



# CS 540 Introduction to Artificial Intelligence

## **Unsupervised Learning I**

Spring 2025

# Announcements

- **Homework:**
  - HW4 to be released today
- Class roadmap:

<b>ML Unsupervised I</b>
ML Unsupervised II
ML Linear Regression
Machine Learning: K - Nearest Neighbors & Naive Bayes

Machine Learning

# Recap of Supervised/Unsupervised

## Supervised learning:

- Make predictions, classify data, perform regression

- Dataset:  $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)$



Features / Covariates / Input

Labels / Outputs

- Goal: find function  $f : X \rightarrow Y$  to predict label on **new** data



indoor

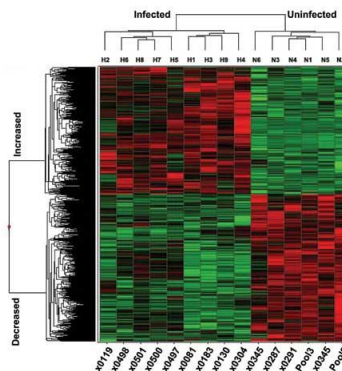
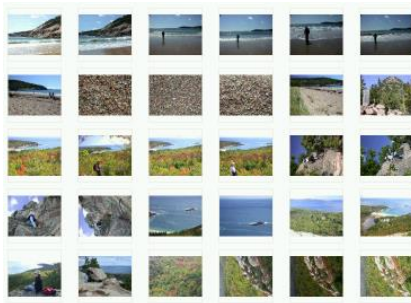
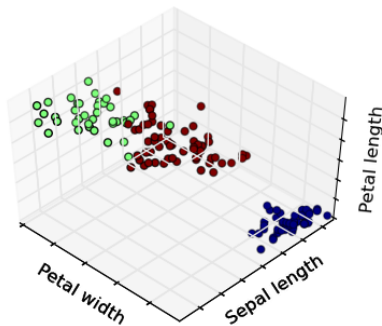


outdoor

# Recap of Supervised/Unsupervised

## Unsupervised learning:

- No labels; generally, won't be making predictions
- Dataset:  $X_1, X_2, \dots, X_n$
- Goal: find patterns & structures that help better understand data.



Mulvey and Gingold

# Recap of Supervised/Unsupervised

Note that there are **other kinds** of ML:

- Mixtures: semi-supervised learning, self-supervised
  - Idea: different types of “signal”
- Reinforcement learning
  - Learn how to act in order to maximize rewards
  - Later on in course...



# Outline

- Intro to Clustering
  - Clustering Types
- Hierarchical Clustering
  - Divisive, agglomerative, linkage strategies
- Centroid-based Clustering
  - k-means

# Unsupervised Learning & Clustering

- Note that clustering is just one type of unsupervised learning (**UL**)
  - PCA is another unsupervised algorithm
- Estimating probability distributions also UL (GANs)
- Clustering is popular & useful!



StyleGAN2 (Karras et al '20)

# Clustering Types

- Several types of clustering

## Hierarchical

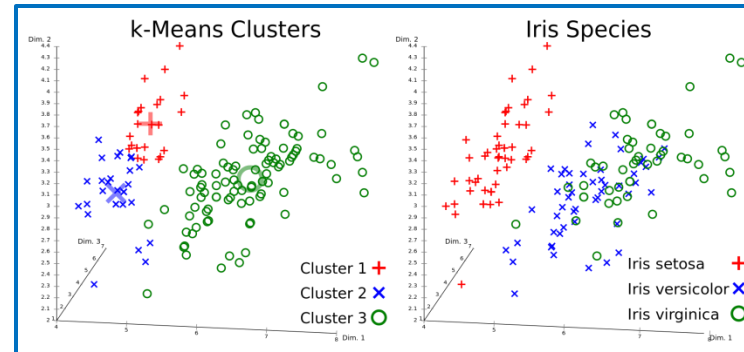
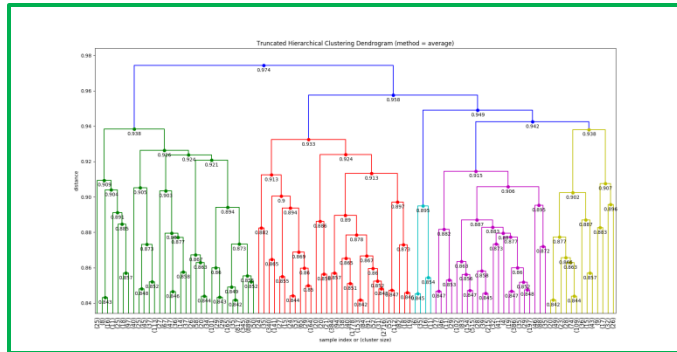
- Agglomerative
- Divisive

## Partitional

- Center-based
- Graph-theoretic
- Spectral

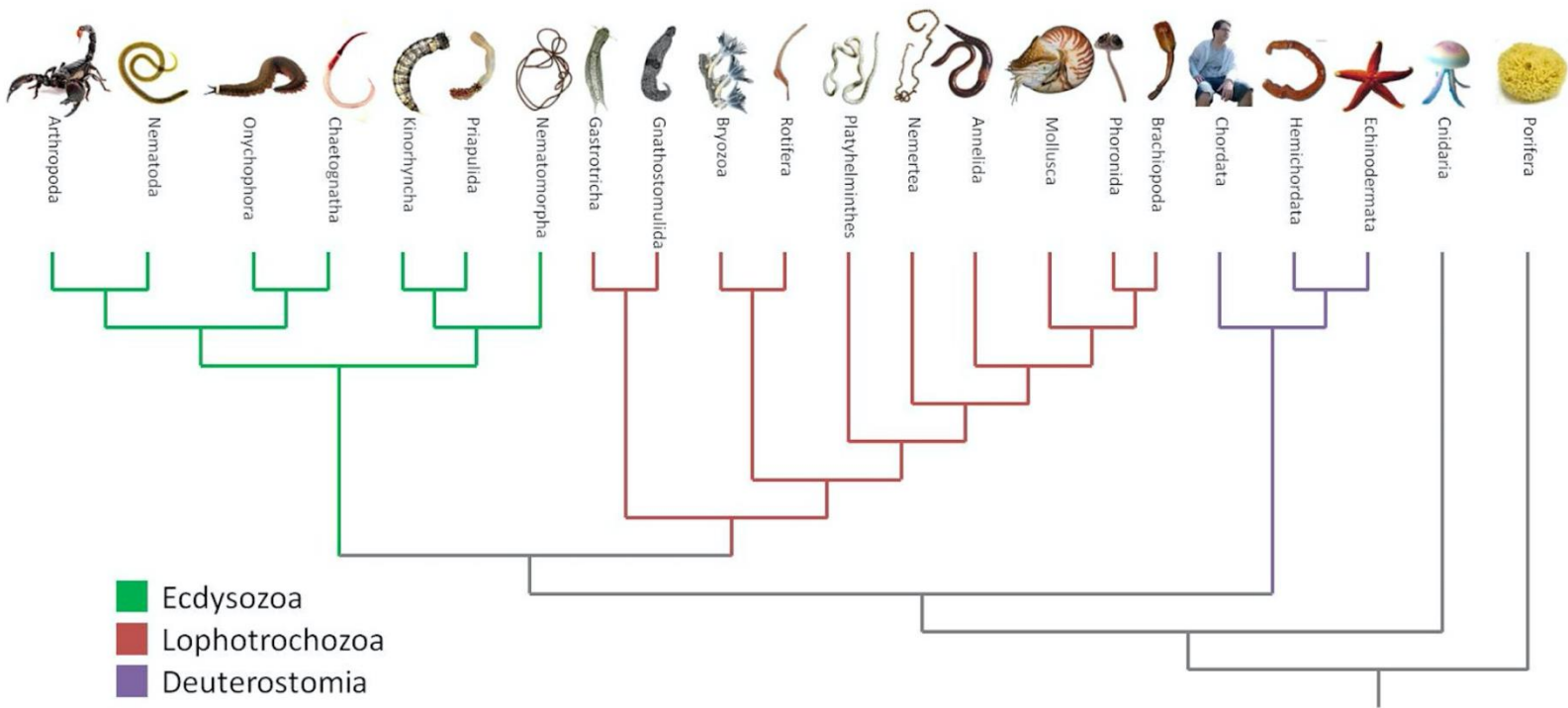
## Bayesian

- Decision-based
- Nonparametric





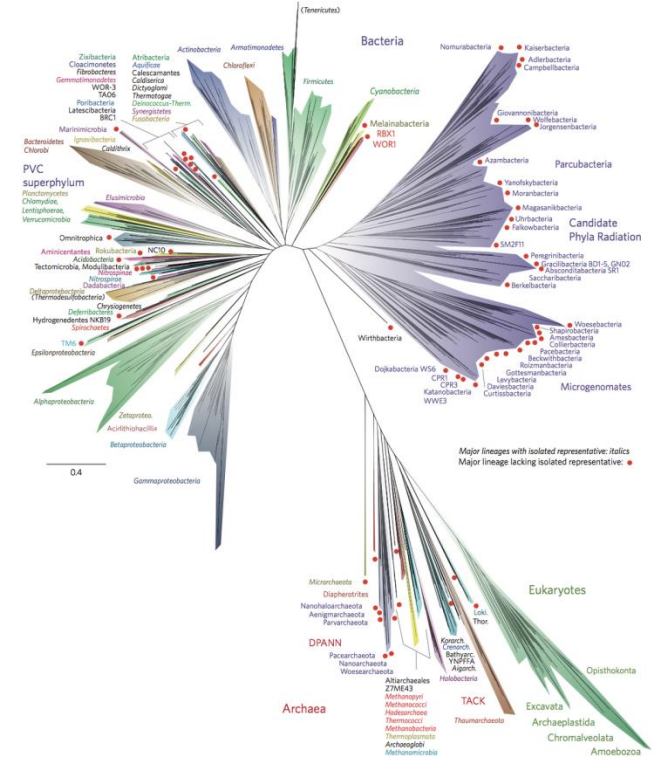
# Hierarchical Clustering



# Hierarchical Clustering

Basic idea: build a “hierarchy”

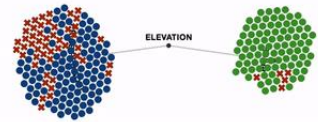
- Want: arrangements from specific to general
- One advantage: no need for k, number of clusters.
- **Input:** points. **Output:** a hierarchy
  - A binary tree



# Agglomerative vs Divisive

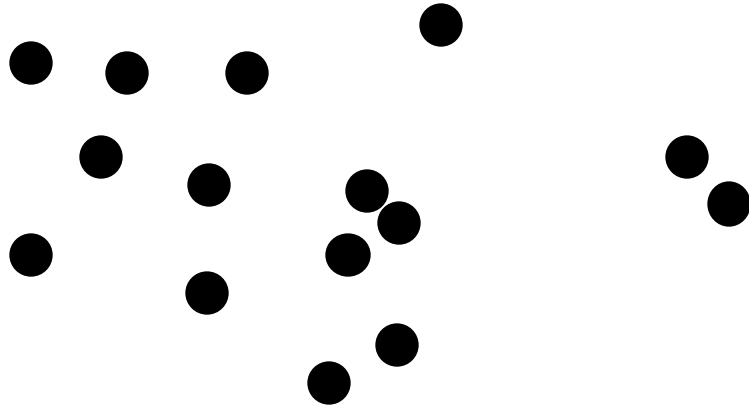
Two ways to go:

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



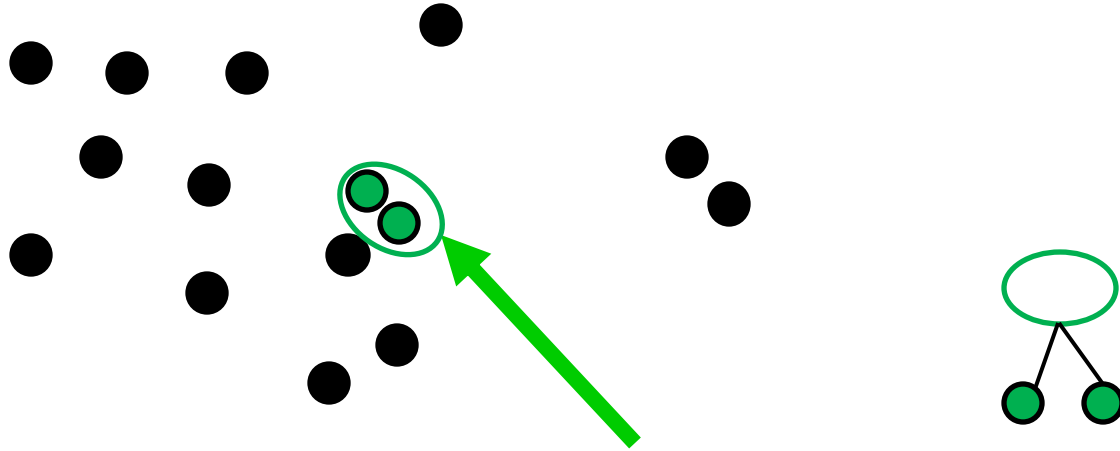
# Agglomerative Clustering Example

**Agglomerative.** Start: every point is its own cluster



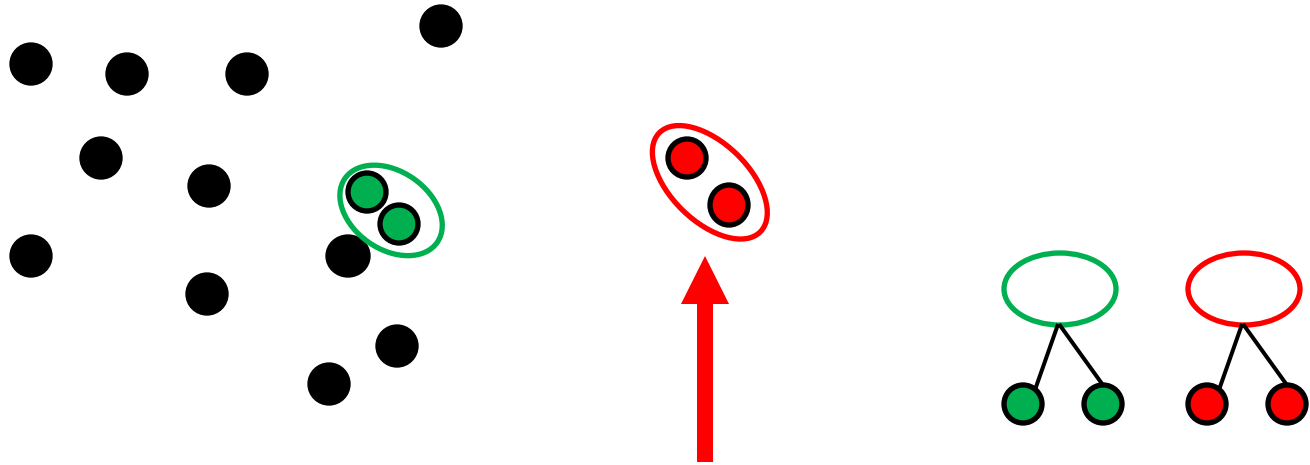
# Agglomerative Clustering Example

**Get** pair of clusters that are closest and merge



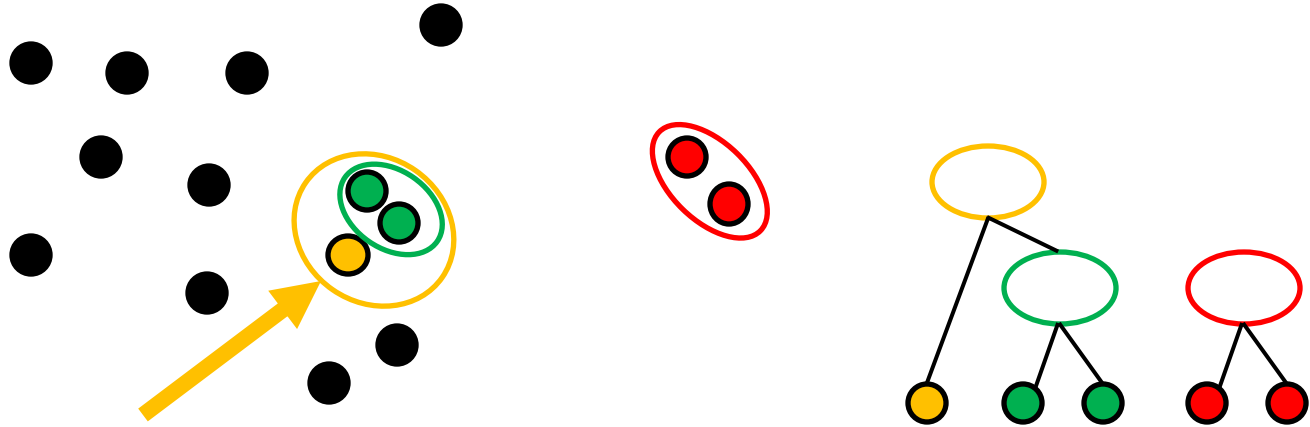
# Agglomerative Clustering Example

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



# Agglomerative Clustering Example

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



# Merging Criteria

Merge: use closest clusters. Define closest?

- 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

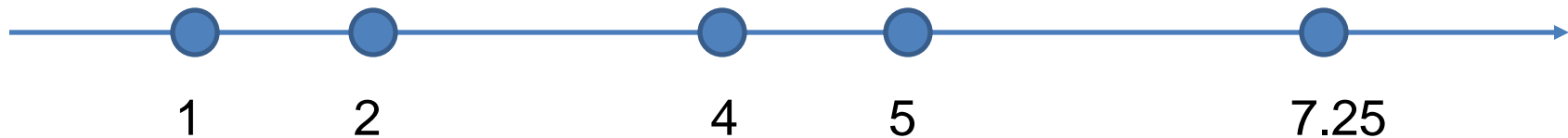
$$d(A, B) = \frac{1}{|A||B|} \sum_{x_1 \in A, x_2 \in B} d(x_1, x_2)$$



# Single-linkage Example

We'll merge using single-linkage

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

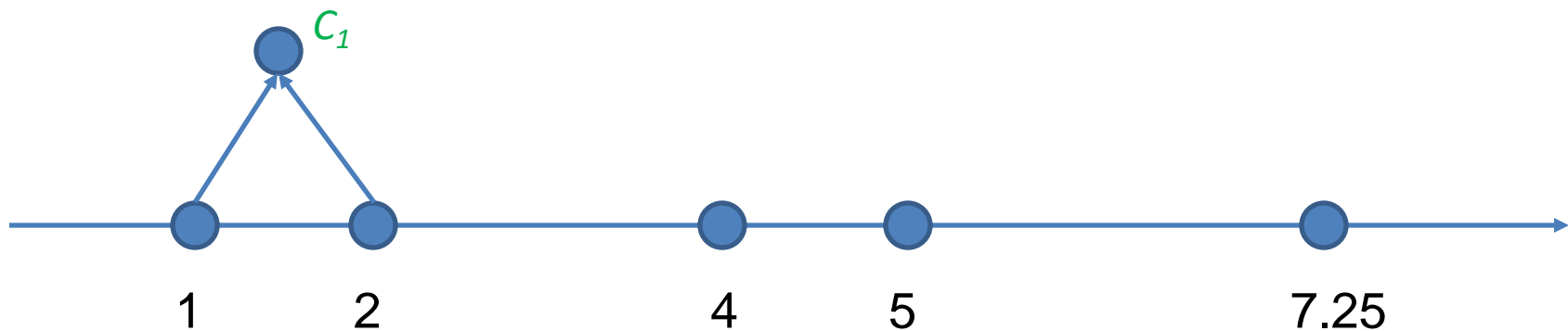


# Single-linkage Example

We'll merge using single-linkage

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

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

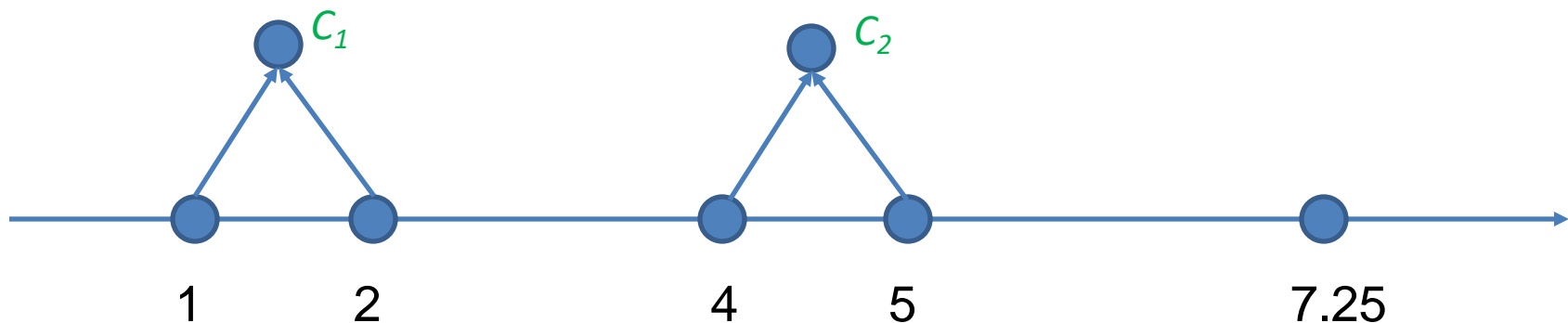


# Single-linkage Example

Continue...

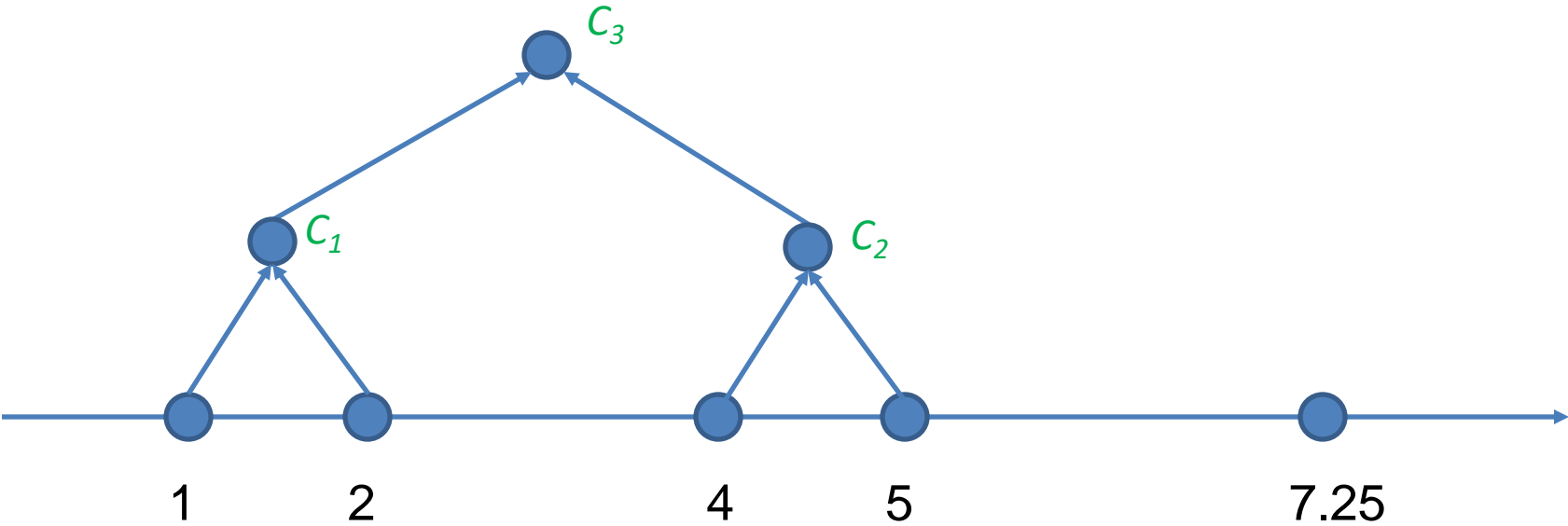
$$d(C_1, C_2) = d(2, 4) = 2$$

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

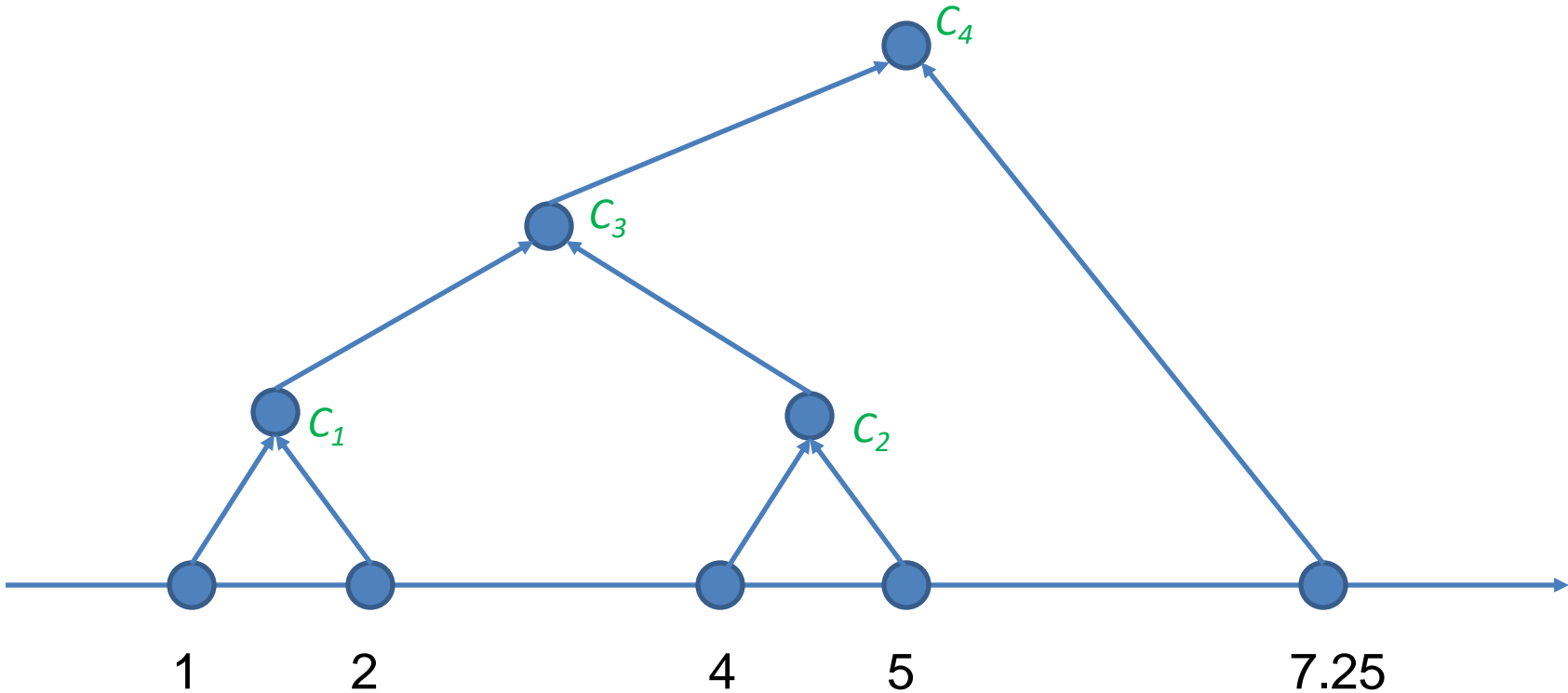


# Single-linkage Example

Continue...



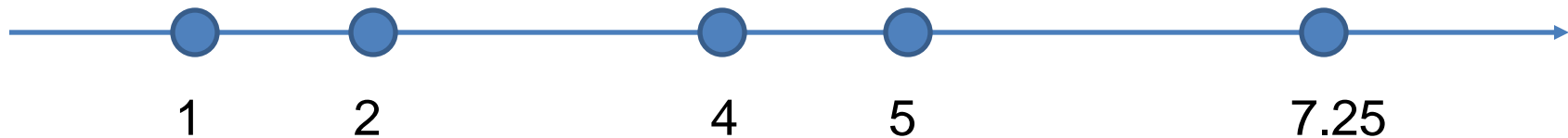
# Single-linkage Example



# Complete-linkage Example

We'll merge using complete-linkage

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

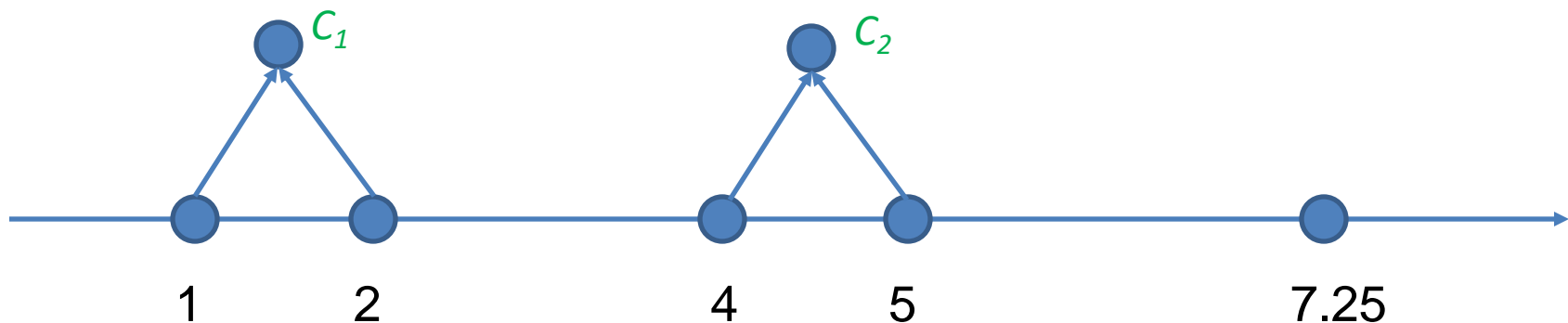


# Complete-linkage Example

Beginning is the same...

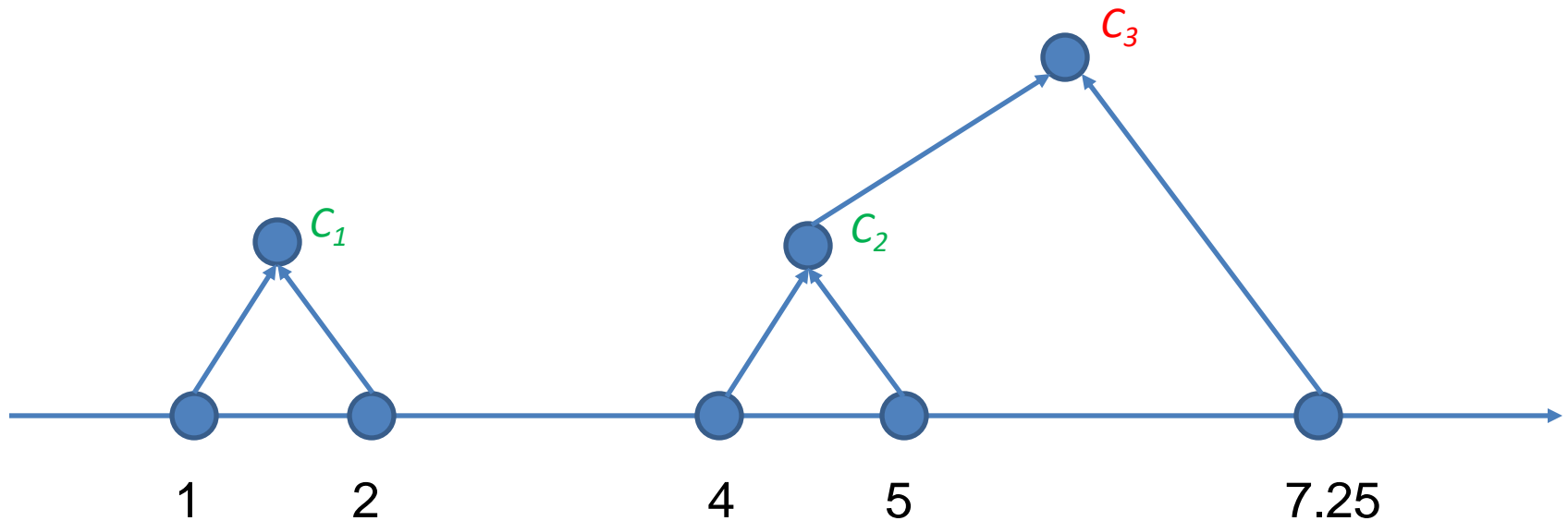
$$d(C_1, C_2) = d(1, 5) = 4$$

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



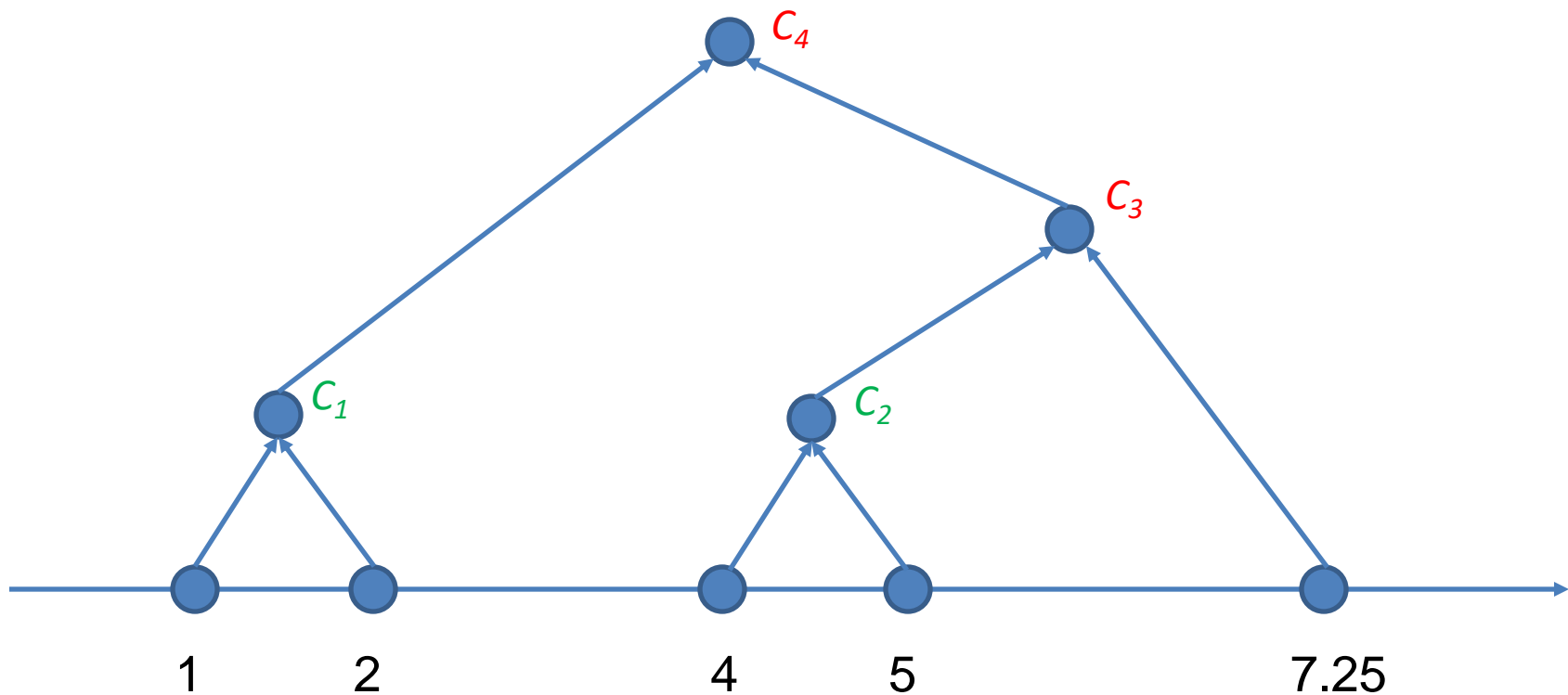
# Complete-linkage Example

Now we diverge:





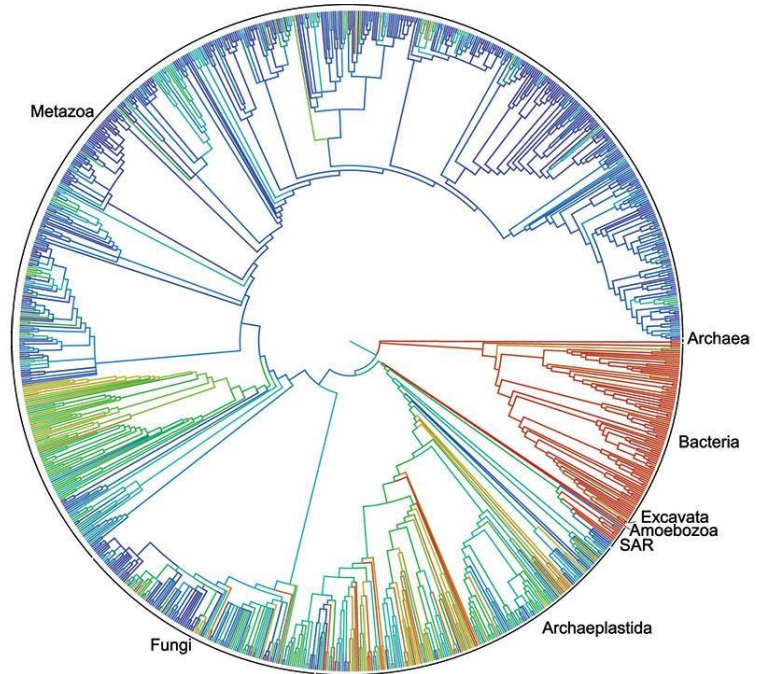
# Complete-linkage Example



# When to Stop?

No simple answer:

- Use the binary tree (a **dendrogram**)
- Cut at different levels (get different heights/depths)

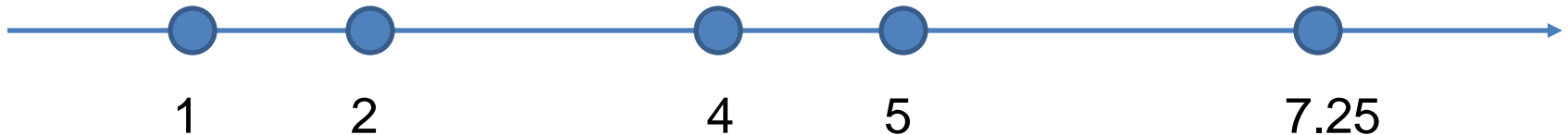


<http://opentreeoflife.org/>

# Break & Quiz

**Q 1.1:** Let's do hierarchical clustering for two clusters with average linkage on the dataset below. What are the clusters?

- A. {1}, {2,4,5,7.25}
- B. {1,2}, {4, 5, 7.25}
- C. {1,2,4}, {5, 7.25}
- D. {1,2,4,5}, {7.25}



# Break & Quiz

**Q 1.1:** Let's do hierarchical clustering for two clusters with average linkage on the dataset below. What are the clusters?

- A. {1}, {2,4,5,7.25}
- **B. {1,2}, {4, 5, 7.25}**
- C. {1,2,4}, {5, 7.25}
- D. {1,2,4,5}, {7.25}

Iteration 1: merge 1 and 2

Iteration 2: merge 4 and 5

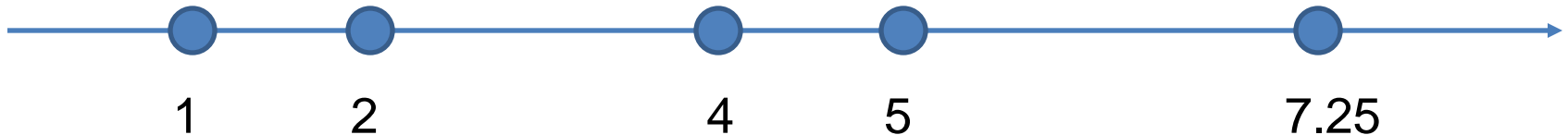
Iteration 3: Now we have clusters {1,2}, {4,5}, {7.25}.

$\text{distance}(\{1,2\}, \{4,5\}) = 3$

$\text{distance}(\{4,5\}, \{7.25\}) = 2.75$

$\text{distance}(\{1,2\}, \{7.25\})$  is clearly larger than the above two.

So average linkage will merge {4,5} and {7.25}



# Break & Quiz

**Q 1.2:** If we do hierarchical clustering on  $n$  points, the maximum depth of the resulting tree is

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

# Break & Quiz

**Q 1.2:** If we do hierarchical clustering on  $n$  points, the maximum depth of the resulting tree is

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

Denote the points as  $x_1, x_2, \dots, x_n$

Suppose:

in iteration 1, we merge points  $x_1$  and  $x_2$

in iteration 2, we merge  $\{x_1, x_2\}$  with  $x_3$

...

in iteration  $t$ , we merge  $\{x_1, x_2, \dots, x_t\}$  with  $x_{t+1}$

...

in iteration  $n-1$ , we merge  $\{x_1, x_{n-1}\}$  with  $x_n$

Then we will get a tree with depth  $n-1$ .

# Center-based Clustering

- k-means is an example of a partitional, **center-based clustering algorithm**.
- Specify a desired number of clusters,  $k$ ; run k-means to find  $k$  clusters.

# K-means clustering

- Very popular clustering method

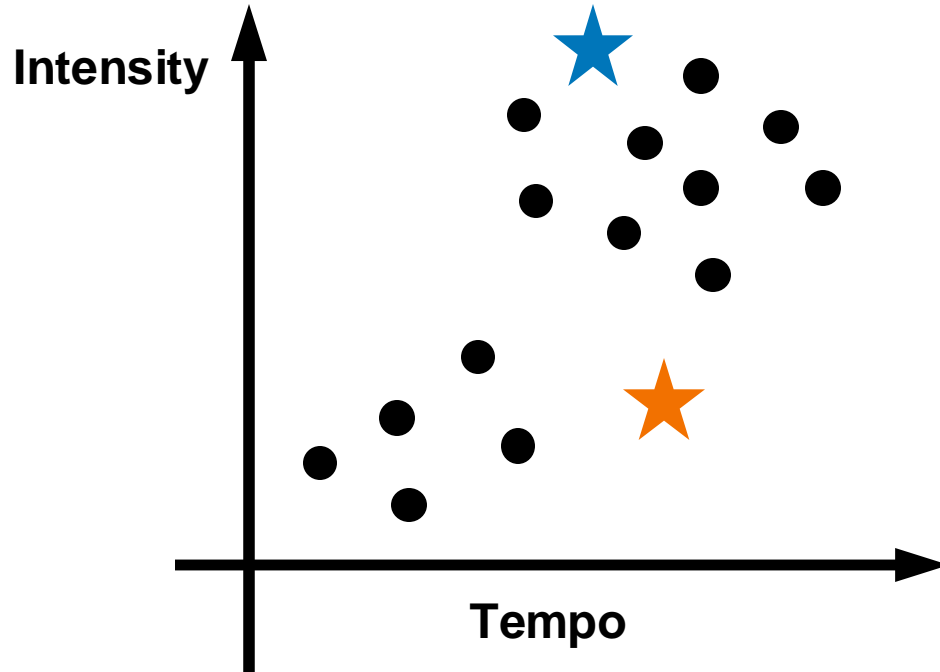
$$x_1, x_2, \dots, x_n$$

- Input: a dataset, and assume the number of clusters **k** is given



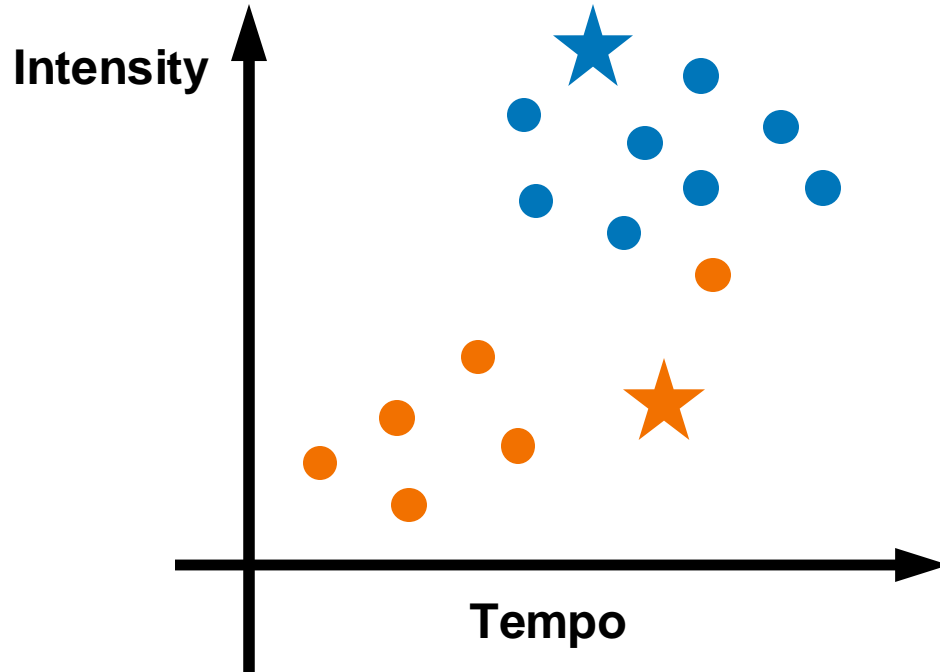
# K-means clustering

Step 1: **Randomly** picking 2 positions as initial cluster centers  
(not necessarily a data point)



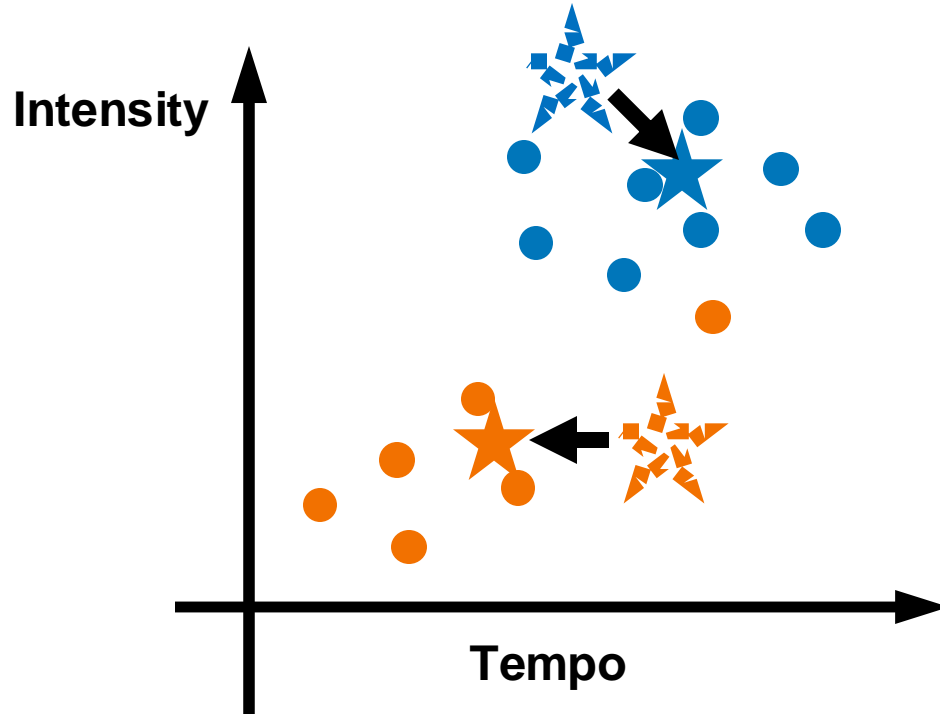
# K-means clustering

Step 2: for each point  $x$ , determine its cluster: find the closest center in Euclidean space



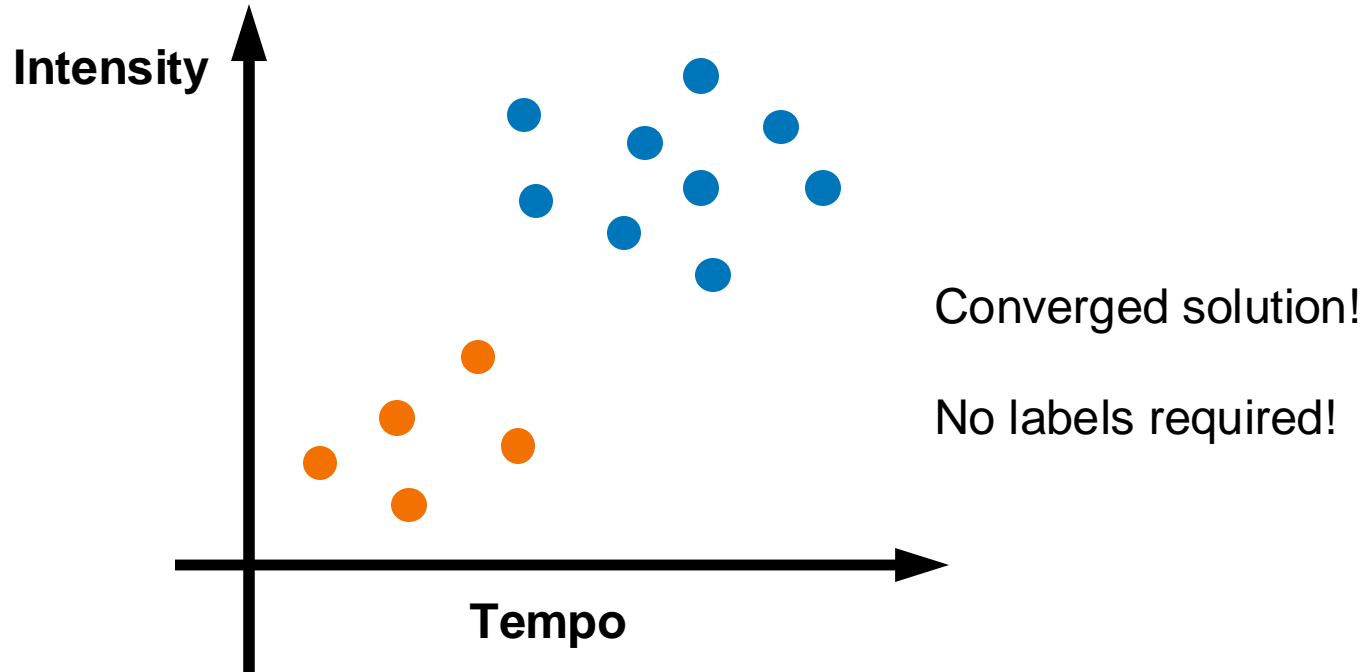
# K-means clustering

Step 3: update all cluster centers as the centroids



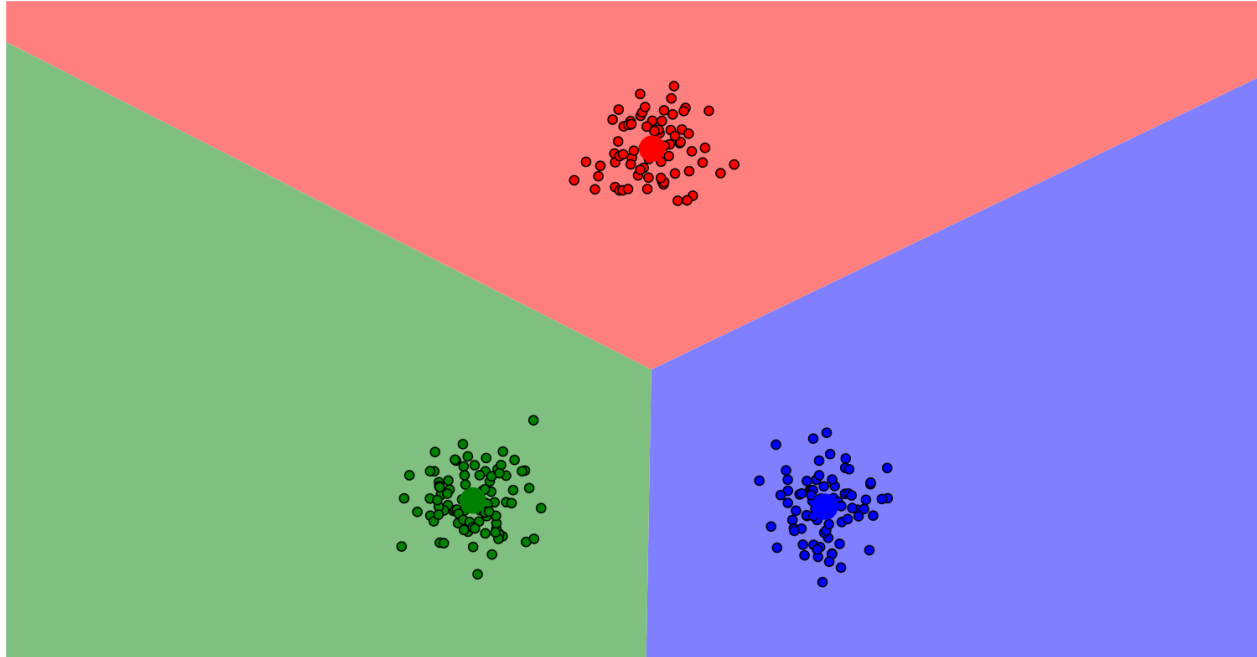
# K-means clustering

Repeat step 2 & 3 until convergence



# K-means clustering: A demo

<https://www.naftaliharris.com/blog/visualizing-k-means-clustering/>



# K-means algorithm

- Input:  $x_1, x_2, \dots, x_n, k$
- Step 1: select  $k$  cluster centers  $c_1, c_2, \dots, c_k$
- Step 2: for each point  $x_i$ , assign it to the closest center in Euclidean distance:

$$y(x_i) = \operatorname{argmin}_j \|x_i - c_j\|$$

- Step 3: update all cluster centers as the centroids:

$$c_j = \frac{\sum_{x:y(x)=j} x}{\sum_{x:y(x)=j} 1}$$

- Repeat Step 2 and 3 until 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 to pick starting cluster centers?
- How many clusters should we use?

# The optimization problem of k-means

- What is k-means trying to optimize?

$$\min_{c,y} \sum_{i=1}^n \|x_i - c_{y(x_i)}\|^2$$



# Questions on k-means

## Will k-means stop (converge)? Yes

Given a fixed dataset and a fixed number of clusters, there are only a **finite number of ways** to assign data points to clusters.

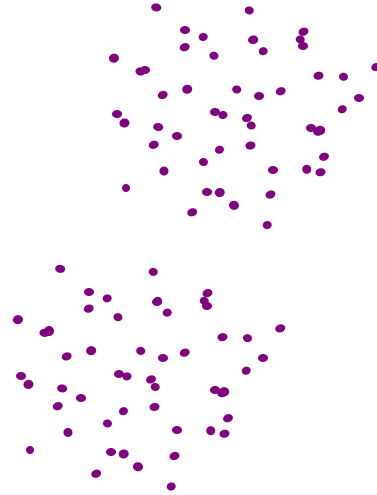
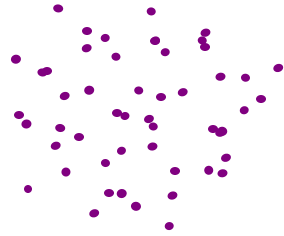
Each iteration consists of:

- Assignment Step: Assign each data point to the closest centroid.
- Update Step: Recompute centroids as the mean of assigned points.

These steps **always reduce or keep the same** the objective function (sum of squared distance), ensuring termination.

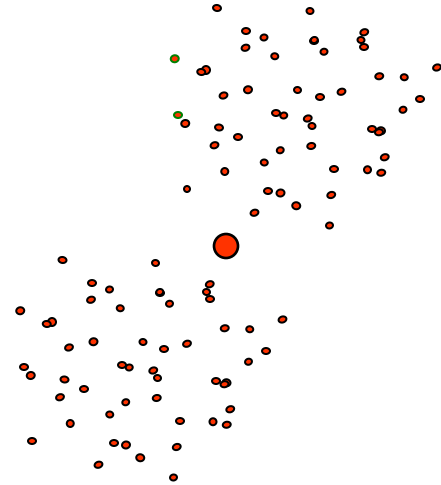
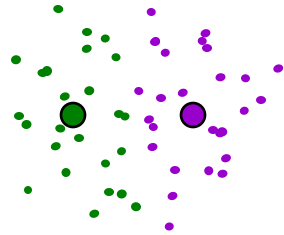
# Questions on k-means

Will it find a global or local optimum? (sadly no guarantee)



# Questions on k-means

Will it find a global or local optimum? (sadly no guarantee)



# Questions on k-means

How many clusters should we use?

- Difficult problem
- Domain knowledge
- Elbow Method
  - Compute the within-cluster sum of squares (WCSS) for different values of  $k$ .
  - Plot WCSS vs.  $k$  and look for the **elbow point** where the reduction in WCSS slows down. The optimal  $k$  is typically at this elbow.

# Break & Quiz

**Q 2.1:** You have seven 2-dimensional points. You run 3-means on it, with initial clusters

$$C_1 = \{(2, 2), (4, 4), (6, 6)\}, C_2 = \{(0, 4), (4, 0)\}, C_3 = \{(5, 5), (9, 9)\}$$

Cluster centroids are updated to?

- A.  $C_1: (4, 4), C_2: (2, 2), C_3: (7, 7)$
- B.  $C_1: (6, 6), C_2: (4, 4), C_3: (9, 9)$
- C.  $C_1: (2, 2), C_2: (0, 0), C_3: (5, 5)$
- D.  $C_1: (2, 6), C_2: (0, 4), C_3: (5, 9)$

# Break & Quiz

**Q 2.1:** You have seven 2-dimensional points. You run 3-means on it, with initial clusters

$$C_1 = \{(2, 2), (4, 4), (6, 6)\}, C_2 = \{(0, 4), (4, 0)\}, C_3 = \{(5, 5), (9, 9)\}$$

Cluster centroids are updated to?

- **A.  $C_1: (4,4), C_2: (2,2), C_3: (7,7)$**
- B.  $C_1: (6,6), C_2: (4,4), C_3: (9,9)$
- C.  $C_1: (2,2), C_2: (0,0), C_3: (5,5)$
- D.  $C_1: (2,6), C_2: (0,4), C_3: (5,9)$

The average of points in C1 is (4,4).  
The average of points in C2 is (2,2).  
The average of points in C3 is (7,7).

# Break & Quiz

**Q 2.2:** We are running 3-means again. We have 3 centers,  $C_1$  (0,1),  $C_2$  (2,1),  $C_3$  (-1,2). Which cluster assignment is possible for the points (1,1) and (-1,1), respectively? Ties are broken arbitrarily:

(i)  $C_1, C_1$  (ii)  $C_2, C_3$  (iii)  $C_1, C_3$

- A. Only (i)
- B. Only (ii) and (iii)
- C. Only (i) and (iii)
- D. All of them

# Break & Quiz

**Q 2.2:** We are running 3-means again. We have 3 centers,  $C_1$  (0,1),  $C_2$ , (2,1),  $C_3$  (-1,2). Which cluster assignment is possible for the points (1,1) and (-1,1), respectively? Ties are broken arbitrarily:

(i)  $C_1, C_1$  (ii)  $C_2, C_3$  (iii)  $C_1, C_3$

- A. Only (i)
- B. Only (ii) and (iii)
- C. Only (i) and (iii)
- **D. All of them**

For the point (1,1): square-Euclidean-distance to  $C_1$  is 1, to  $C_2$  is 1, to  $C_3$  is 5  
So it can be assigned to  $C_1$  or  $C_2$

For the point (-1,1): square-Euclidean-distance to  $C_1$  is 1, to  $C_2$  is 9, to  $C_3$  is 1  
So it can be assigned to  $C_1$  or  $C_3$



# Break & Quiz

**Q 2.3:** If we run K-means clustering twice with random starting cluster centers, are we guaranteed to get same clustering results? Does K-means always converge?

- A. Yes, Yes
- B. No, Yes
- C. Yes, No
- D. No, No

# Break & Quiz

**Q 2.3:** If we run K-means clustering twice with random starting cluster centers, are we guaranteed to get same clustering results? Does K-means always converge?

- A. Yes, Yes
- **B. No, Yes**
- C. Yes, No
- D. No, No

The clustering from k-means will depend on the initialization. Different initialization can lead to different outcomes.

K-means will always converge on a finite set of data points:

1. There are finite number of possible partitions of the points
2. The assignment and update steps of each iteration will only decrease the sum of the distances from points to their corresponding centers.
3. If it run forever without convergence, it will revisit the same partition, which is contradictory to item 2.



Thanks!