

EE2211 Introduction to Machine Learning

Lecture 11

Wang Xinchao
xinchao@nus.edu.sg

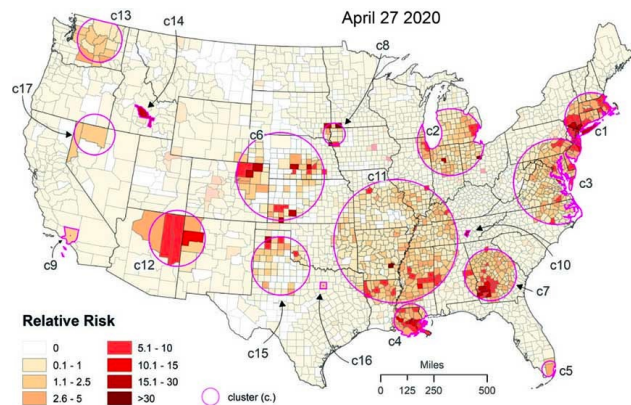
Course Contents

- Introduction and Preliminaries (Xinchao)
 - Introduction
 - Data Engineering
 - Introduction to Linear Algebra, Probability and Statistics
 - Fundamental Machine Learning Algorithms I (Yueming)
 - Systems of linear equations
 - Least squares, Linear regression
 - Ridge regression, Polynomial regression
 - Fundamental Machine Learning Algorithms II (Yueming)
 - Over-fitting, bias/variance trade-off
 - Optimization, Gradient descent
 - Decision Trees, Random Forest
 - Performance and More Algorithms (Xinchao)
 - Performance Issues
 - **K-means Clustering**
 - Neural Networks
- [Important] In the Final, no coding questions for Xinchao's part!**
- Despite you will see some in the tutorial, they won't be tested.

Outline

- Introduction of unsupervised learning
- K-means Clustering
 - The most popular clustering technique
- Fuzzy Clustering

Unsupervised Learning



Discovering Covid clusters



Business analysis

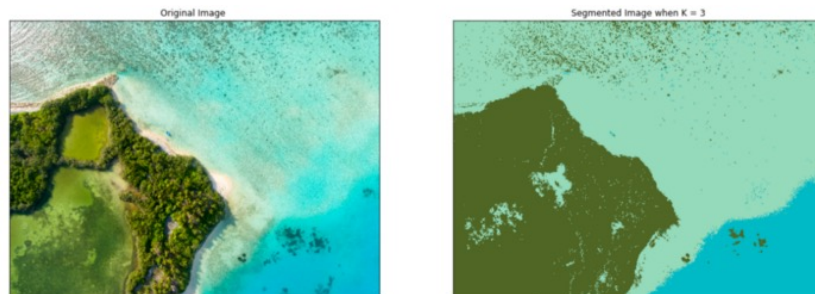
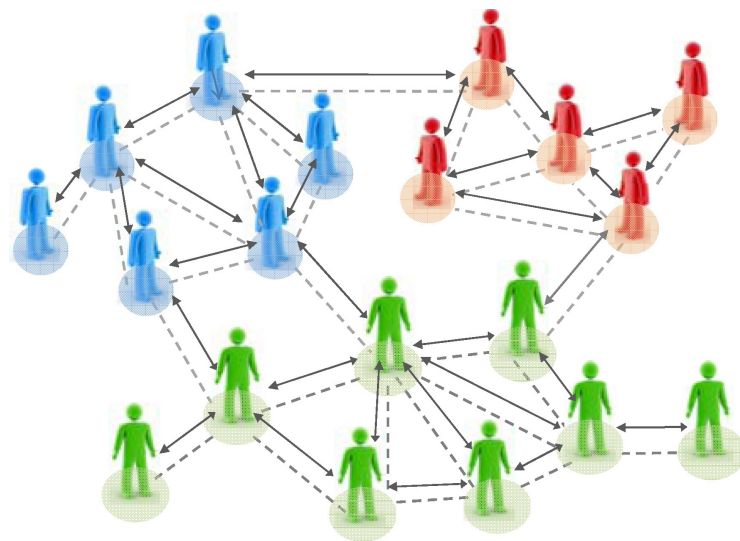


Image segmentation



Community detection in social networks

Unsupervised Learning

Introduction

Motivation: we do not always have labeled data.

In **unsupervised learning**, the dataset is a collection of **unlabeled examples** $\{\mathbf{x}_i\}_{i=1}^M$.

Unsupervised Learning

Introduction

Evaluation of unsupervised learning is hard:

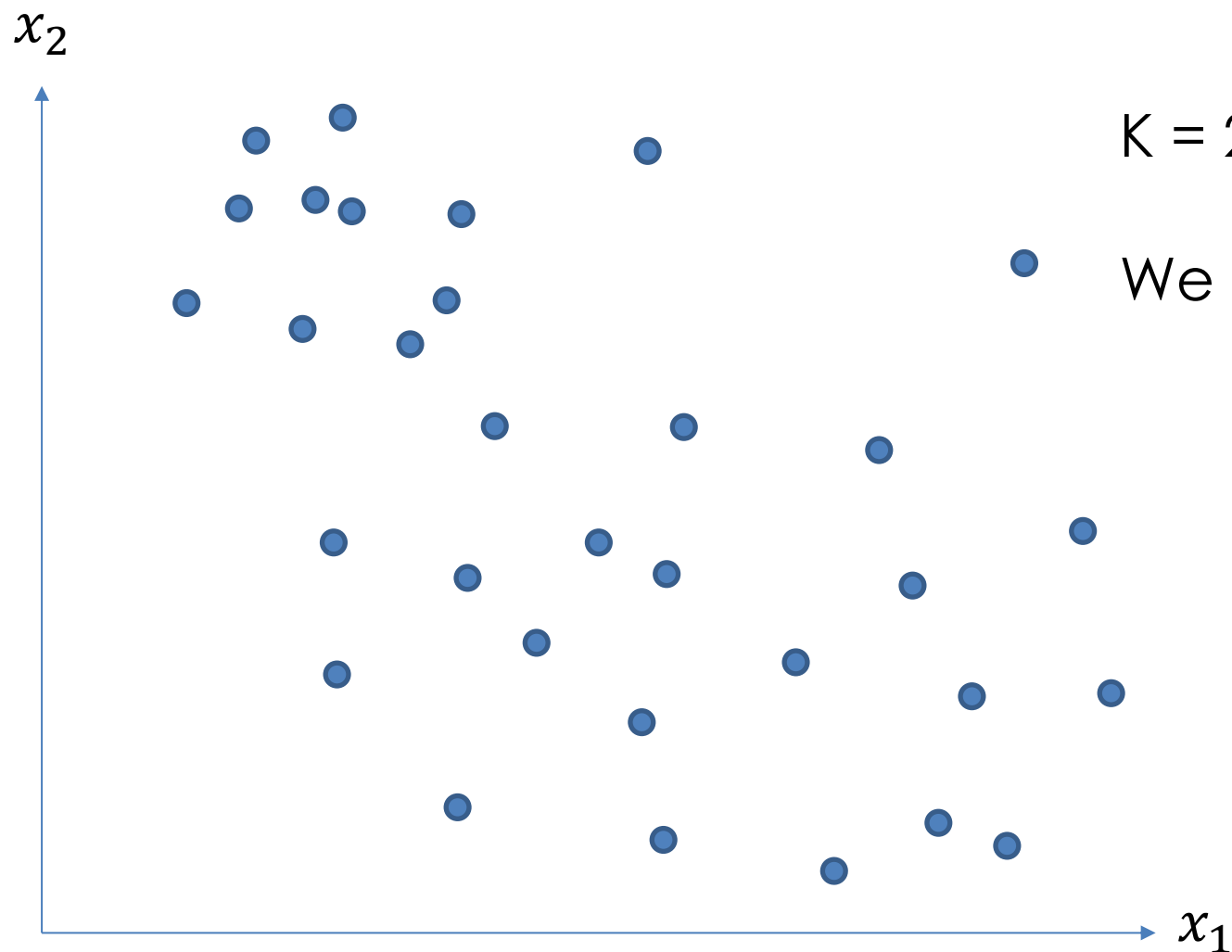
- The **absence of labels** representing the desired behavior for your model means **the absence of a solid reference point to judge the quality of your model.**

Unsupervised Learning

Main Approaches

- **Clustering**
 - ✓ Groups a set of objects in such a way that objects in the same group (called a **cluster**) are **more similar** (in some sense) to each other than to those in other groups (clusters).
- **Density Estimation**
 - ✓ Models the probability density function (pdf) of the unknown probability distribution from which the dataset has been drawn.
- **Component Analysis**
 - ✓ Breaks down the data from the perspective of signal analysis.
- **Unsupervised Neural Networks**
 - ✓ Autoencoder

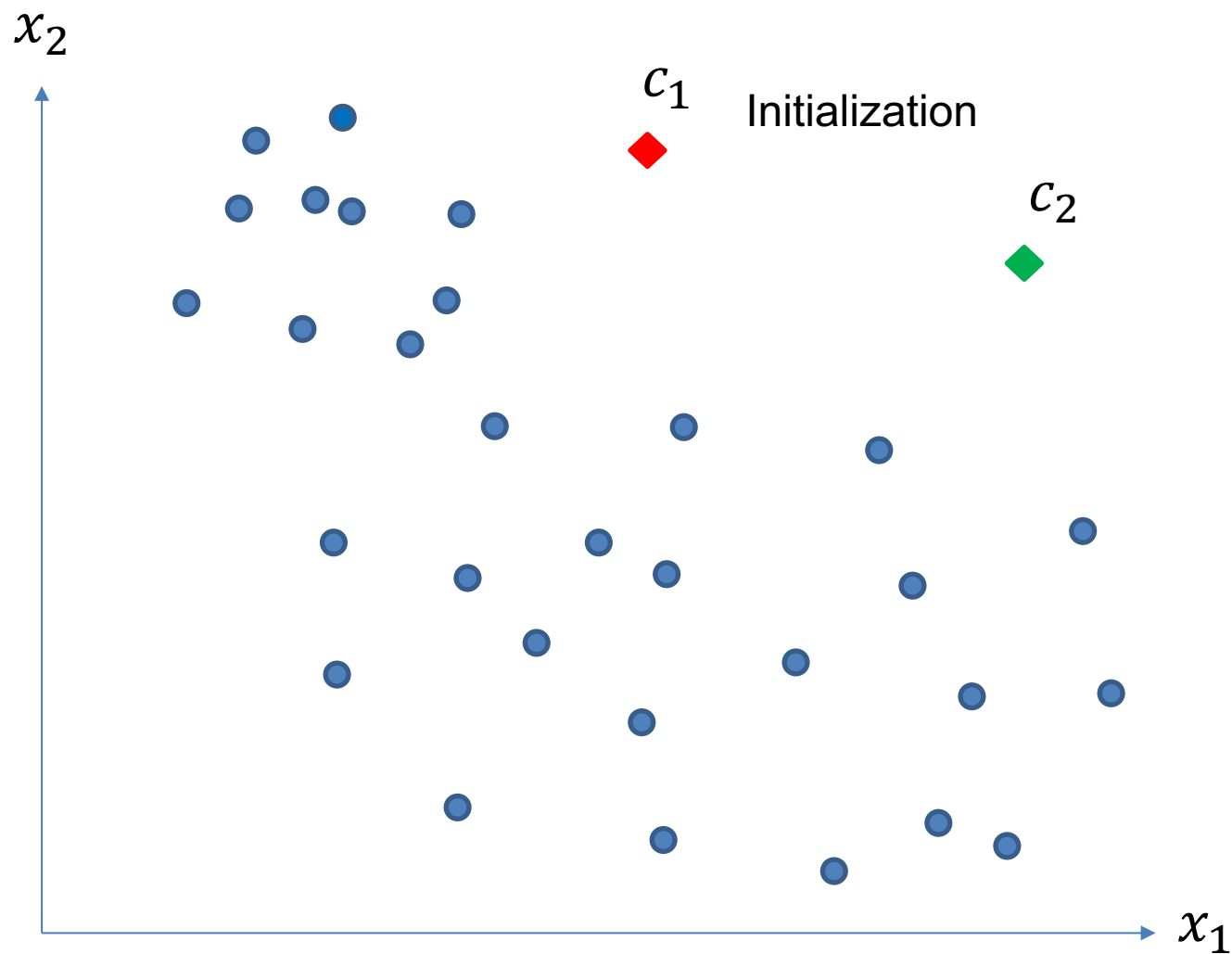
K-means Clustering (2D)



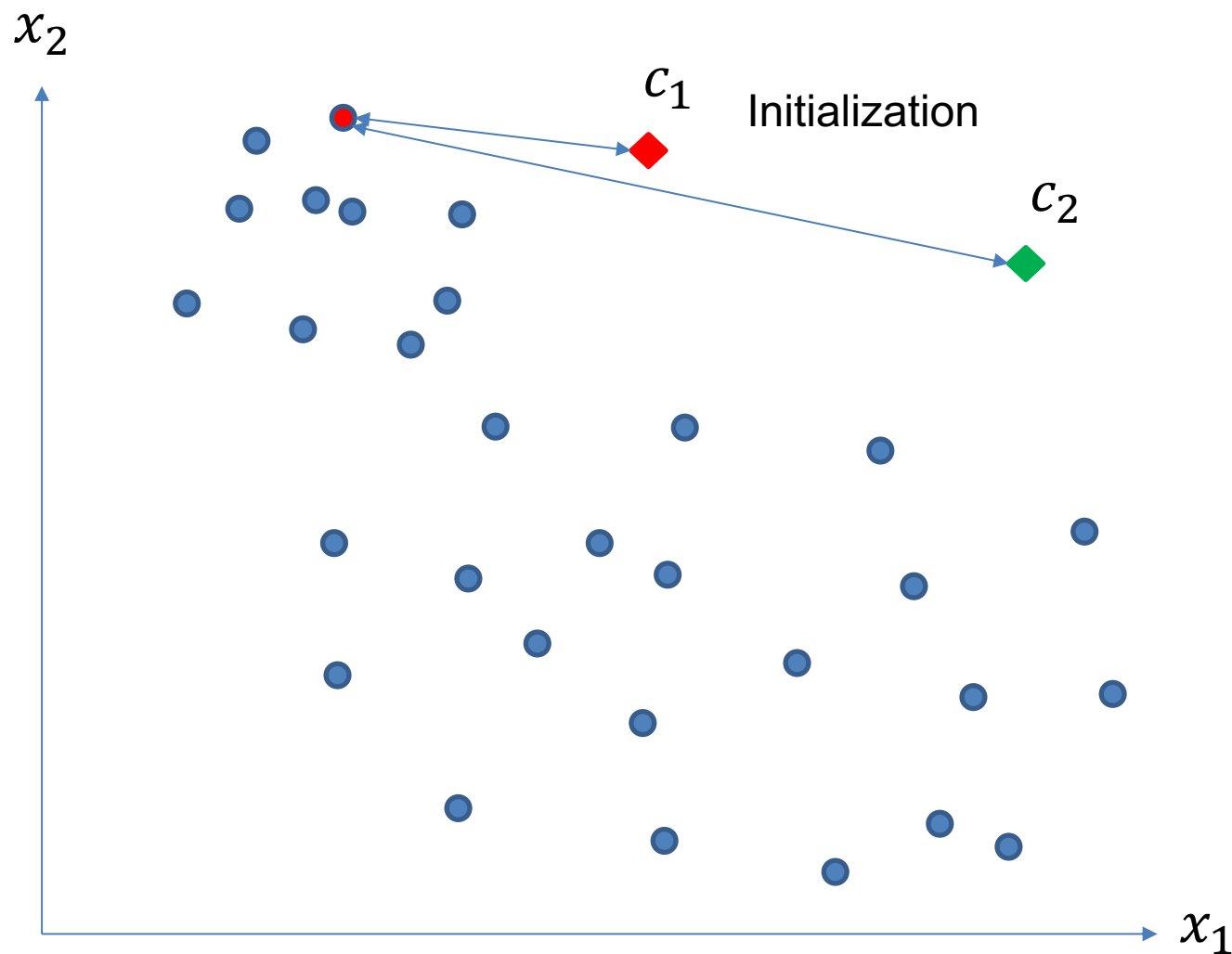
$K = 2$

We seek 2 clusters

K-means Clustering

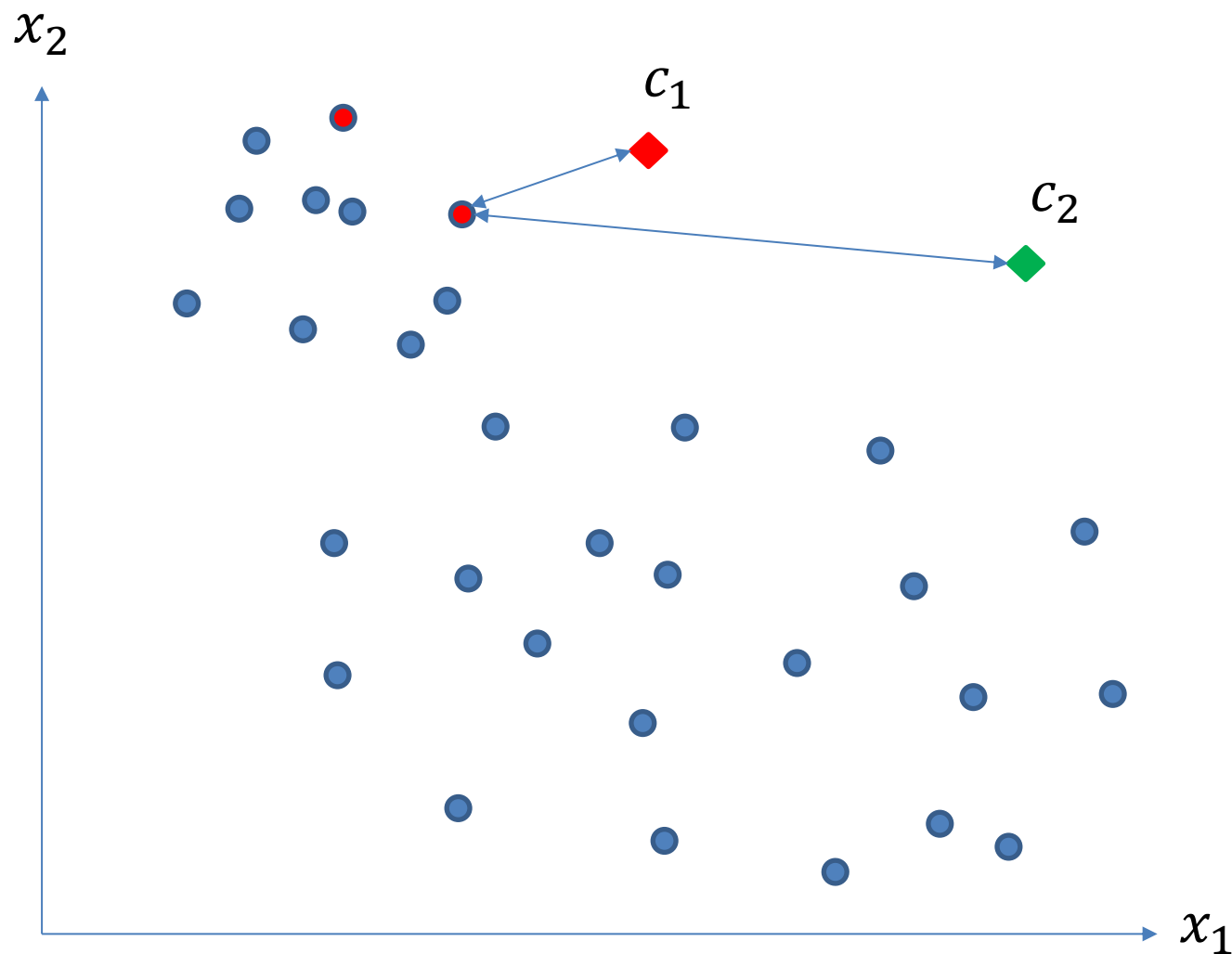


K-means Clustering



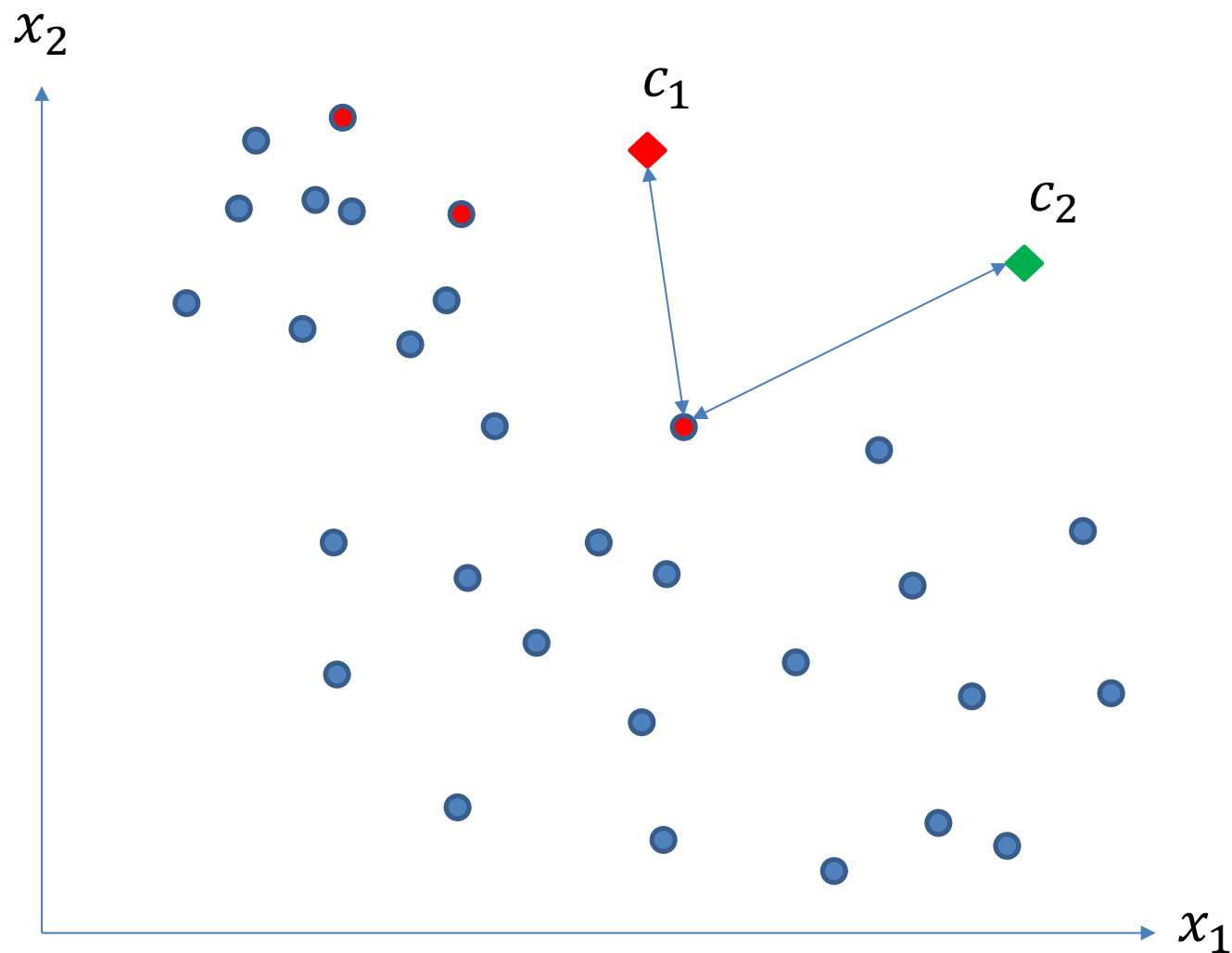
Assignment

K-means Clustering



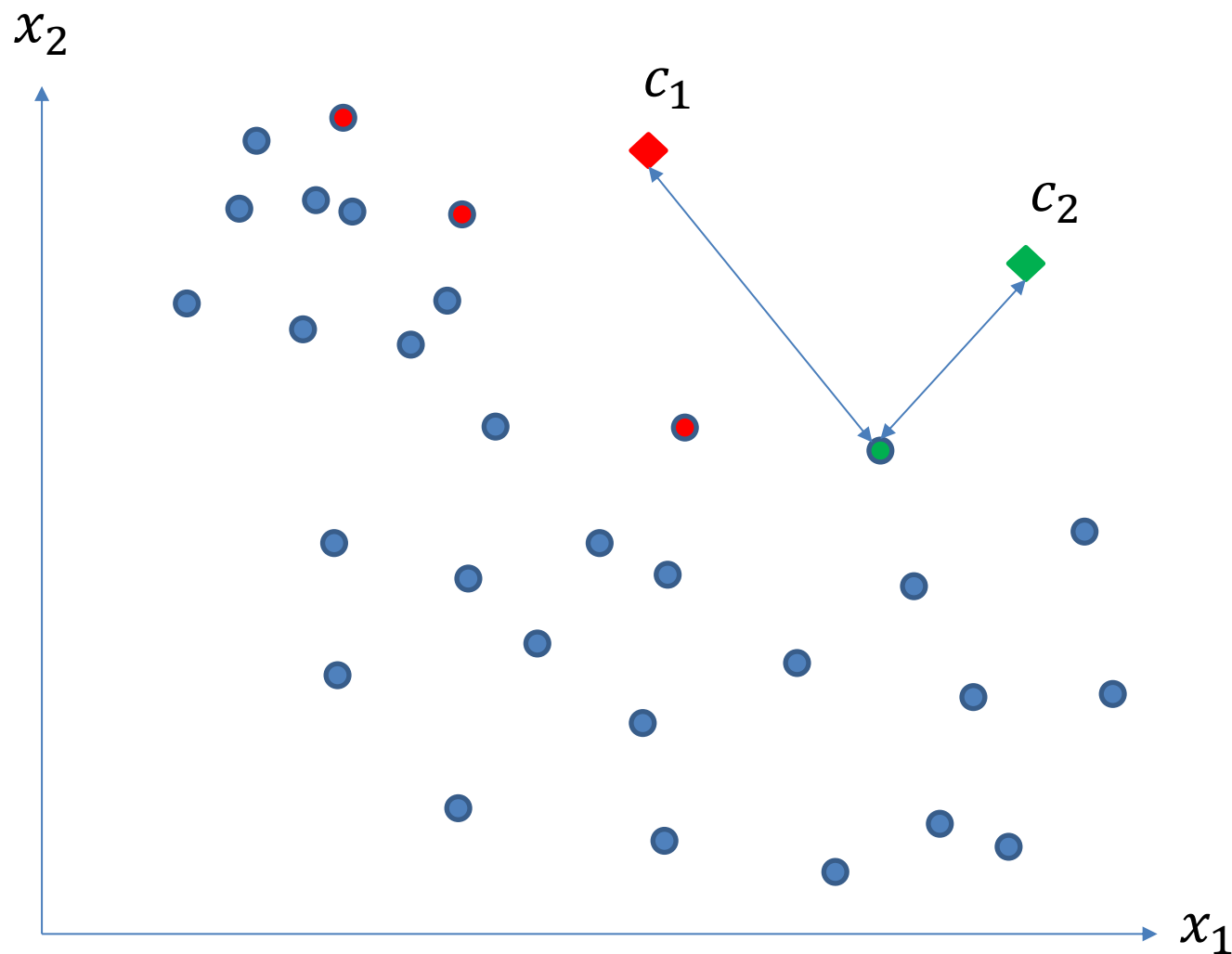
Assignment

K-means Clustering



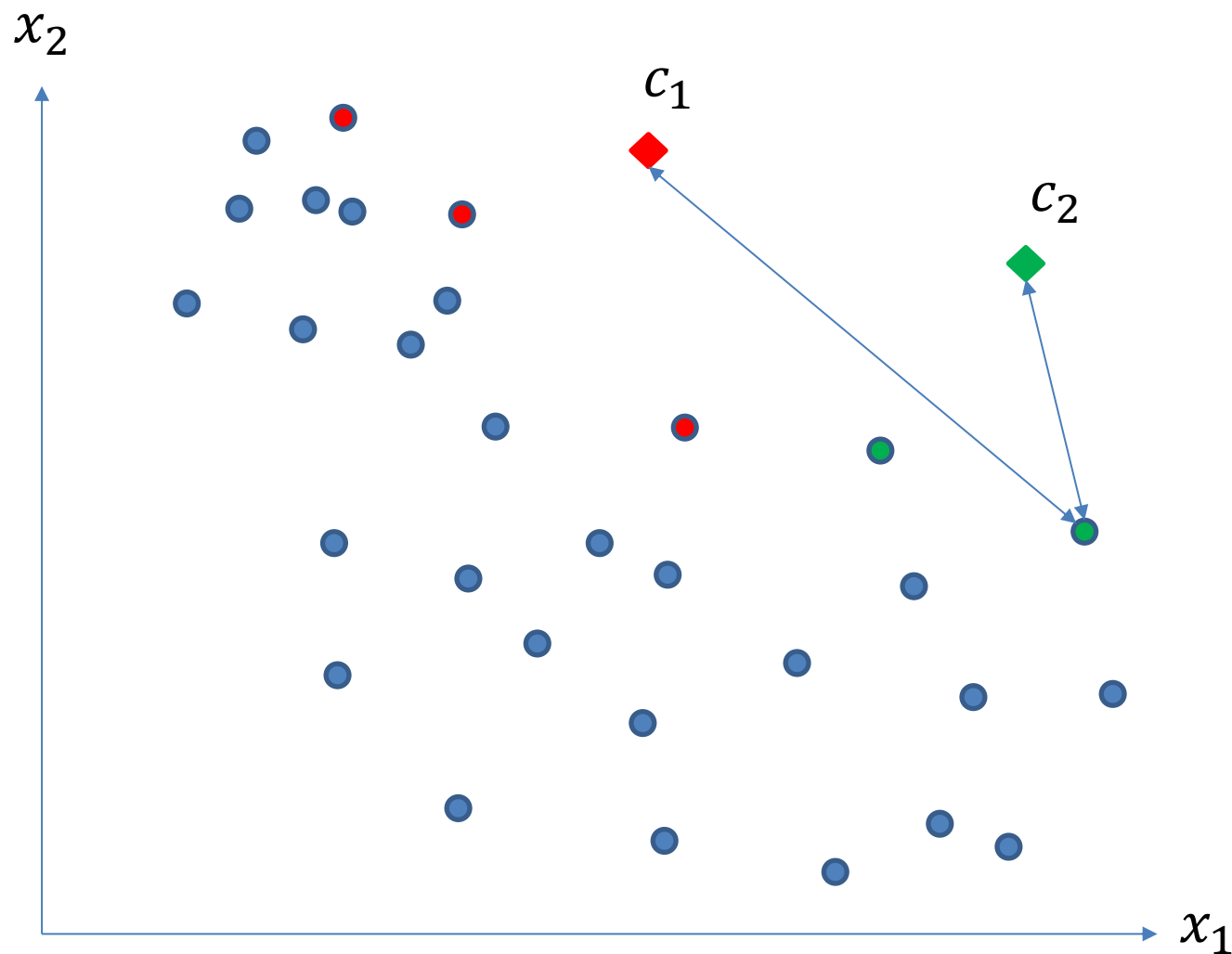
Assignment

K-means Clustering



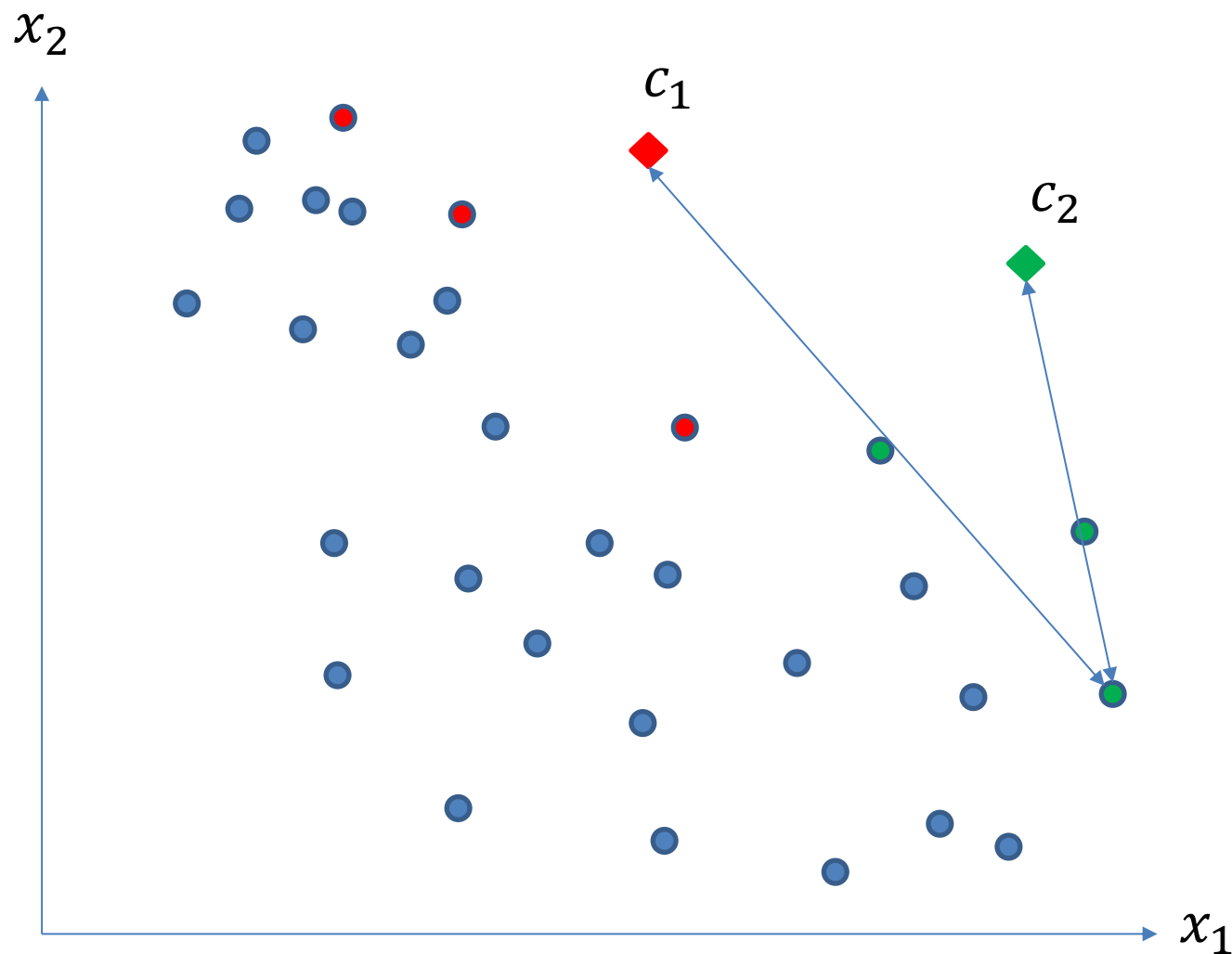
Assignment

K-means Clustering



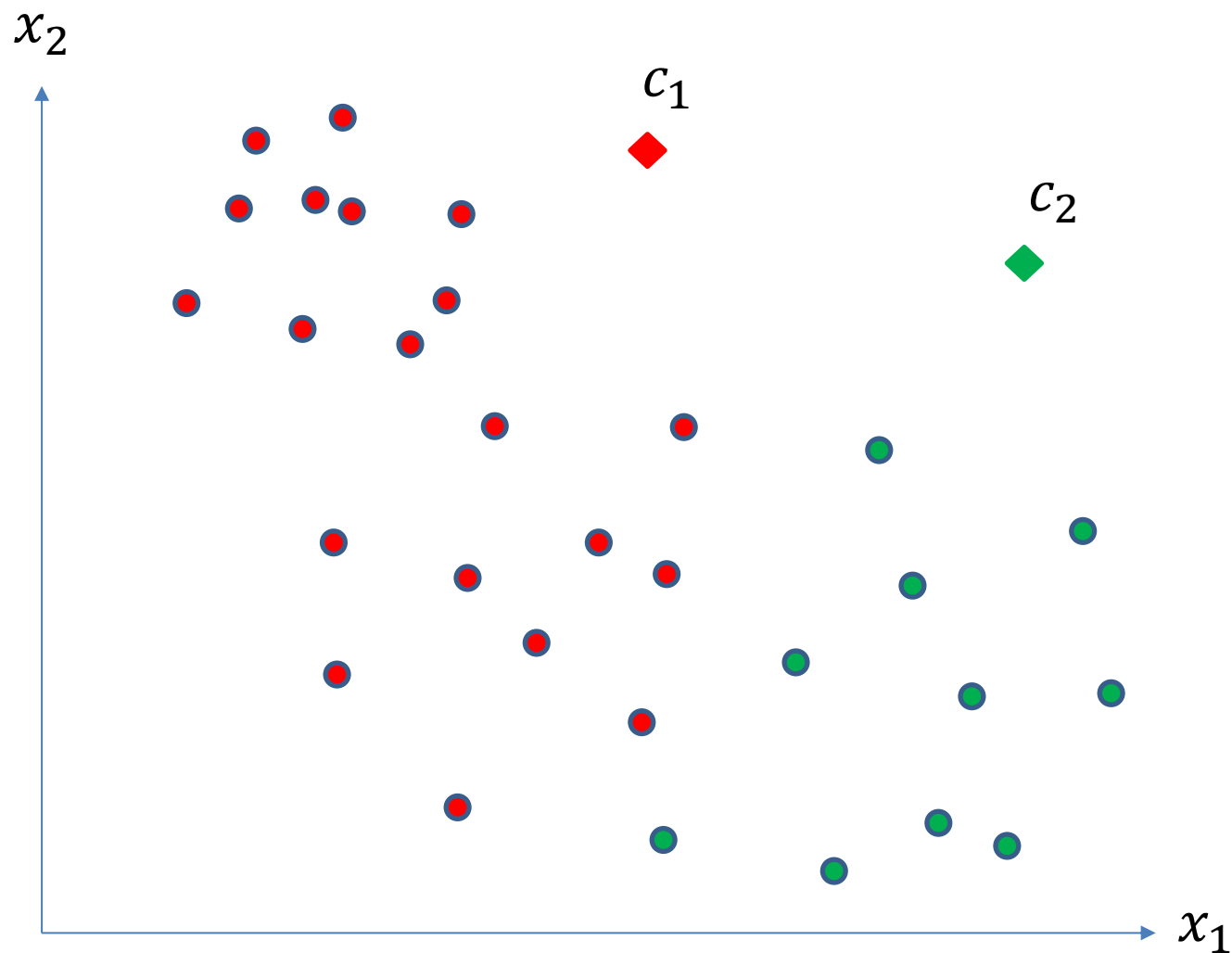
Assignment

K-means Clustering



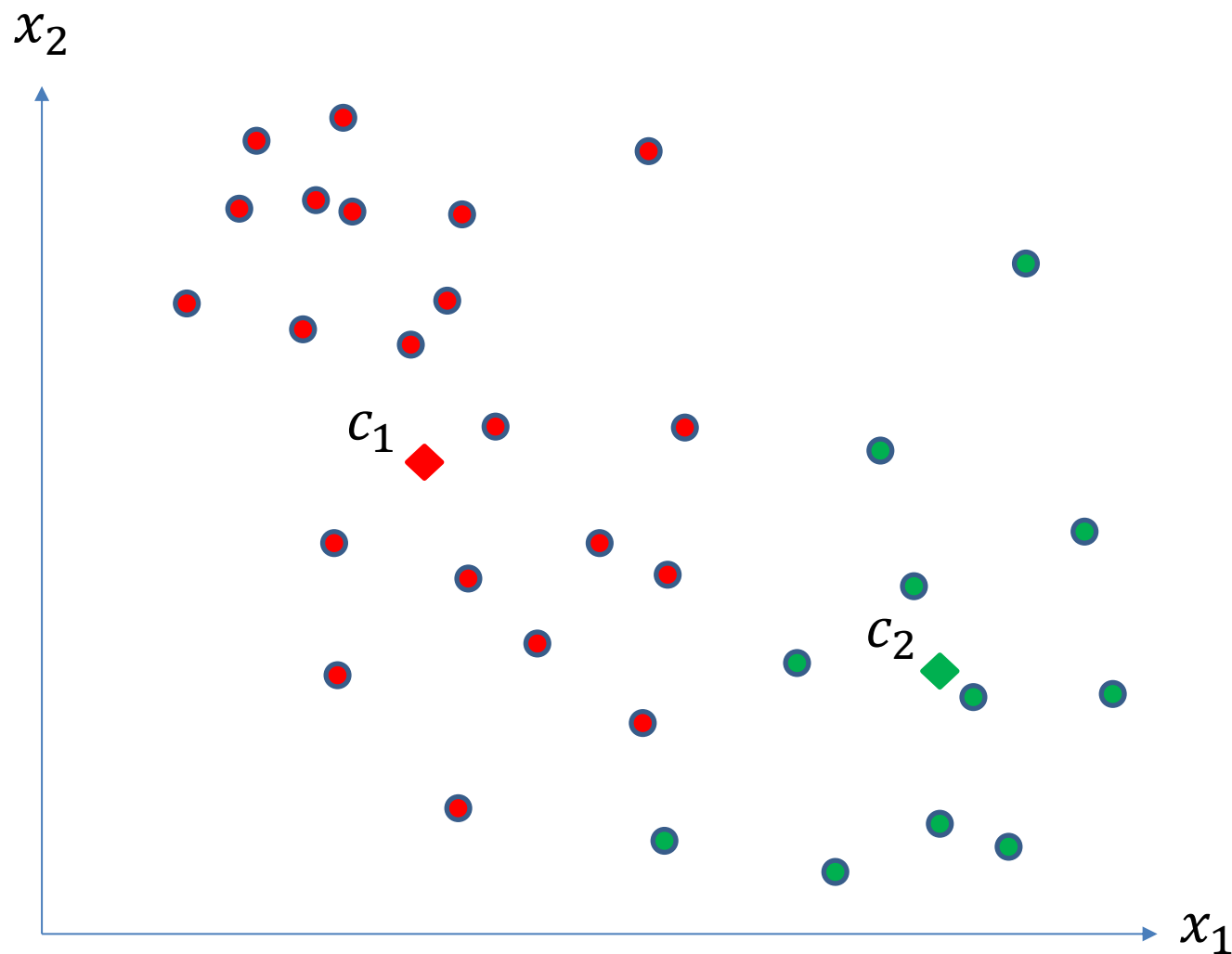
Assignment

K-means Clustering



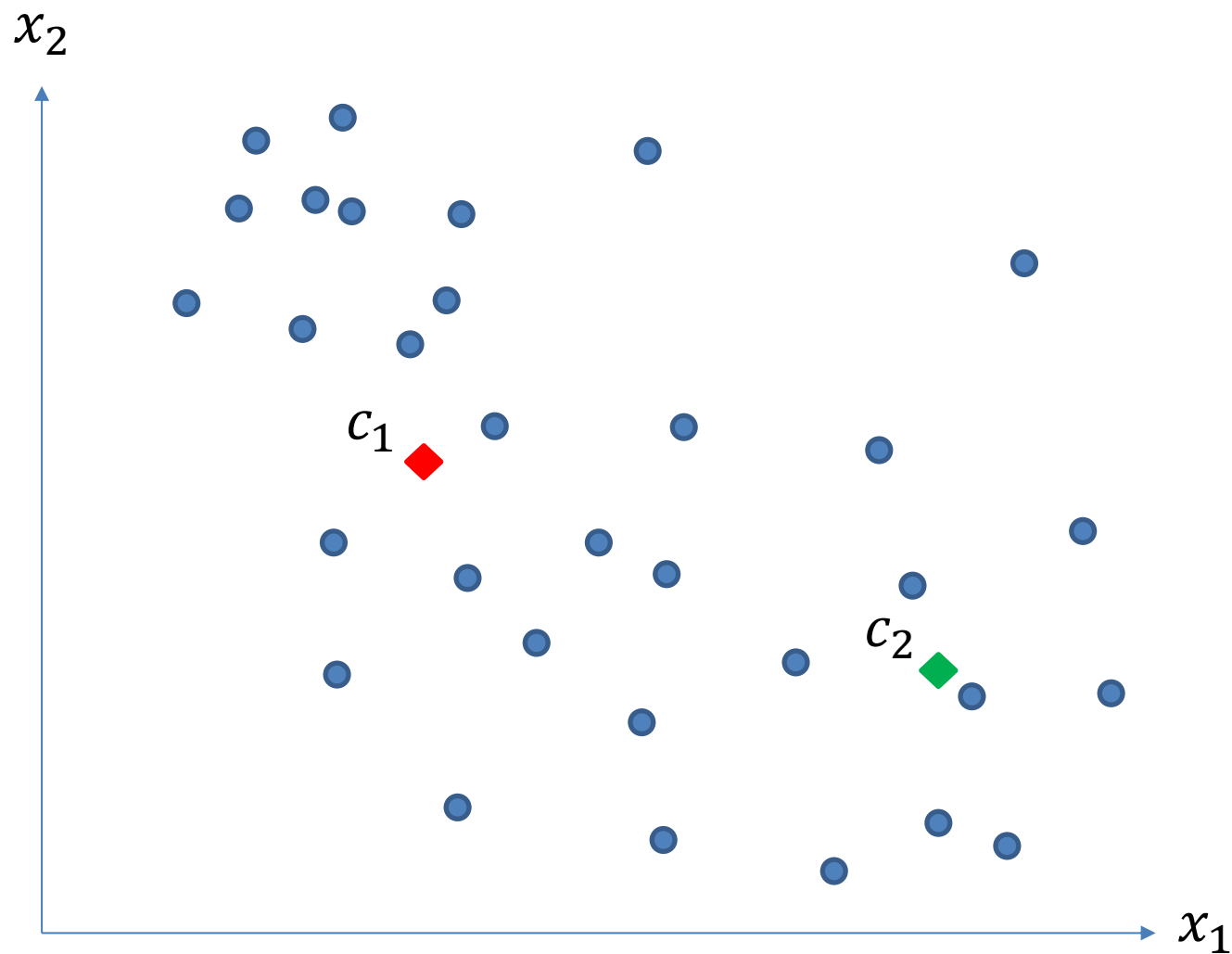
Assignment

K-means Clustering

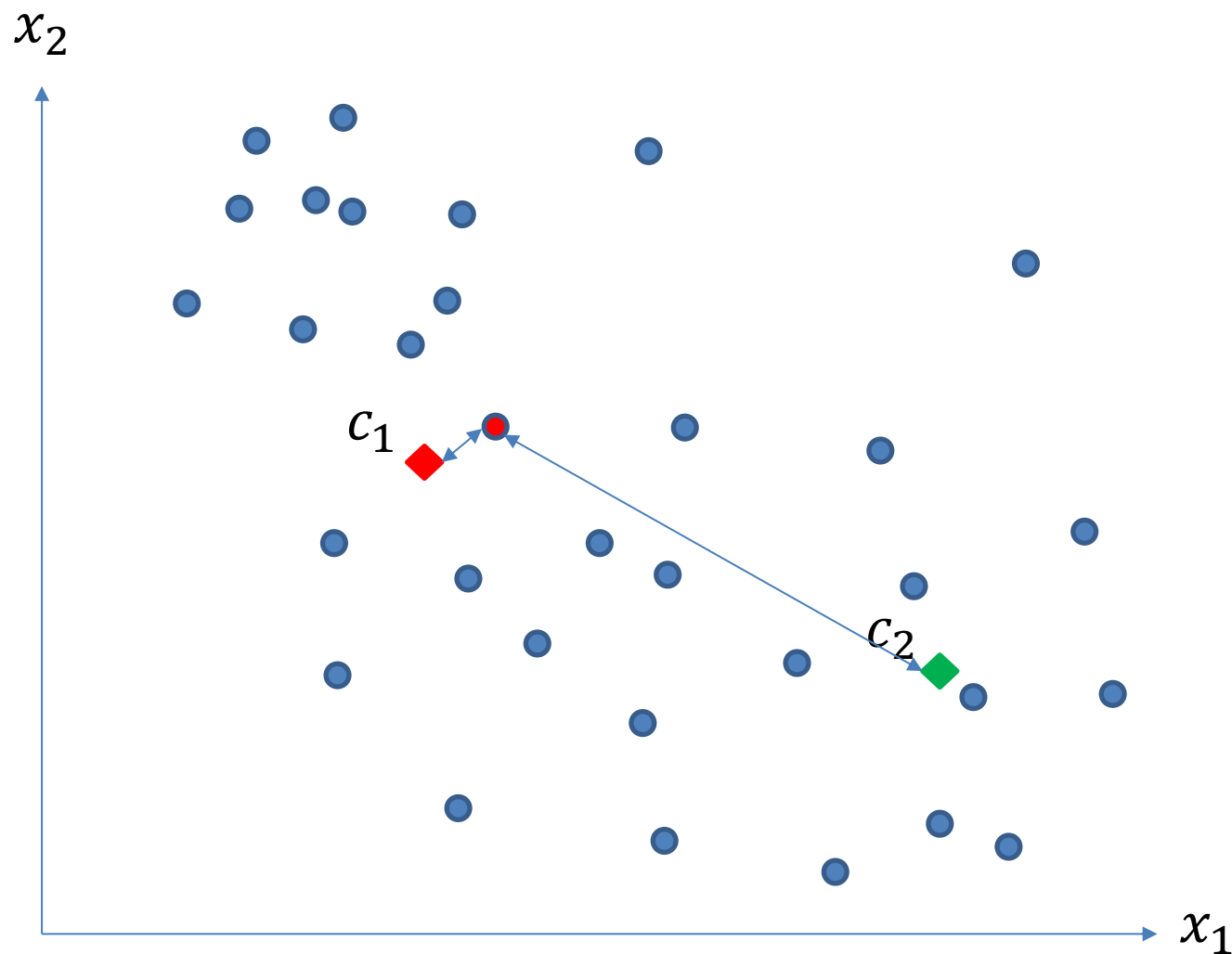


Centroid
Update

K-means Clustering

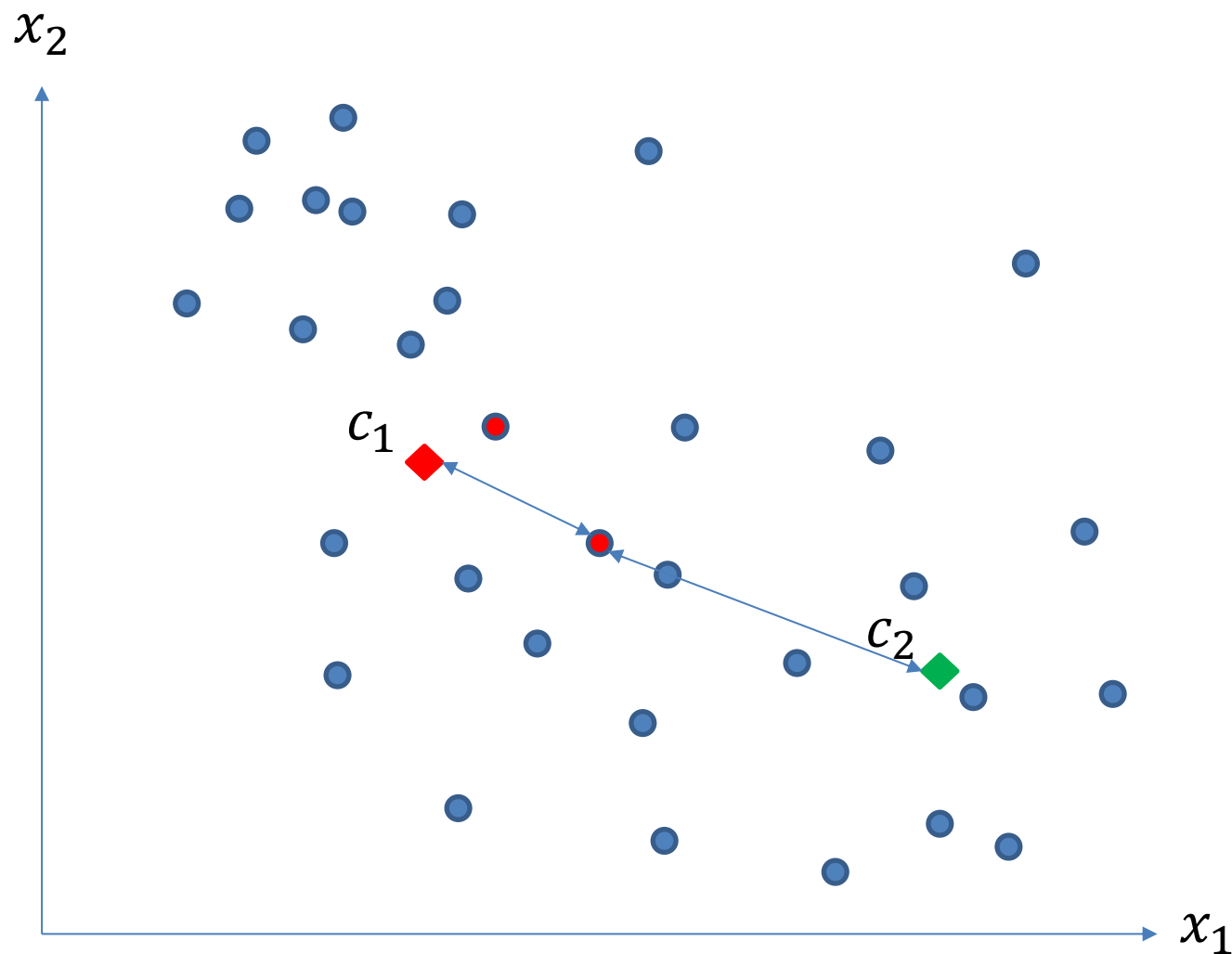


K-means Clustering



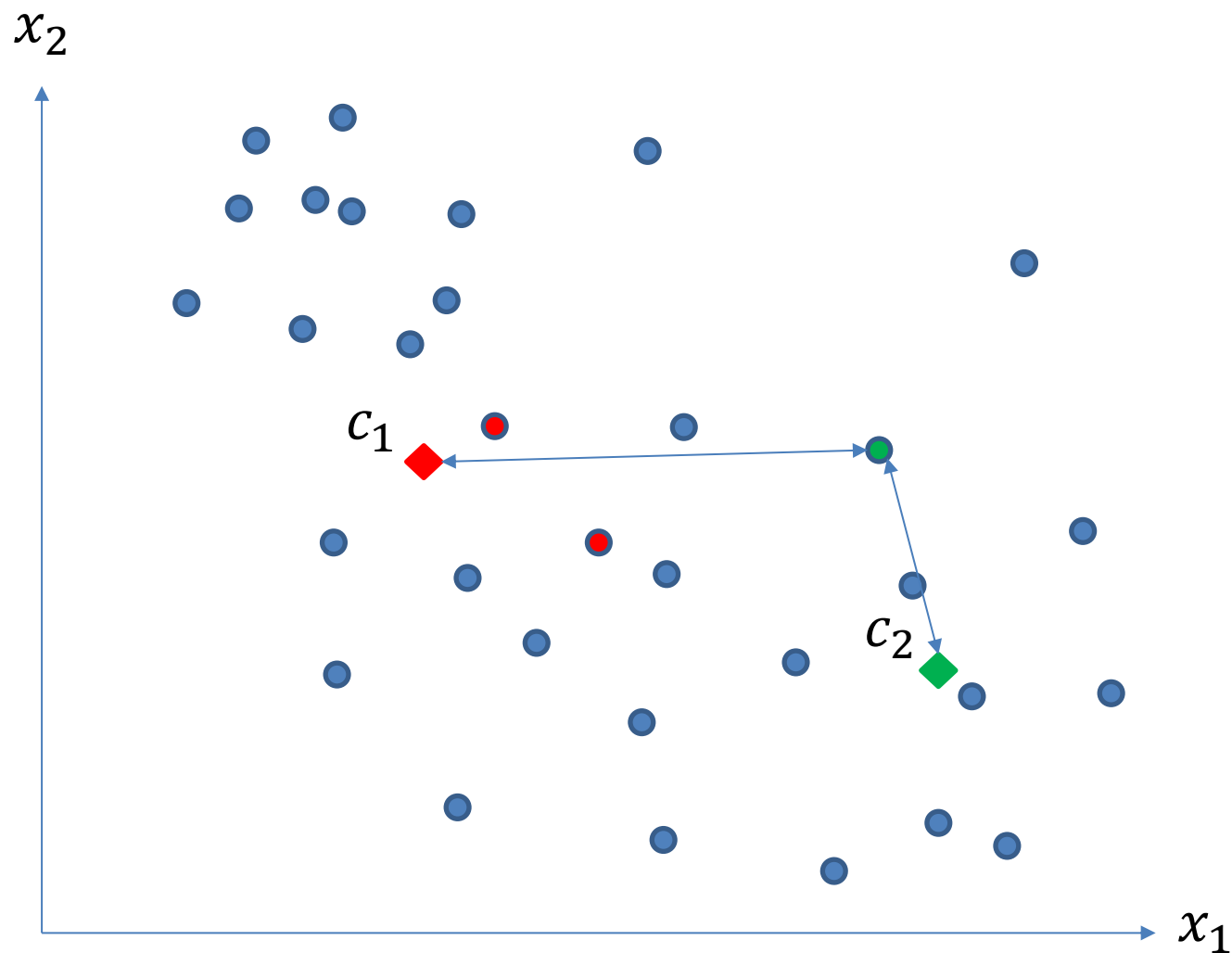
Assignment

K-means Clustering



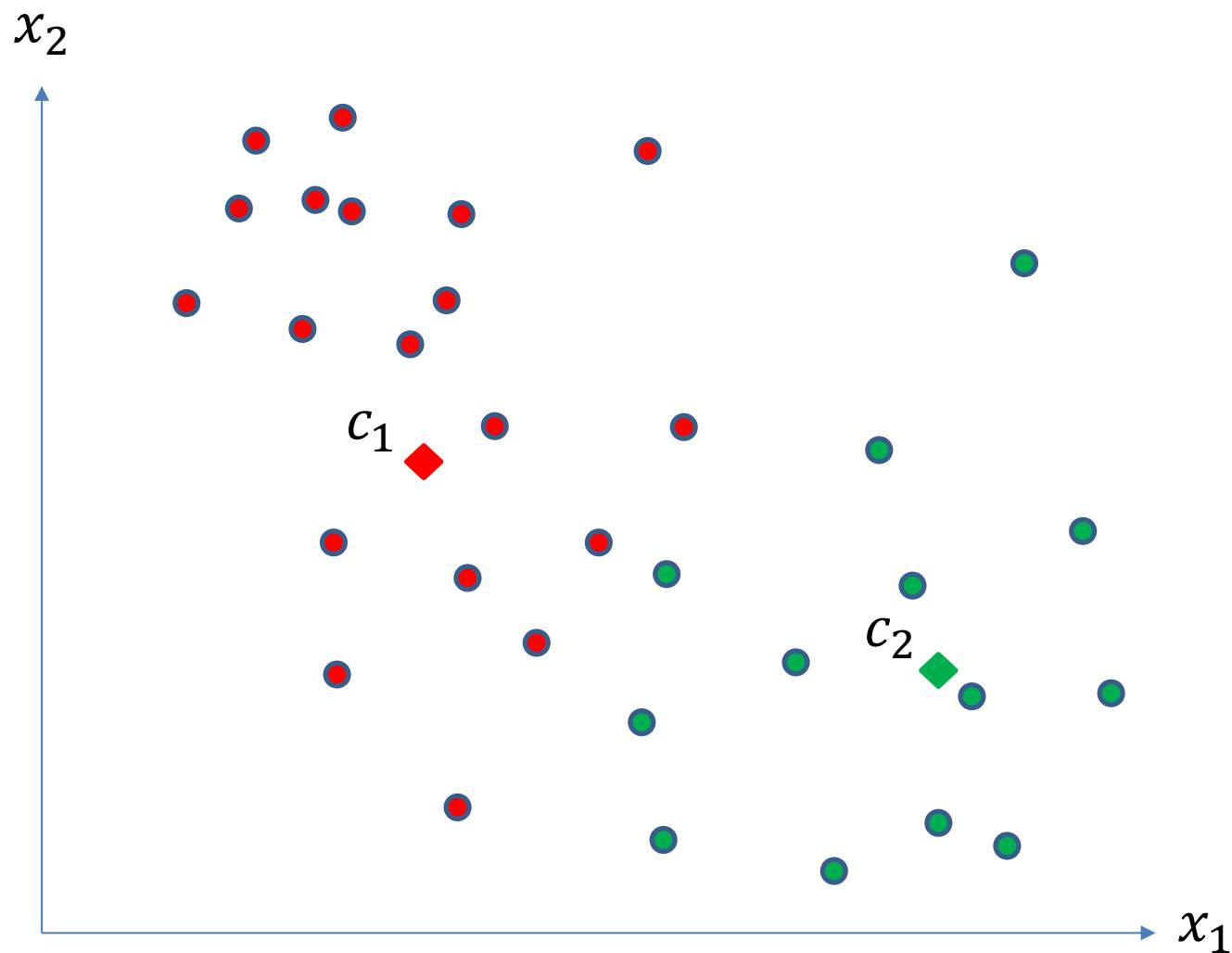
Assignment

K-means Clustering



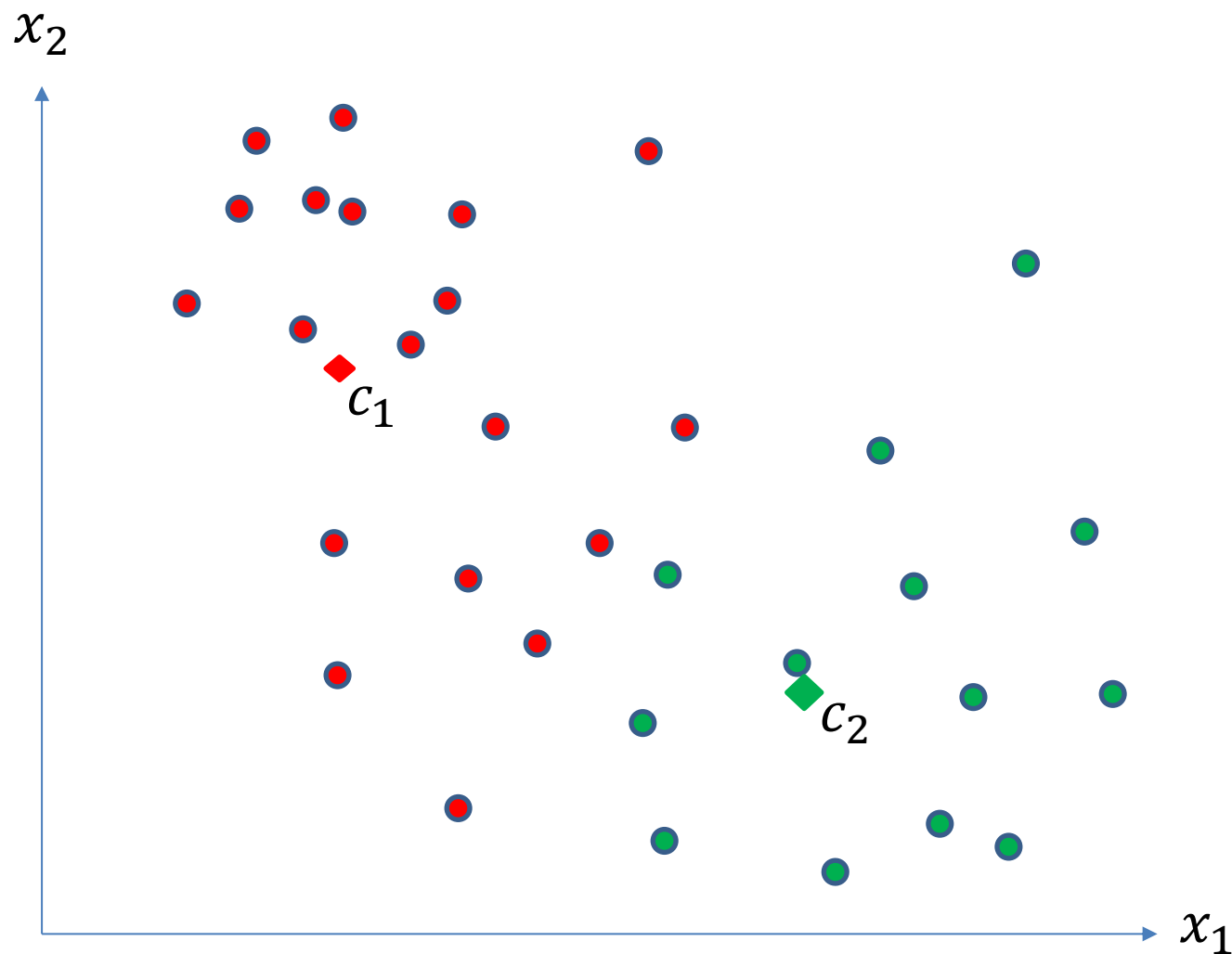
Assignment

K-means Clustering



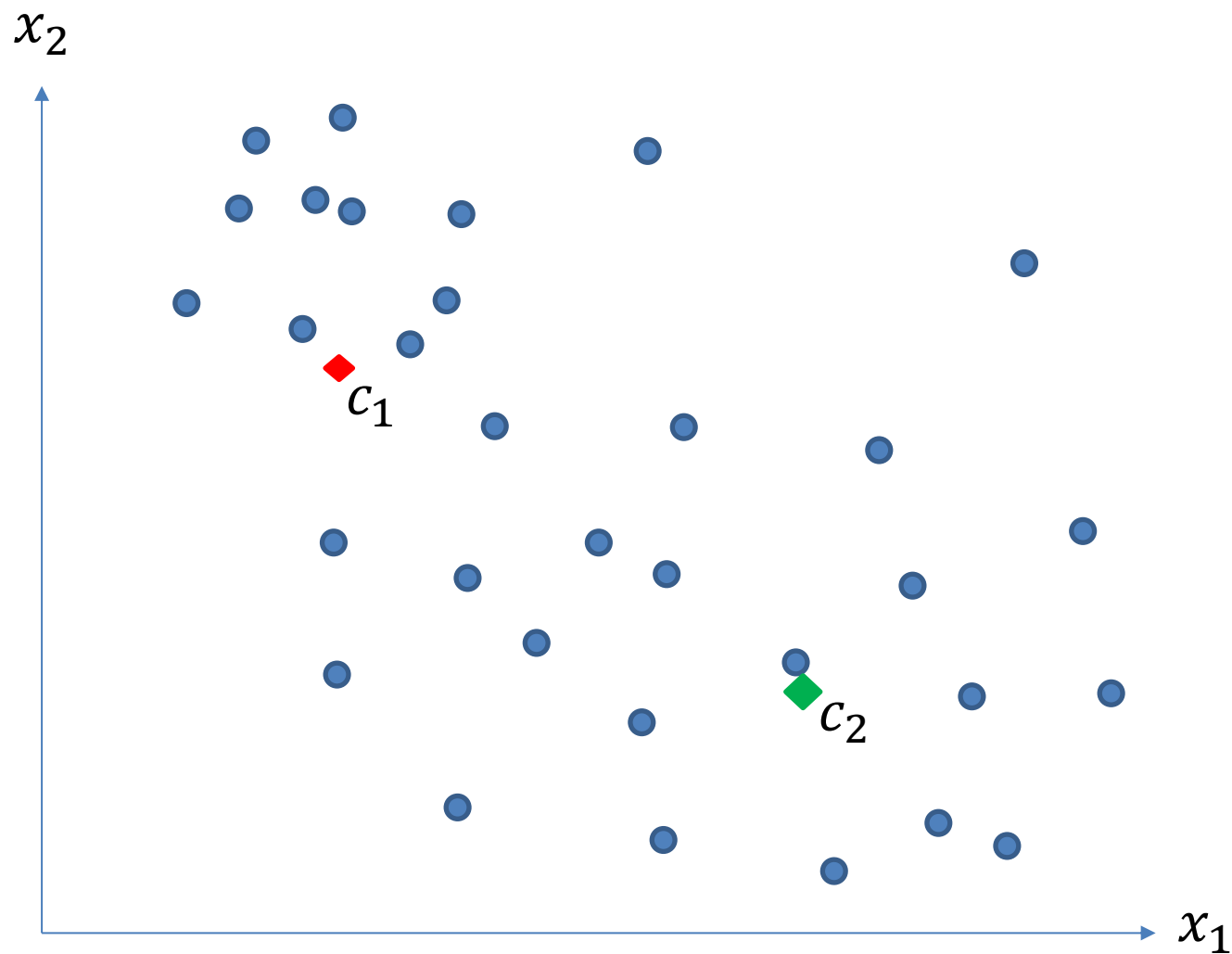
Assignment

K-means Clustering

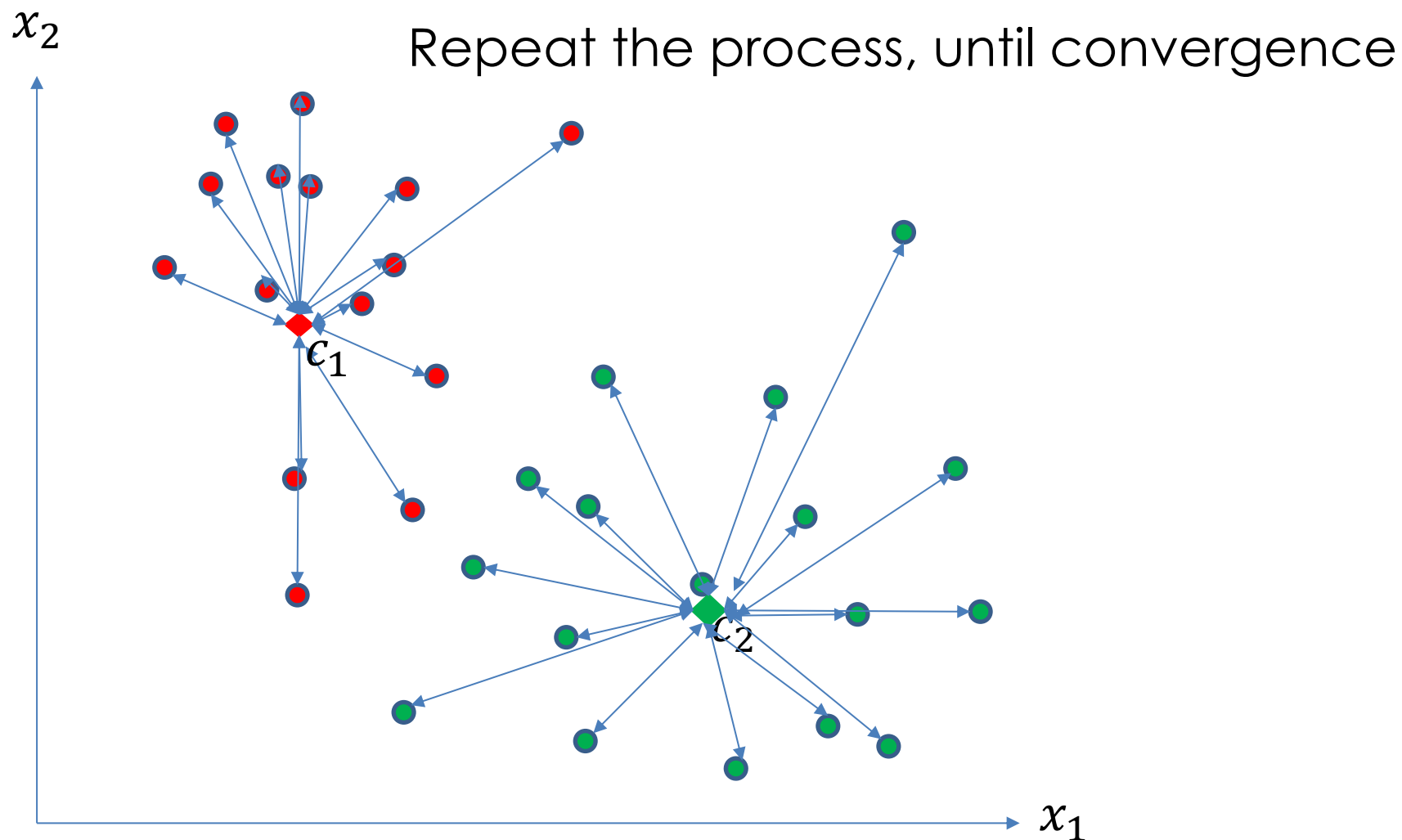


Centroid
Update

K-means Clustering



K-means Clustering



K-means Clustering

Basic/Naïve K-means Clustering

Looping between
Assignment and Centroid Update

1. First, we **choose** K — the number of clusters. Then we randomly select K feature vectors, called **centroids**, to the feature space.
2. Next, **compute the distance from each example x to each centroid c** using some metric, like the Euclidean distance. Then we **assign the closest centroid to each example** (like if we labeled each example with a centroid id as the label).
3. For each centroid, we **calculate the average feature vector** of the examples labeled with it. These average feature vectors become the **new** locations of the **centroids**.
4. We **recompute** the distance from each example to each centroid, modify the assignment and repeat the procedure until **the assignments don't change after the centroid locations are recomputed**.
5. Finally, we **conclude** the clustering with a list of assignments of centroids IDs to the examples.



```
# Define the k-means function
def kmeans_step(data, k, centroids):

    # Assign each data point to the closest centroid
    distances = np.sqrt(((data - centroids[:, np.newaxis])**2).sum(axis=2))
    labels = np.argmin(distances, axis=0)

    # Update centroids to be the mean of the data points assigned to them
    new_centroids = np.zeros_like(centroids)
    for j in range(k):
        new_centroids[j] = np.mean(data[labels == j], axis=0)

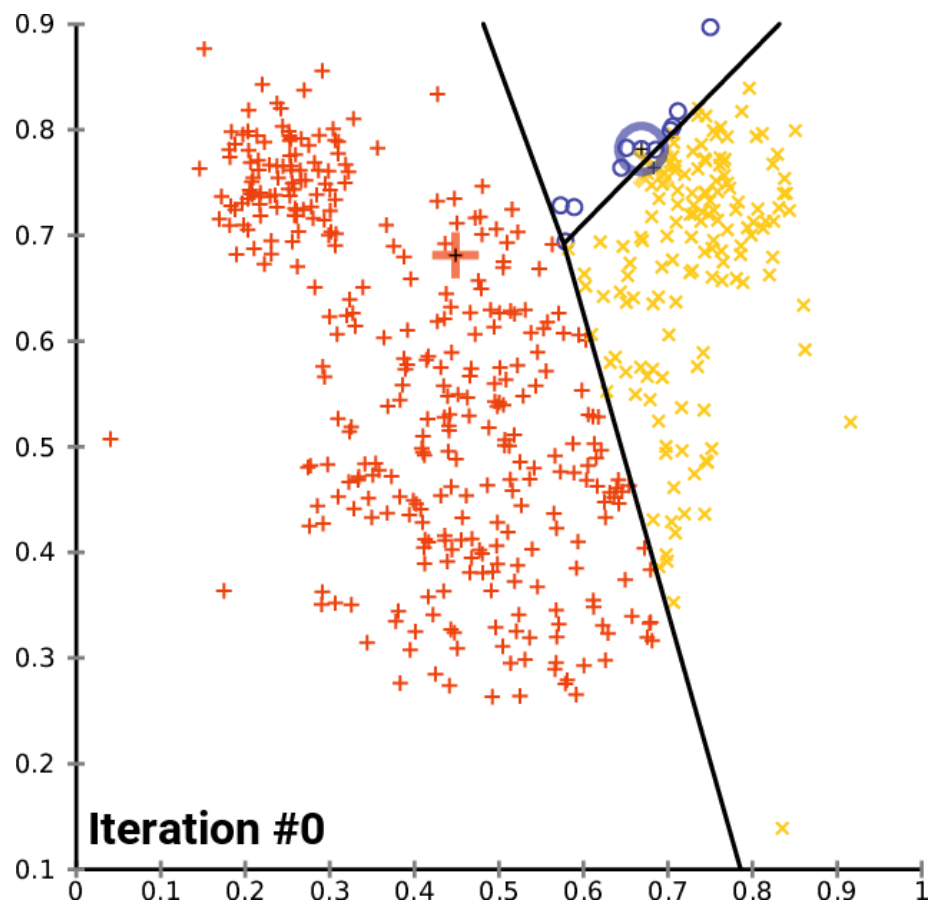
    # End if centroids no longer change
    if np.linalg.norm(new_centroids - centroids) < tolerance:
        print("End Clustering, Centroids no change.")
        # Return the original centroids and labels, and set end to True
        return centroids, labels, True
    else:
        # Return the centroids and labels, and set end to False
        return new_centroids, labels, False
```

- See Python code:
 - lec11.ipynb
- See live demo at:
 - lec11_kmeans.html

All available in Canvas
Files\For Students\Lecture Notes

In the Final, no coding questions for Xinchao's part!

K-means Clustering



https://en.wikipedia.org/wiki/K-means_clustering

K-means Clustering

Optimization Objective Function (within-cluster variance)

m : # of samples; i : index of samples

K : # of clusters; k : index of clusters

Minimize J

$$J = \sum_{i=1}^m \sum_{k=1}^K w_{ik} \|\mathbf{x}_i - \mathbf{c}_k\|^2 \quad (1)$$

The term w_{ik} is equal to 1 for data point \mathbf{x}_i if the data point belongs to cluster S_k , else $w_{ik} = 0$.

Note: The optimization objective function was called $C(\mathbf{w})$ in Lecture 8. Here, we use J (with parameters w_{ik} and \mathbf{c}_k) so that it is differentiated from the centroids \mathbf{c}_k .

Ref: <https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a>
https://en.wikipedia.org/wiki/K-means_clustering

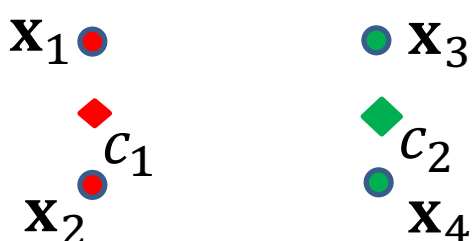
K-means Clustering

Optimization Objective Function (within-cluster variance)

Minimize J

$$J = \sum_{i=1}^m \sum_{k=1}^K w_{ik} \|\mathbf{x}_i - \mathbf{c}_k\|^2 \quad (1)$$

$w_{11} = ?$
 $w_{12} = ?$



$w_{41} = ?$
 $w_{42} = ?$

The term w_{ik} is equal to 1 for data point \mathbf{x}_i if the data point belongs to cluster S_k , else $w_{ik} = 0$.

Note: The optimization objective function was called $C(\mathbf{w})$ in Lecture 8. Here, we use J (with parameters w_{ik} and \mathbf{c}_k) so that it is differentiated from the centroids \mathbf{c}_k .

Ref: <https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a>
https://en.wikipedia.org/wiki/K-means_clustering

K-means Clustering

Naïve K-means Algorithm

1. Assignment Step (fix \mathbf{c} and update w):

Computing distances to
all centroids

$$\mathbf{x}_i \in \mathcal{S}_k \text{ } (w_{ik} = 1) \text{ if } \|\mathbf{x}_i - \mathbf{c}_k\|^2 < \|\mathbf{x}_i - \mathbf{c}_j\|^2 \text{ (else } w_{ik} = 0),$$

$$i = 1, \dots, m; \quad j, k = 1, \dots, K.$$

2. Update Step (fix w and update \mathbf{c}):

$$\frac{\partial J}{\partial \mathbf{c}_k} = -2 \sum_{i=1}^m w_{ik} (\mathbf{x}_i - \mathbf{c}_k) = 0 \quad \Rightarrow \quad \mathbf{c}_k = \frac{\sum_{i=1}^m w_{ik} \mathbf{x}_i}{\sum_{i=1}^m w_{ik}}$$

Solving an optimization, i.e., setting derivative to 0

Note: $\|\mathbf{x} - \mathbf{c}\| = \sqrt{\sum_{d=1}^D (x_d - c_d)^2}$ is called the Euclidean distance.
where $\mathbf{x} = (x_1, x_2, \dots, x_D)$, $\mathbf{c} = (c_1, c_2, \dots, c_D)$

K-means Clustering

1. Assignment Step (fix \mathbf{c} and update w):

$$\mathbf{x}_i \in S_k \text{ (} w_{ik} = 1 \text{) if } \|\mathbf{x}_i - \mathbf{c}_k\|^2 < \|\mathbf{x}_i - \mathbf{c}_j\|^2 \text{ (else } w_{ik} = 0),$$

$$i = 1, \dots, m; \quad j, k = 1, \dots, K.$$

2. Update Step (fix w and update \mathbf{c}):

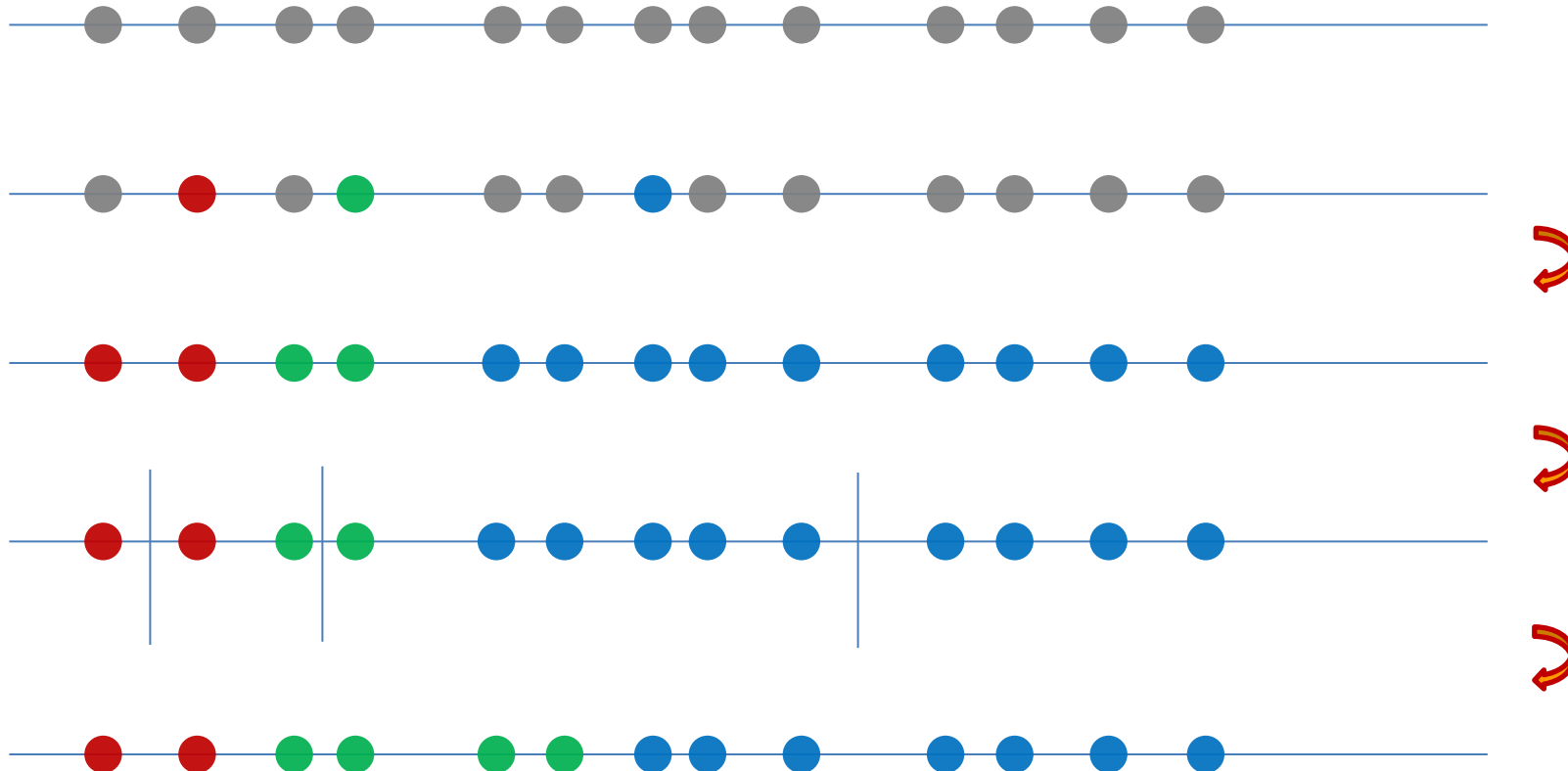
$$\frac{\partial J}{\partial \mathbf{c}_k} = -2 \sum_{i=1}^m w_{ik} (\mathbf{x}_i - \mathbf{c}_k) = 0 \Rightarrow \mathbf{c}_k = \frac{\sum_{i=1}^m w_{ik} \mathbf{x}_i}{\sum_{i=1}^m w_{ik}}$$

By repeating this two steps, the total loss $J = \sum_{i=1}^m \sum_{k=1}^K w_{ik} \|\mathbf{x}_i - \mathbf{c}_k\|^2$, is **guaranteed to NOT increase (i.e., remain the same or decrease)** until convergence.

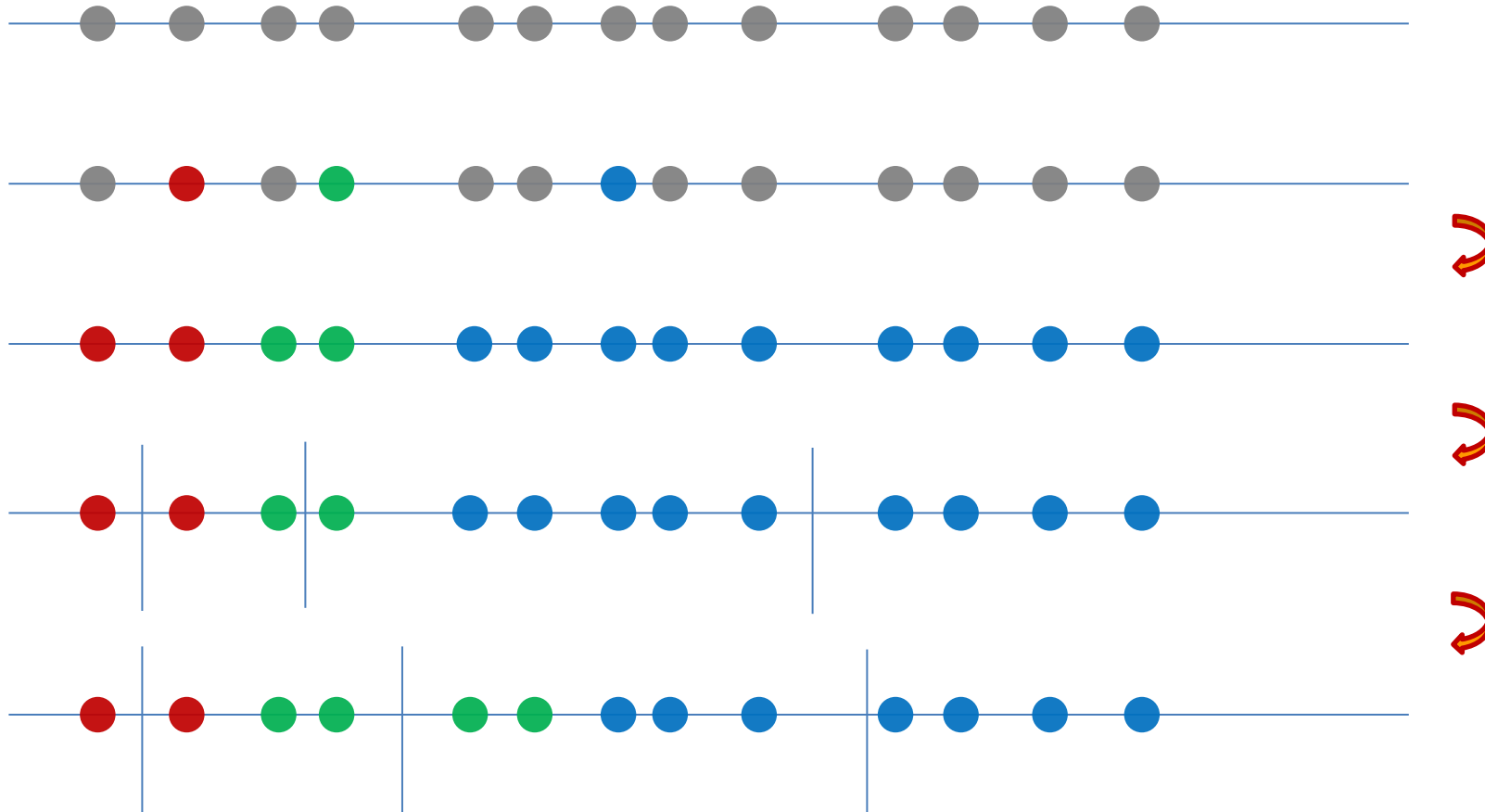
Why? **At Step 2:** we compute the new mean, by solving an optimization, i.e., compute the derivative and set to zero, and solve \mathbf{c}_k . This means that, the new \mathbf{c}_k is guaranteed to give a smaller J value.

At Step 1: we only change the assignment, if the distance to the new centroid is smaller! In other words, we either remain in the old group, or change to a new group that is closer (i.e., gives a smaller J)

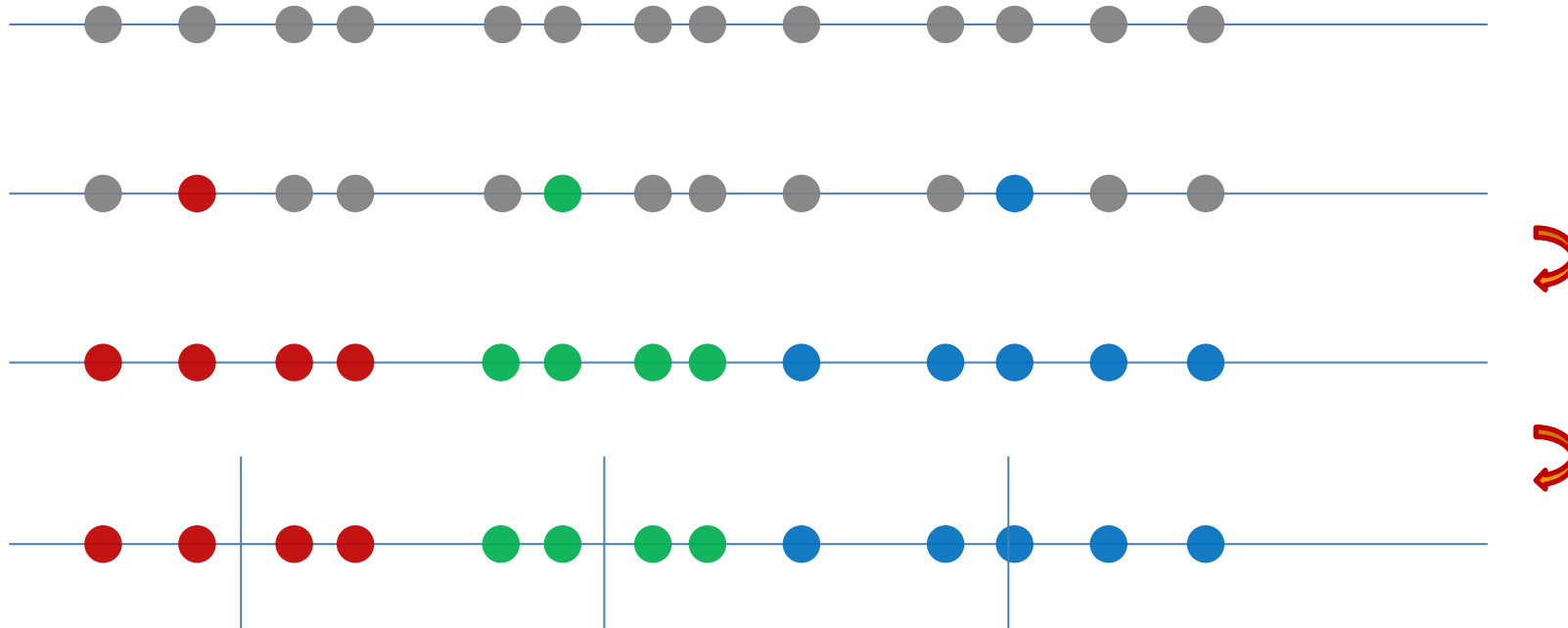
K-means Clustering (1 D)



K-means Clustering (1 D)



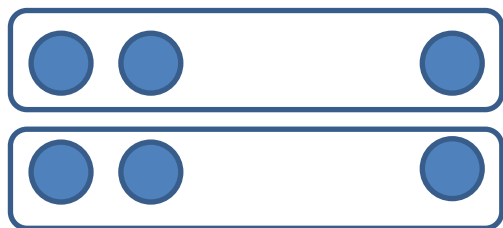
K-means Clustering (1 D)



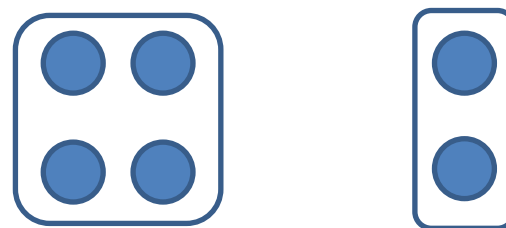
Different initializations give different clusters!

K-means Clustering

- Unfortunately, k-means is not guaranteed to find a global minimum, it finds only local minimum.
- Example:



K-means



Optimal J

- Finding the optimal J is NP-hard*
- In practice, k-means clustering usually performs well
- It can be very efficient, and its solution can be used as a starting point for other clustering algorithms

*<https://en.wikipedia.org/wiki/NP-hardness>

K-means Clustering

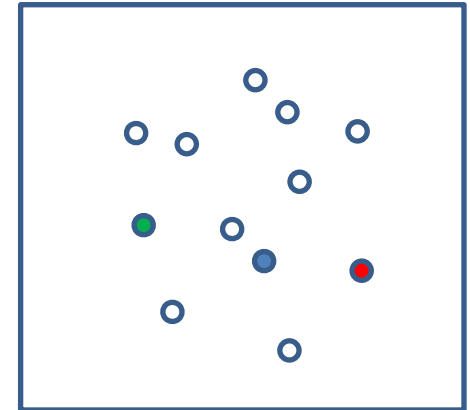
• Initialization

Initialization by centroid

Forgy method:

- Randomly chooses k observations from the dataset and uses these as the initial means.

$k=3$

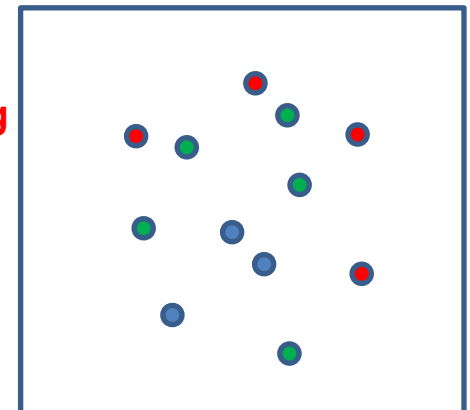


Random partition:

- First randomly assigns a cluster to each observation and then proceeds to the update step, thus computing the initial mean to be the centroid of the cluster's randomly assigned points

Initialization by grouping

$k=3$



Ref: [https://en.wikipedia.org/wiki/K-means_clustering#Standard_algorithm_\(naive_k-means\)](https://en.wikipedia.org/wiki/K-means_clustering#Standard_algorithm_(naive_k-means))

Hard vs Soft Clustering

Hard clustering:

Each data point can belong only one cluster, e.g. K-means

- For example, an apple can be red **OR** green (hard clustering)

Soft clustering (also known as Fuzzy clustering):

Each data point can belong to more than one cluster.

- For example, an apple can be red **AND** green (fuzzy clustering)
- Here, the apple can be red to a certain degree as well as green to a certain degree.
- Instead of the apple belonging to green [green = 1] and not red [red = 0], the apple can belong to green [green = 0.5] and red [red = 0.5]. These values are normalized between 0 and 1; however, they do not represent probabilities, so the two values **do not need to add up to 1**.

Ref: https://en.wikipedia.org/wiki/Fuzzy_clustering

Hard vs Soft Clustering

Objective Function for Fuzzy C-means

Minimize J

$$J = \sum_{i=1}^m \sum_{k=1}^C (w_{ik})^r \|\mathbf{x}_i - \mathbf{c}_k\|^2$$

$$\text{where } w_{ik} = \frac{1}{\sum_{j=1}^C \left(\frac{\|\mathbf{x}_i - \mathbf{c}_k\|}{\|\mathbf{x}_i - \mathbf{c}_j\|} \right)^{\frac{2}{r-1}}}$$

**No need to memorize
the equation!**

Each element, $w_{ik} \in [0,1]$, tells the degree to which element, \mathbf{x}_i , belongs to cluster \mathbf{c}_k .

The fuzzifier $r > 1$ determines the level of cluster fuzziness; usually $1.25 \leq r \leq 2$.

Hard vs Soft Clustering

Objective Function for Fuzzy C-means

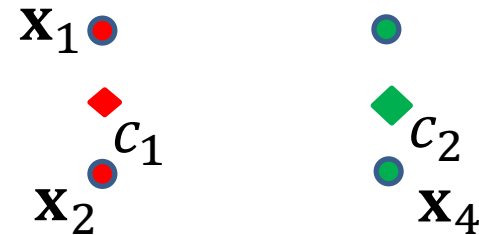
Minimize J

$$J = \sum_{i=1}^m \sum_{k=1}^C (w_{ik})^r \|\mathbf{x}_i - \mathbf{c}_k\|^2$$

$$\text{where } w_{ik} = \frac{1}{\sum_{j=1}^C \left(\frac{\|\mathbf{x}_i - \mathbf{c}_k\|}{\|\mathbf{x}_i - \mathbf{c}_j\|} \right)^{\frac{2}{r-1}}}$$

$$w_{11} = 0.6$$

$$w_{12} = 0.2$$



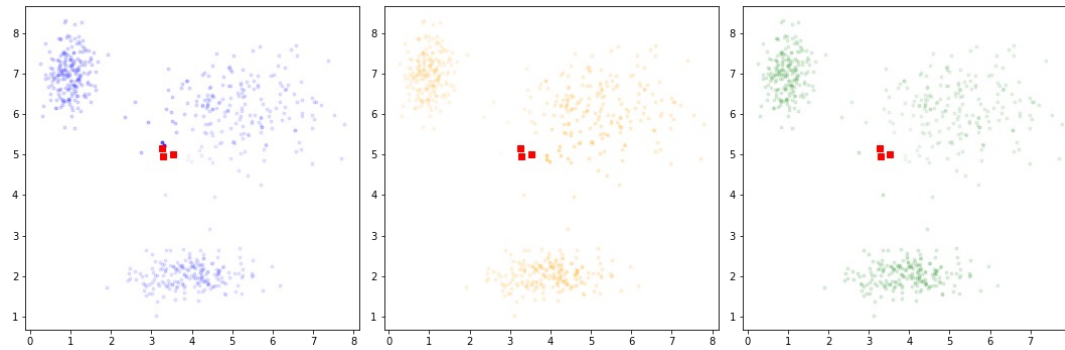
$$w_{41} = 0.18$$

$$w_{42} = 0.75$$

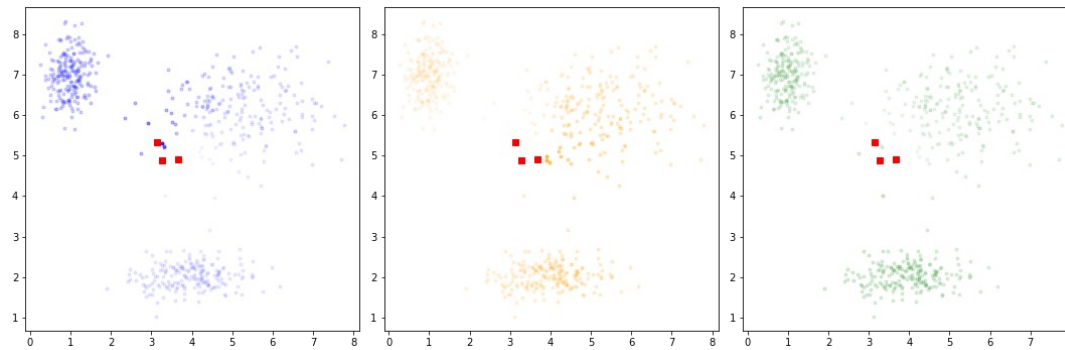
Each element, $w_{ik} \in [0,1]$, tells the degree to which element, \mathbf{x}_i , belongs to cluster \mathbf{c}_k .

The fuzzifier $r > 1$ determines the level of cluster fuzziness; usually $1.25 \leq r \leq 2$.

Iteration 1

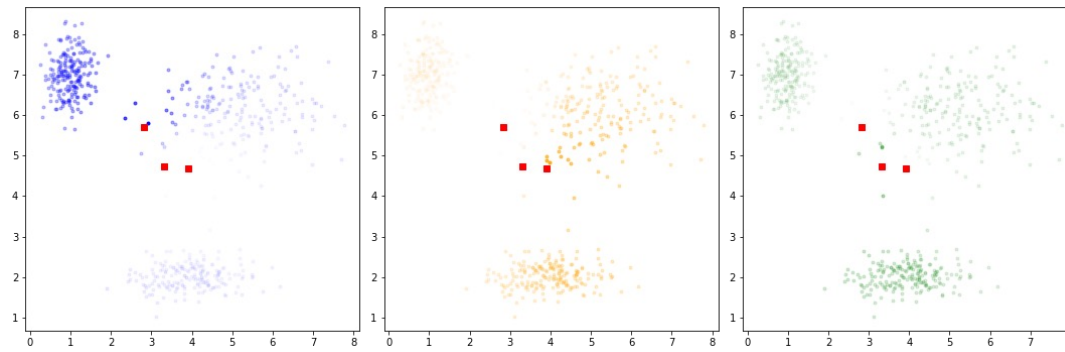


Iteration 2

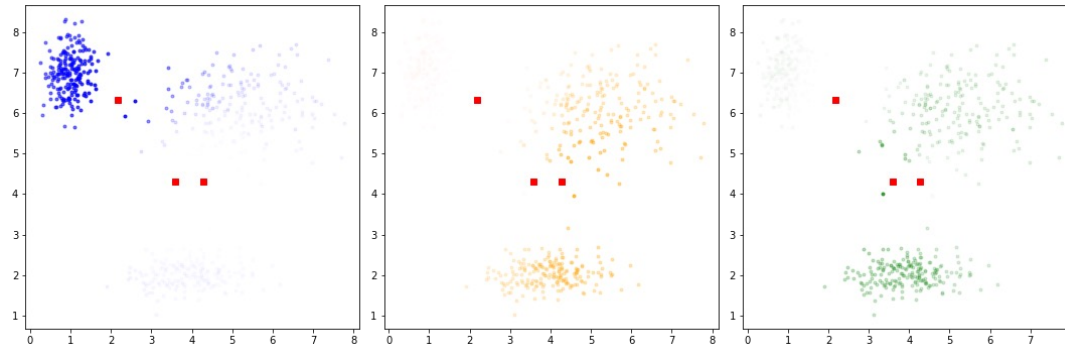


Visualization of Fuzzy C-means Iterations

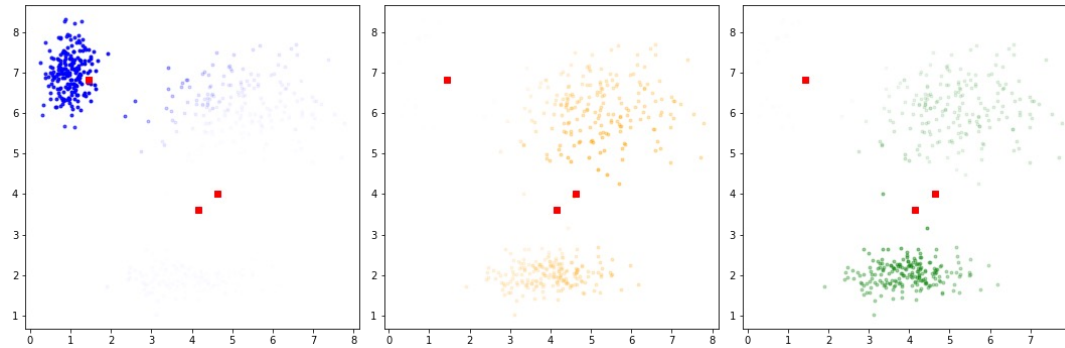
Iteration 3



Iteration 4

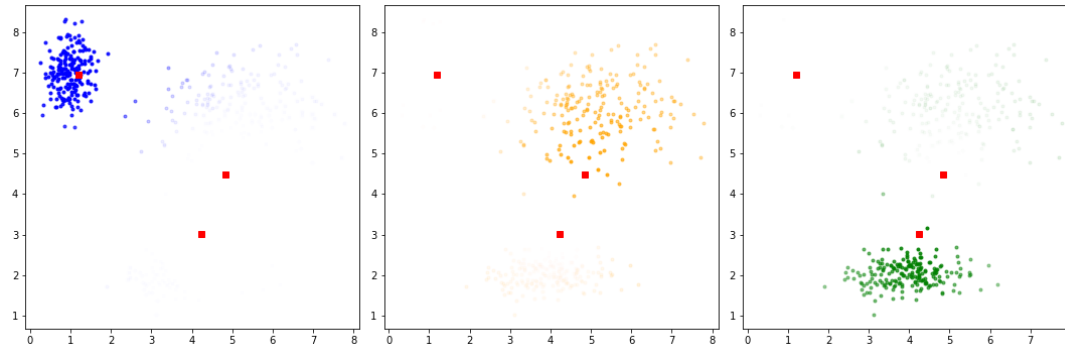


Iteration 5

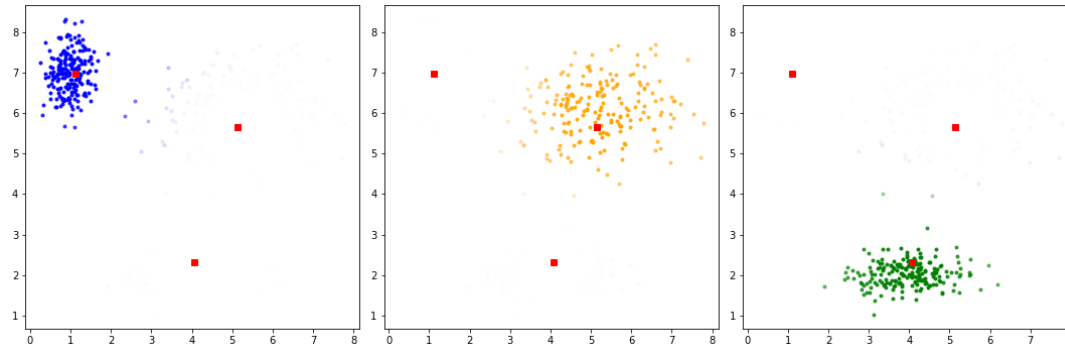


Visualization of Fuzzy C-means Iterations

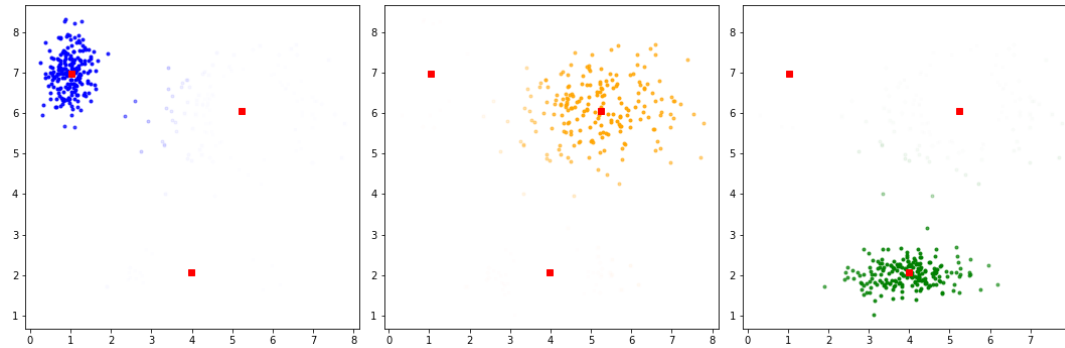
Iteration 6



Iteration 7



Iteration 8



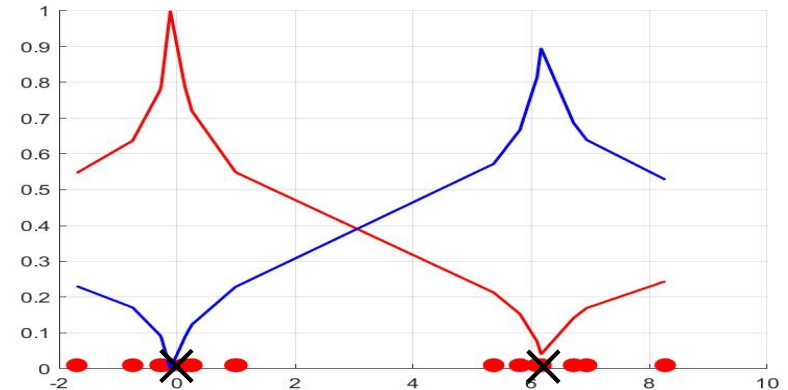
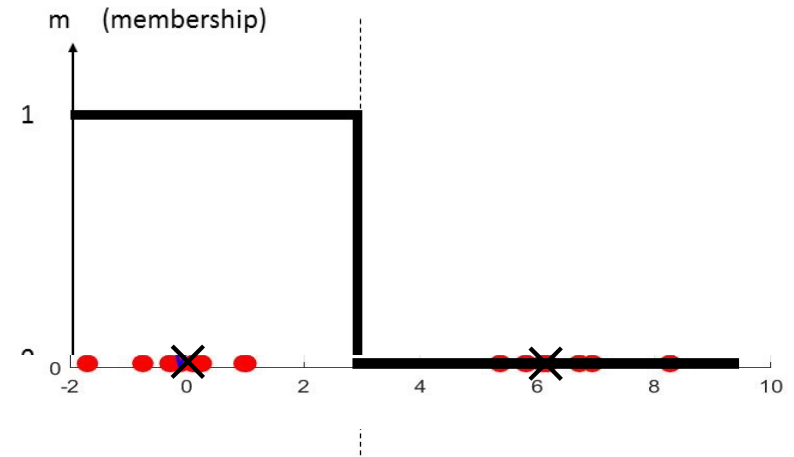
Visualization of Fuzzy C-means Iterations

Hard vs Soft Clustering

Naïve K-means versus Fuzzy C-means

Naïve K-means: $w_{ik} \in \{0,1\}$

Fuzzy C-means: $w_{ik} \in [0,1]$



Ref: https://en.wikipedia.org/wiki/Fuzzy_clustering

Summary

- Introduction of unsupervised learning
- K-means Clustering
 - The most popular clustering technique
- Fuzzy Clustering

Practice Question

We have a collection of 9 foreign coins. We measure their radius in millimeters and summarize them as follows.

Coin ID	01	02	03	04	05	06	07	08	09
Radius (mm)	10	11	12	15	16	17	20	21	22

We'd like to group the coins into three groups according to their radius.

Assume we pick coin 01 as the initial centroid for Group A, coin 04 for Group B, and coin 07 for Group C. We would like to assign the coins to the three groups using Euclidean distance. Before updating the new centroid, we will have BLANK1 coins in Group A (please enter an integer here).

