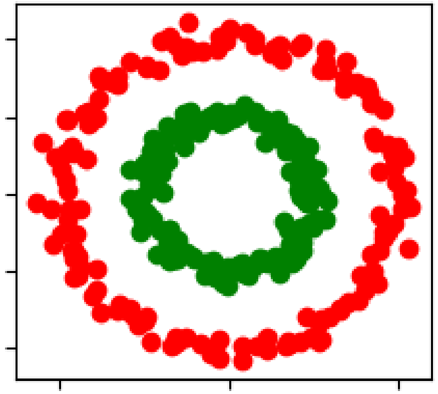
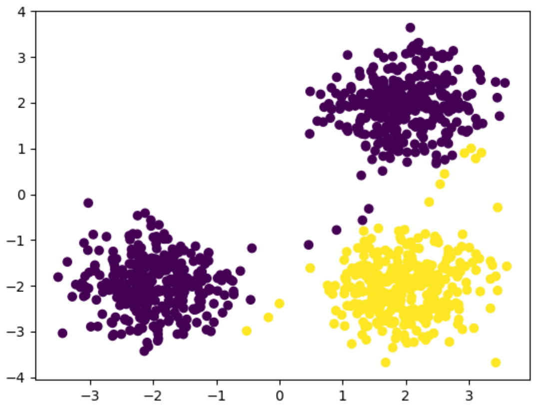
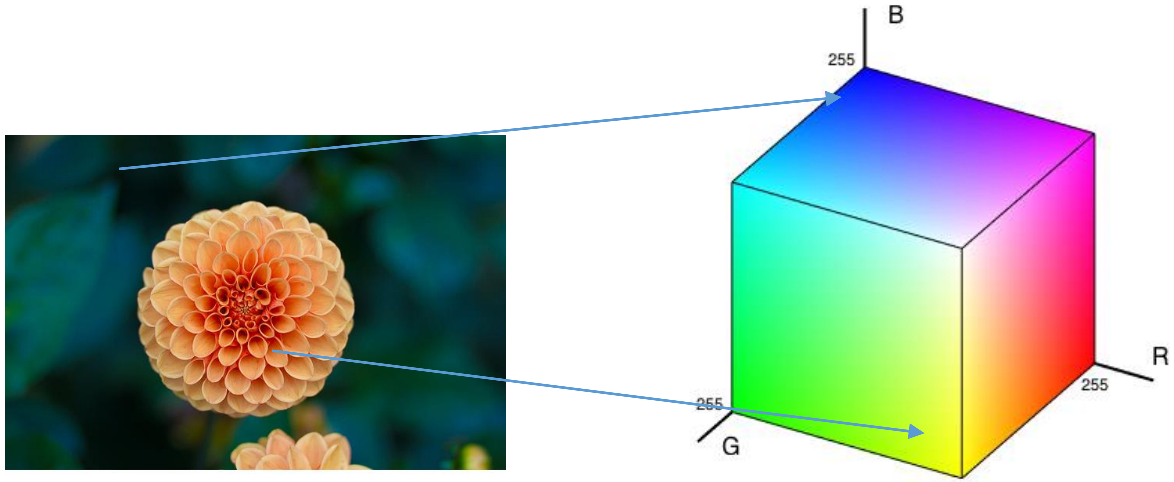


Clustering Algorithms
an introduction

Mustafa Hajij





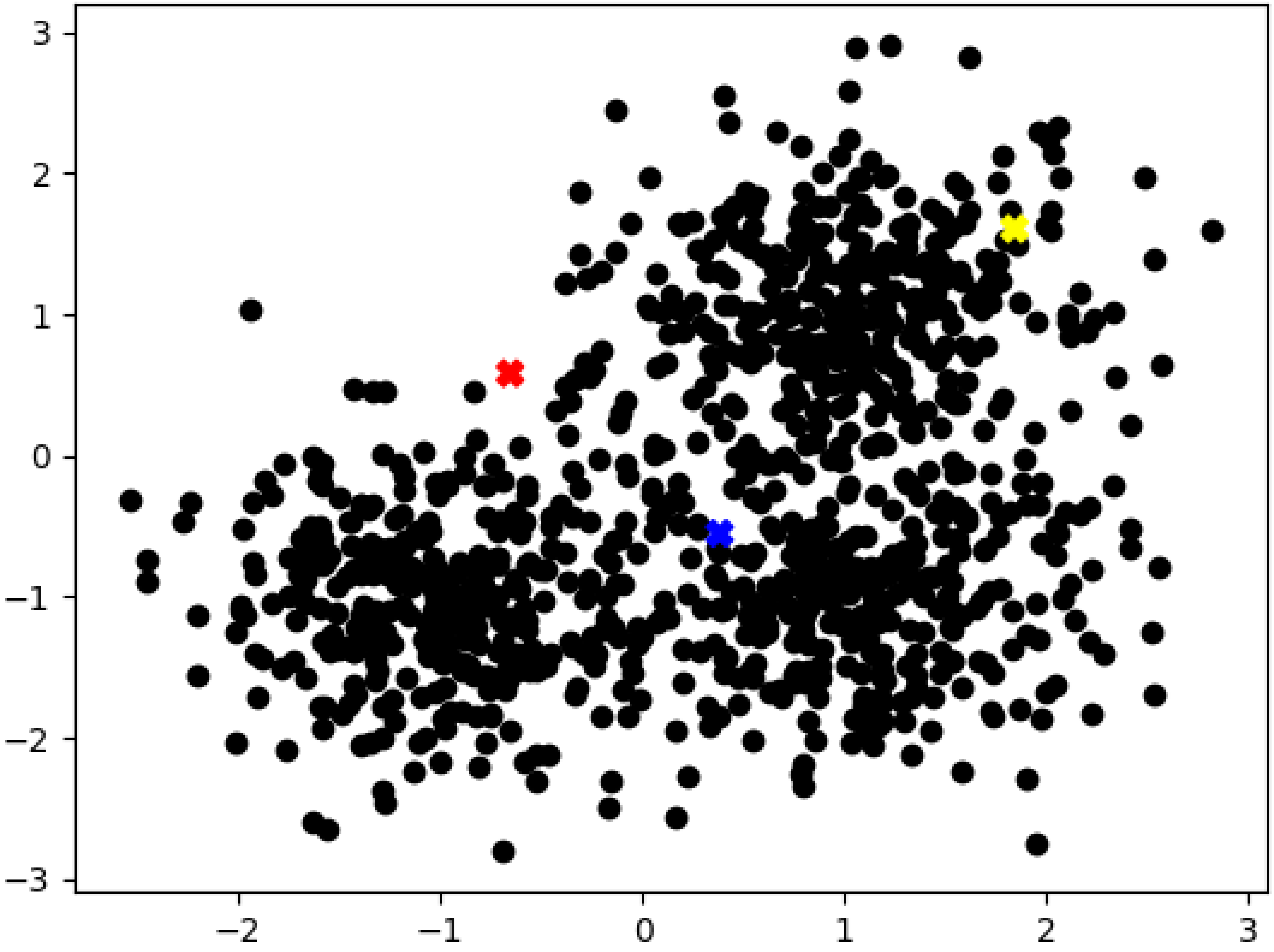
Clustering Algorithms

Recall:

- 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 ?

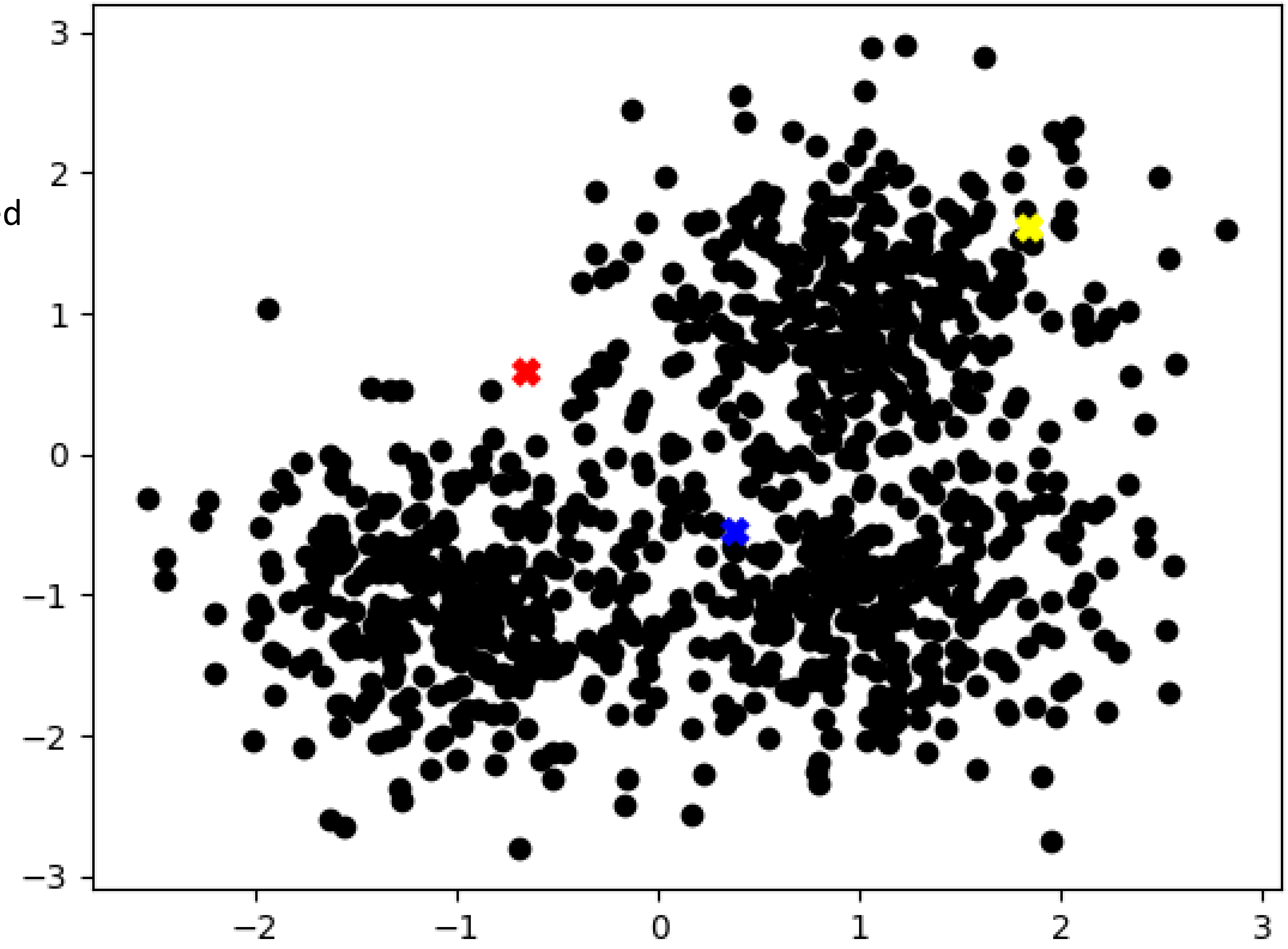
K-Means Algorithm: Example

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



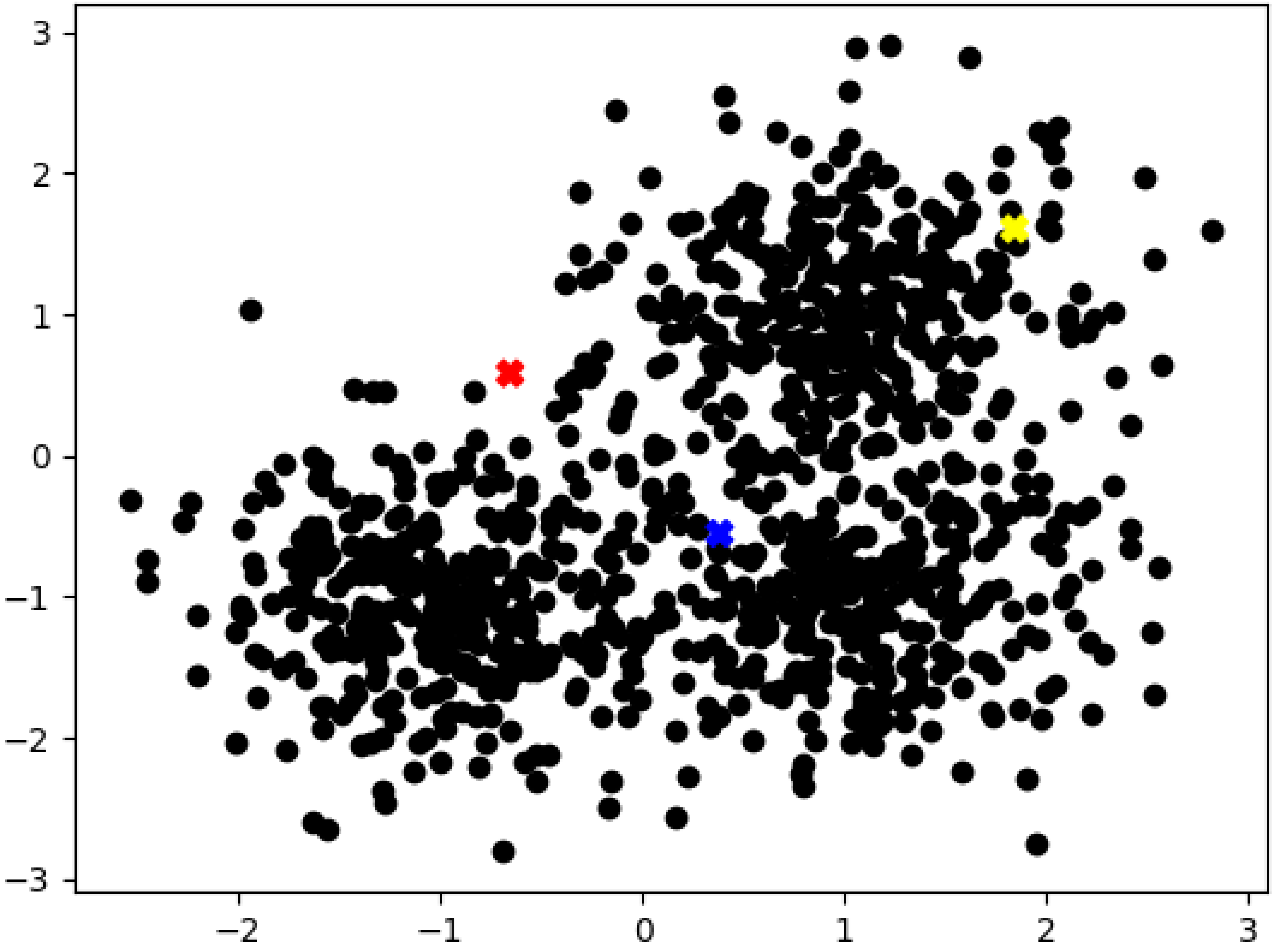
K-Means Algorithm: Example

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



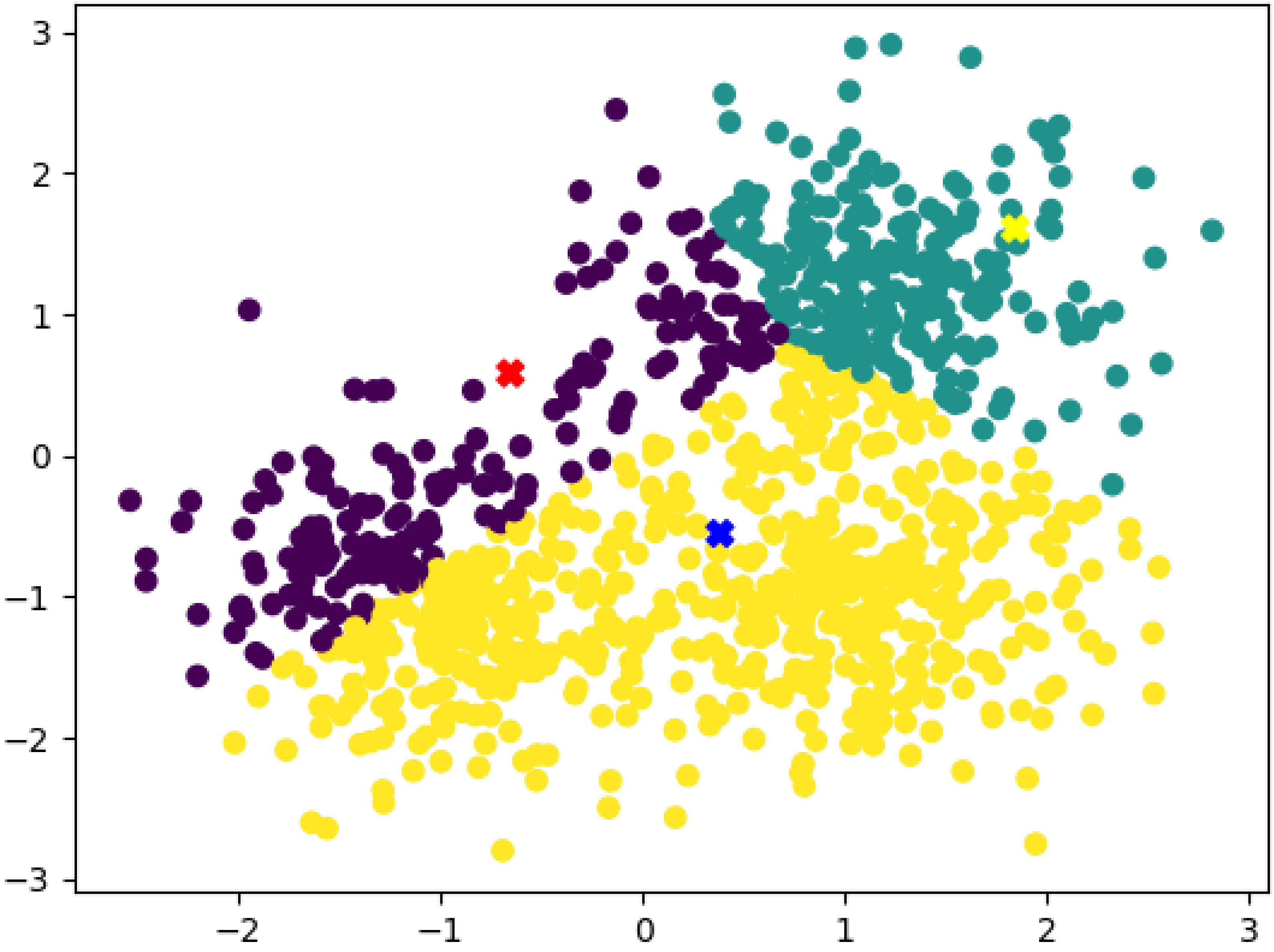
K-Means Algorithm: Example

Choose randomly 3 *centroids* : c_1, c_2, c_3 (the points appear in blue, red and yellow)



K-Means Algorithm: Example

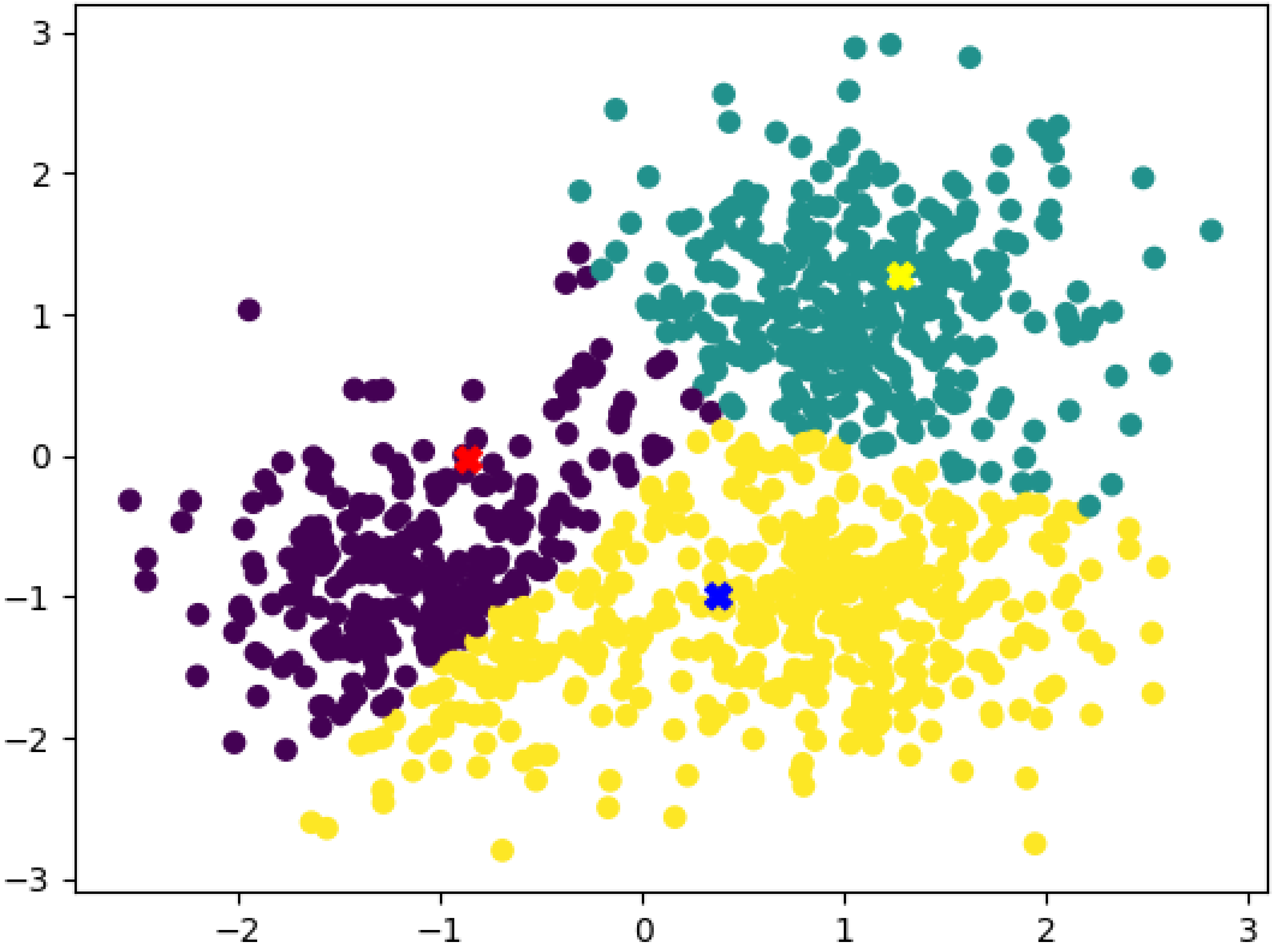
assign each point x
in the set to the
closest centroid



K-Means Algorithm: Example

Update the centroids c_1, c_2, c_3 as follows :

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



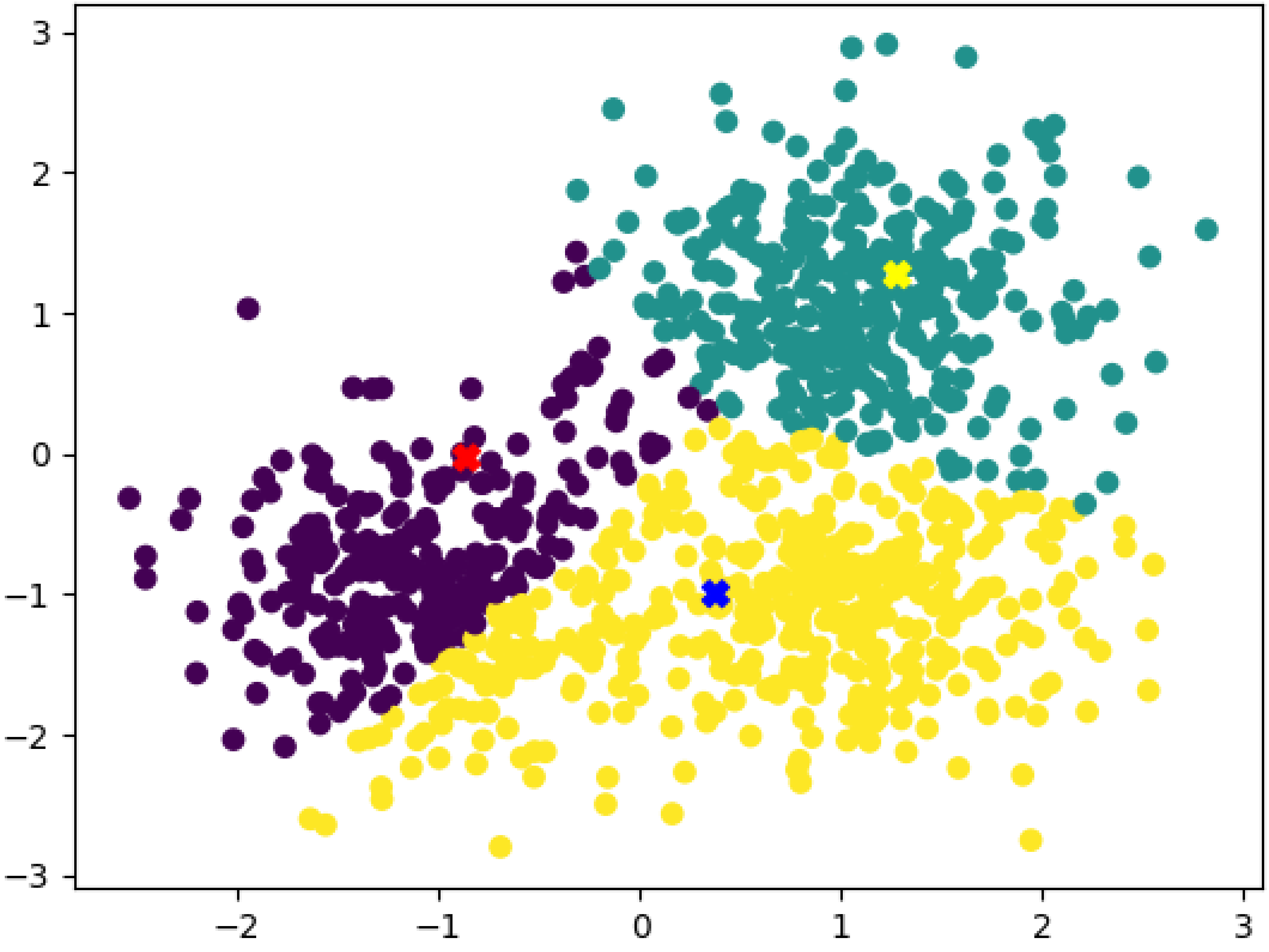
K-Means Algorithm: Example

Update the centroids c_1, c_2, c_3 as follows :

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

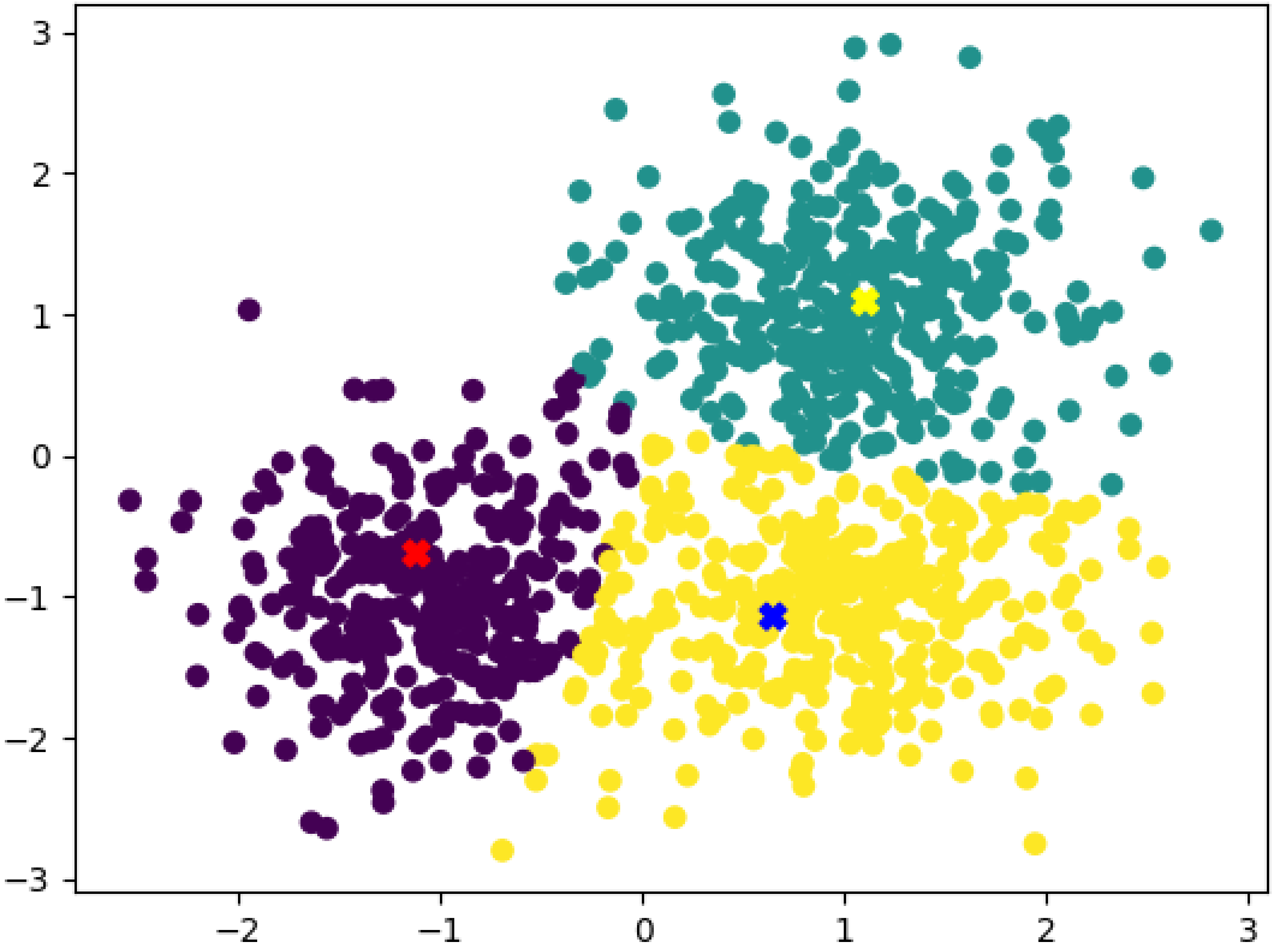
In other words, the new center is the average of the cluster members

assign each point x in the set to the closest centroid



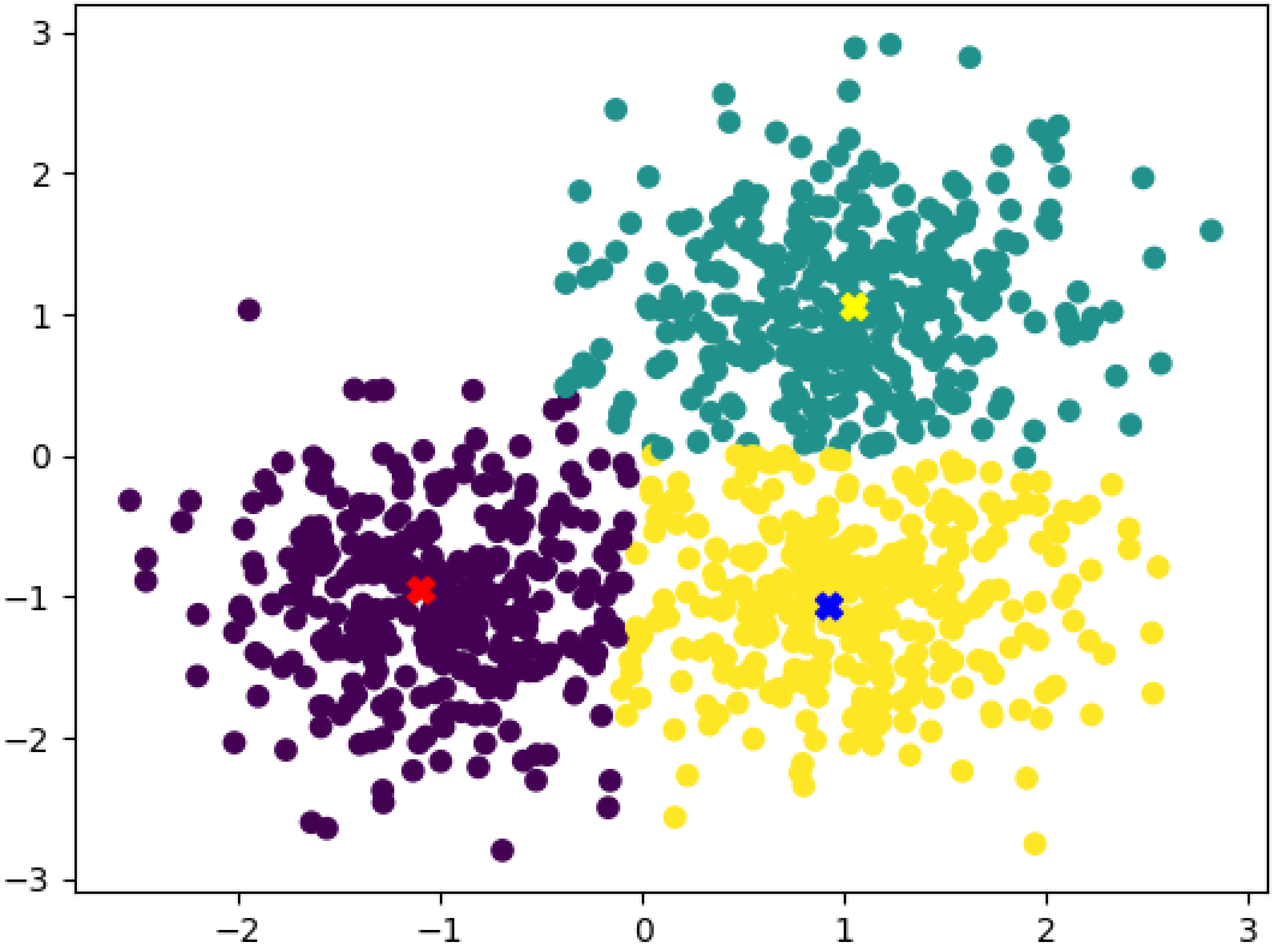
K-Means Algorithm: Example

Repeat until convergence



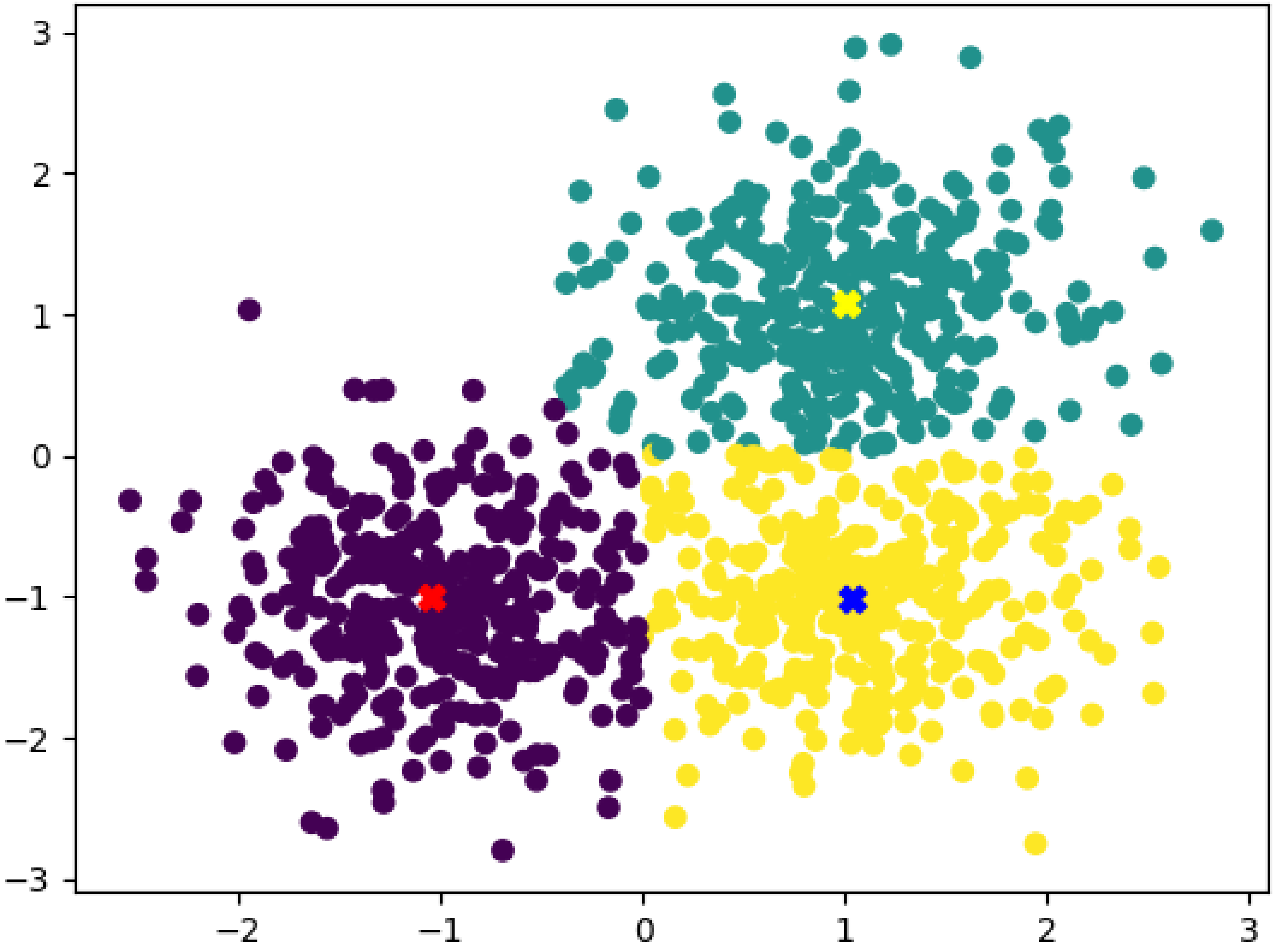
K-Means Algorithm: Example

Repeat until convergence



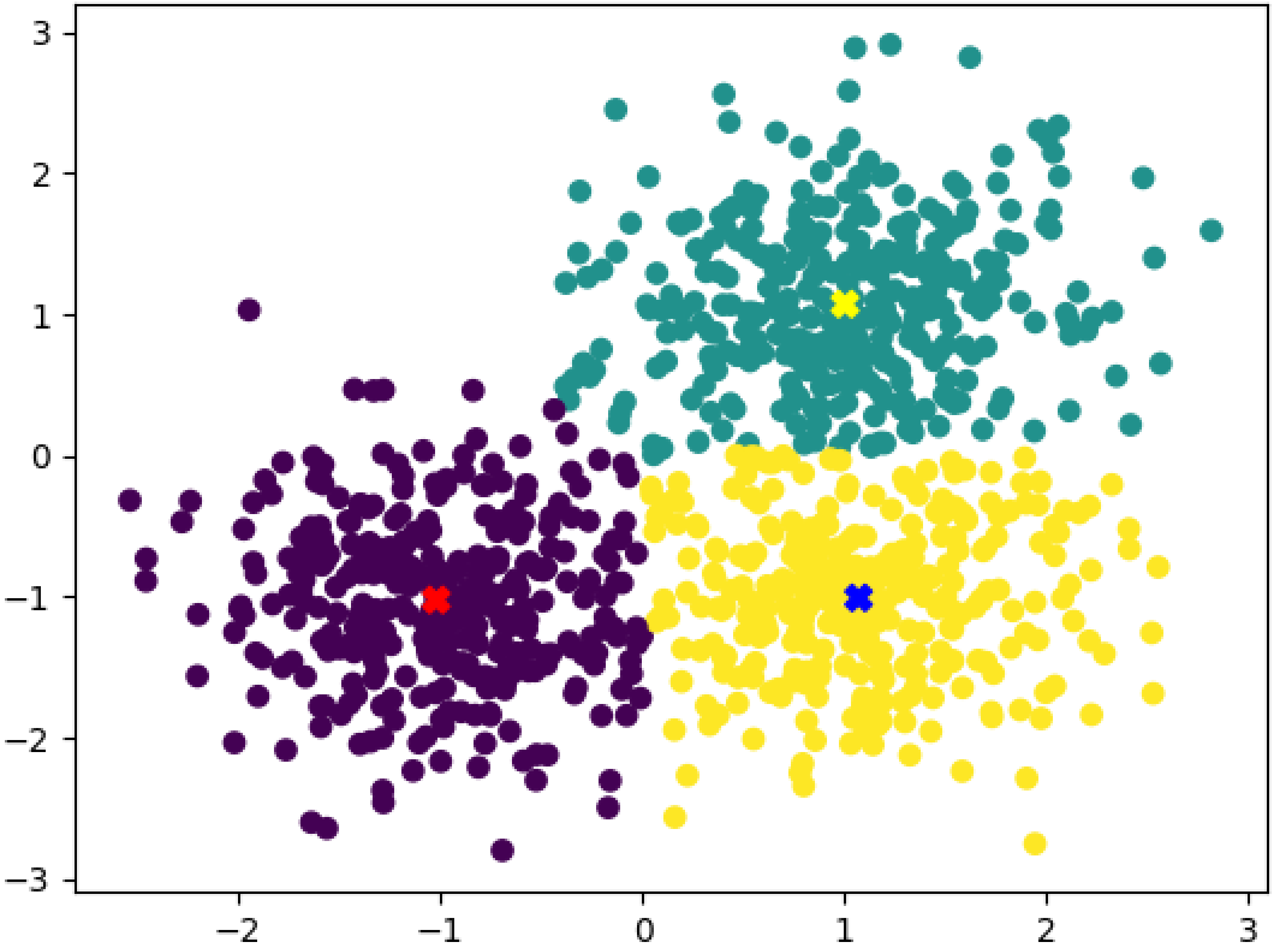
K-Means Algorithm: Example

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, $X = \{x^{(1)}, \dots, x^{(m)}\}$ where $x^{(i)}$ $is\ a\ point\ in\ R^n$

The Euclidian distance between two points x and y in R^n will be denoted by $d(x, y)$.

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, $X = \{x^{(1)}, \dots, x^{(m)}\}$ where $x^{(i)}$ $is\ a\ point\ in\ R^n$

The Euclidian distance between two points x and y in R^n will be denoted by $d(x, y)$.

Q: what is the formula for the Euclidian distance between two points in R^n?

K-Means Algorithm

The K-means algorithm takes two inputs:

1. A parameter K, which is the number of clusters one wants to find in the data.

2. The training set X of the points. $Here\ X = \{x^{(1)}, \dots, x^{(m)}\}$ where $x^{(i)}\ is\ a\ point\ in\ R^n$.

The algorithm returns the data X partitioned into K-clusters.

K-Means Algorithm

1-Choose randomly k centroids : $c_1, c_2, \dots, c_K$ in R^n

2-Repeat until convergence

a : We assign each point x in the set to the closest centroid. ← Cluster assignment step

b : Update the centroids $c_1, c_2, \dots, c_K$ as follows : ← Centroid move step

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

Here S_i is the cluster associated with the centroid c_i

Convergence:

- none of the cluster assignments change
- The centroids do not change

K-Means Algorithm

1-Choose randomly k centroids : $c_1, c_2, \ldots., c_K$ in R^n

2-Repeat until convergence

a : We assign each point x in the set to the closest centroid.

Cluster assignment step

b : Update the centroids $c_1, c_2, \ldots., c_K$ as follows :

Centroid move step

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

Here S_i is the cluster associated with the centroid c_i

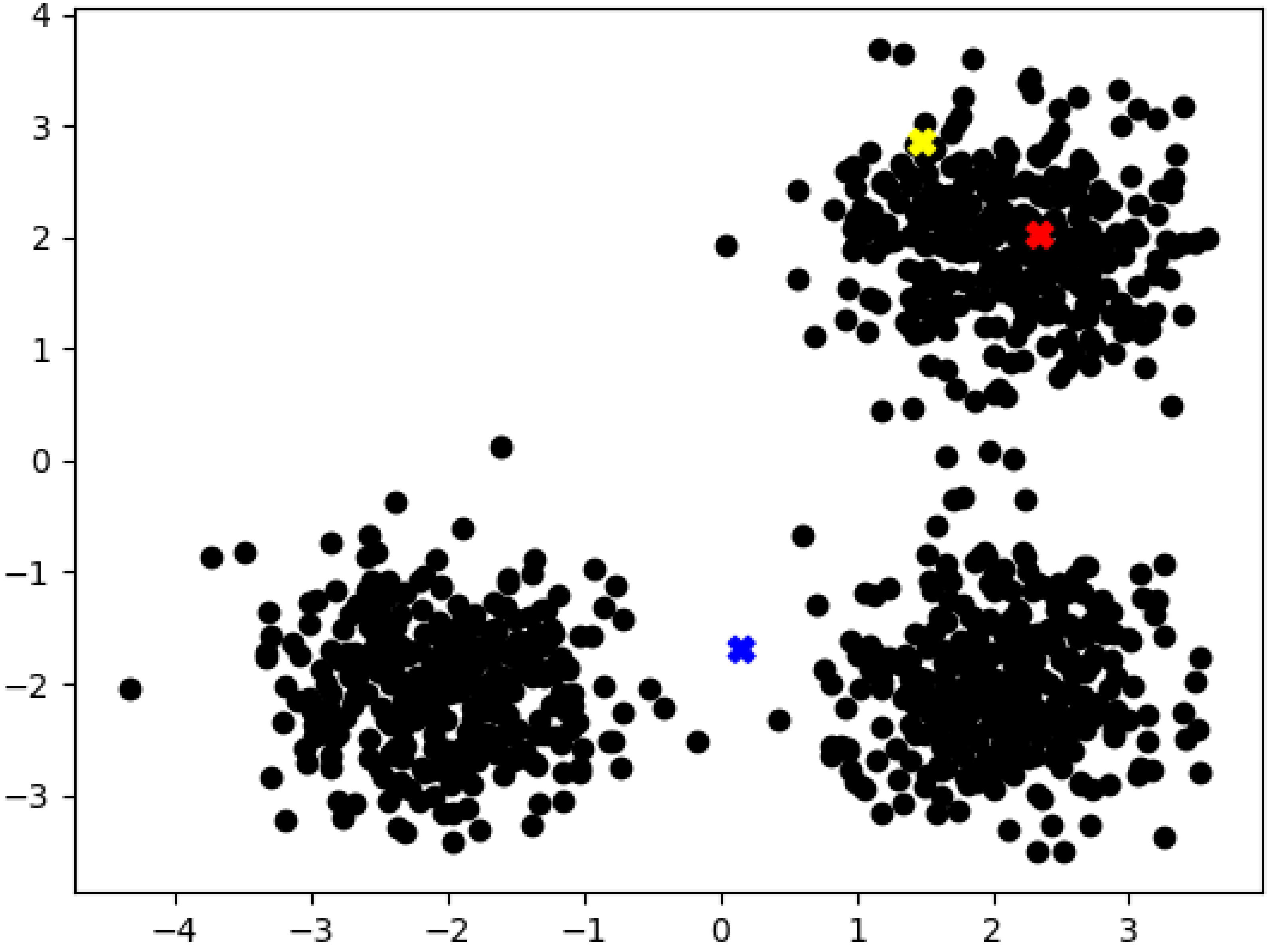
This is where *mean* comes from

Convergence:

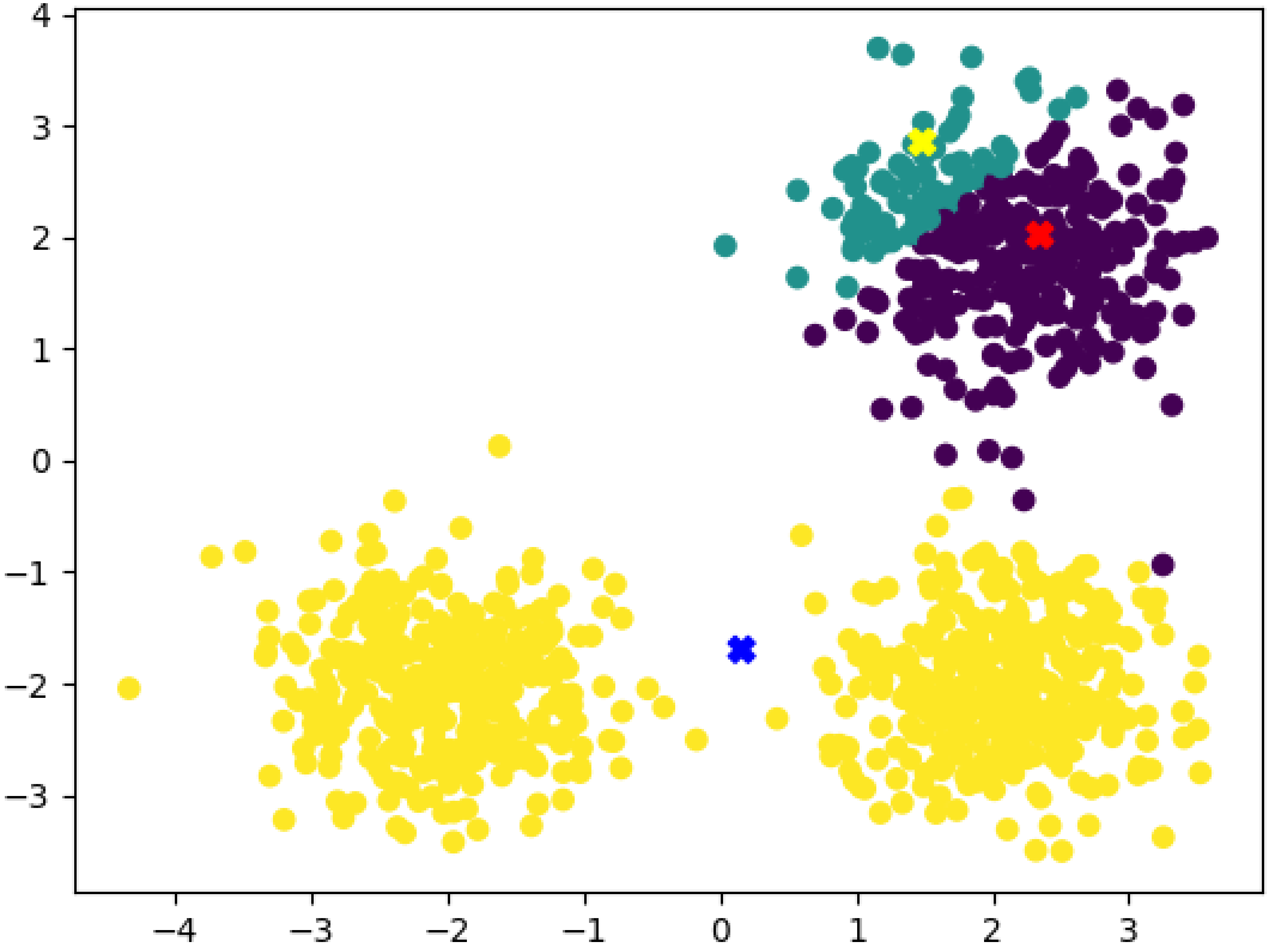
- none of the cluster assignments change
- The centroids do not change

K-Means Algorithm: Example

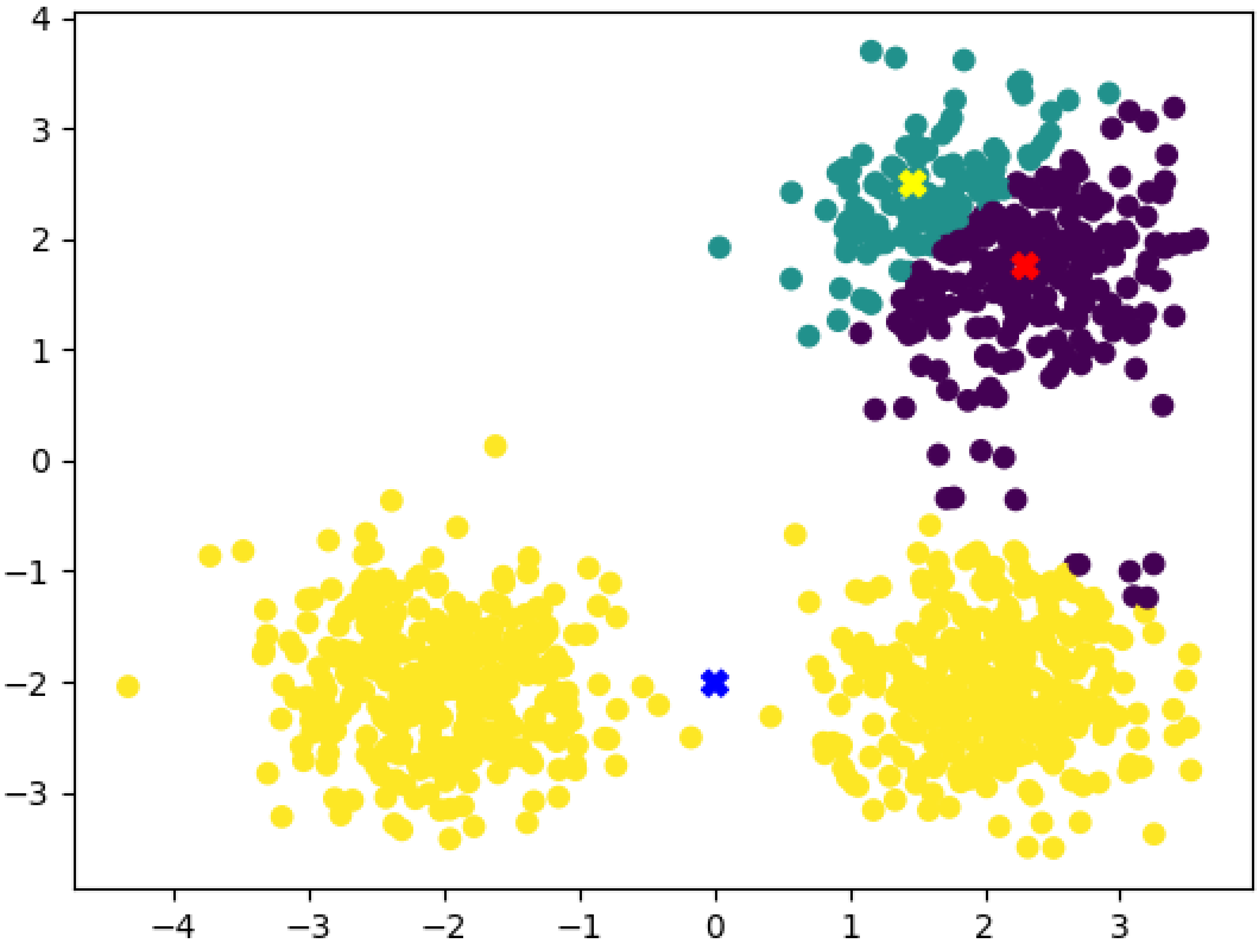
Lets consider another example



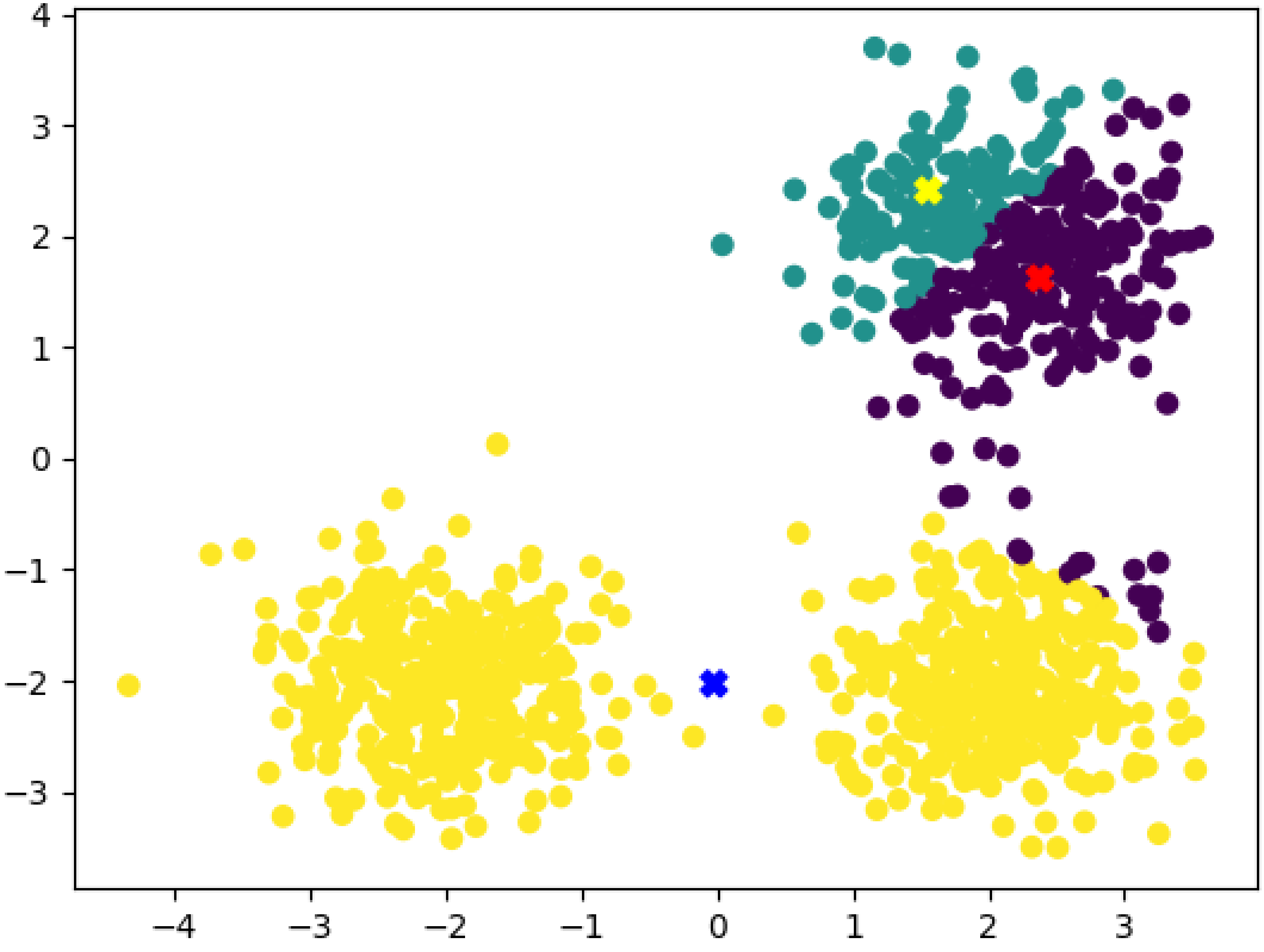
K-Means Algorithm: Example



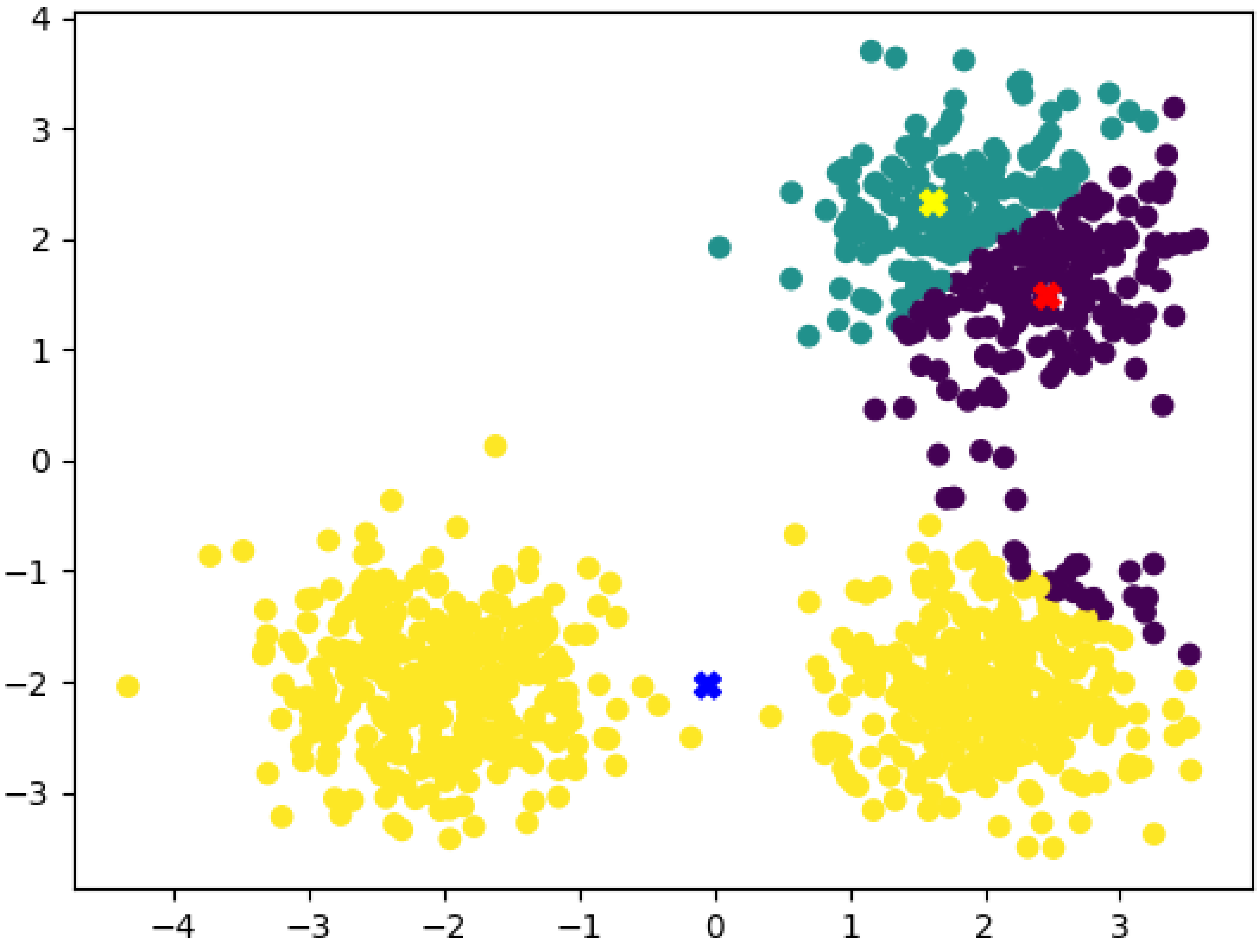
K-Means Algorithm: Example



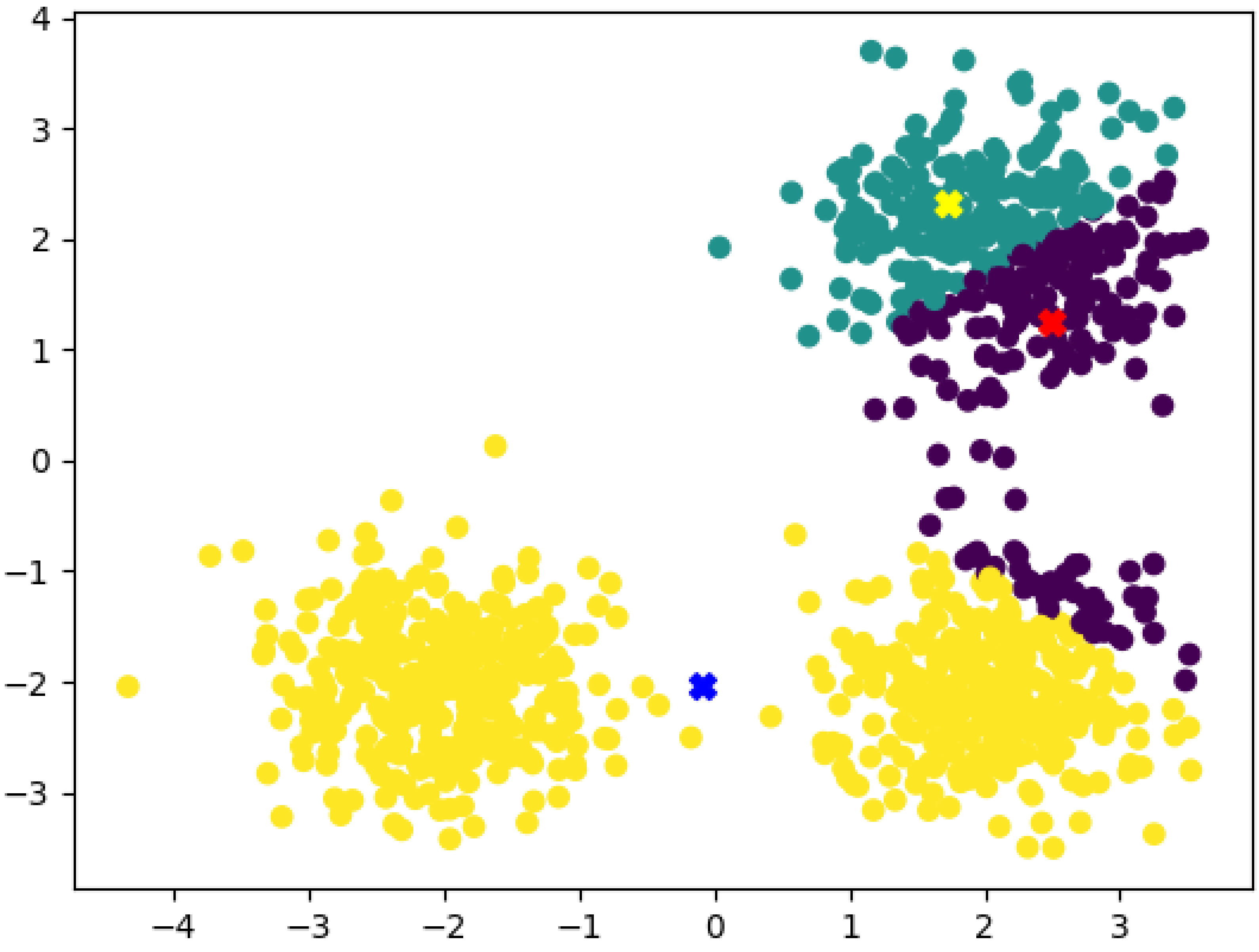
K-Means Algorithm: Example



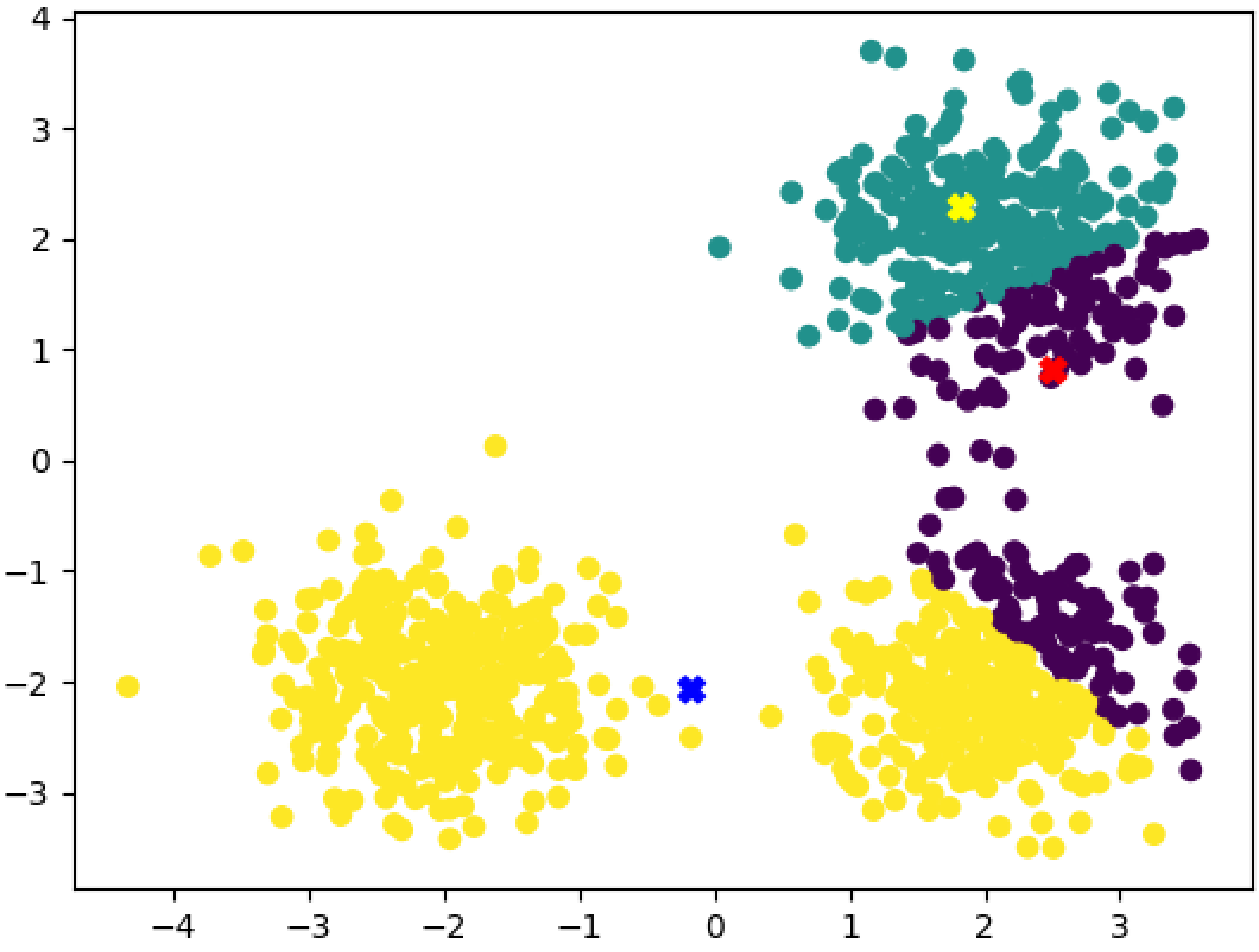
K-Means Algorithm: Example



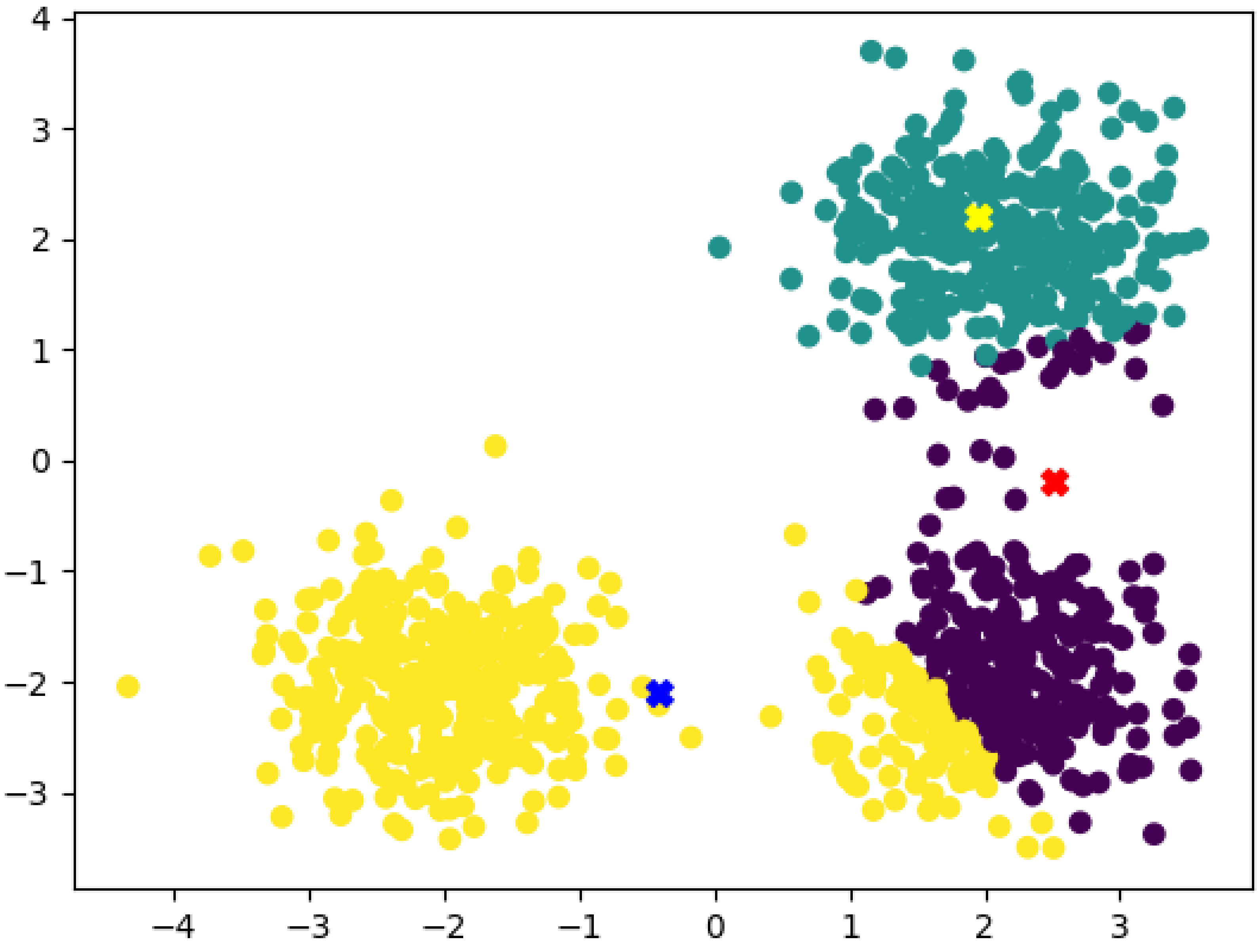
K-Means Algorithm: Example



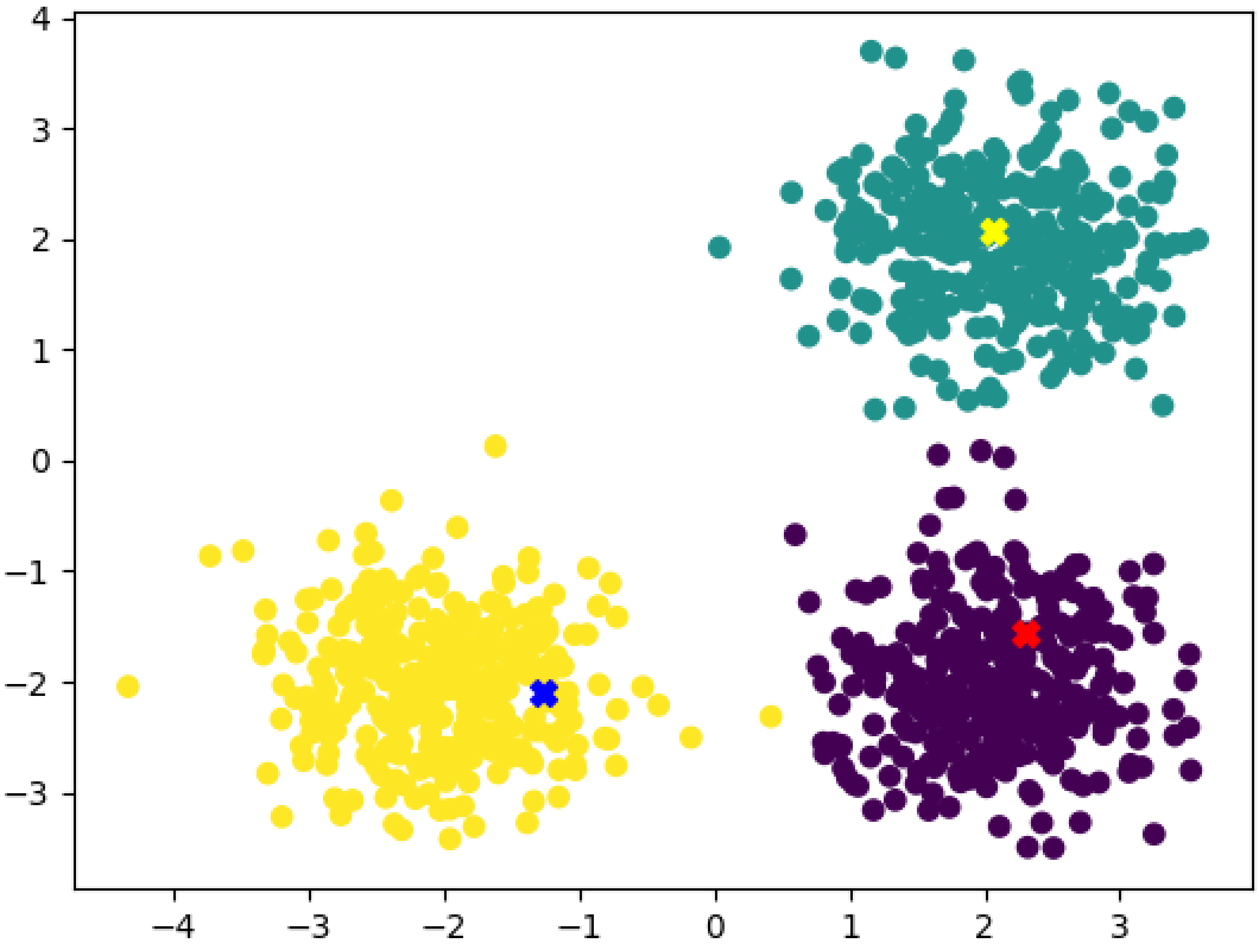
K-Means Algorithm: Example



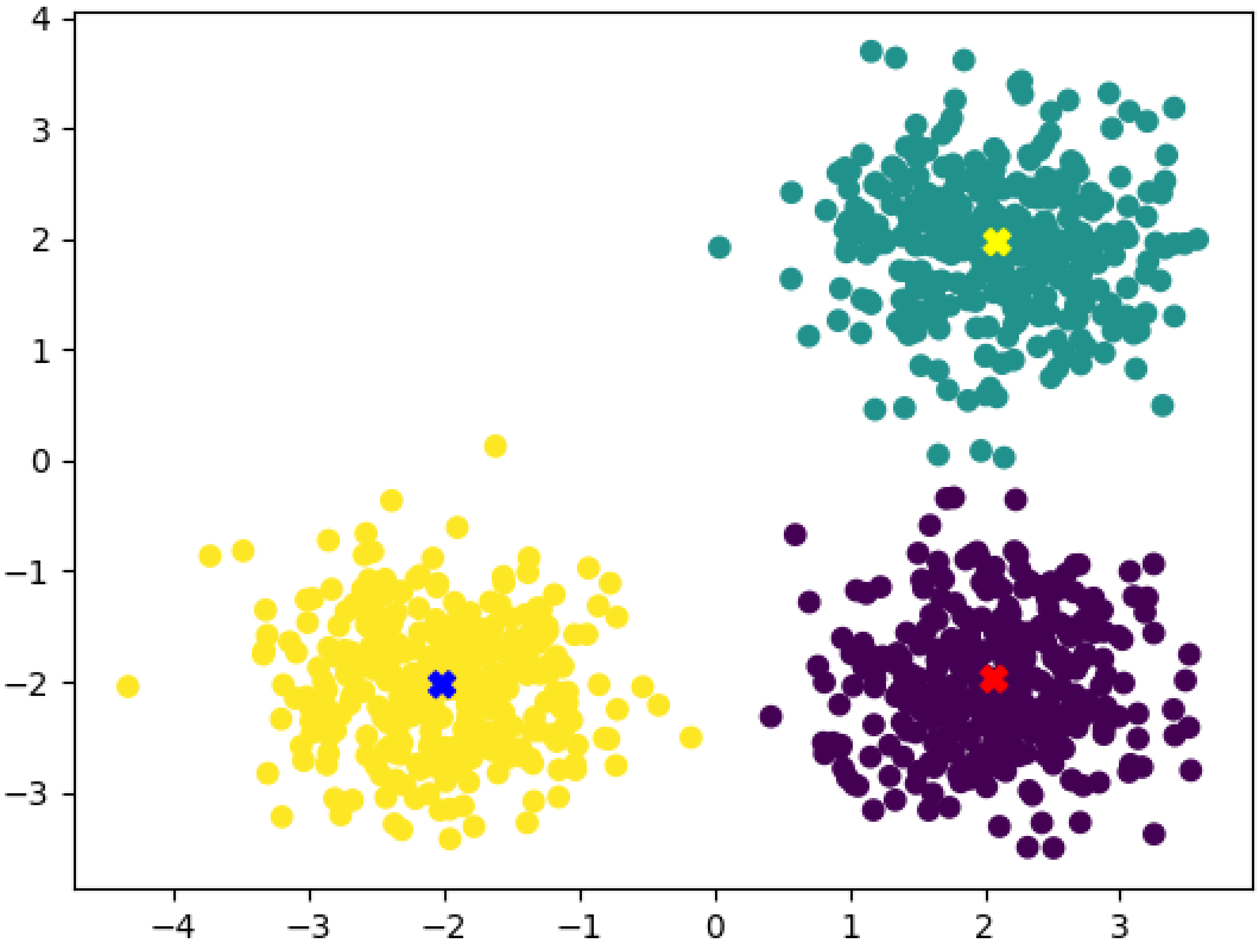
K-Means Algorithm: Example



K-Means Algorithm: Example



K-Means Algorithm: Example



K-Means Algorithm: Complexity

1-Choose randomly k centroids : $c_1, c_2, \dots.., c_K$ in R^n

2-Repeat until convergence

a : We assign each point x in the set to the closest centroid.

b : Update the centroids $c_1, c_2, \dots.., c_K$ as follows :

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

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

1-Choose randomly k centroids : $c_1, c_2, \ldots., c_K$ in R^n

2-Repeat until convergence

a : We assign each point x in the set to the closest centroid.

b : Update the centroids $c_1, c_2, \ldots., c_K$ as follows :

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

Here S_i is the cluster associated with the centroid c_i

K-Means Algorithm: convergence

- What exactly is the optimization function of this algorithm ?

Cost function of K-means $$\sum_{i} \sum_{x \in S_i} D(x, c_i)^2$$

This is the total squared distance from the centroid to the points of the cluster associated to the centroid

K-Means Algorithm: convergence

Cost function of K-means $$\sum_i \sum_{x \in S_i} D(x, c_i)^2$$

This is the total squared distance from the center to the points of the cluster associated to the center

- 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 ?

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

Cost function of K-means $\sum_i \sum_{x \in S_i} D(x, c_i)^2$

This is the total squared distance from the center to the points of the cluster associated to the center

- Which part of the algorithm guarantees the algorithm tried to minimize the above function ?

1-Choose randomly k centroids : $c_1, c_2, \ldots.., c_K$ in R^n

2-Repeat until convergence

a : We assign each point x in the set to the closest centroid.

b : Update the centroids $c_1, c_2, \ldots.., c_K$ as follows :

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

Here S_i is the cluster associated with the centroid c_i

K-Means Algorithm: convergence

Cost function of K-means $$\sum_i \sum_{x \in S_i} D(x, c_i)^2$$

This is the total squared distance from the center to the points of the cluster associated to the center

- Which part of the algorithm guarantees the algorithm tried to minimize the above function ?

1-Choose randomly k centroids : $c_1, c_2, \ldots, c_K$ in R^n

2-Repeat until convergence

a : We assign each point x in the set to the closest centroid.

b : Update the centroids $c_1, c_2, \ldots, c_K$ as follows :

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

Here S_i is the cluster associated with the centroid c_i

Minimize the cost function with respect to the clusters

Minimize the cost function with respect to the centroids

K-Means Algorithm: problems

K-means has the following problems :

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

sklearn example [here]

K-Means Algorithm: problems

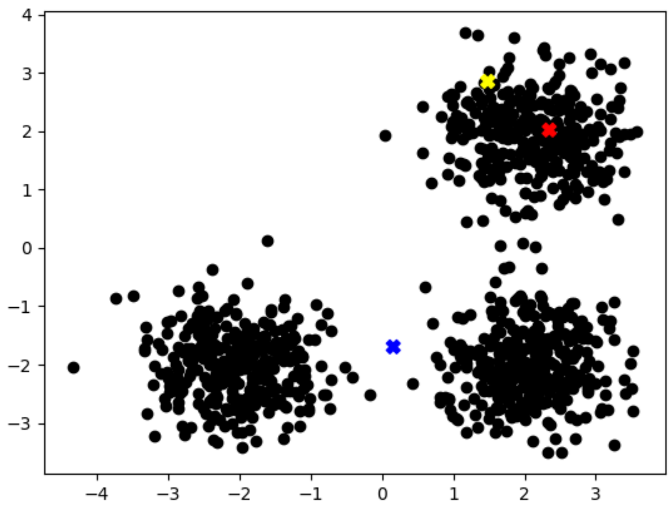
K-means has the following problems :

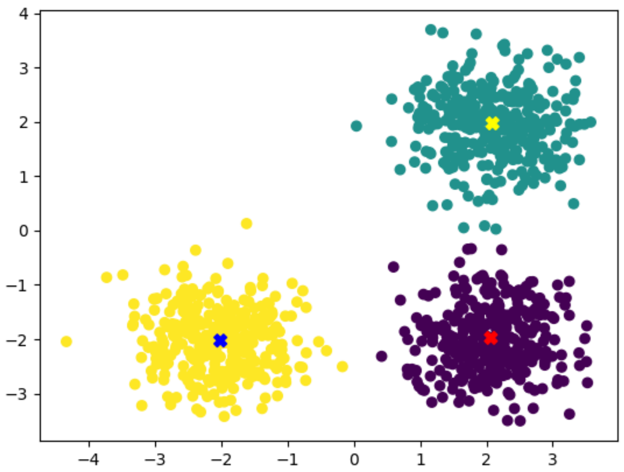
- Outliers
- Clusters with different densities
- Non-convex shapes
- Clusters with different sizes
- May converge to local optimum

Look up what the term *convex* means exactly!

sklearn example here

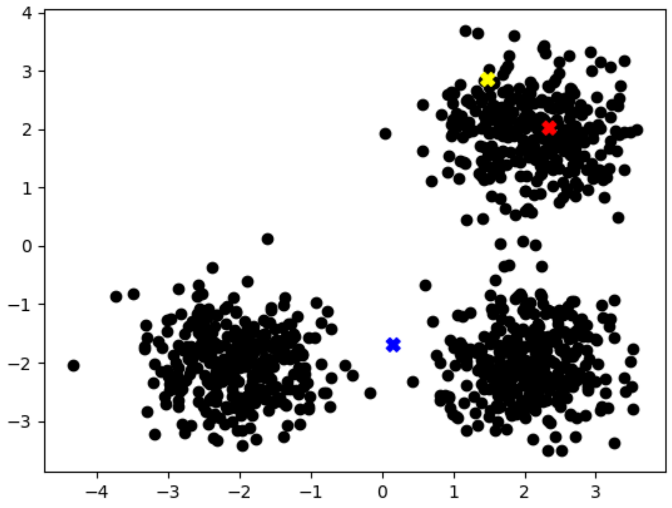
K-Means Algorithm: remarks

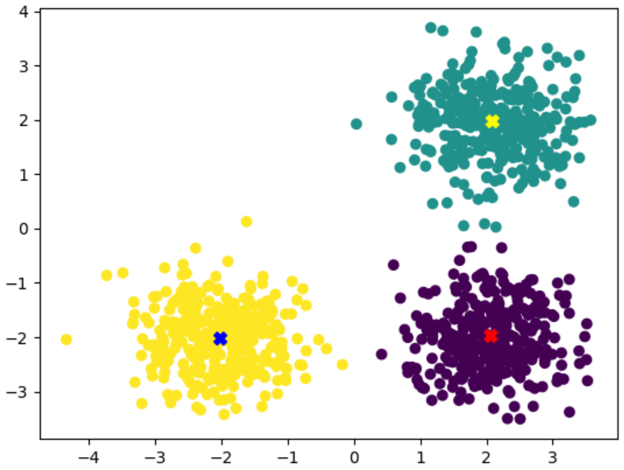




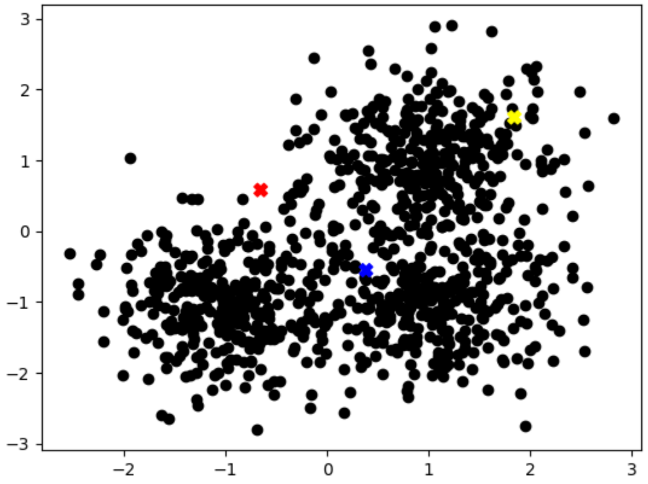
K-means is clearly applicable to data sets where the clusters are very well-separated

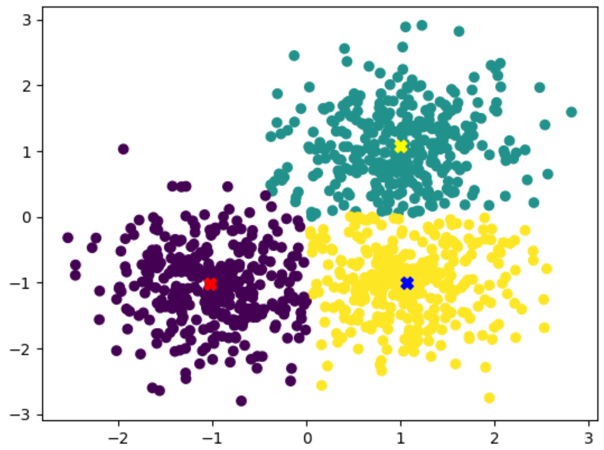
K-Means Algorithm: remarks





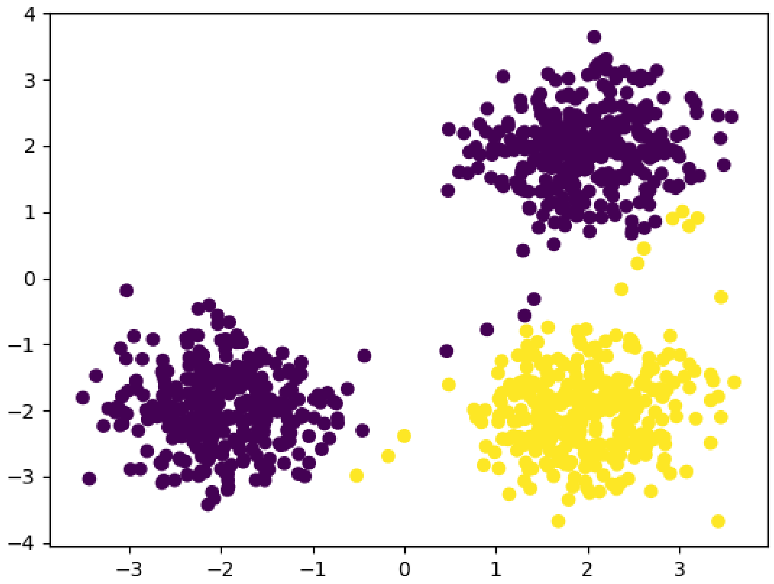
K-means is clearly applicable to data sets where the clusters are very well-separated

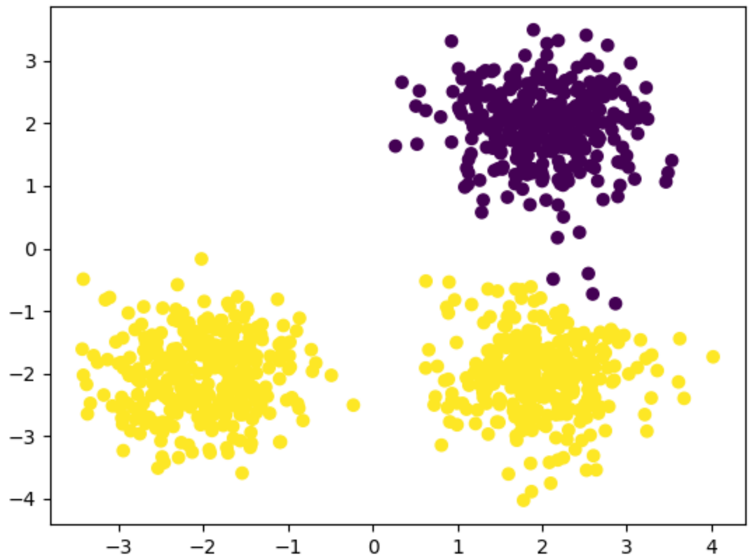




K-Means is often applied to data that have no clear clustering structure

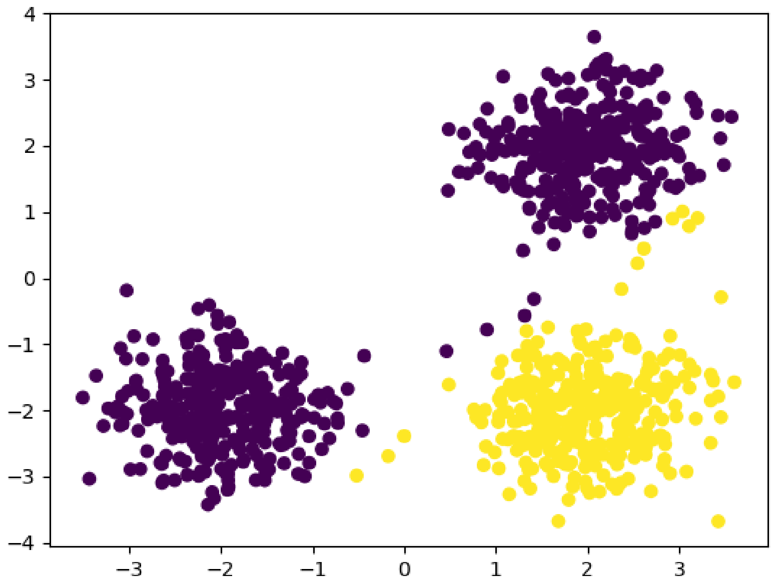
K-Means Algorithm: remarks

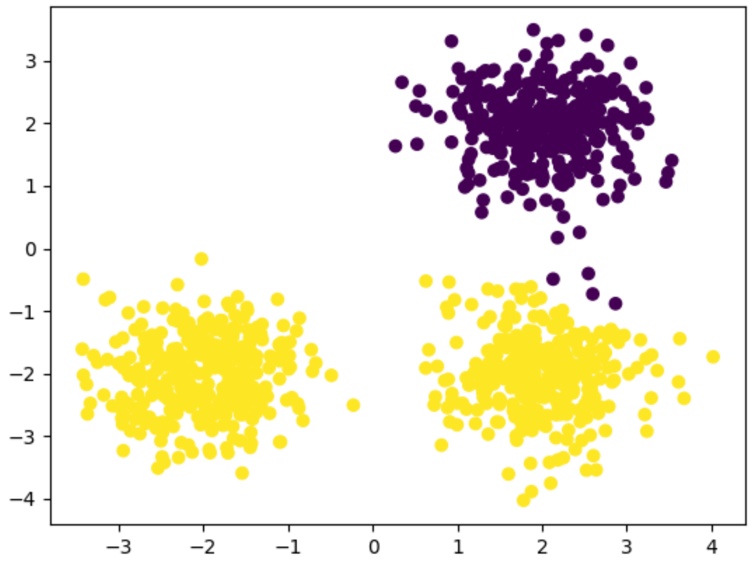




K-means may get stuck in a local optimum

K-Means Algorithm: remarks





K-means may get stuck in a local optimum

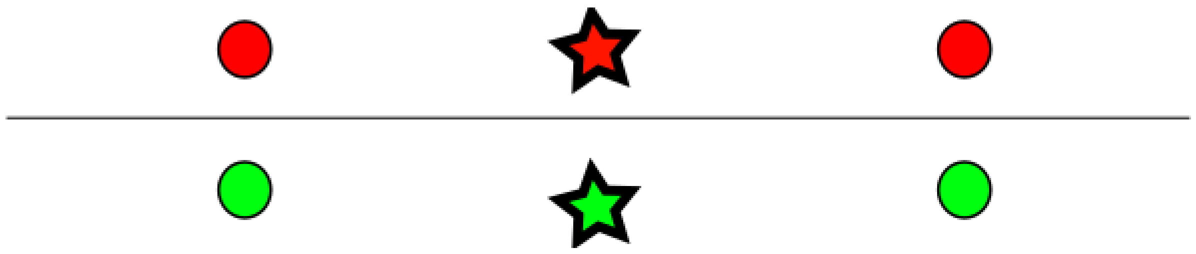
One solution for this is to run K-means several times and pick the attempt that minimizes the cost function

K-Means Algorithm: remarks

Nearby points may not end in the same cluster

K-Means Algorithm: remarks

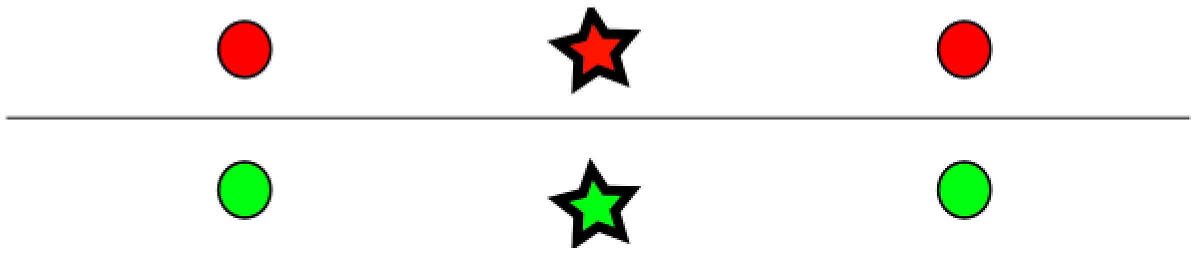
Nearby points may not end in the same cluster



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

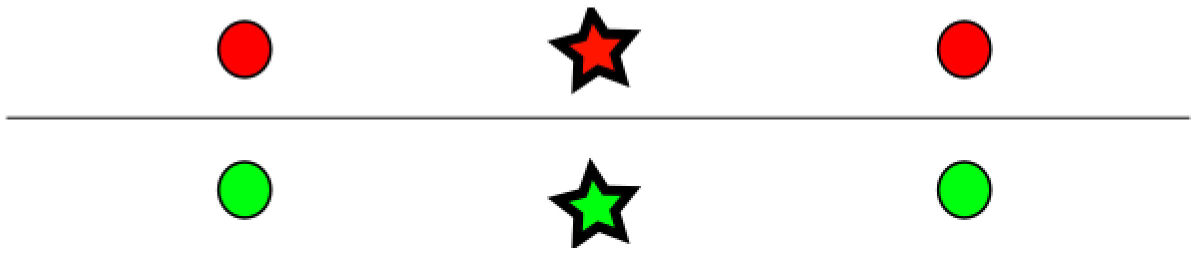


It is possible that K-means gets stuck in this local optimum.

How would you solve this problem ?

K-Means Algorithm: remarks

Nearby points may not end in the same cluster

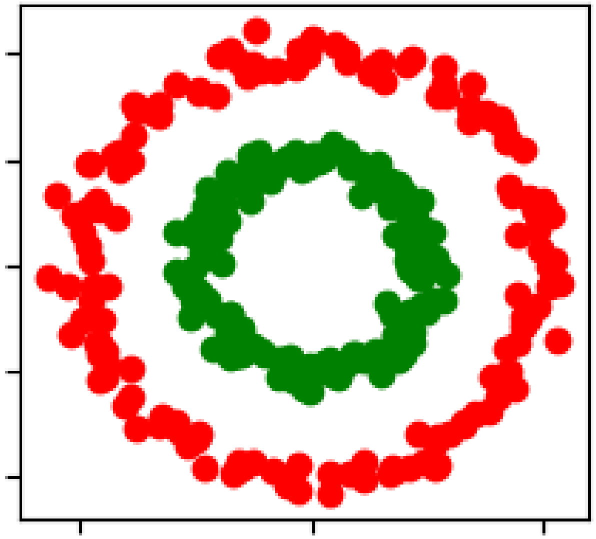


It is possible that K-means gets stuck in this local optimum.

How would you solve this problem ?

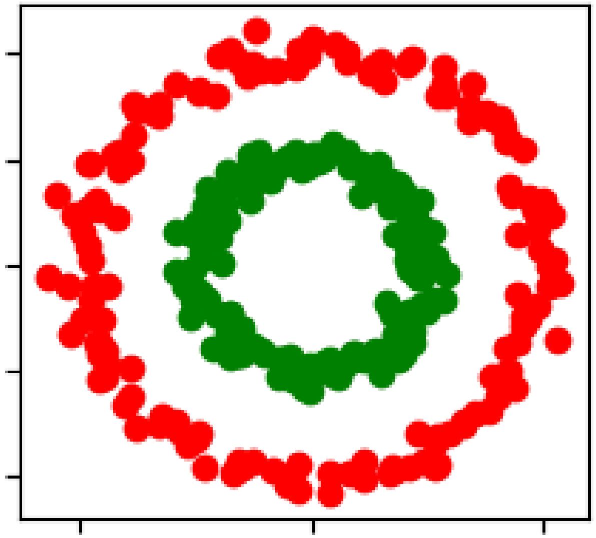
Try different initialization and choose the one that produces the best result (wrt the cost function).

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 :

$$(x, y) \to \left(\sqrt{x^2 + y^2}, \tan^{-1}\left(\frac{y}{x}\right)\right) \quad (convert\ the\ feature\ vector\ to\ polar\ coordinates)$$

or change the distance function used in the K-means algorithm.

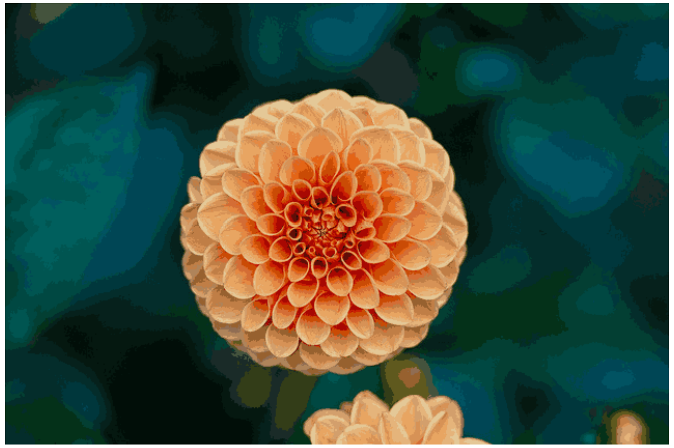
Application: Color Quantization

K-means can be used to reduce the number of colors needed to in image. In this example we can reduce the number of colors from 62941 unique colors to 128, while maintain the overall quality.

62941 colors

128 colors

64 colors



32 colors

16 colors

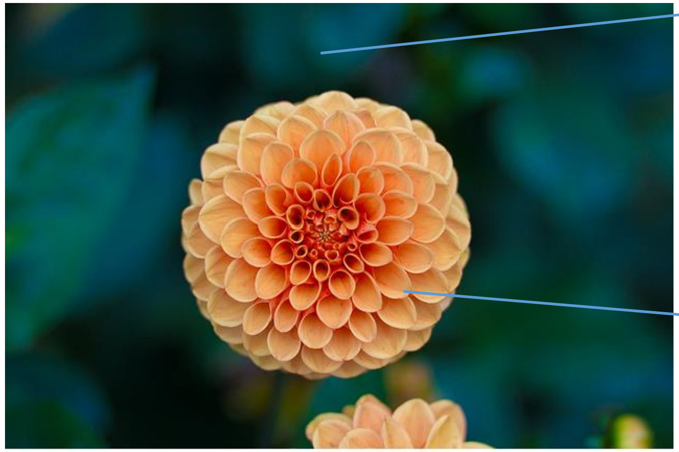
Application: Color Quantization

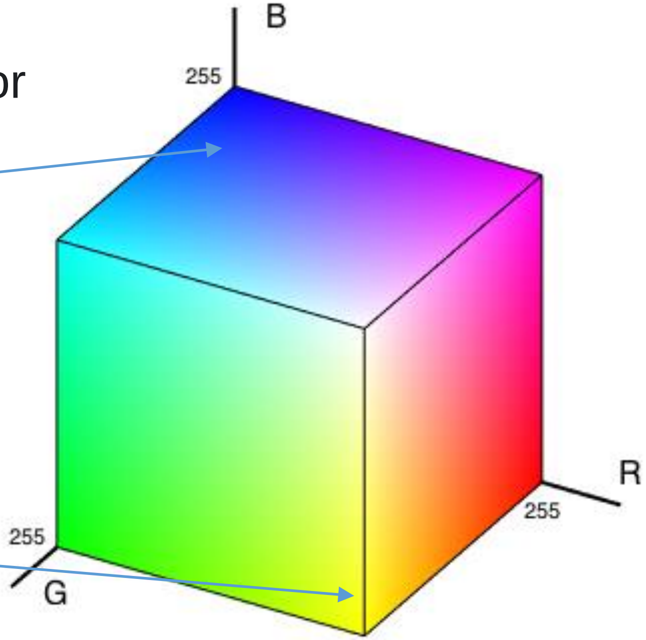
Main idea:

Recall that every pixel in an 8-bit color image is represented by three numbers (r, g, b) where r, g and b are integers between 0 and $2^8 = 255$ (the range that a single 8-bit can offer).

The idea is to consider only the pixel colors of the image and think of them as being points in RGB cube in R^3.
The K-means clustering is then performed on this data set consisting of the all points (r, g, b) in R^3corresponding to the pixels in the image.

Each cluster center is then chosen to be the representative color of that cluster and mapped back to the image.





sklearn code example [here]

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