CS57300
PURDUE UNIVERSITY

NOVEMBER 8, 2021

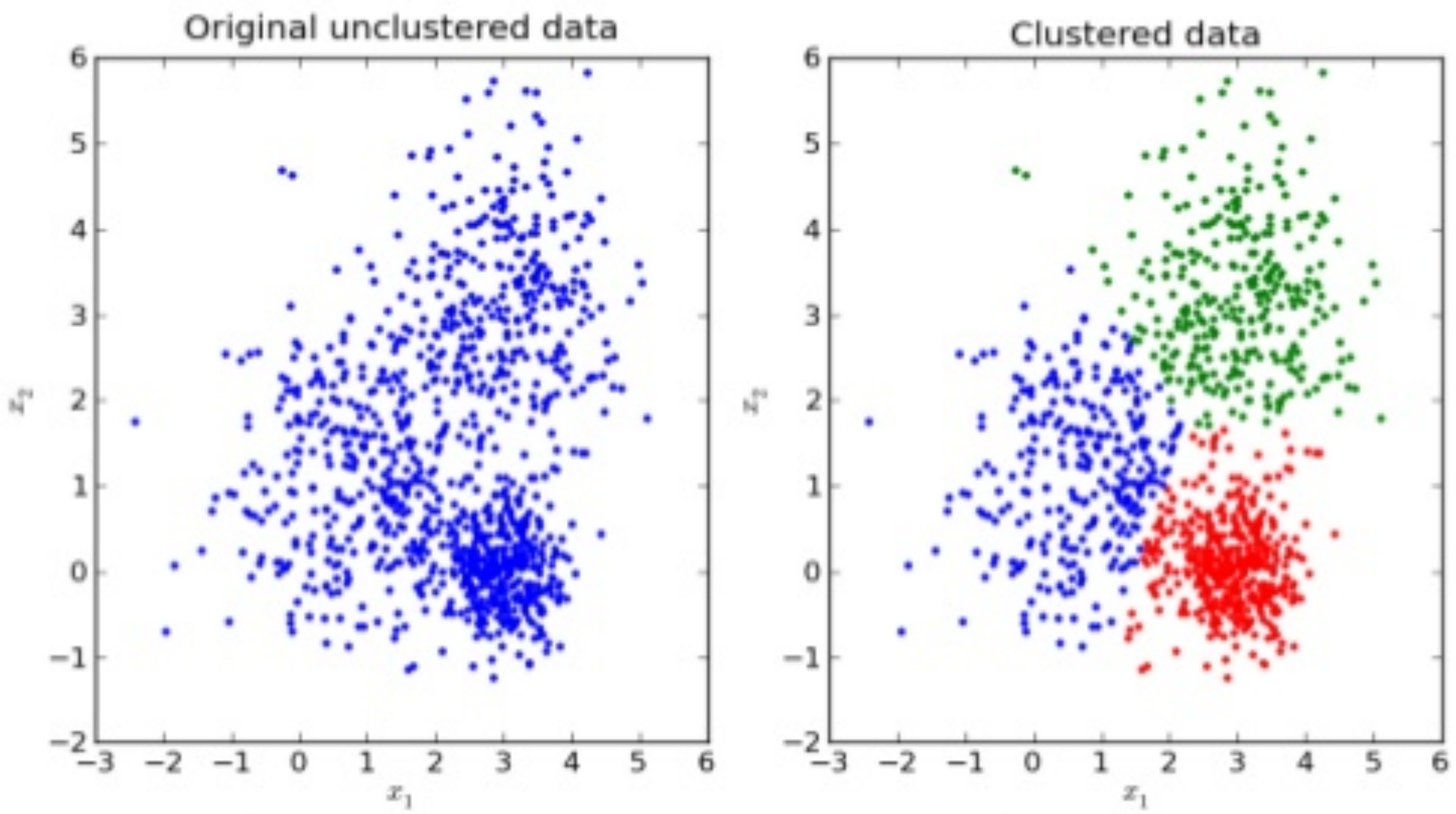
DATA MINING

DESCRIPTIVE MODELING

DATA MINING COMPONENTS

- Task specification: **Description**
- **Knowledge representation**
- Learning technique
- Evaluation and interpretation

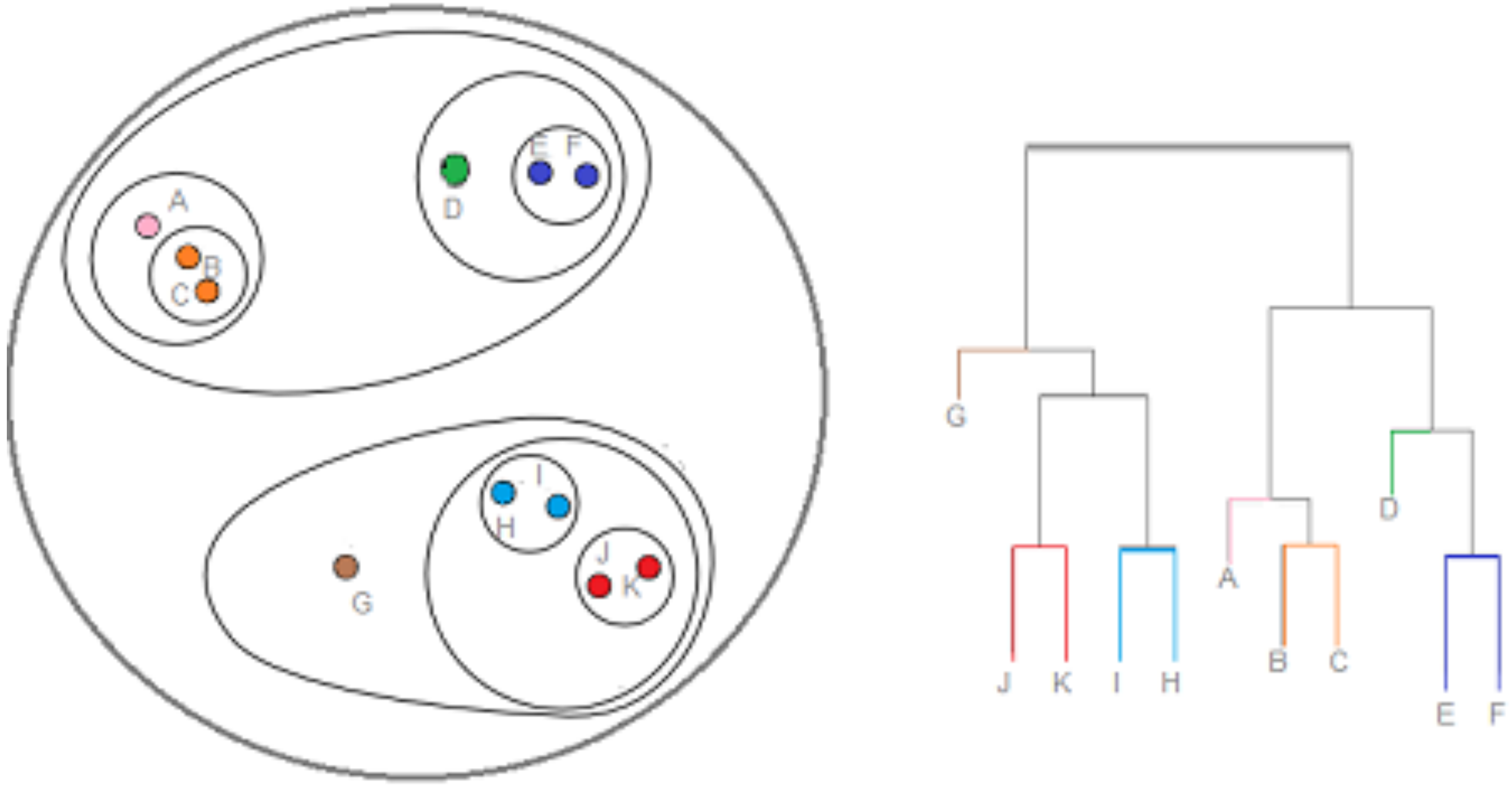
PARTITION-BASED CLUSTERING



- Partition data instances into a fixed number of groups
- Representative algorithm: K-means

Model space:
all possible assignments of data instance to group

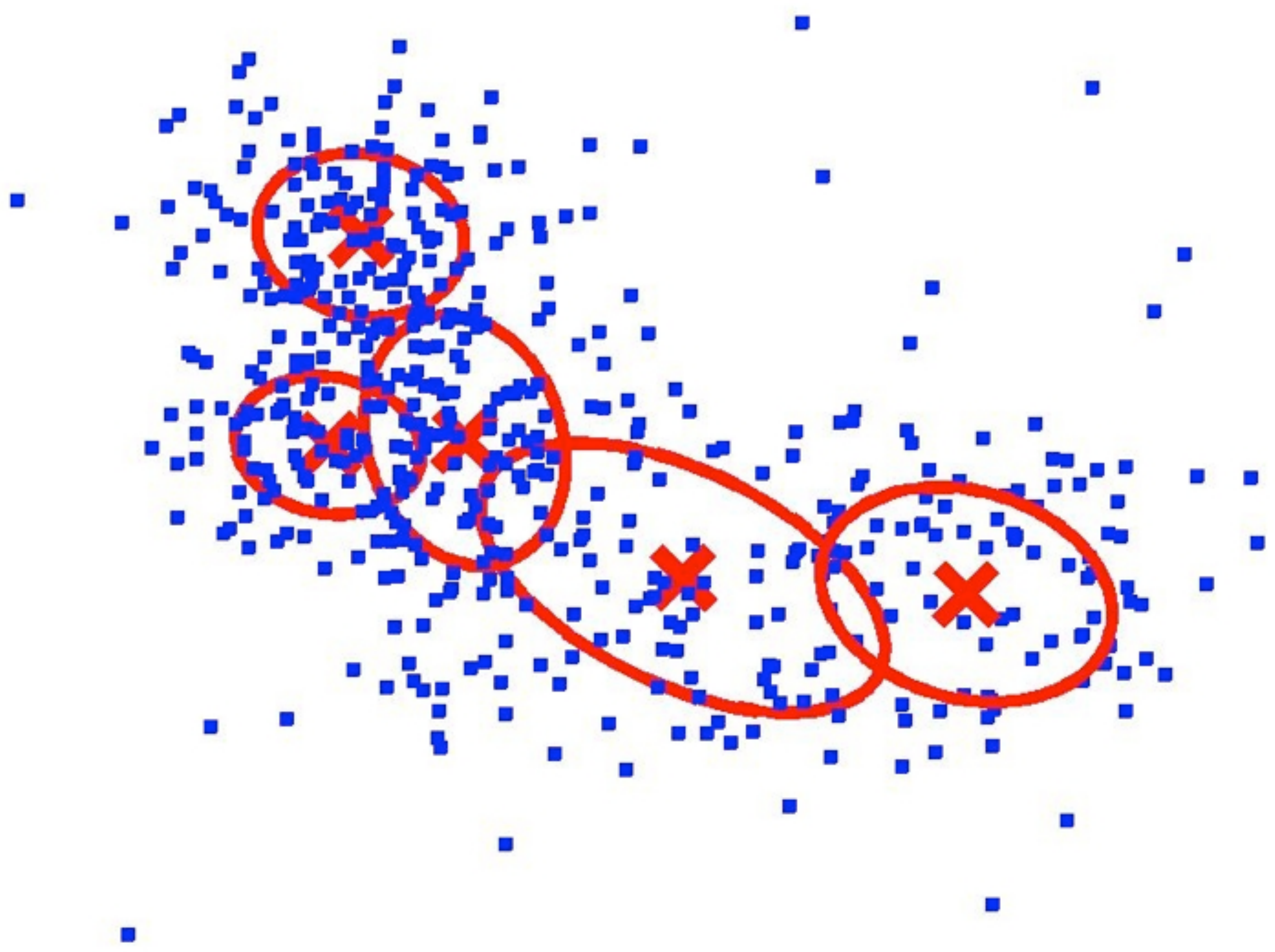
HIERARCHICAL CLUSTERING



- Build a hierarchy of clusters given the data
- Can be agglomerative (“bottom-up”) or divisive (“top-down”)

Model space:
all possible hierarchies

PROBABILISTIC MODEL-BASED CLUSTERING



$$f(x) = \sum_{k=1}^{K} w_k f_k(x;\theta)$$

probability of observing x

likelihood of x being generated from cluster k

likelihood of point belonging to cluster k

Model space:

w_k and $f_k(x;\theta)$

DATA MINING COMPONENTS

- Task specification
- Knowledge representation
- **Learning technique**
- Evaluation and interpretation

LEARNING DESCRIPTIVE MODELS

- Select a **knowledge representation** (a "model")
  - Defines a **space** of possible models $M=\{M_1, M_2, \ldots, M_k\}$
- Define **scoring functions** to "score" different models
- Use **search** to identify "best" model(s)
  - Search the space of models
  - Evaluate possible models with **scoring function** to determine the model which best fits the data

DESCRIPTIVE SCORING FUNCTIONS

- Clustering: What makes a good cluster?
  - High intra-group similarity, low inter-group similarity
  - Scoring function is often a function of within-cluster similarity and between-cluster similarity
- Example scoring functions

cluster centroid:

$$r_k = \frac{1}{n_k} \sum_{x(i) \in C_k} x(i)$$

between-cluster distance:

$$bc(C) = \sum_{1 \leq j < k \leq K} d(r_j, r_k)^2$$

within-cluster distance:

$$wc(C) = \sum_{k=1}^{K} wc(C_k) = \sum_{k=1}^{K} \sum_{x(i) \in C_k} d(x(i), r_k)^2$$

DESCRIPTIVE SCORING FUNCTIONS

- Structure learning and density estimation: Does the model representation capture the observed data well?
  - Likelihood of the observed data is often used as the scoring function
  - Also applicable to probabilistic model-based clustering

SEARCHING OVER MODELS

- Search over the model space to find the model structure / parameters that optimize the scoring function
- Discrete model space example: partition-based clustering
    - Find k clusters among n data instances: k^n possible allocations
    - Exhaustive search is intractable
    - Most approaches use iterative improvement algorithms to search the model space heuristically

SEARCHING OVER MODELS

- Continuous model space example: probabilistic model-based clustering
  - Searching for the cluster weight (i.e., w_k) and cluster parameters (i.e., $f_k(x, \theta)$) that gives the highest likelihood of observing the current data
  - Solution: **Expectation-maximization** to iteratively infer cluster member and estimate cluster parameters

DATA MINING COMPONENTS

- Task specification
- Knowledge representation
- Learning technique
- **Evaluation and interpretation**

DESCRIPTIVE MODEL EVALUATION

- Clustering evaluation
  - **Supervised**: Measures the extent to which clusters match external class label values, e.g., how likely a cluster contains only data instances of a particular class?
  - **Unsupervised**: Measures goodness of fit without class labels, e.g., how closely related instances within each cluster are and distinct instances across different clusters are?

DESCRIPTIVE MODEL EVALUATION

- Describe the current data precisely vs. Generalize to new data
- Example: in partition-based clustering, the model captures the data the best when $k=n$
- Strike a balance between between how well the model fits and the data and the simplicity of the model

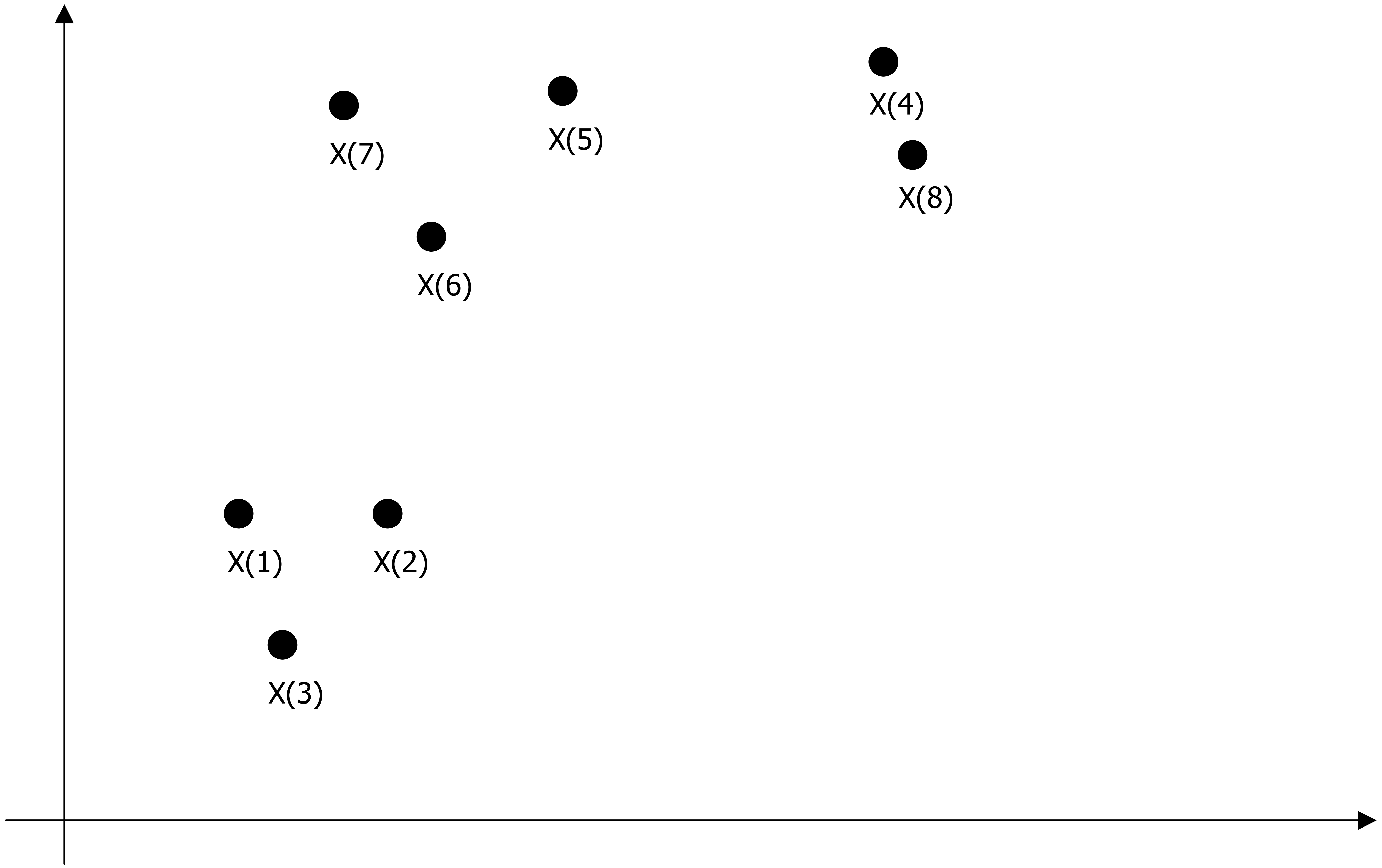
PARTITION-BASED CLUSTERING

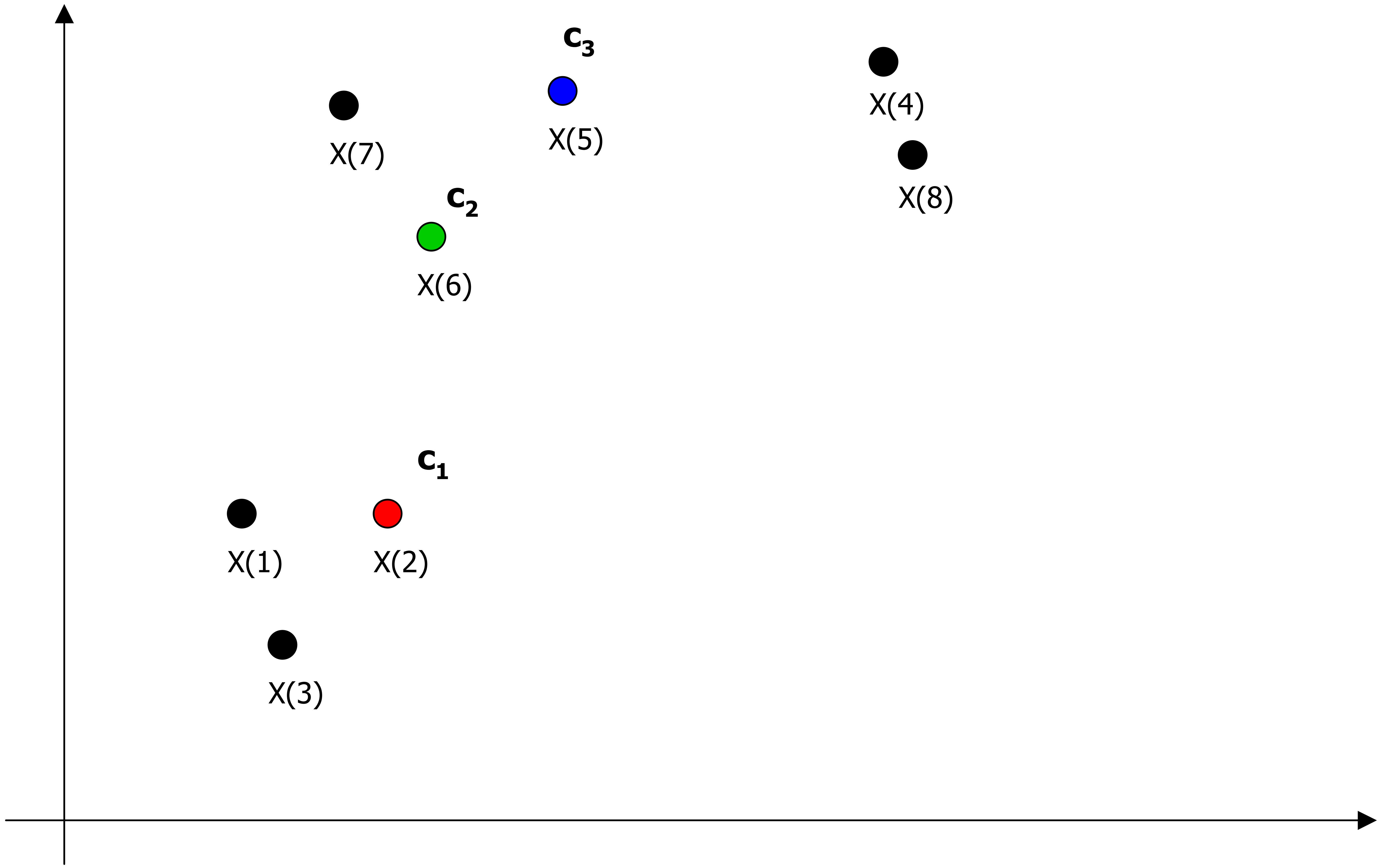
PARTITION-BASED

- Input: data D={**x**(1),**x**(2),...,**x**(n)}
- Output: k clusters C={C_1,...,C_k} such that each **x**(i) is assigned to a unique C_j
- Evaluation: Score(C,D) is maximized/minimized
    - Combinatorial optimization: search among k^n allocations of n objects into k classes to maximize score function
    - Exhaustive search is intractable
    - Most approaches use iterative improvement algorithms

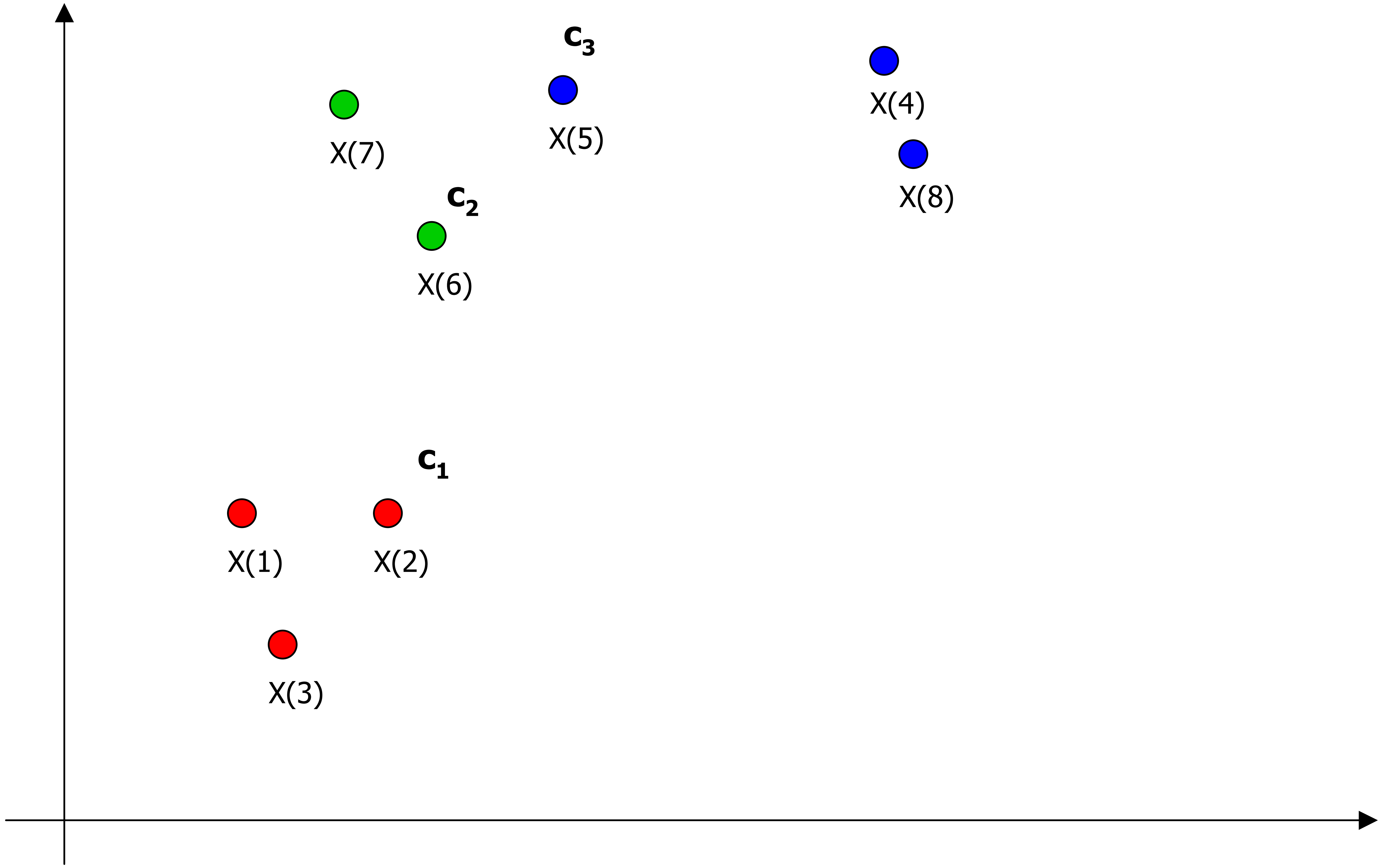
EXAMPLE: K-MEANS

- Algorithm idea:
  - Start with k randomly chosen centroids
  - Repeat until no changes in assignments
    - Assign instances to closest centroid
    - Recompute cluster centroids

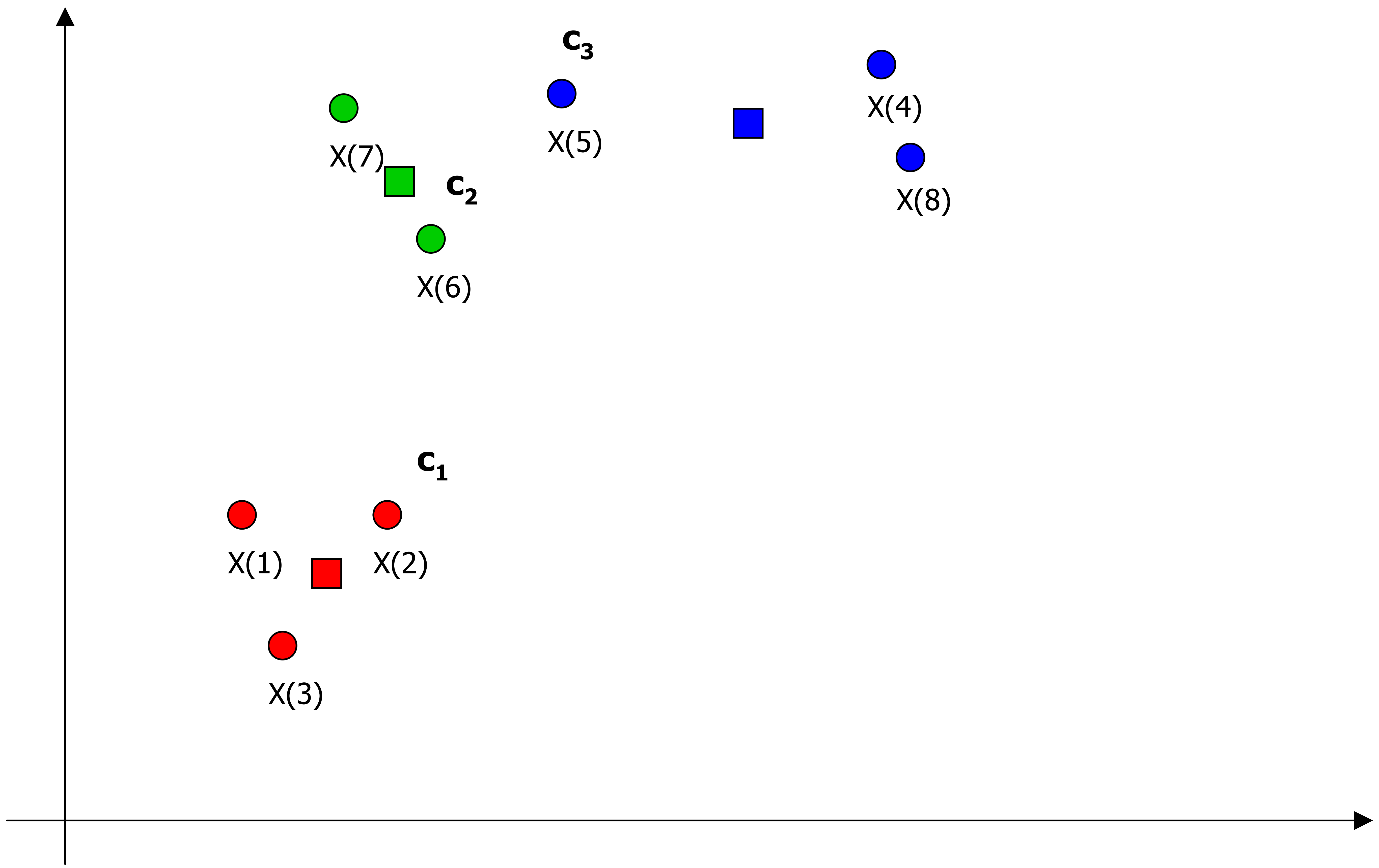

X(4)
X(7)
X(5)
X(8)
X(6)
X(1)
X(2)
X(3)



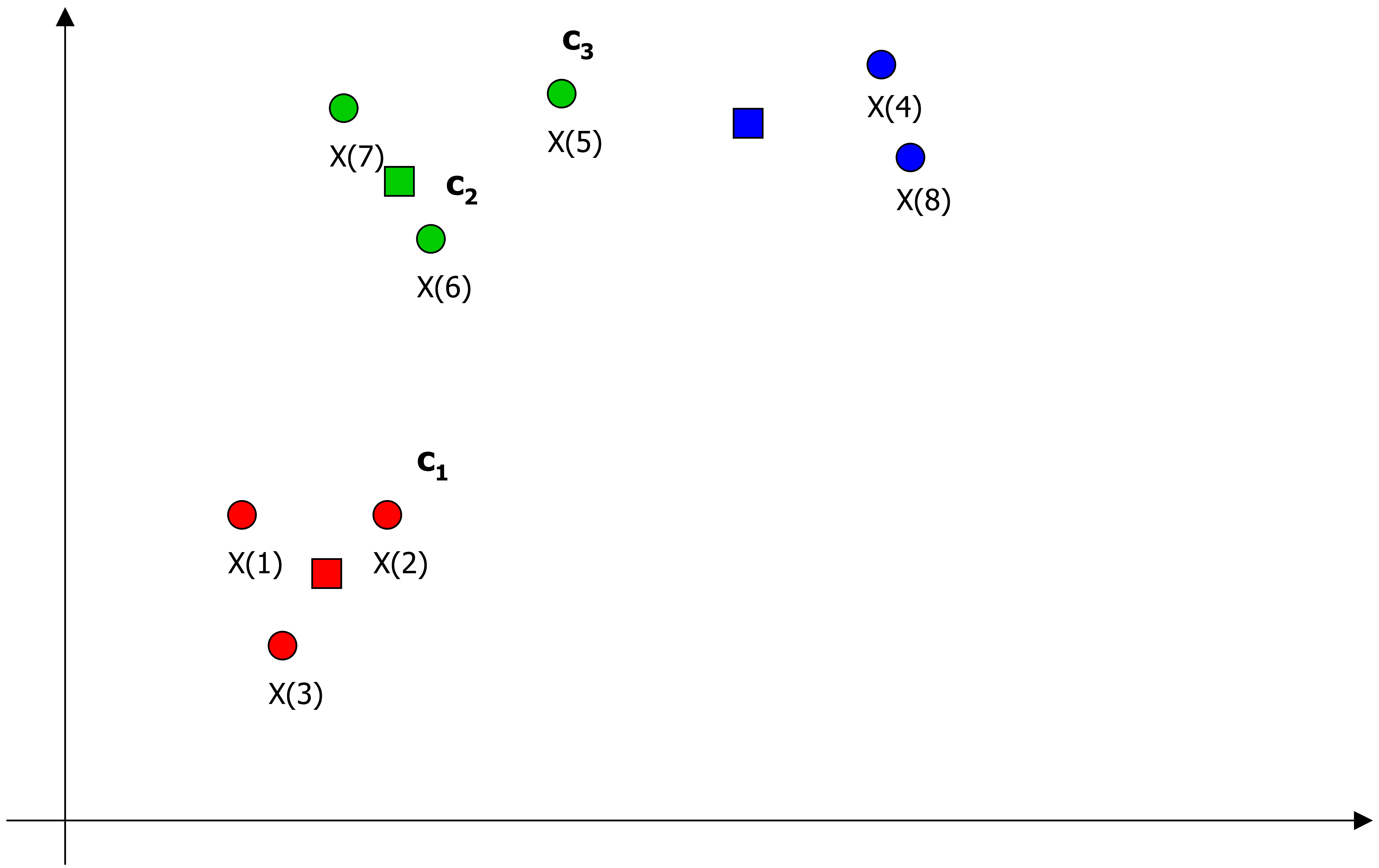
c3
X(7)
X(5)
X(4)
X(8)
c2
X(6)
c1
X(1)
X(2)
X(3)

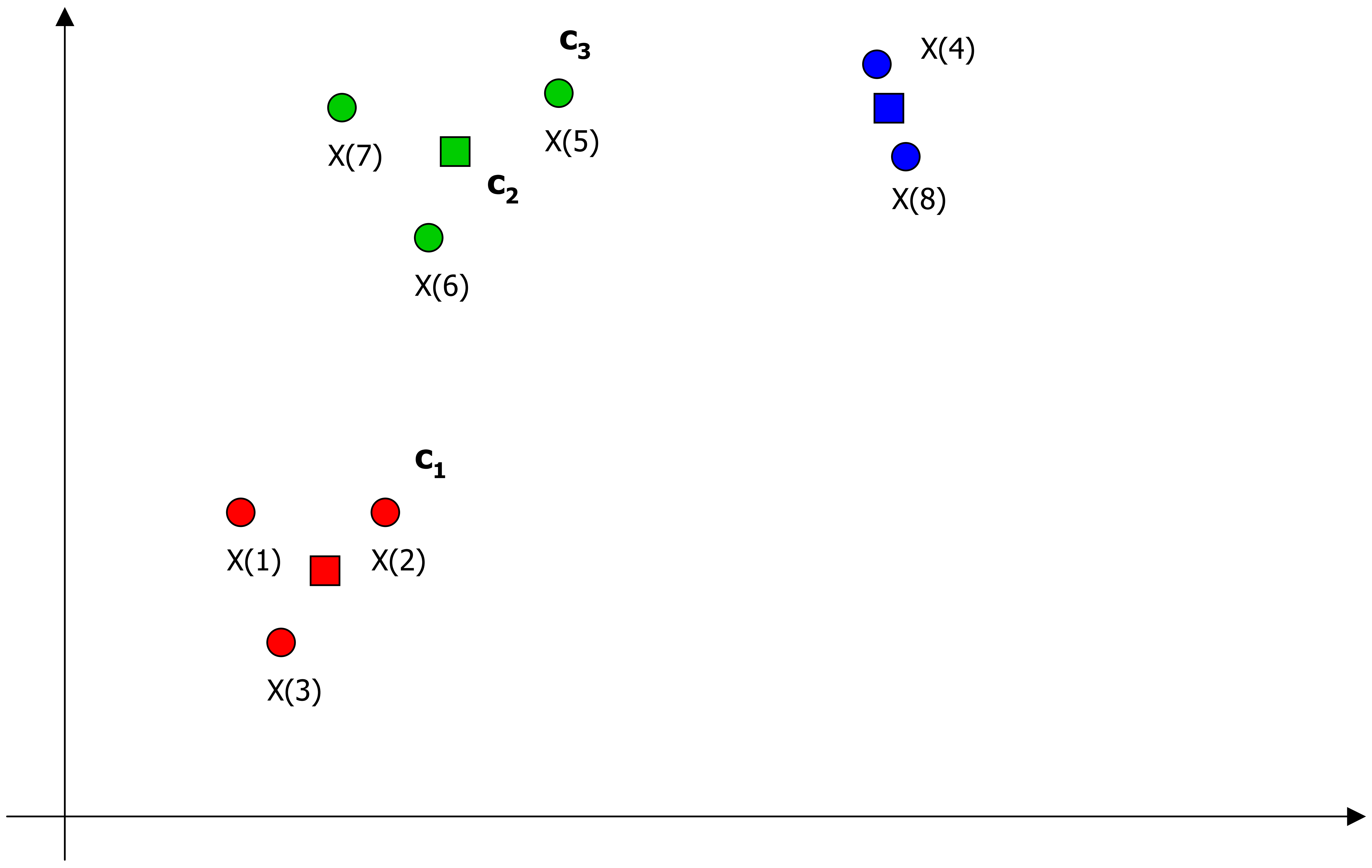


c_3
X(4)
X(7)
X(5)
X(8)
c_2
X(6)
c_1
X(1)
X(2)
X(3)



c3
X(7)
X(5)
X(4)
c2
X(8)
X(6)
c1
X(1)
X(2)
X(3)


c3
X(4)
X(5)
X(7)
c2
X(8)
X(6)
c1
X(1)
X(2)
X(3)


c3
X(4)
X(7)
X(5)
c2
X(8)
X(6)
c1
X(1)
X(2)
X(3)

Algorithm 2.1 The `k-means` algorithm

Input: Dataset D, number clusters k

Output: Set of cluster representatives C, cluster membership vector $\mathbf{m}$

/* Initialize cluster representatives C */

Randomly choose k data points from D

Use these k points as initial set of cluster representatives C

repeat

/* Data Assignment */

Reassign points in D to closest cluster mean

Update $\mathbf{m}$ such that m_i is cluster ID of ith point in D

/* Relocation of means */

Update C such that c_j is mean of points in jth cluster

until convergence

SCORING FUNCTION OF K-MEANS

- What scoring function is K-means trying to optimize for?

Score function: $$wc(C) = \sum_{k=1}^{K} wc(C_k) = \sum_{k=1}^{K} \sum_{x(i) \in C_k} d(x(i), r_k)^2$$

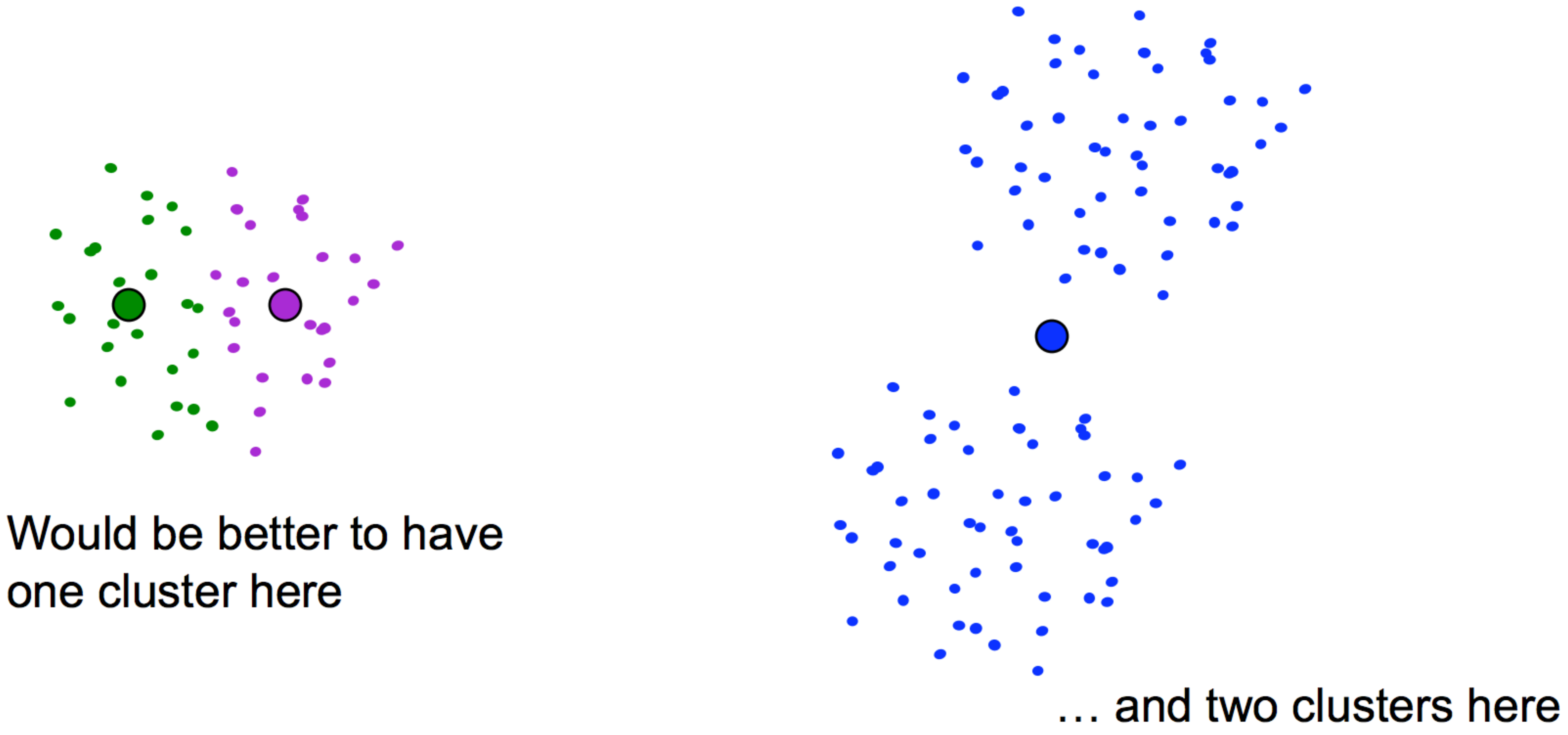
- An alternating optimization approach
  - Fix r_k, optimize for membership of C(x(i)): $min \sum_{i=1}^{N} (x(i) - r_{C(x(i))})^2$
  - Fix C(x(i)), optimize for r_k: $min_{r_k} \sum_{i=1}^{N} (x(i) - r_{C(x(i))})^2 = \sum_{k=1}^{K} \sum_{x \in C_k} (x - r_k)^2$
    - Take derivative with respect to r_k and set to 0 leads to $r_k = \frac{1}{|C_k|} \sum_{x \in C_k} x$

ALGORITHM DETAILS

- Does it terminate?
  - Yes, the objective function decreases on each iteration. It usually converges quickly.
- Does it converge to an optimal solution?
  - No, the algorithm terminates at a local optima which depends on the starting seeds.

K-MEANS IS SENSITIVE TO INITIAL SEEDS

A local optimum:



K-MEANS

- Strengths:
  - Relatively efficient (time complexity is $O(K \cdot N \cdot i)$, where i is the number of iterations)
  - Finds spherical clusters
- Weaknesses:
  - Terminates at local optimum (sensitive to initial seeds)
  - Applicable only when mean is defined
  - Need to specify K
  - Susceptible to outliers/noise