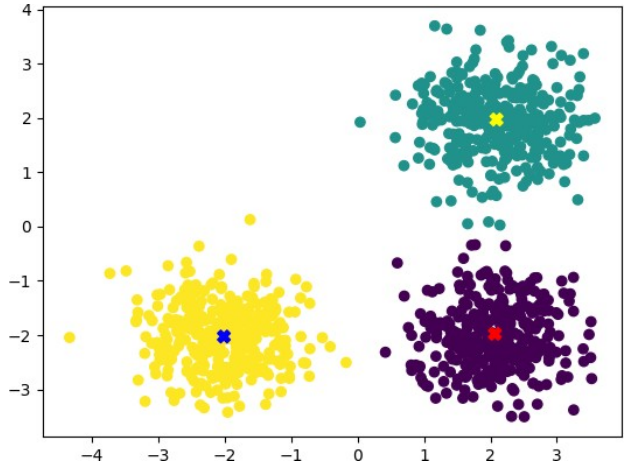
K-Means Algorithm: problems

- Outliers
- Clusters with different densities
- Non-convex shapes
- Clusters with different sizes
- May converge to local optimum

sklearn example HERE

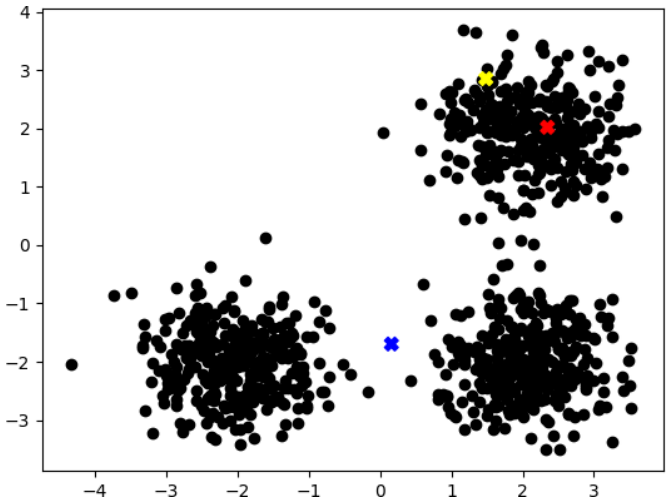
K-MEANS ALGORITHM: REMARKS

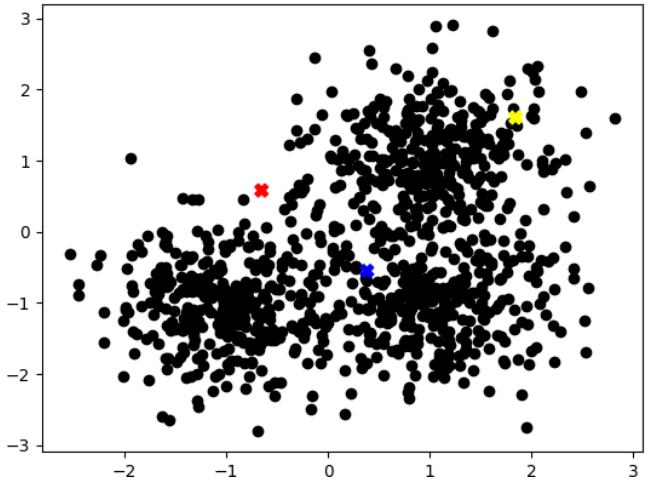


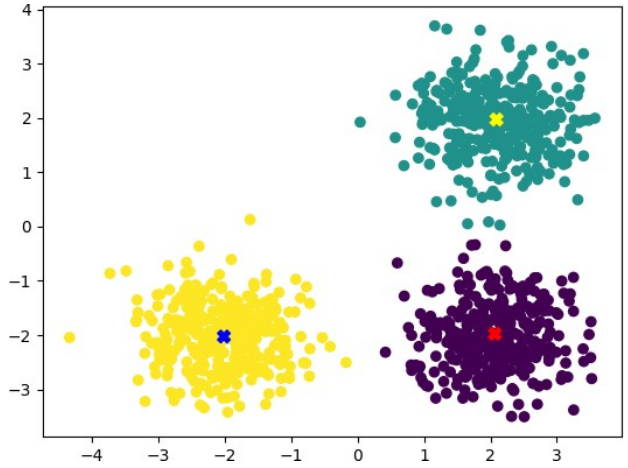


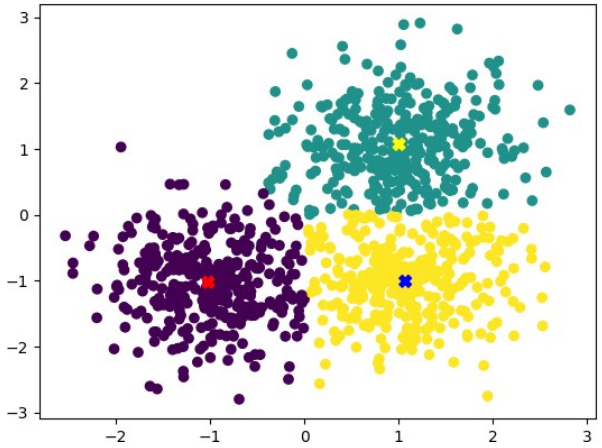
K-means is clearly applicable to data sets where the clusters are very well-separated

K-MEANS ALGORITHM: REMARKS





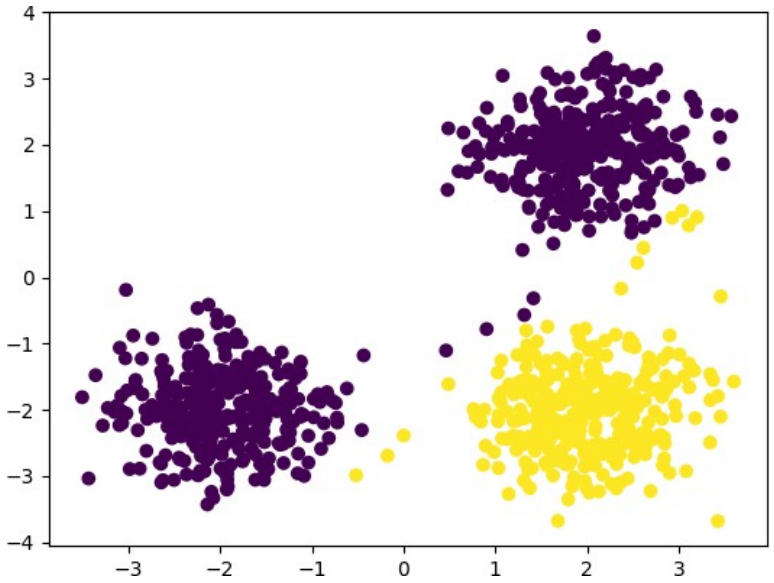


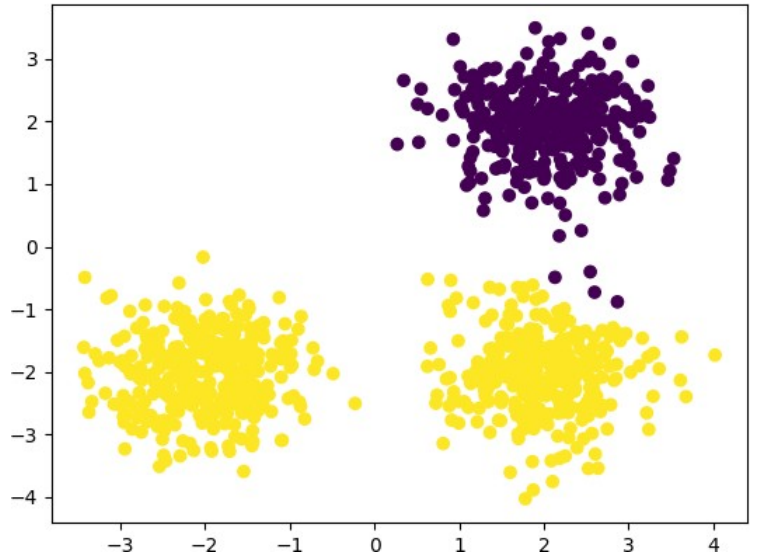


K-means is clearly applicable to data sets where the clusters are very well-separated

K-Means is often applied to data that have no clear clustering structure

K-MEANS ALGORITHM: REMARKS





- SELECTING K IS CHALLENGING
- ONE SOLUTION FOR THIS IS TO RUN K-MEANS SEVERAL TIMES AND PICK THE ATTEMPT THAT MINIMIZES THE COST FUNCTION

K-Means Algorithm: remarks

- Nearby points may not end in the same cluster

K-Means Algorithm: remarks

- Nearby points may not end in the same cluster

- It is possible that K-means gets stuck in this local optimum.

K-Means Algorithm: remarks

- Nearby points may not end in the same cluster

- It is possible that K-means gets stuck in this local optimum.
- How would you solve this problem ?

K-MEANS ALGORITHM: REMARKS

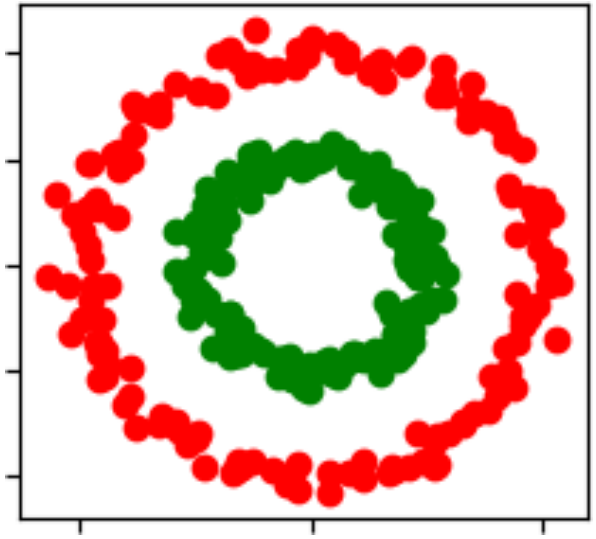
- NEARBY POINTS MAY NOT END IN THE SAME CLUSTER

- IT IS POSSIBLE THAT K-MEANS GETS STUCK IN THIS LOCAL OPTIMUM.

- HOW WOULD YOU SOLVE THIS PROBLEM ?
  - Try different initialization and choose the one that produces the best result (wrt the cost function).

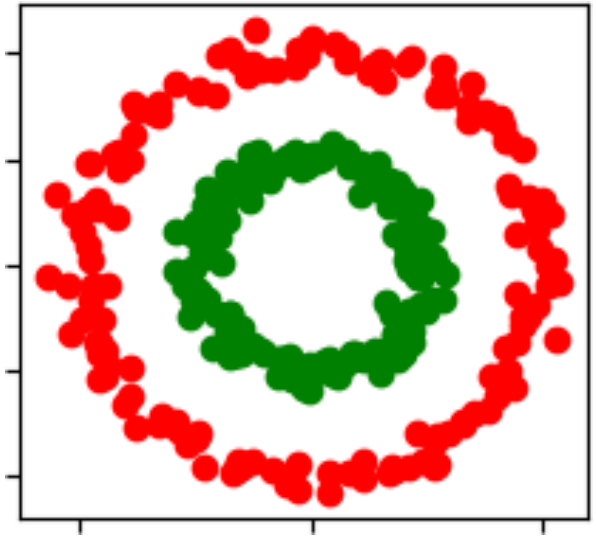
K-MEANS ALGORITHM: REMARKS

- USING EUCLIDIAN DISTANCE K-MEANS WILL NOT GIVE THE NATURAL CLUSTERS FOR THIS SET.



K-MEANS ALGORITHM: REMARKS

- USING EUCLIDIAN DISTANCE K-MEANS WILL NOT GIVE THE NATURAL CLUSTERS FOR THIS SET.



- ONE CAN TRY TO CHANGE THE FEATURES:

$$(x, y) \rightarrow \left(\sqrt{x^2 + y^2}, \text{TAN}^{-1}\frac{y}{x}\right)$$

(CONVERT THE FEATURE VECTOR TO POLAR COORDINATES)

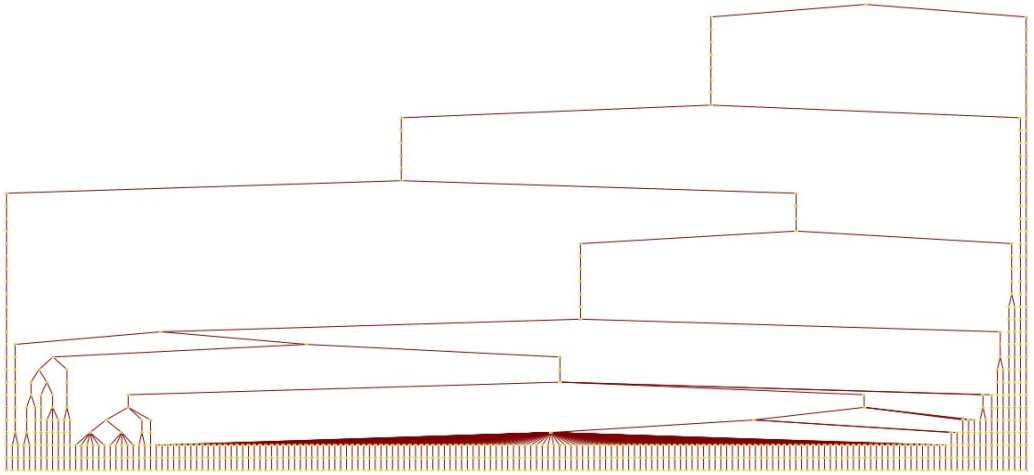
- OR CHANGE THE DISTANCE FUNCTION USED IN THE K-MEANS ALGORITHM.

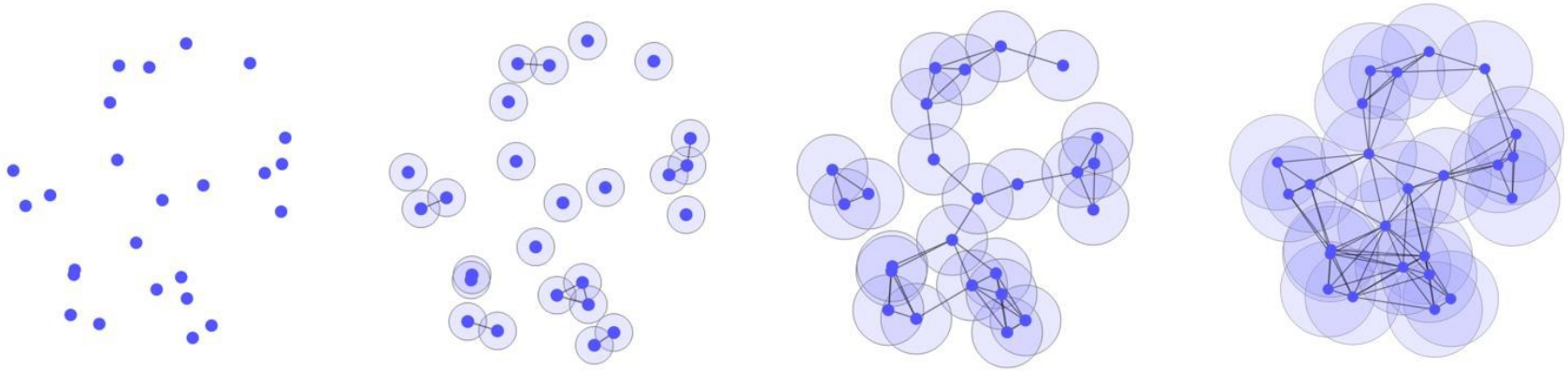
K-MEANS VARIATIONS

- ALMOST EVERY ASPECT OF K-MEANS HAS BEEN ALTERED AND CHANGED TO PERFORM OTHER CLUSTERING TASKS.
  - Distance function: Any function that satisfy distance axioms can be used instead of the Euclidian distance.
  - Cost function
  - Initialization heuristics
  - Efficiency
  - Centroid definition: K-Medians, K-mediods

SEE WIKIPEDIA PAGE HERE

Hierarchical Clustering





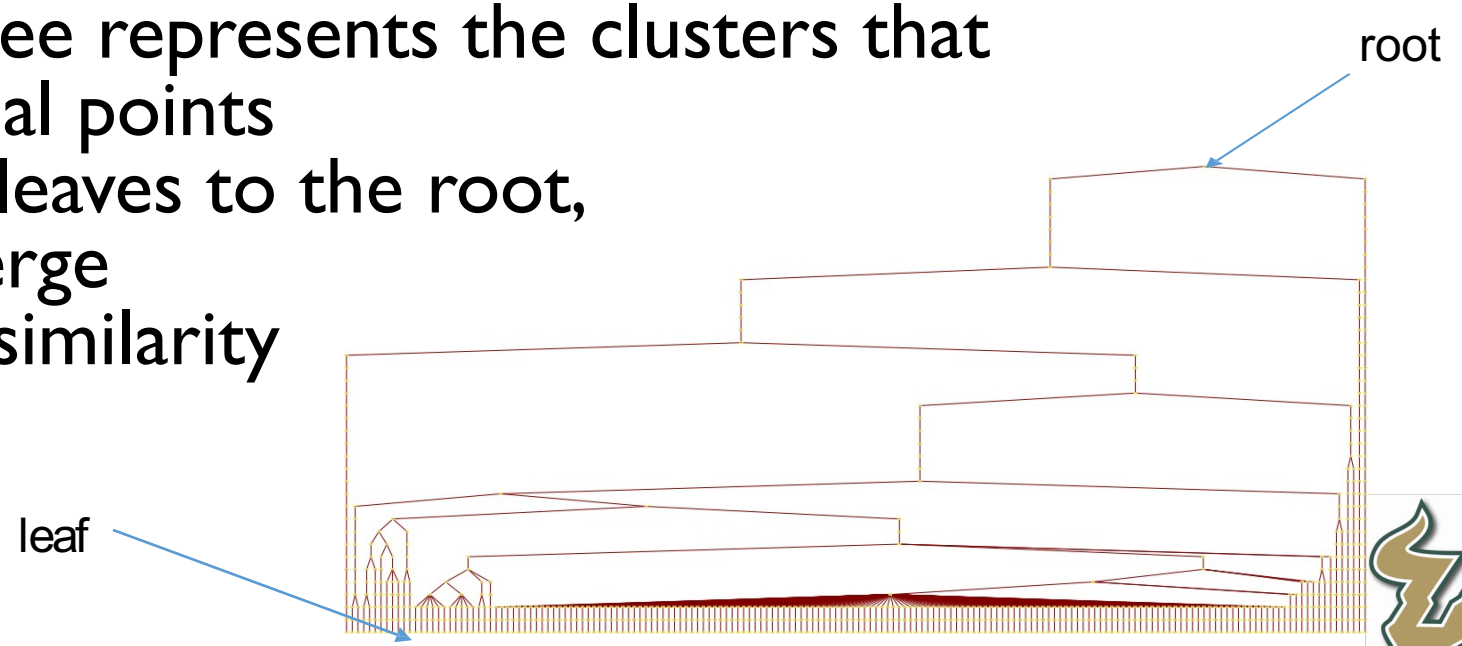
Hierarchical Clustering

- Hierarchical clustering is a family of clustering algorithms that build a tree clusters.

- It is usually done by merging or splitting the clusters successively.
  - Agglomerative: bottom up (Merge)
  - Divisive: top down (Split)

Hierarchical Clustering

- Hierarchical clustering is usually represented by a tree called the dendrogram that represents the clustering at all levels.
- Each node in the tree represents a cluster
  - The root of the tree represents the cluster that contains all points
  - The leaves of the tree represents the clusters that contain the individual points
  - As we go from the leaves to the root, clusters start to merge according to some similarity criterion.



General Steps

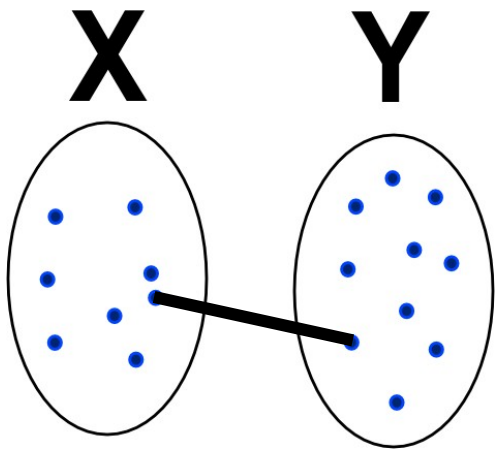
- General Steps in a standard hierarchical agglomerative clustering algorithm:
  1. Compute the distance matrix, or the dissimilarity, between all points in the input data.
     - Choice of metric will impact the results largely.
  2. Initialize every point in the dataset to be its own cluster
  3. Compute the distance between all clusters
  4. Combine the closest two clusters: the two clusters c_i and c_j with $\min_{i,j} d(c_i, c_j)$
  5. Remove the clusters c_i and c_j and add the cluster $c_i + c_j$.
  6. Go back to 3 and repeat until we have a single cluster.

General Steps

- Question : Given two clusters X and Y. How do we measure the distance between them?
  - This is important because step 3 we need to measure the distance between clusters rather than points.

Distance between clusters

- Given two clusters X and Y.
- How do we measure the distance between them ?

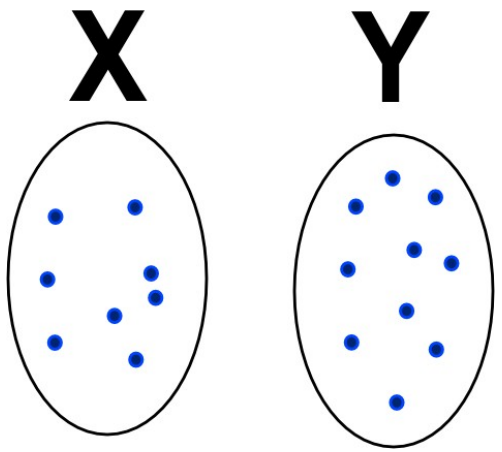


$$D(X,Y) = \min_{x \in X, y \in Y} d(x,y)$$

- One way is to measure the minimal distance between all points of X and Y.
- This distance induce single linkage clustering.

Distance between clusters

- Given two clusters X and Y.
- How do we measure the distance between them ?

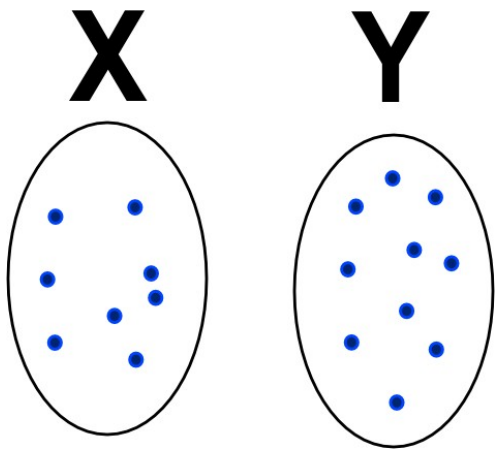


$$D(X,Y) = \frac{1}{|X||Y|} \sum_{x \in X, y \in Y} d(x,y)$$

- We could also consider the mean distance between the points of the clusters

Distance between clusters

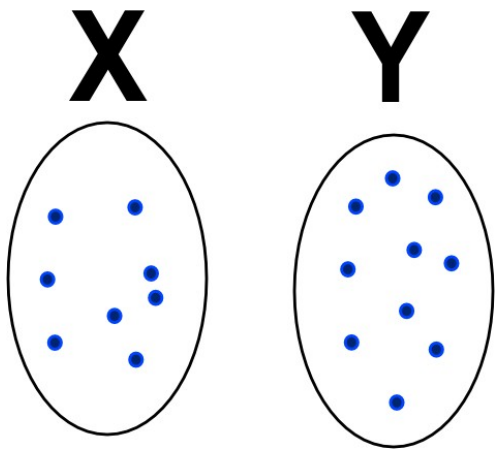
- Given two clusters X and Y.
- How do we measure the distance between them ?



- There are other measures as well : The minimal energy criterion, the distance between the centroids of the clusters

Distance between clusters

- Given two clusters X and Y. How do we measure the distance between them ?



  - For efficient calculations we usually require the following condition:
  - Knowing the distance $d(A, C), d(B, C)$
  - Implies we can calculate in constant time the distance $d(A + B, C)$

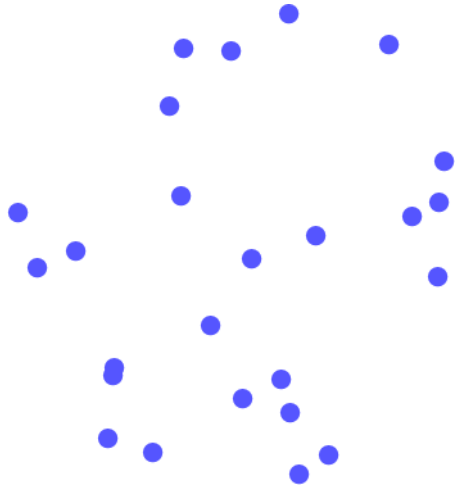
- There are other measures as well : The minimal energy criterion, the distance between the centroids of the clusters

Single-linkage clustering algorithm

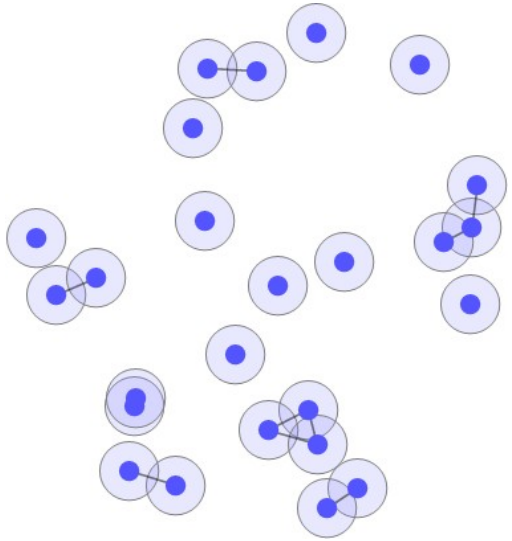
- The input of the algorithm is a distance matrix D, contains all distances $d(i, j)$ between the points. At the beginning each point is its own cluster.

Single Linkage Hierarchical Clustering and the ε-Neighborhood Graph

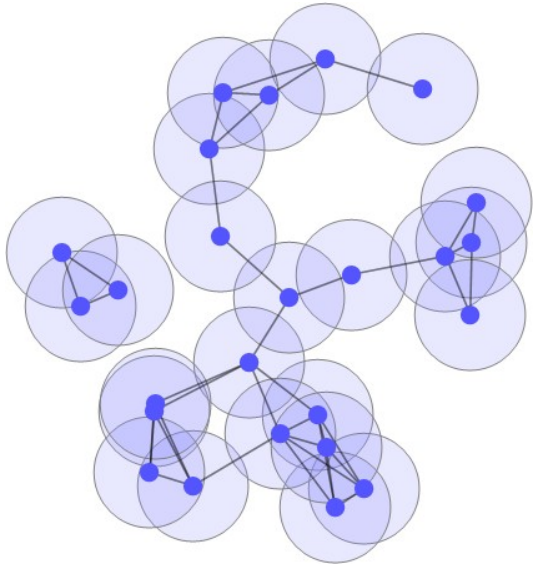
- Suppose that we are given a set of points $X = \{p_1, p_2, \ldots, p_n\}$ in R^d with a distance function d defined one them.
- Consider the connected components of the ε-neighborhood graph as we continuously increase ε from zero to infinity.



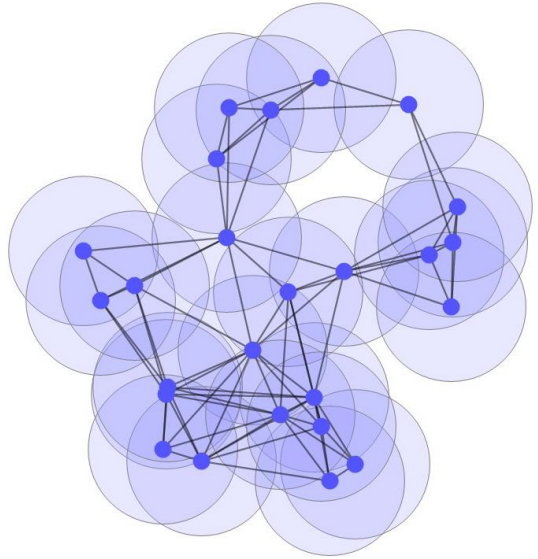
Every point is a connected component



When ε is a little larger we start some clusters starts to get form



When ε is even larger we have few clusters. As the clusters get larger and larger



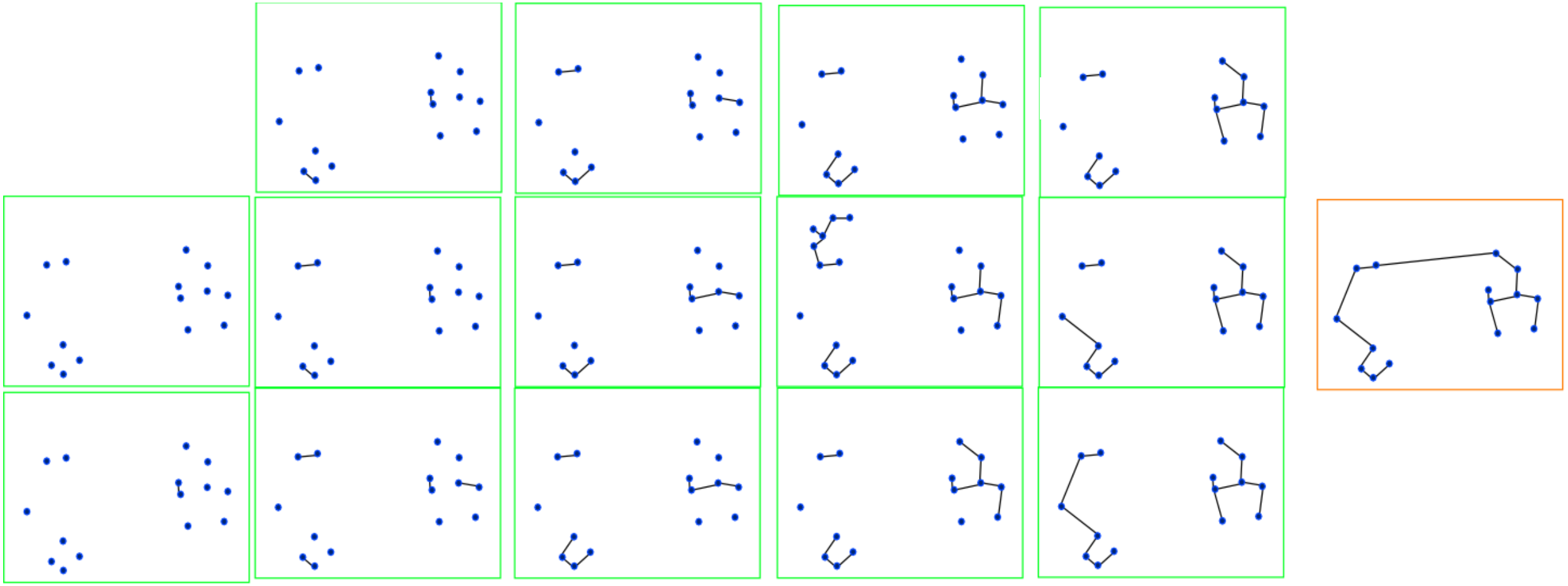
At some point all points become a part to of a single cluster

Kruskal's Algorithm

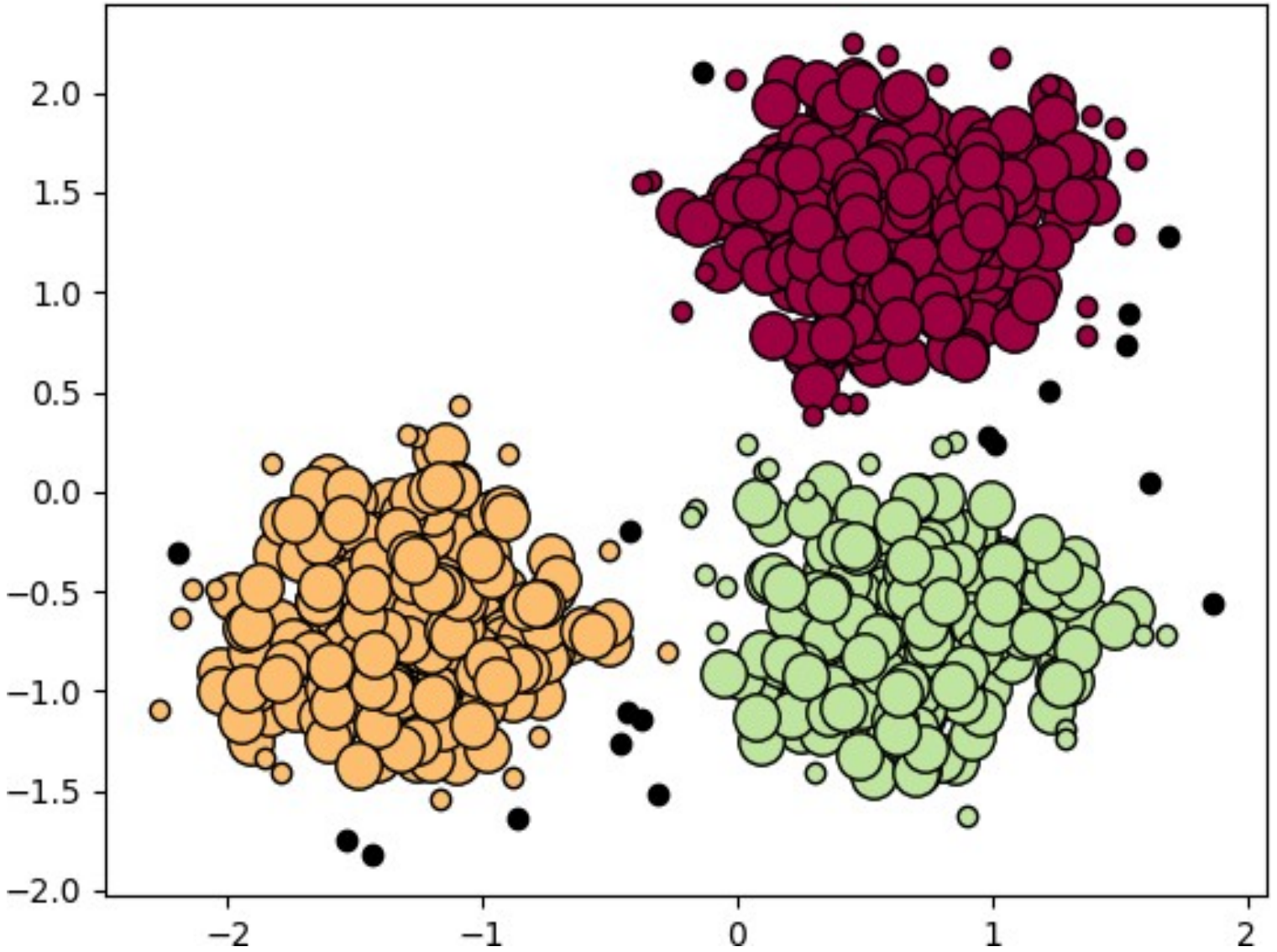
- Let $G = (V, E, w)$ be a connected weighted graph.
- Informally, the algorithm can be given by the following steps:

1. Set $VT\ to\ be\ V, Set\ ET = \{\}$. Let $S = E$
2. While S is not empty and T is not a spanning tree
    1. Select an edge e from S with the minimum weight and delete e from S.
    2. If e connects two separate trees of T then add e to ET

Single Linkage Hierarchical Clustering and the and Kruskal's algorithm



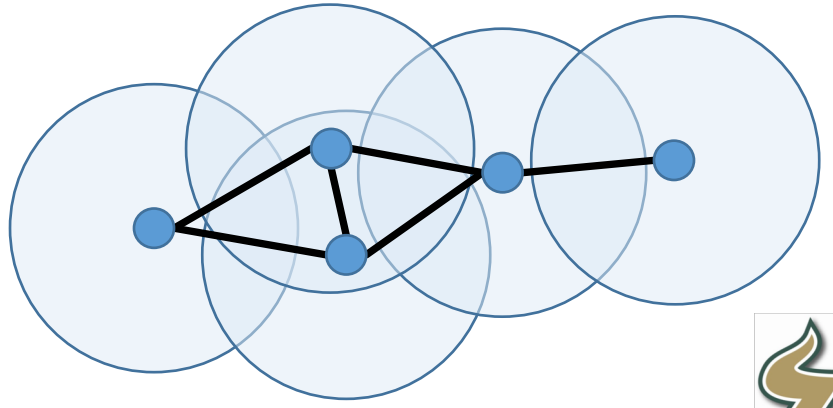
DBSCAN
(Density-Based Spatial Clustering of Applications with Noise)



ε NEIGHBOR GRAPH

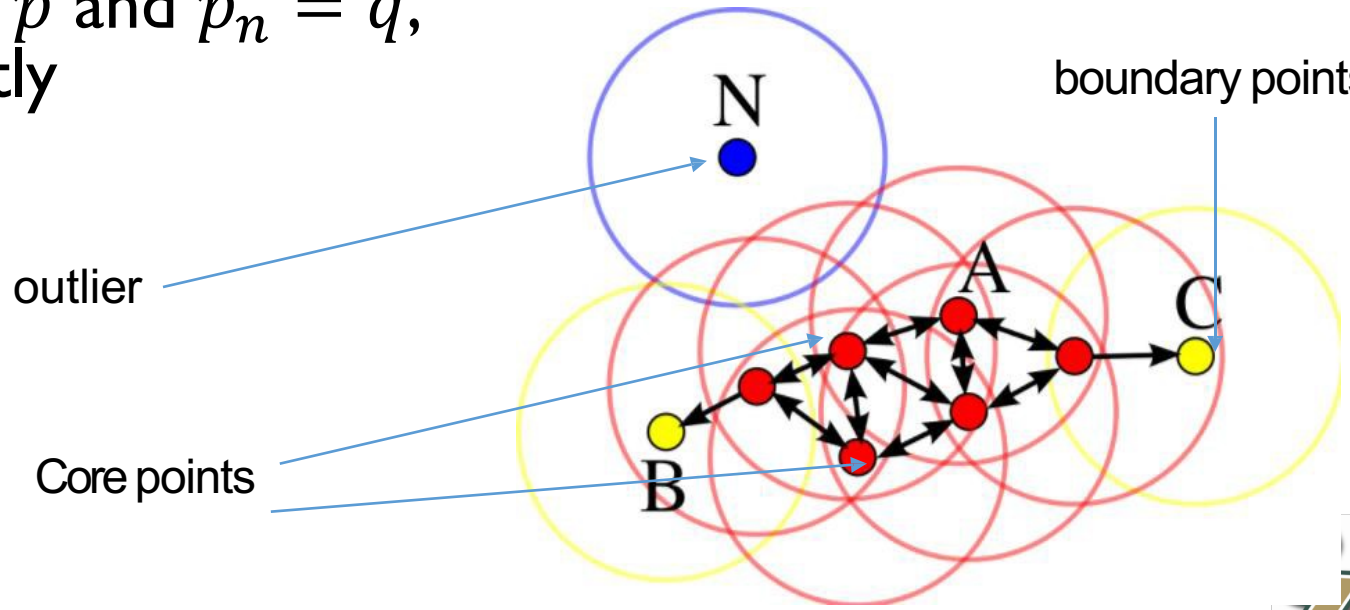
- DBSCAN USE AN ε NEIGHBOR GRAPH TO IDENTIFY CLUSTERS.

- FOR THE ε NEIGHBOR GRAPH:
  - Points from data are vertices
  - Edges are formed if $d(x_i, x_j) \leq \varepsilon$ for all $i \neq j$



DBSCAN

- In DBSCAN points are classified as follows :
  - core points: A point p is said to be a core point if at least **min_samples** points are within a distance ε
  - directly reachable: A point q is directly reachable from p if the point q is within distance ε from point p, and p is a core point.
  - Density-reachable points: A point q is 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 pi (all the points on the path must be core points, with the possible exception of q).



DBSCAN

- In DBSCAN points are classified as follows :
  - Outliers: All points not reachable from any other point are called outliers.
  - Boundary points: Boundary points cannot reach other points but other points core points can reach them

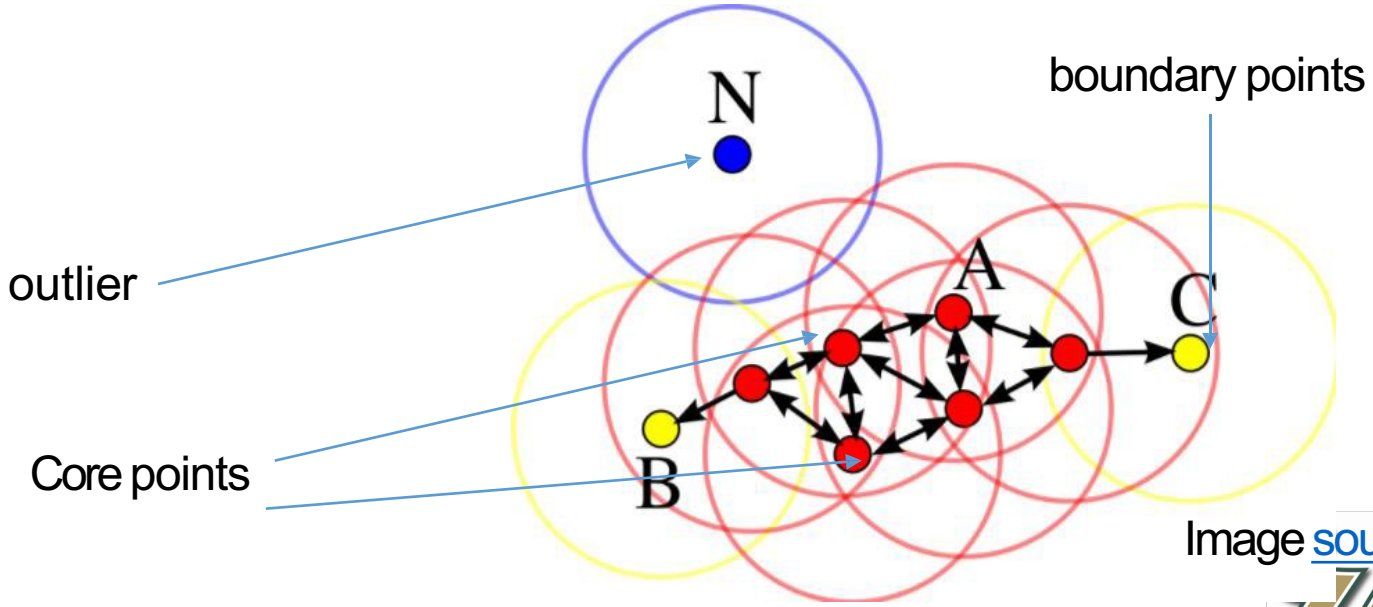
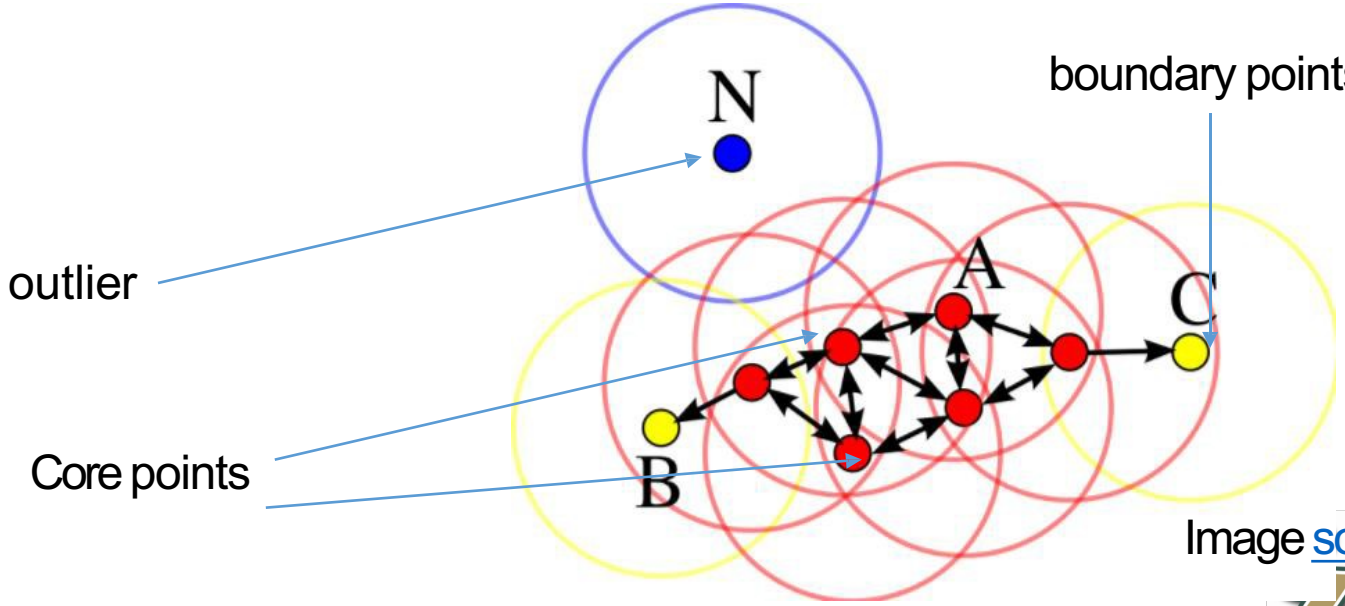


Image source

DBSCAN-Reachability

- If p is reachable from q. Does that mean q is reachable from p? Explain. Can you give an example from the points below?



DBSCAN-Density-connectedness

- Two points p and q are density-connected if there is a point o such that both p and q are reachable from o.
- A cluster then satisfies two properties:
  - All points within the cluster are mutually density-connected.
  - If a point is density-reachable from any point of the cluster, it is part of the cluster as well.

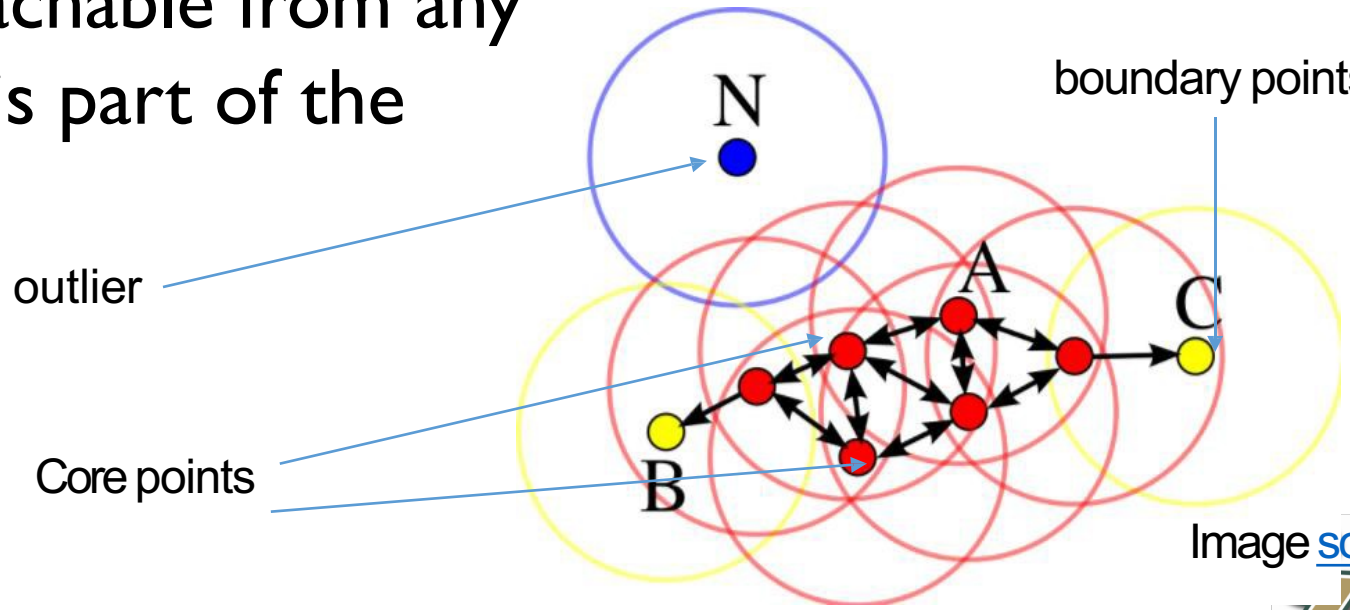
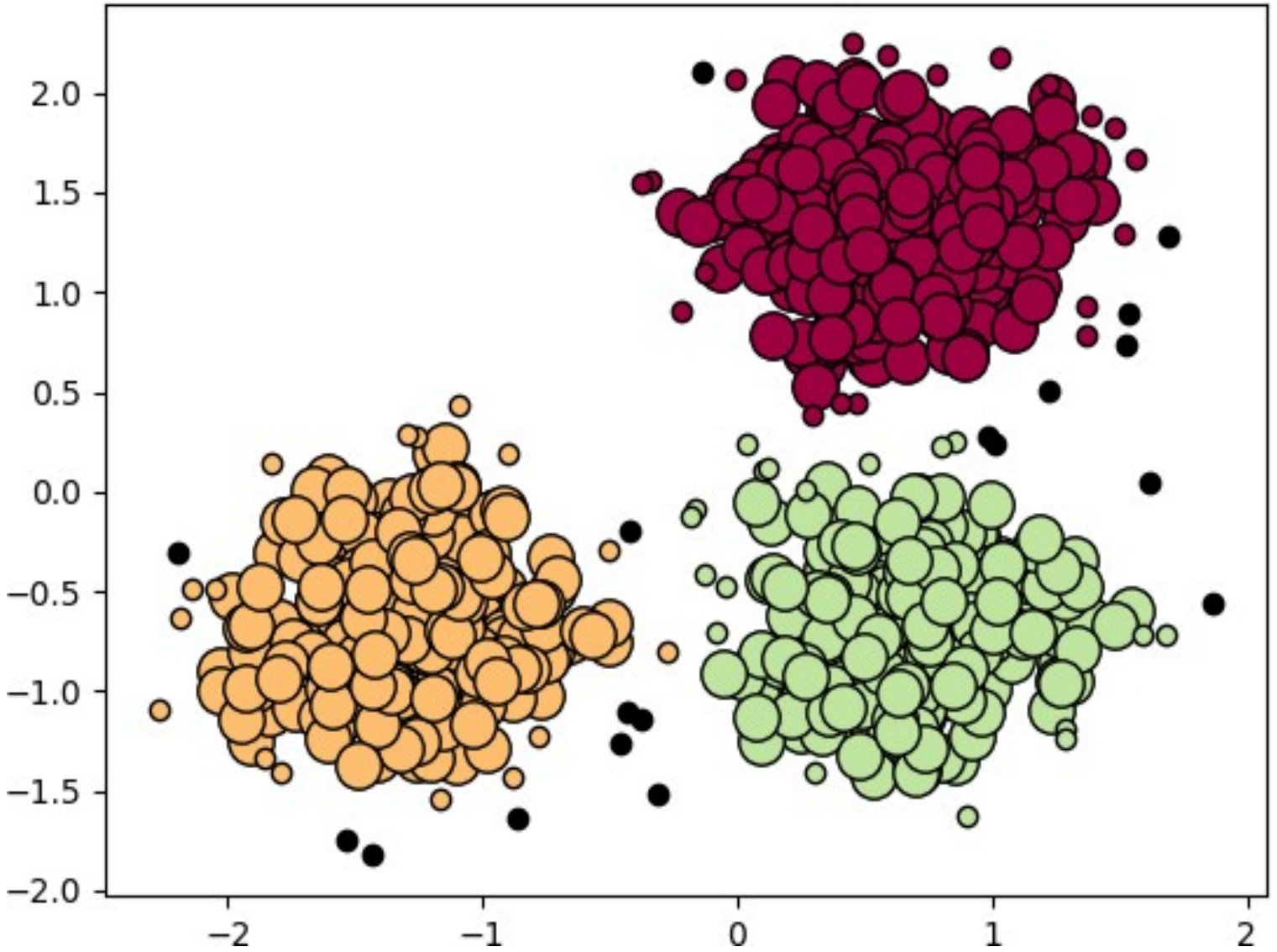


Image source

DBSCAN-Algorithm

- Input a data X, a positive real number ε and a positive integer **MIN_SAMPLES**
  1. Find the ε neighbor graph.
  2. Identify the core points with more than **MIN_SAMPLES** neighbors.
  3. Find the connected components of core points on the neighbor graph, ignoring all non-core points.
  4. Assign each non-core point to a nearby cluster if the cluster is an ε neighbor, otherwise assign it to noise.

DBSCAN-SKLEARN



SKLEARN DBSCAN / SKLEARN EXAMPLE

COMPARISON BETWEEN CLUSTERING ALGORITHMS

Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large `n_samples`, medium `n_clusters` with MiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with `n_samples`	Many clusters, uneven cluster size, non-flat geometry	Distances between points
Spectral clustering	number of clusters	Medium `n_samples`, small `n_clusters`	Few clusters, even cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints	Distances between points
Agglomerative clustering	number of clusters, linkage type, distance	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, non Euclidean distances	Any pairwise distance
DBSCAN	neighborhood size	Very large `n_samples`, medium `n_clusters`	Non-flat geometry, uneven cluster sizes	Distances between nearest points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation	Mahalanobis distances to centers
Birch	branching factor, threshold, optional global clusterer.	Large `n_clusters` and `n_samples`	Large dataset, outlier removal, data reduction.	Euclidean distance between points

SOURCE