

CS 760: Machine Learning

Unsupervised Learning: Clustering

Announcements

- **Logistics:**

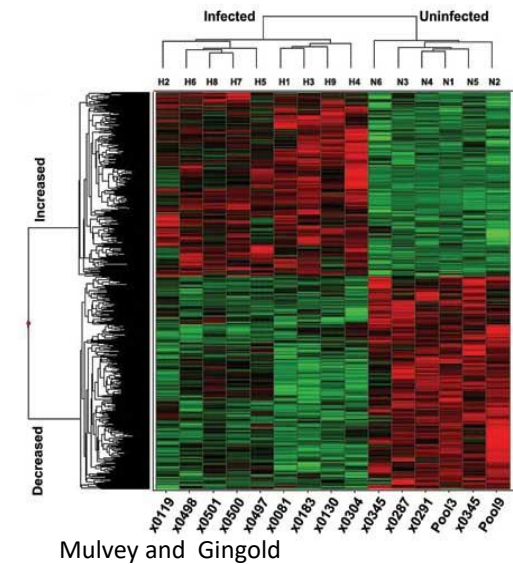
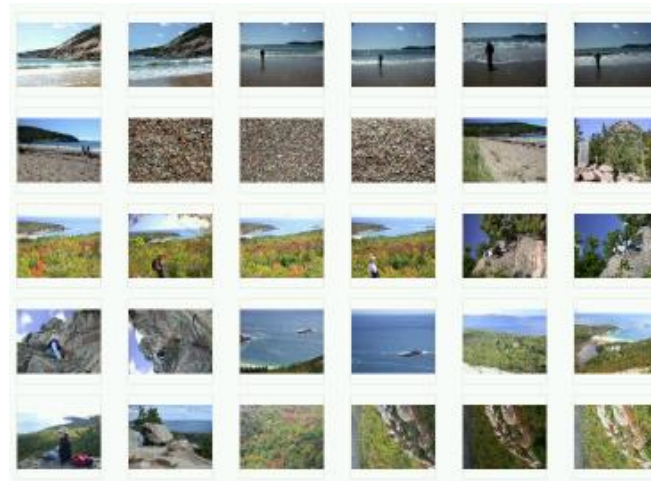
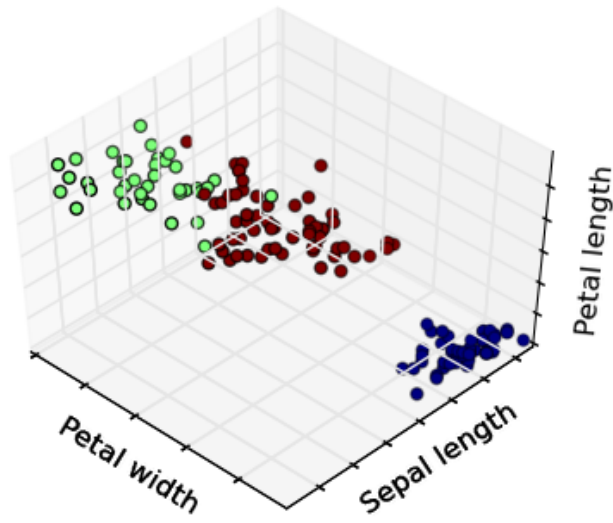
- HW1 due Wednesday before class
- HW2 out the same day

- **Class roadmap:**

- Two lectures on unsupervised learning
- Then five on neural networks
- Then midterm

Unsupervised Learning

- Goal: find patterns & structures that help (us OR a machine) better understand data.
- No labels; usually won't be making predictions
- Sometimes model a distribution, but not always



Outline

- **K-means clustering**
- **Gaussian Mixture Models**
 - Mixtures, Expectation-Maximization algorithm
- **Advanced clustering methods**
 - hierarchical, spectral clustering

Outline

- **K-means clustering**
- **Gaussian Mixture Models**
 - Mixtures, Expectation-Maximization algorithm
- **Advanced clustering methods**
 - hierarchical, spectral clustering

Clustering

Several types:

Partitional

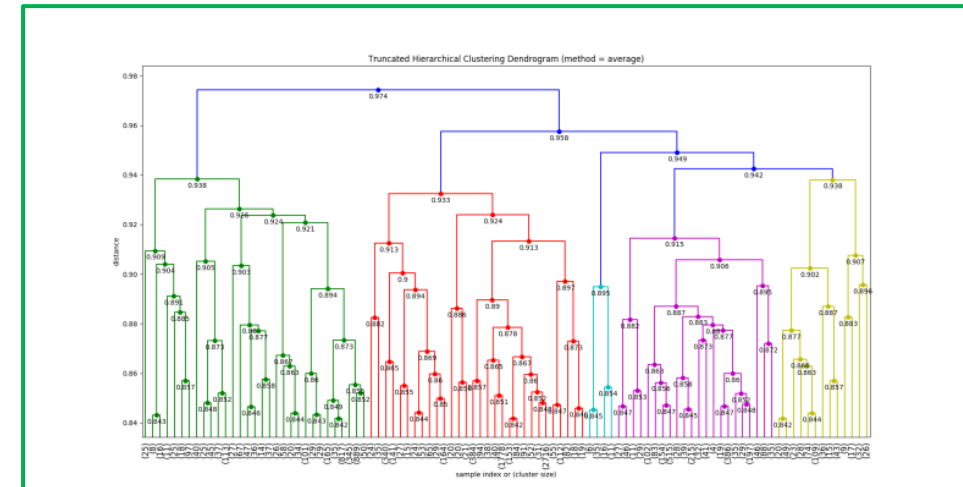
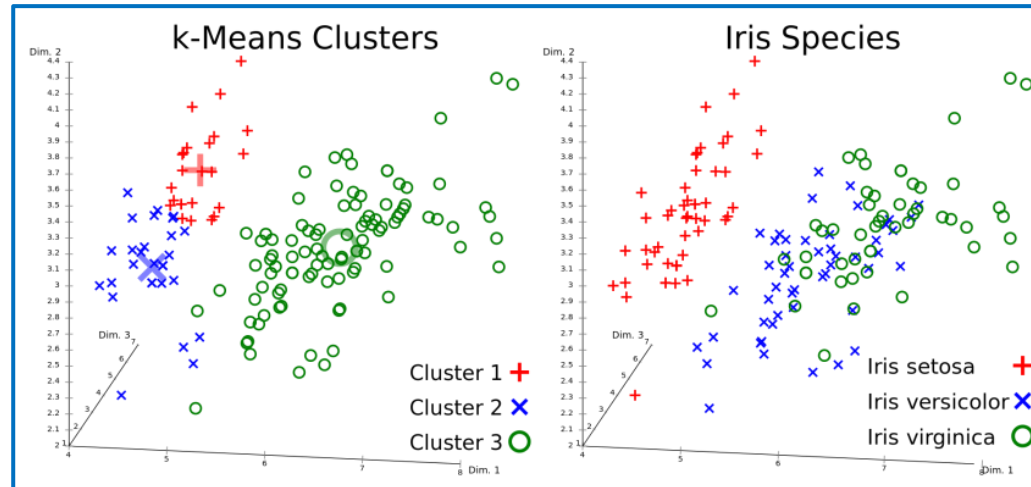
- Centroid
- Graph-theoretic
- Spectral

Hierarchical

- Agglomerative
- Divisive

Bayesian

- Decision-based
- Nonparametric

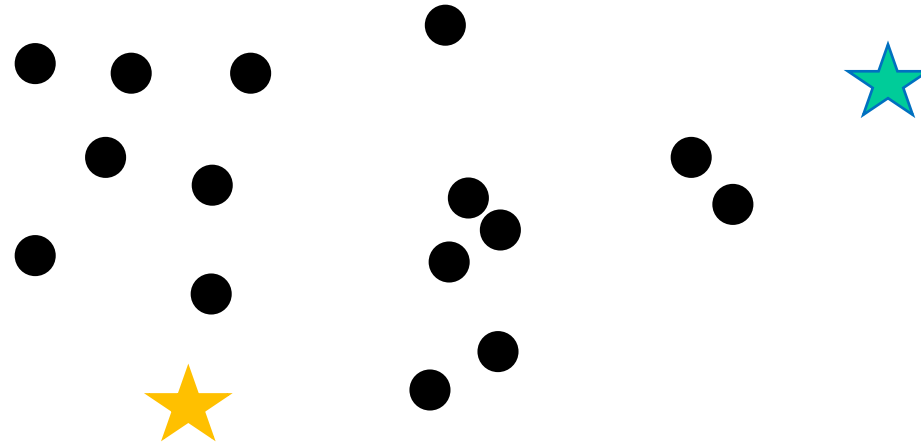


K-Means Clustering

k-means is a type of partitional **centroid-based clustering**

Algorithm:

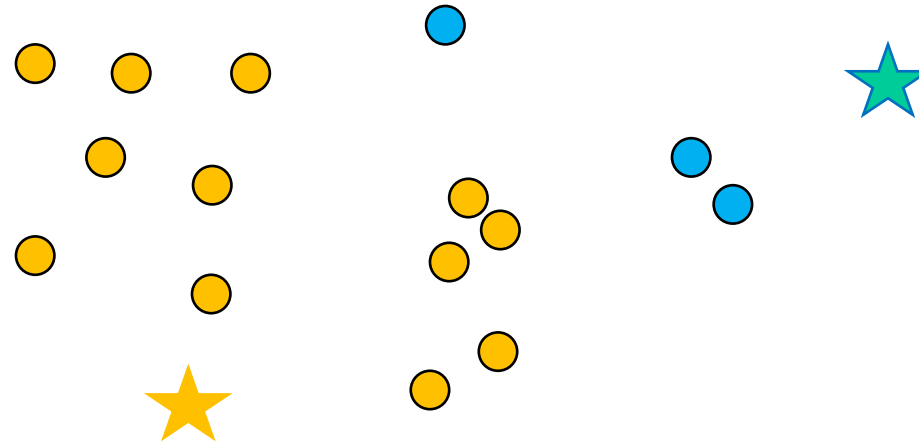
1. Randomly pick k cluster centers



K-Means Clustering: Algorithm

k-means clustering

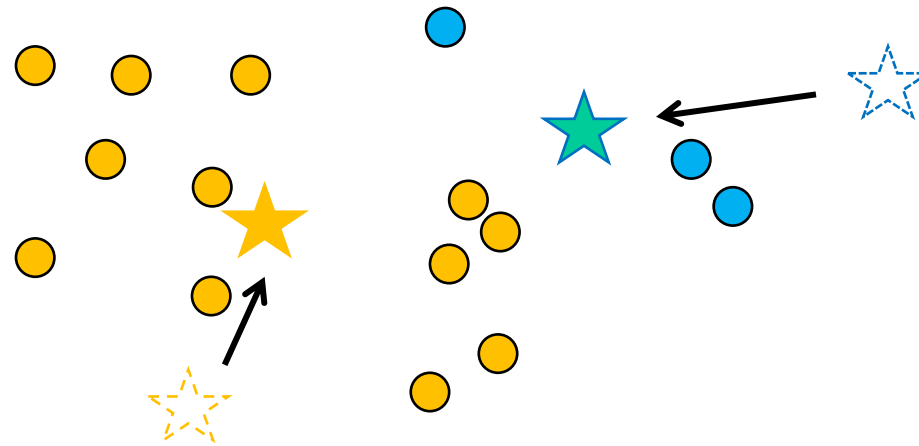
2. Find closest center for each point



K-Means Clustering: Algorithm

k-means clustering

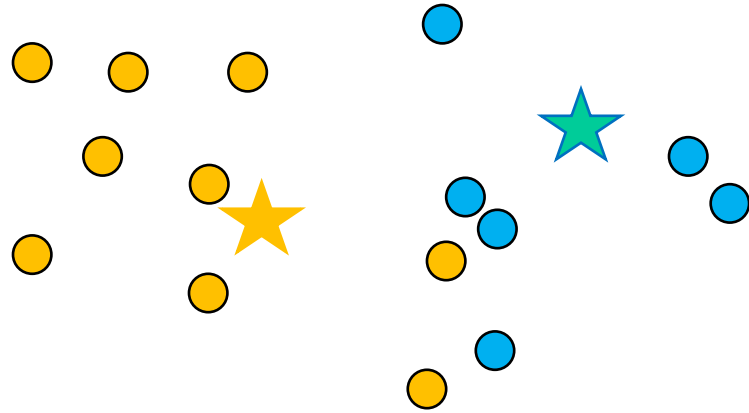
3. Update cluster centers by computing **centroids**



K-Means Clustering: Algorithm

k-means clustering

Repeat Steps 2 & 3 until convergence



K-means clustering (Lloyd's) algorithm

Input: # clusters k , points x_1, \dots, x_n

Step 1: select k cluster centers c_1, \dots, c_k

Step 2: for each point $x \in \{x_1, \dots, x_n\}$ determine its nearest cluster center:

$$i_x = \operatorname{argmin}_i \|x - c_i\|_2$$

Step 3: update cluster centers as the **centroids**:

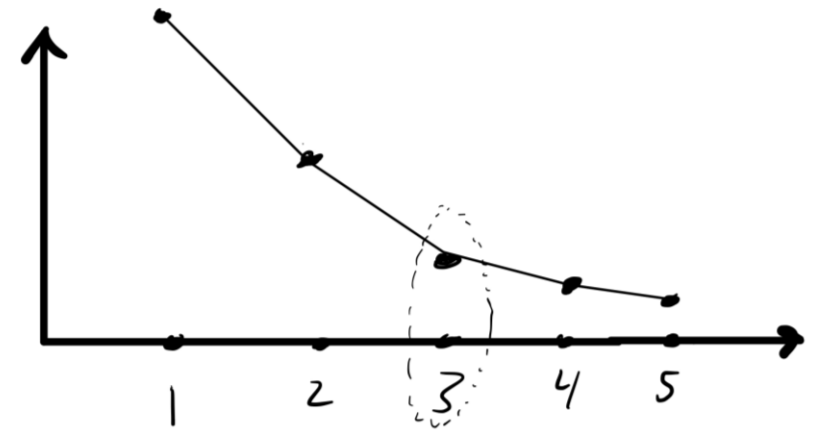
$$c_i = \frac{\sum_{x:i_x=i} x}{|\{x : i_x = i\}|}$$

Repeat step 2 and 3 until the cluster centers no longer change

Questions on k-means

- What is k-means trying to optimize?
- Will k-means stop (converge)?
- Will it find a global or local optimum?
- How many clusters should we use?
- How to pick starting cluster centers?

$$\sum_{x \in \{x_1, \dots, x_n\}} \|x - c_{i_x}\|^2$$



How to pick starting cluster centers?

- Randomly choosing starting centers can lead to poor performance.
- A smarter strategy: k-means ++ (Arthur & Vassilivitski '07)
 - has an $O(\log k)$ -approximation guarantee in expectation
 - commonly used in practice

Choose c_1 randomly from $X = \{X_1, \dots, X_n\}$. Let $C = \{c_1\}$.

For $j = 2, \dots, k$:

(a) Compute $D(X_i) = \min_{c \in C} \|X_i - c\|$ for each X_i .

(b) Choose a point X_i from X with probability

$$p_i = \frac{D^2(X_i)}{\sum_{j=1}^n D^2(X_j)}.$$

(c) Call this randomly chosen point c_j . Update $C \leftarrow C \cup \{c_j\}$.



Break & Quiz

Break & Quiz

Q: True or False: finding an initialization that beats kmeans++ and optimizes the k-means objective exactly is a great open research direction for ML researchers.

Break & Quiz

Q: True or False: finding an initialization that beats kmeans++ and optimizes the k-means objective exactly is a great open research direction for ML researchers.

False: optimizing the k-means objective is NP-Hard.
Leave it to theoretical CS.

Outline

- K-means clustering
- **Gaussian Mixture Models**
 - Mixtures, Expectation-Maximization algorithm
- **Advanced clustering methods**
 - hierarchical, spectral clustering

Mixture Models

- Let us get back to modeling densities in unsupervised learning.
- Have dataset: $\{ (x^{(1)}, x^{(2)}, \dots, x^{(n)}) \}$
- One type of model: **mixtures**
 - A function of a **latent variable** z
 - Model:

$$p(x^{(i)} | z^{(i)}) p(z^{(i)})$$

Mixture Models: Gaussians

- Many different types of mixtures, but let us focus on Gaussians.
- What does this mean?
- Latent variable z has some multinomial distribution, $\sum_{i=1}^k \phi_i = 1$

$$z^{(i)} \sim \text{Multinomial}(\phi)$$

- Then, let us make x be Gaussian conditioned on z

$$x^{(i)} | (z^{(i)} = j) \sim \mathcal{N}(\mu_j, \Sigma_j)$$



Mean Covariance Matrix

Gaussian Mixture Models: Likelihood

- How should we learn the parameters? ϕ, μ_j, Σ_j
- Could try our usual way: maximum likelihood
 - Log likelihood:

$$\ell(\phi, \mu, \Sigma) = \sum_{i=1}^n \log \sum_{z^{(i)}=1}^k p(x^{(i)} | z^{(i)}; \mu, \Sigma) p(z^{(i)}; \phi)$$

- Turns out to be **hard** to solve... inner sum leads to problems!

GMMs: Supervised Setting

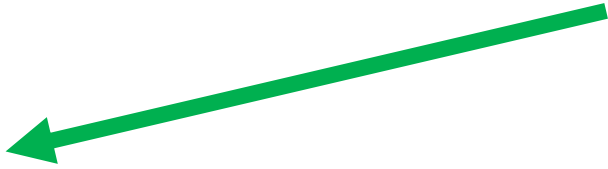
- What if we knew the z 's?
 - “Supervised” setting...
- First, empirically estimate the multinomial parameters:

$$\phi_j = \frac{1}{n} \sum_{i=1}^n 1\{z^{(i)} = j\}$$

- Next the Gaussian components:

$$\mu_j = \frac{\sum_{i=1}^n 1\{z^{(i)} = j\} x^{(i)}}{\sum_{i=1}^n 1\{z^{(i)} = j\}}$$

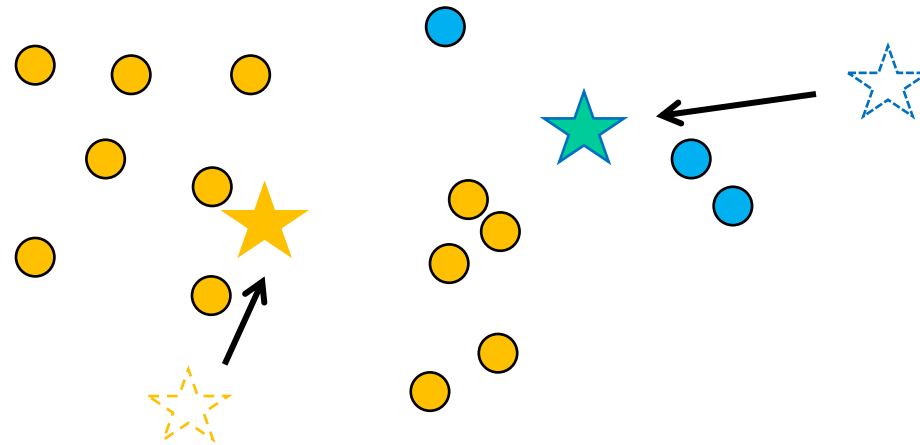
Average of x 's
where $z = j$



$$\Sigma_j = \frac{\sum_{i=1}^n 1\{z_j^{(i)} = j\} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^n 1\{z_j^{(i)} = j\}}$$

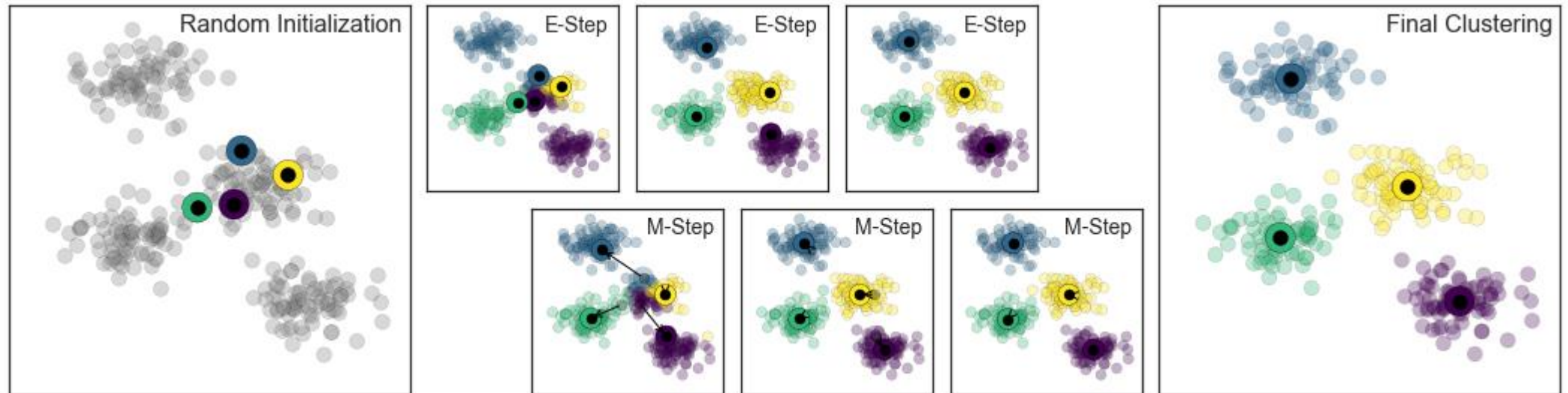
GMMs: Back to Latent Setting

- But, we don't get to see the z 's!
- What could we do instead?
- Recall our **k-means** approach: we don't know the centers, but we pretend we do, perform a clustering, re-center, iterate



GMMs: Expectation Maximization

- EM :an algorithm for dealing with latent variable problems
- Iterative, alternating between two steps:
 - **E**-step: estimate latent variable (probabilities) based on current model
 - **M**-step: update the parameters of $p(x|z)$
 - Note similarity to k-means clustering.



GMM EM: E-Step

- Let us write down the formulas.
- **E-step**: fix parameters, compute posterior:

$$w_j^{(i)} = p(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma)$$

- These w 's are “soft” assignments of the z terms, i.e. probabilities over the values z could take. Concretely:

$$w_j^{(i)} = p(z^{(i)} = j | x^{(i)}; \phi, \mu, \Sigma) = \frac{p(x^{(i)} | z^{(i)} = j; \mu, \Sigma) p(z^{(i)} = j; \phi)}{\sum_{\ell=1}^k p(x^{(i)} | z^{(i)} = \ell; \mu, \Sigma) p(z^{(i)} = \ell; \phi)}$$

GMM EM: M-Step

- Let's write down the formulas.
- **M-step:** fix w , update parameters:

$$\phi_j = \frac{1}{n} \sum_{i=1}^n w_j^{(i)}$$

Soft version of our counting estimator for the supervised case.

$$\mu_j = \frac{\sum_{i=1}^n w_j^{(i)} x^{(i)}}{\sum_{i=1}^n w_j^{(i)}}$$

Soft version of our empirical mean and covariances.

$$\Sigma_j = \frac{\sum_{i=1}^n w_j^{(i)} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^n w_j^{(i)}}$$

EM through the lens of maximum likelihood estimation

- Why is EM a sensible idea?
- Let us write out the log likelihood for our problem

$$\mathcal{L}(\theta) = \sum_{i=1}^n \log p_{\theta}(x^{(i)}) = \sum_{i=1}^n \log \left(\sum_{j=1}^k p_{\theta}(x^{(i)}, z^{(i)} = j) \right)$$

- Letting $Q^{(i)} = [Q_1^{(i)}, \dots, Q_k^{(i)}]$ be any distribution over $z^{(i)}$

$$\mathcal{L}(\theta) = \sum_{i=1}^n \log \left(\sum_{j=1}^k Q_j^{(i)} \frac{p_{\theta}(x^{(i)}, z^{(i)} = j)}{Q_j^{(i)}} \right)$$

EM through the lens of maximum likelihood estimation

- Letting $Q^{(i)} = [Q_1^{(i)}, \dots, Q_k^{(i)}]$ be any distribution over $z^{(i)}$

$$\mathcal{L}(\theta) = \sum_{i=1}^n \log \left(\sum_{j=1}^k Q_j^{(i)} \frac{p_{\theta}(x^{(i)}, z^{(i)} = j)}{Q_j^{(i)}} \right)$$

- By an application of Jensen's inequality: $f(E[X]) \geq E[f(X)]$ for concave f

$$\mathcal{L}(\theta) \geq \sum_{i=1}^n \sum_{j=1}^k Q_j^{(i)} \log \left(\frac{p_{\theta}(x^{(i)}, z^{(i)} = j)}{Q_j^{(i)}} \right)$$

EM through the lens of maximum likelihood estimation

- We have a lower bound on the log likelihood:

$$\mathcal{L}(\theta) \geq \sum_{i=1}^n \sum_{j=1}^k Q_j^{(i)} \log \left(\frac{p_{\theta}(x^{(i)}, z^{(i)} = j)}{Q_j^{(i)}} \right)$$

- If this lower bound is **tight**, by maximizing the lower bound, we can hope to do well in maximizing the likelihood.
- A good choice is $Q_j^{(i)} = p_{\theta}(z^{(i)} = j | x^{(i)})$ (guarantees likelihood increases every step)

General EM Algorithm

On round t of EM:

- E-Step (Expectation): Update $Q_j^{(i)}$ for all i and j (This effectively computes the lower bound)

$$Q_j^{(i)} \leftarrow p_{\theta_t}(z^{(i)} = j | x^{(i)})$$

- M-step: Maximize lower bound with respect to parameters θ_t

$$\theta_{t+1} \leftarrow \arg \max_{\theta} \sum_{i=1}^n \sum_{j=1}^k Q_j^{(i)} \log \left(\frac{p_{\theta}(x^{(i)}, z^{(i)} = j)}{Q_j^{(i)}} \right)$$

Do at home: Show that this corresponds to the GMM update equations



Break & Quiz

Q: State if the following sentences are true or false.

A. In a Gaussian mixture model, the log likelihood is concave in the parameters.

B. We can maximize the likelihood of a mixture model using gradient descent.

C. EM is always guaranteed to find a global maximum

$$\mathcal{L}(\theta) = \sum_{i=1}^n \log p_{\theta}(x^{(i)}) = \sum_{i=1}^n \log \left(\sum_{j=1}^k p_{\theta}(x^{(i)}, z^{(i)} = j) \right)$$

Ans: A: false, B: true, C: false

We use EM over GD because it is more efficient than GD.

Q: Which of the following sentences are true.

- A. GMMs are generative models
- B. When you learn a GMM, you are estimating the density of the data.
- C. GMMs can be used for clustering.

Ans: All are true

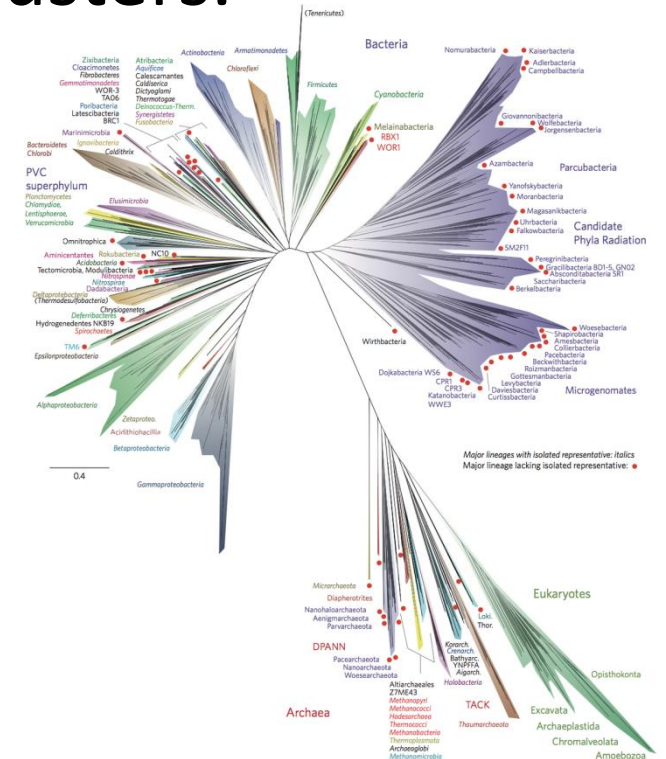
Outline

- K-means clustering
- Gaussian Mixture Models
 - Mixtures, Expectation-Maximization algorithm
- **Advanced clustering methods**
 - hierarchical, spectral

Hierarchical Clustering

Basic idea: build a “hierarchy”

- Want: arrangements from specific to general
- One advantage: no need for k, number of clusters.
- **Input:** points.
- **Output:** a hierarchy (a binary tree)



Credit: Wikipedia

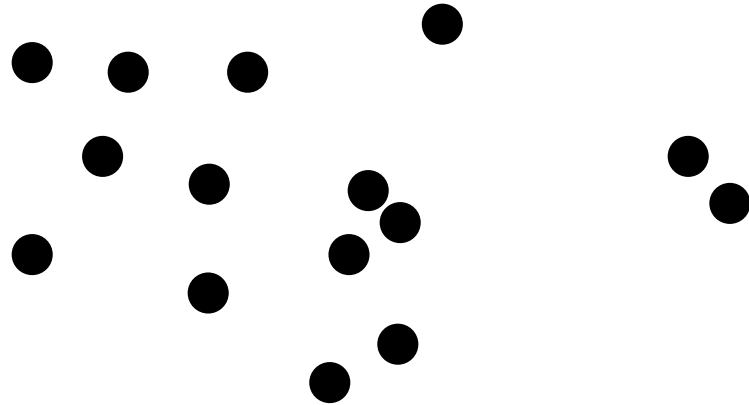
HC: Agglomerative vs Divisive

Two ways to go:

- **Agglomerative:** bottom up.
 - Start: each point a cluster.
 - Progressively merge clusters
- **Divisive:** top down
 - Start: all points in one cluster.
 - Progressively split clusters

HC: Agglomerative Clustering Example

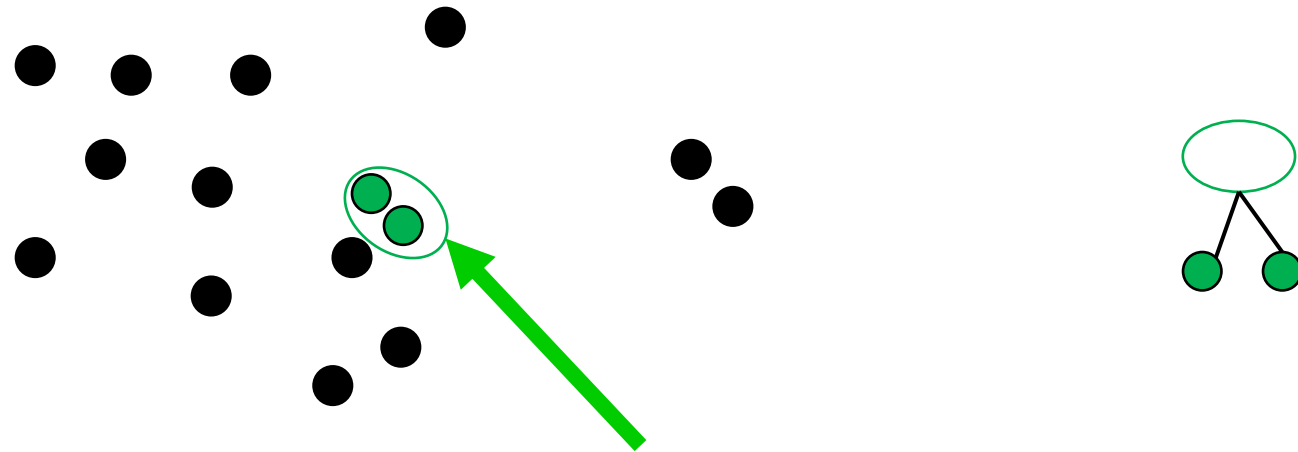
Agglomerative: Start: every point is its own cluster



HC: Agglomerative Clustering Example

Basic idea: build a “hierarchy”

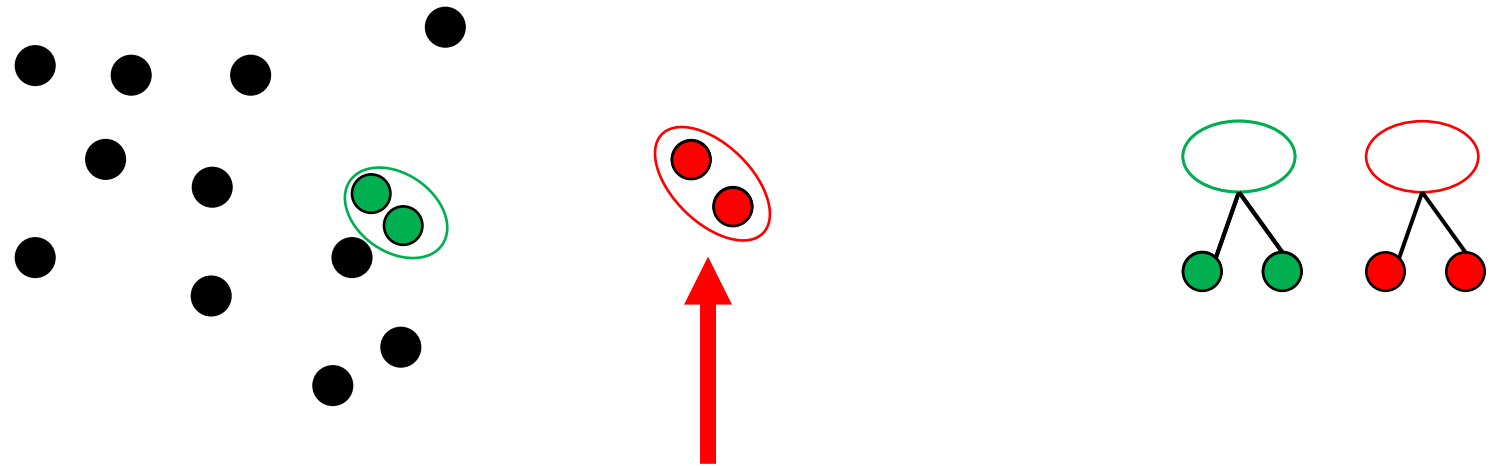
- Get pair of clusters that are closest and merge



HC: Agglomerative Clustering Example

Basic idea: build a “hierarchy”

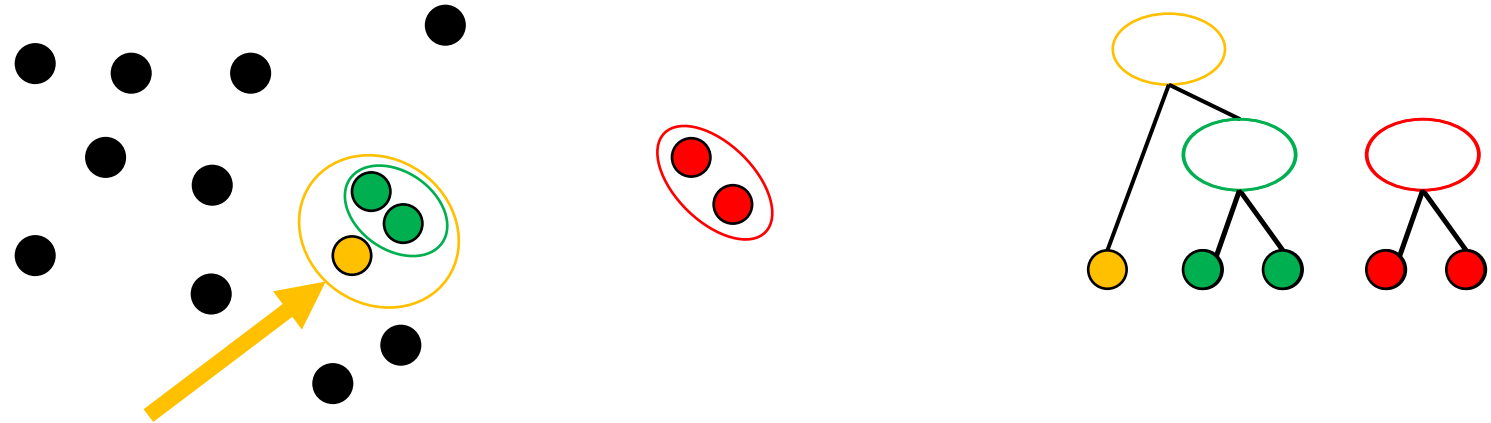
- **Repeat:** Get pair of clusters that are closest and merge



HC: Agglomerative Clustering Example

Basic idea: build a “hierarchy”

- **Repeat:** Get pair of clusters that are closest and merge



HC: Merging Criteria

Merge: use closest clusters. Define closest?

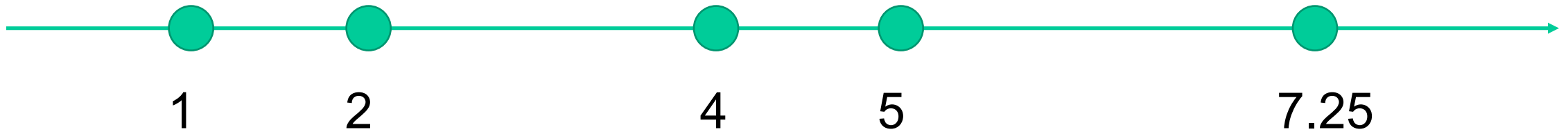
First define a distance between points $d(x_1, x_2)$. Then, define distance between clusters.

- Single-linkage
$$d(A, B) = \min_{x_1 \in A, x_2 \in B} d(x_1, x_2)$$
- Complete-linkage
$$d(A, B) = \max_{x_1 \in A, x_2 \in B} d(x_1, x_2)$$
- Average-linkage
$$d(A, B) = \frac{1}{|A||B|} \sum_{x_1 \in A, x_2 \in B} d(x_1, x_2)$$

Single-linkage Example

We'll merge using single-linkage

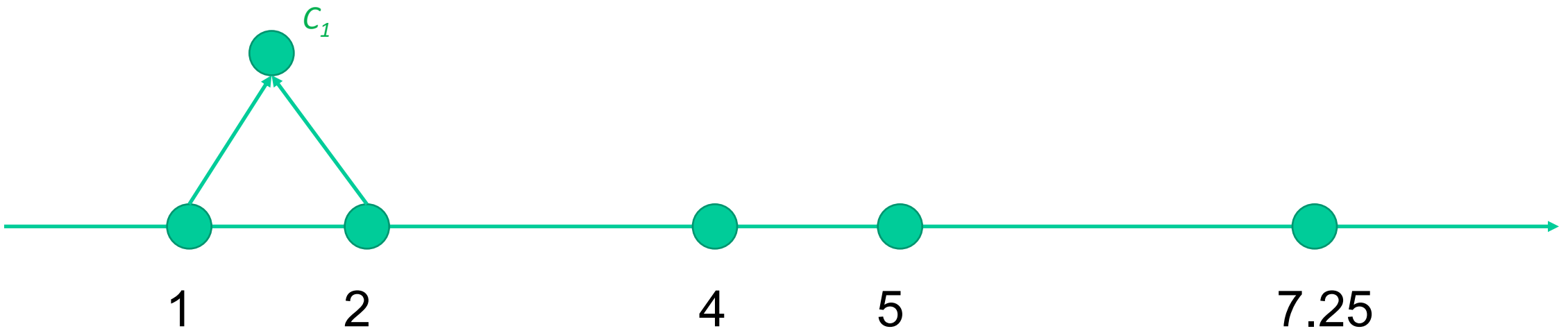
- 1-dimensional vectors.
- Initial: all points are clusters



Single-linkage Example

$$d(C_1, \{4\}) = d(2, 4) = 2$$

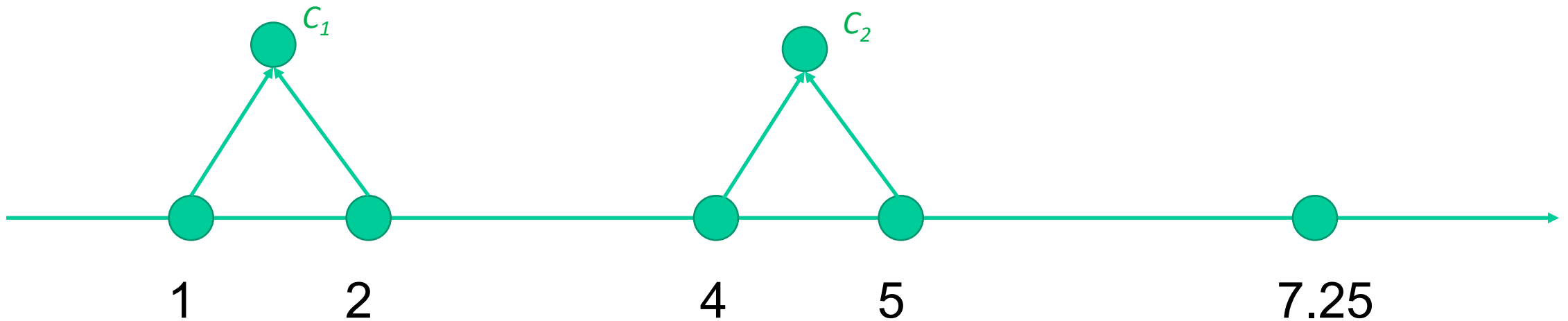
$$d(\{4\}, \{5\}) = d(4, 5) = 1$$



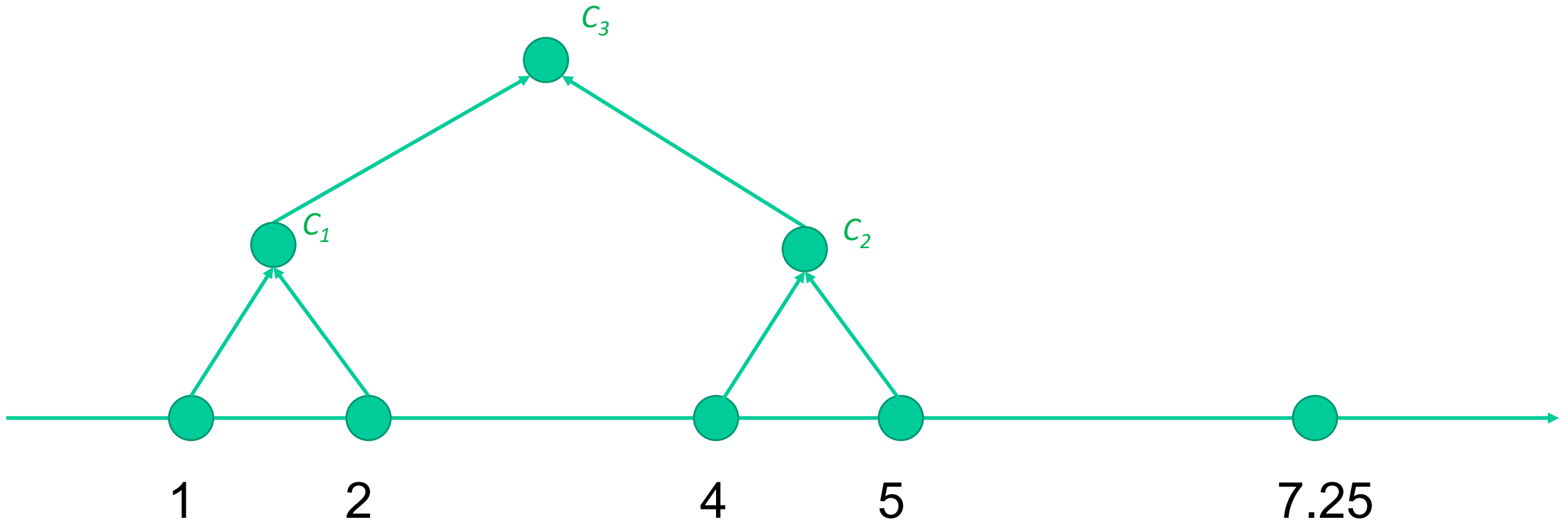
Single-linkage Example

$$d(C_1, C_2) = d(2, 4) = 2$$

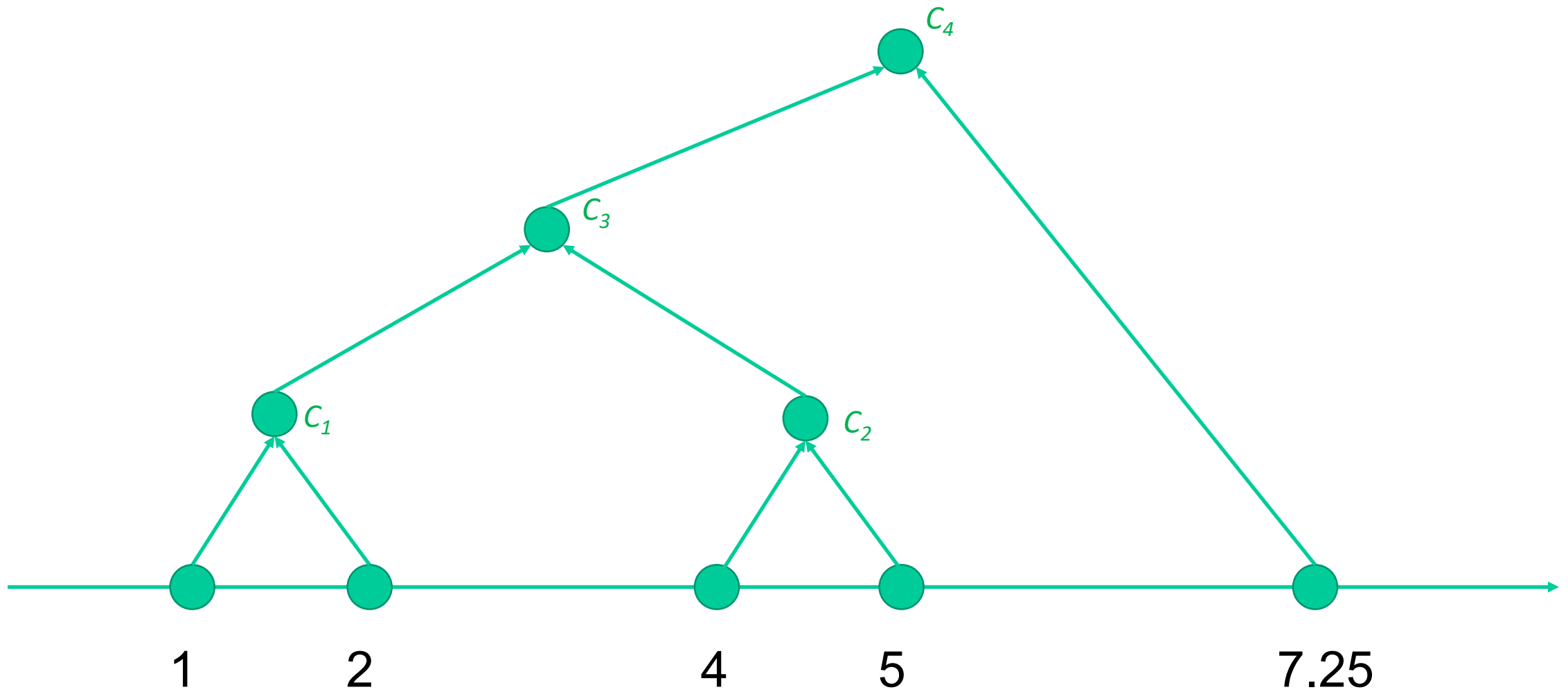
$$d(C_2, \{7.25\}) = d(5, 7.25) = 2.25$$



Single-linkage Example



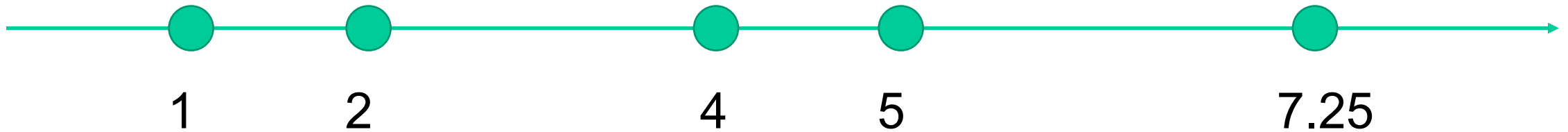
Single-linkage Example



Complete-linkage Example

We'll merge using complete-linkage

- 1-dimensional vectors.
- Initial: all points are clusters

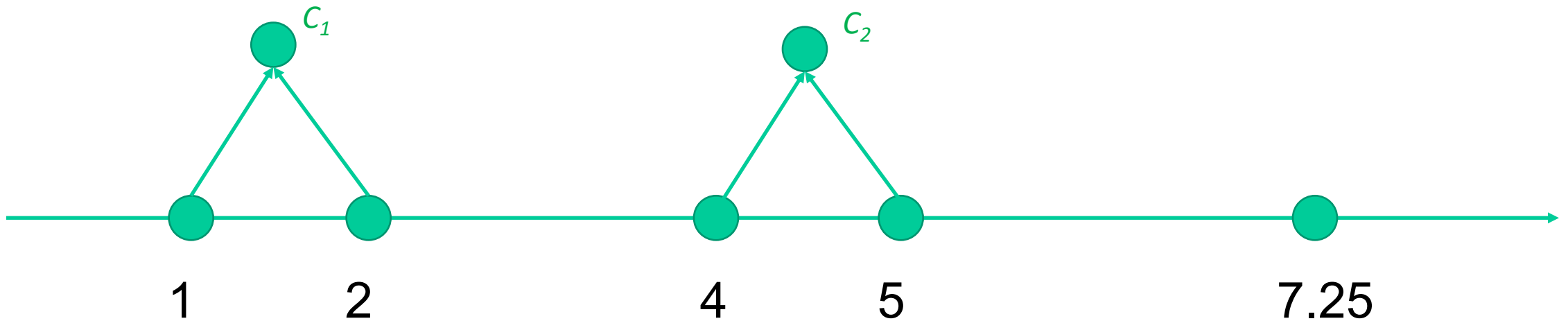


Complete-linkage Example

Beginning is the same...

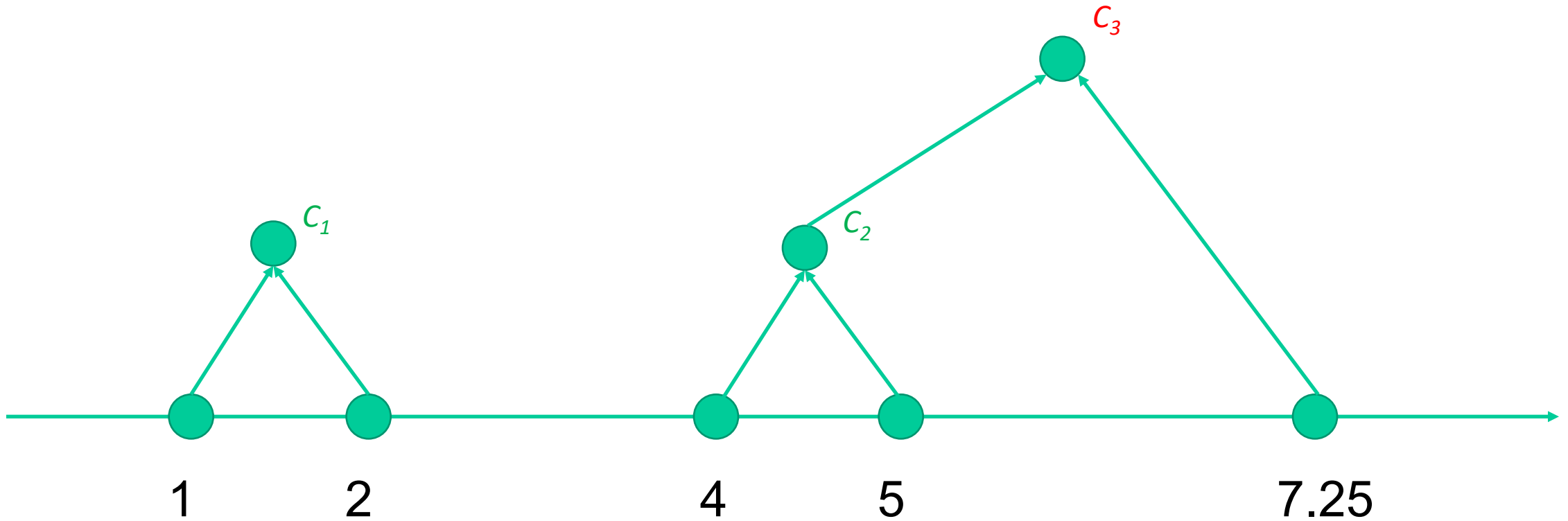
$$d(C_1, C_2) = d(1, 5) = 4$$

$$d(C_2, \{7.25\}) = d(4, 7.25) = 3.25$$

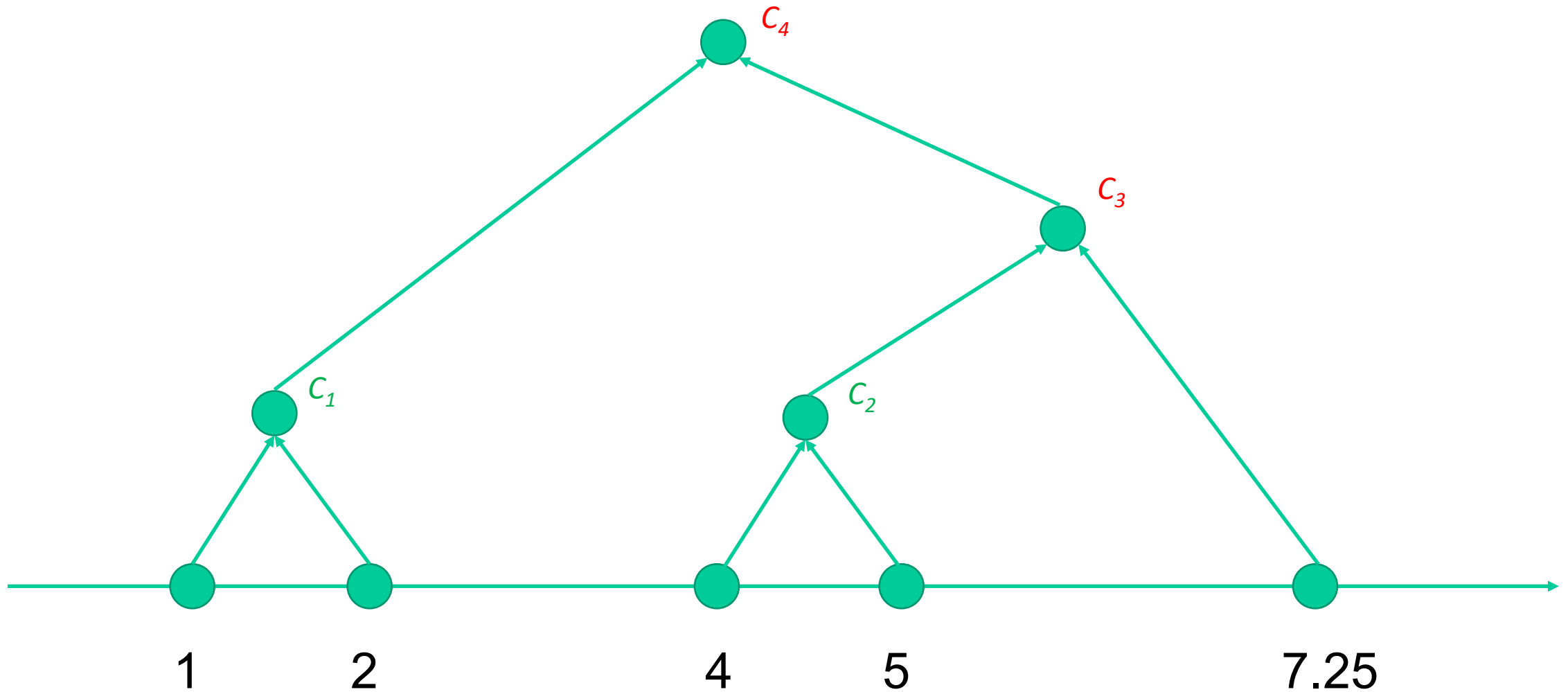


Complete-linkage Example

Now different from single linkage:



Complete-linkage Example





Break & Quiz

Break & Quiz

Q: If we do hierarchical clustering on n points, the maximum depth of the resulting tree is

- A. 2
- B. $\log_2 n$
- C. $n/2$
- D. $n-1$

Break & Quiz

Q: If we do hierarchical clustering on n points, the maximum depth of the resulting tree is

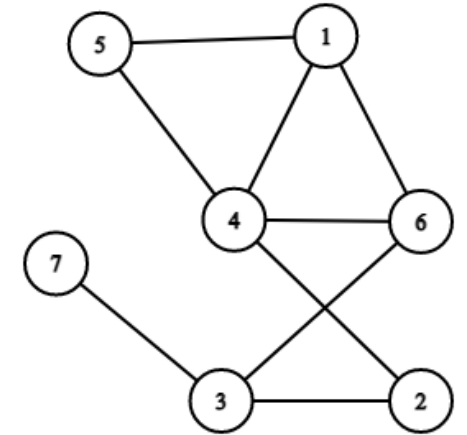
- A. 2
- B. $\log_2 n$
- C. $n/2$
- **D. $n-1$**

Graph/proximity based clustering

- Recall: graph $G = (V, E)$ has vertex set V , edge set E .
 - Edges can be weighted or unweighted
 - Encode **similarity**
- Treat each data point as a node in a graph.
- Edges based on similarity of data points
- e.g. for Euclidean vectors:

$$w_{ij} = e^{-\alpha \|x_i - x_j\|^2}$$

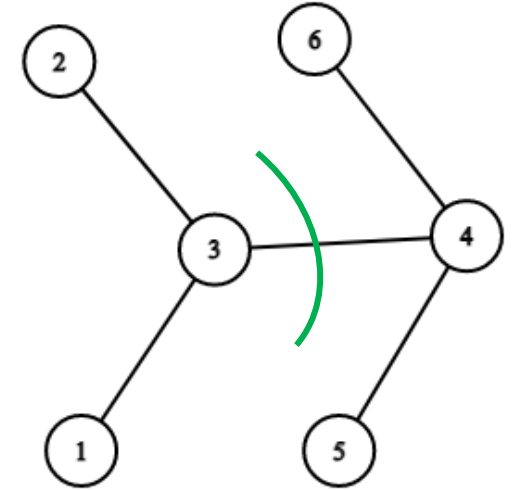
- But they don't need to be in Euclidean space!



Graph-Based Clustering

Want: partition V into k groups

- Implies a graph “cut”
- One idea: minimize the **weight** of the cut



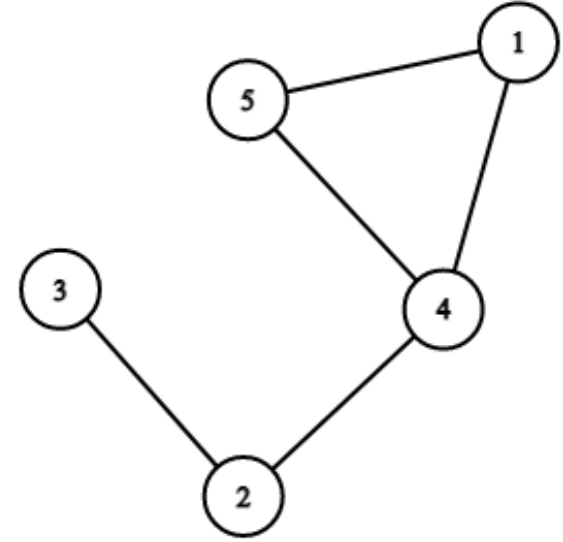
$$W(A, B) = \sum_{i \in A, j \in B} w_{ij}$$

$$\text{cut}(A_1, \dots, A_k) := \frac{1}{2} \sum_{i=1}^k W(A_i, \bar{A}_i).$$

Partition-Based Clustering

How do we compute these?

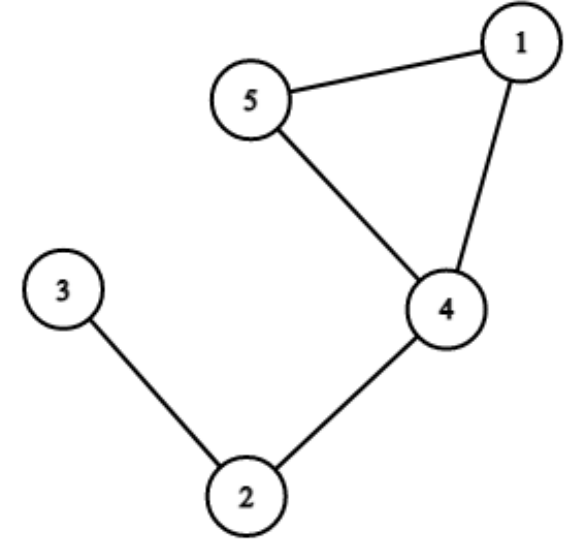
- Hard problem → heuristics
 - Greedy algorithm
 - “Spectral” approaches
- Spectral clustering approach:
 - **Adjacency** matrix



$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Partition-Based Clustering

- Spectral clustering approach:
 - **Adjacency** matrix
 - **Degree** matrix

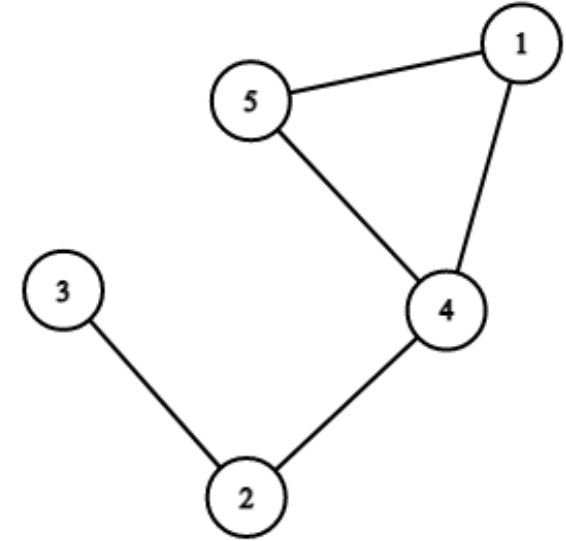


$$D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Spectral Clustering

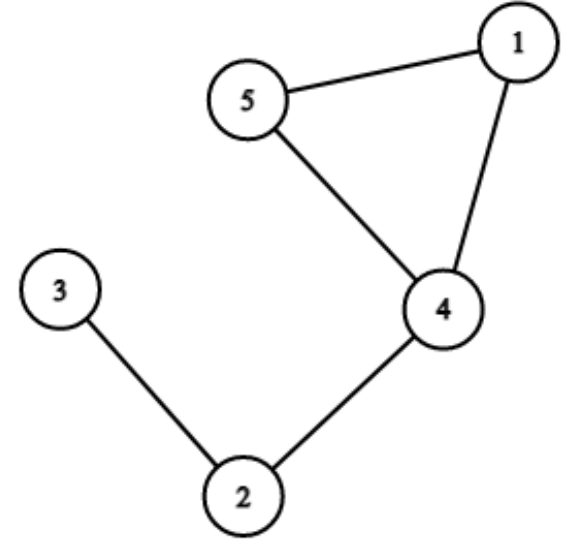
- Spectral clustering approach:
 - 1. Compute **Laplacian** $L = D - A$
(Important tool in graph theory)



$$L = \underbrace{\begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}}_{\text{Degree Matrix}} - \underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{Adjacency Matrix}} = \underbrace{\begin{bmatrix} 2 & 0 & 0 & -1 & -1 \\ 0 & 2 & -1 & -1 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ -1 & -1 & 0 & 3 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{bmatrix}}_{\text{Laplacian}}$$

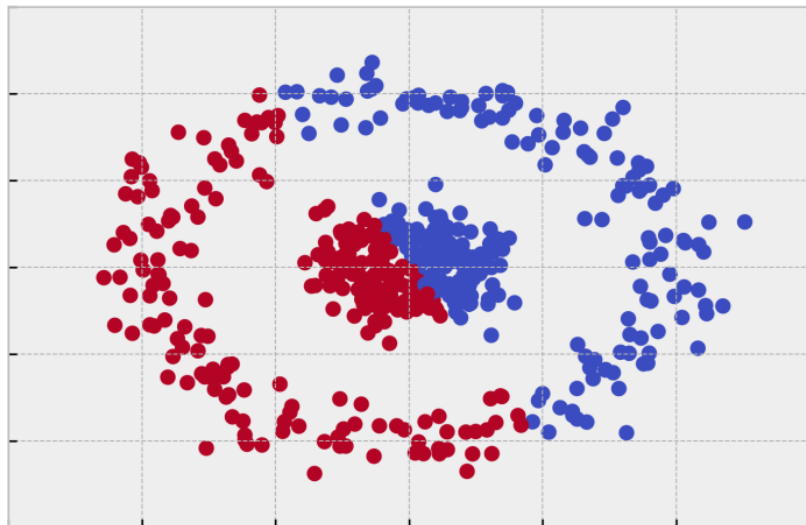
Spectral Clustering

- Spectral clustering approach:
 1. Compute **Laplacian** $\mathbf{L} = \mathbf{D} - \mathbf{A}$
 - 1a (optional): compute normalized Laplacian:
 $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$, or $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1} \mathbf{A}$
 2. Compute k **smallest** eigenvectors u_1, \dots, u_k of \mathbf{L}
 - Why? Connected to graph cuts via the **Fiedler vector**
 3. Set U to be the $n \times k$ matrix with u_1, \dots, u_k as columns. Take the n rows formed as points
 4. Run k-means on the representations



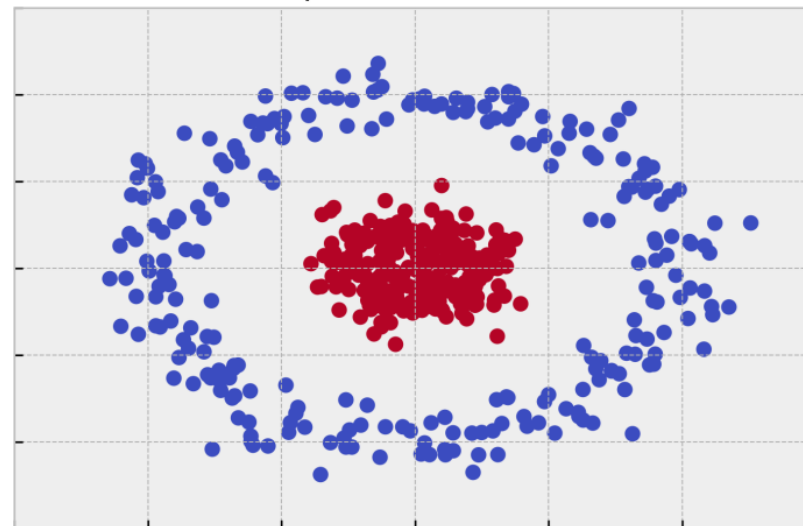
Spectral Clustering

K-Means Circles



Credit: William Fleshman

Spectral Clustering





Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov, Fred Sala, Kirthy Kandasamy, Tengyang Xie