



CS 540 Introduction to Artificial Intelligence

Unsupervised Learning I

University of Wisconsin-Madison
Fall 2026 Sections 1 & 2

Announcements

- HW2 due on **Friday Sep 25th at 11:59 PM**
- HW3 will be out tomorrow.

- Class roadmap:

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

Machine Learning

Recap of Supervised vs Unsupervised Learning

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



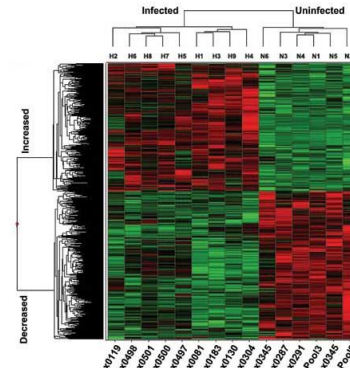
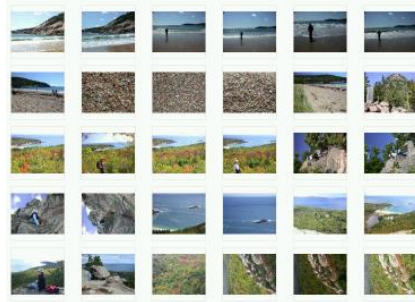
indoor



outdoor

Unsupervised learning:

-

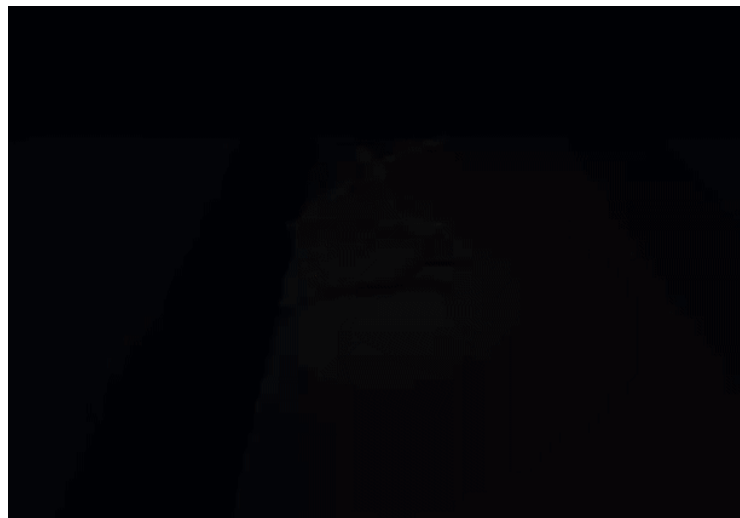


Mulvey and Gingold

Recap of Supervised/Unsupervised

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

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



DeepMind

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
- Clustering is popular & useful!



StyleGAN2 (Keras et al '20)

Clustering

- **Clustering:** group the data so that data in the same group are similar to each other than to data in other groups.
- Several types of clustering

Hierarchical

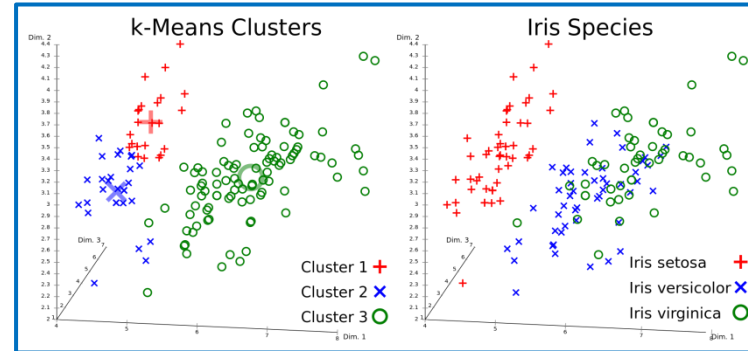
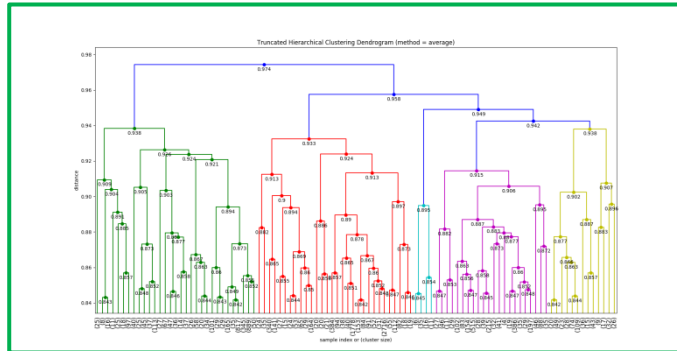
- Agglomerative
- Divisive

Partitional

- Center-based
- Graph-theoretic
- Spectral

Bayesian

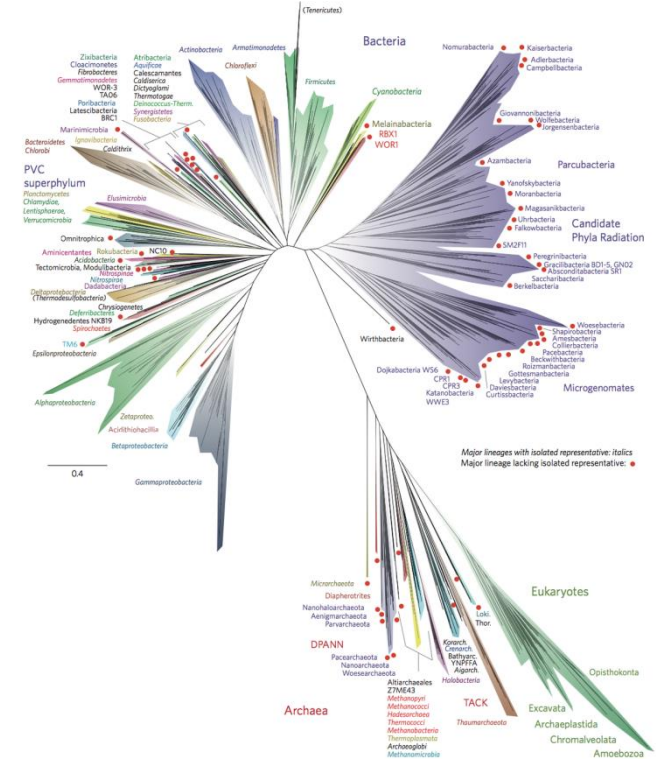
- Decision-based
- Nonparametric



Hierarchical Clustering

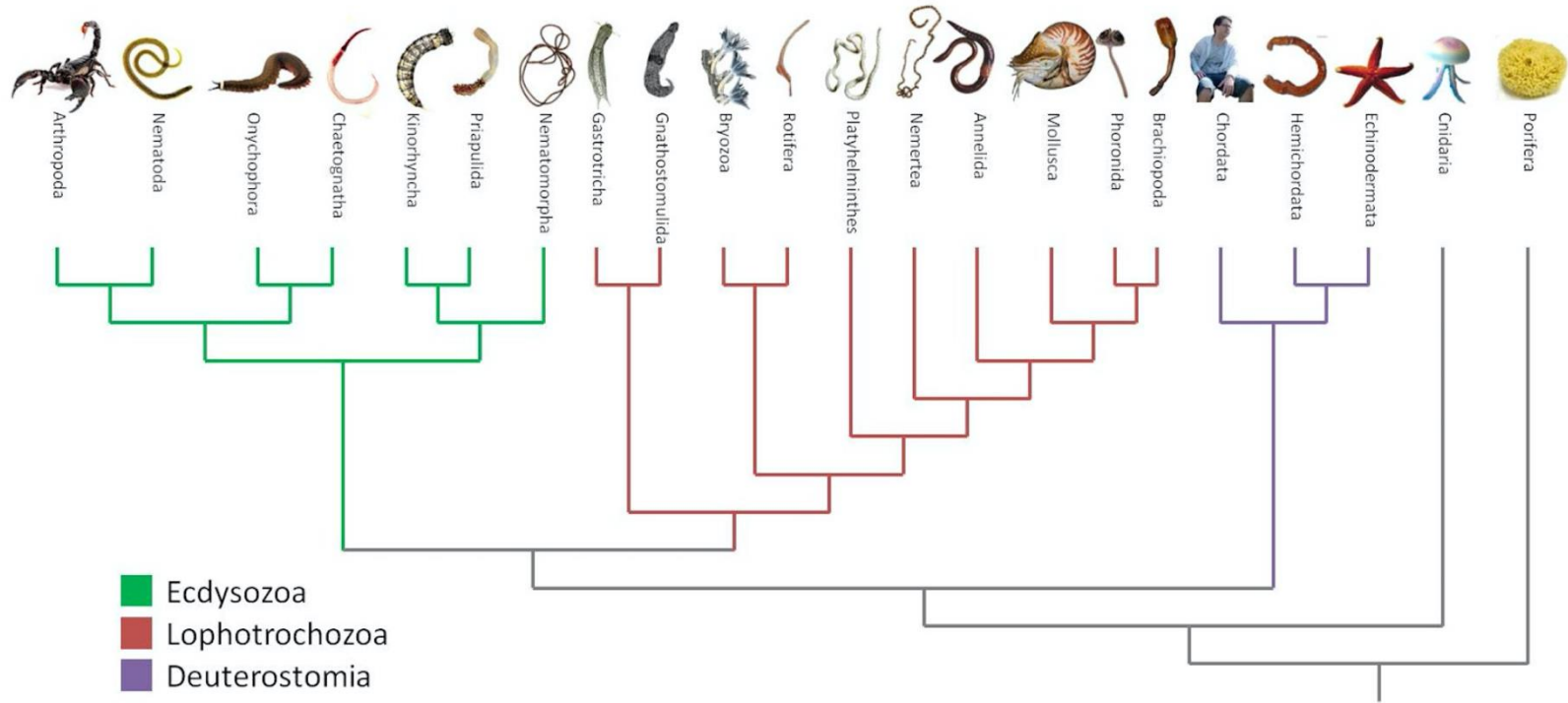
Basic idea: build a “hierarchy”

- Want: arrangements from specific to general
- One advantage: no need to specify the number of clusters.
- **Input:** points. **Output:** a hierarchy
 - E.g. A binary tree



Credit: Wikipedia

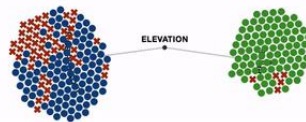
Hierarchical Clustering



Agglomerative vs Divisive

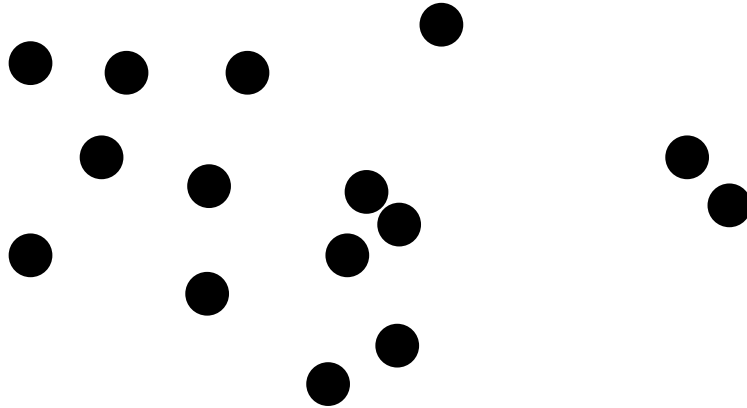
Two ways to go:

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



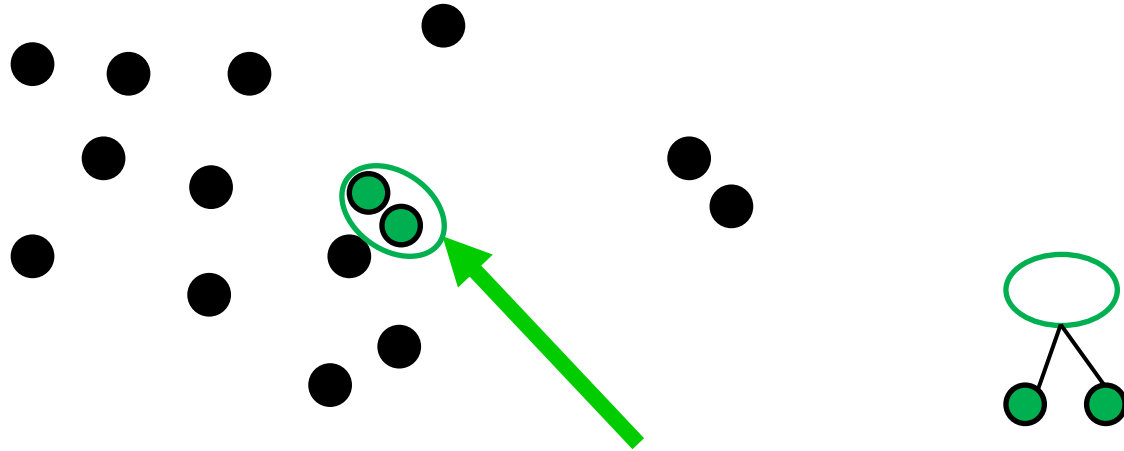
Agglomerative Clustering Example

Agglomerative. Start: every point is its own cluster



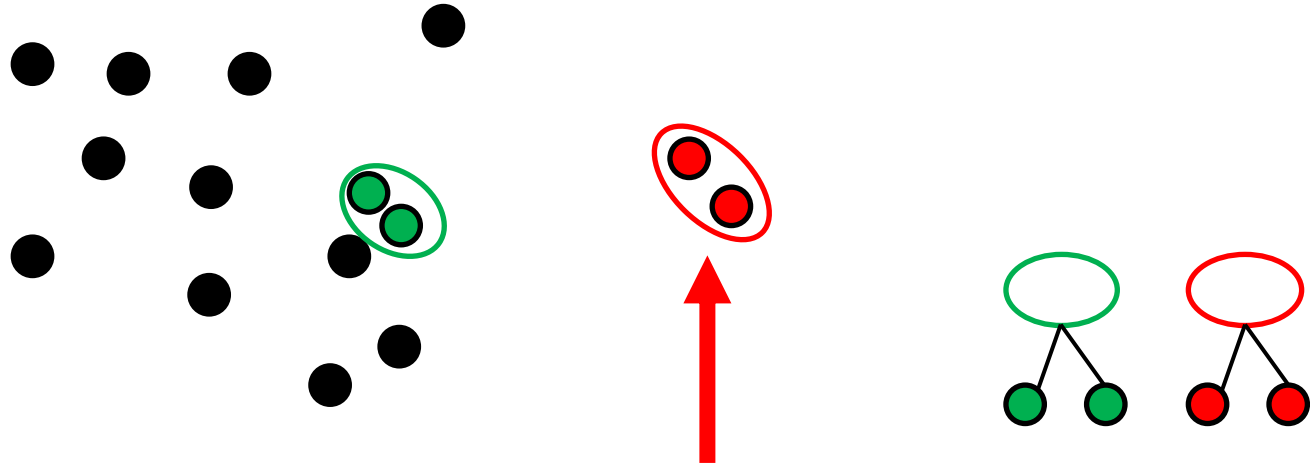
Agglomerative Clustering Example

Get pair of clusters that are closest and merge



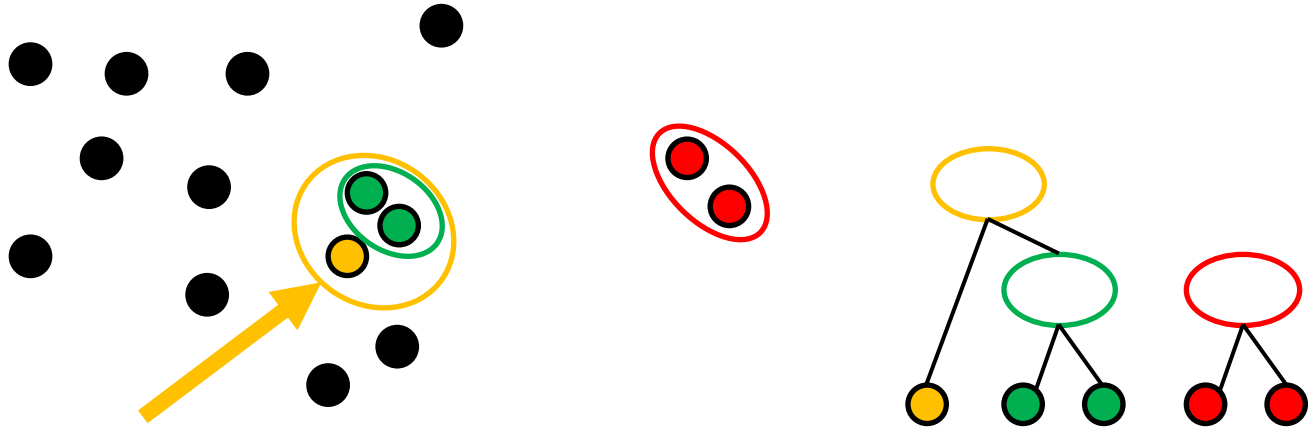
Agglomerative Clustering Example

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

$$d(A, B) = \max_{x_1 \in A, x_2 \in B} d(x_1, x_2)$$

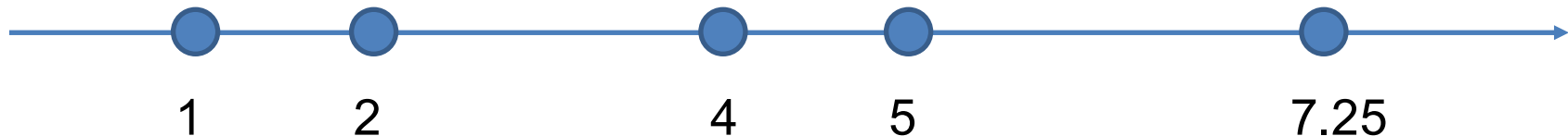
- Average-linkage

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

Single-linkage Example

We'll merge using single-linkage

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

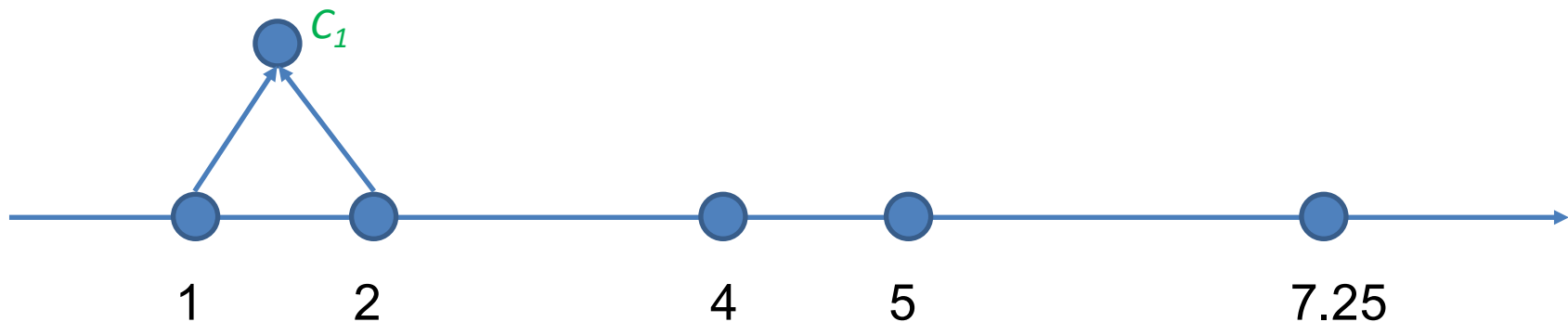


Single-linkage Example

We'll merge using single-linkage

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

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

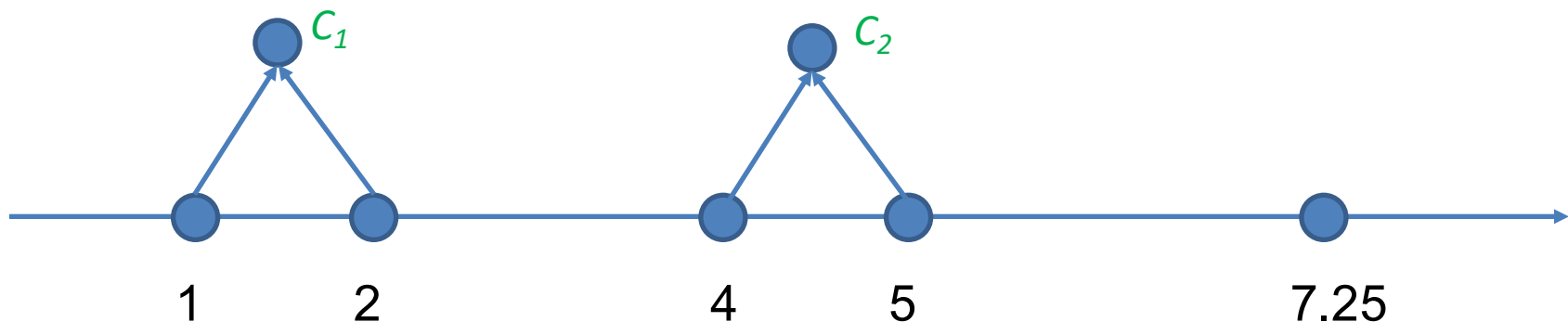


Single-linkage Example

Continue...

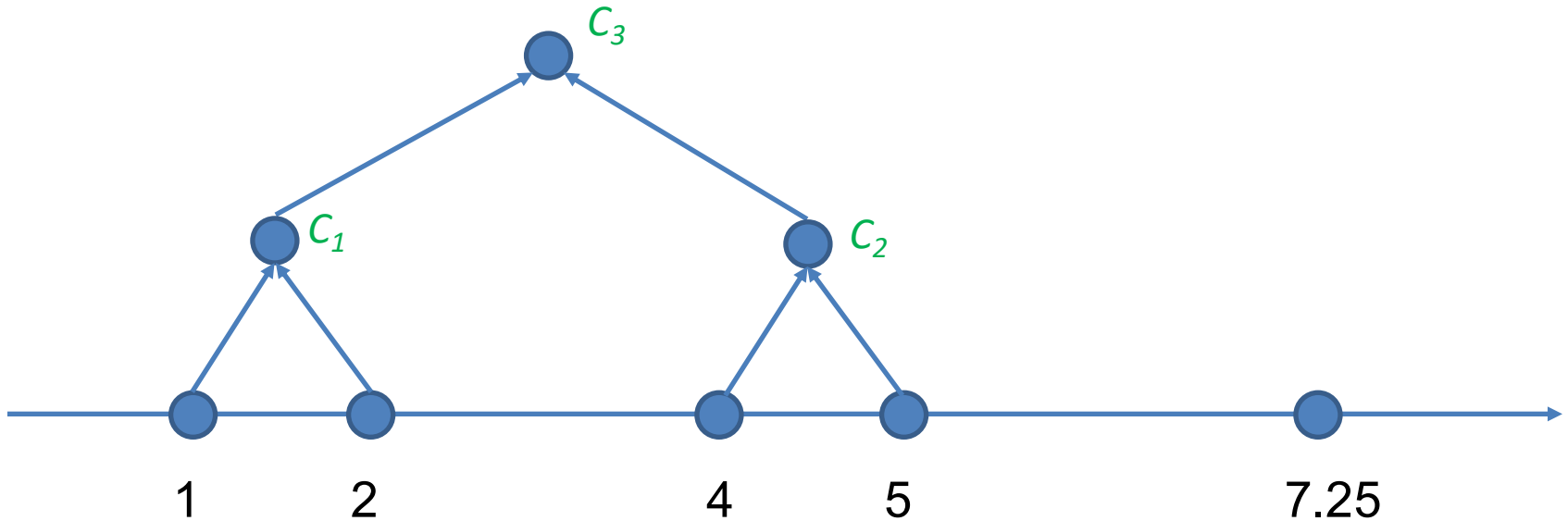
$$d(C_1, C_2) = d(2, 4) = 2$$

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

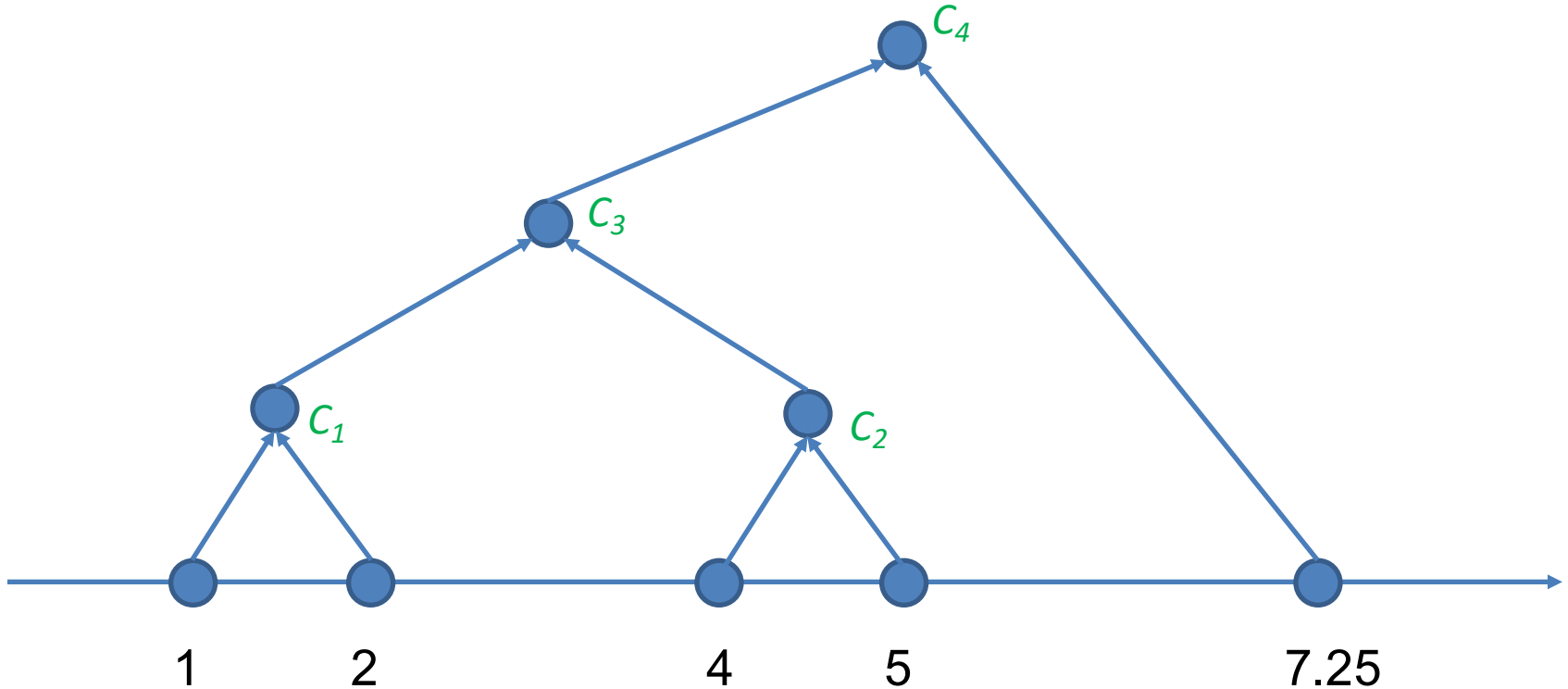


Single-linkage Example

Continue...



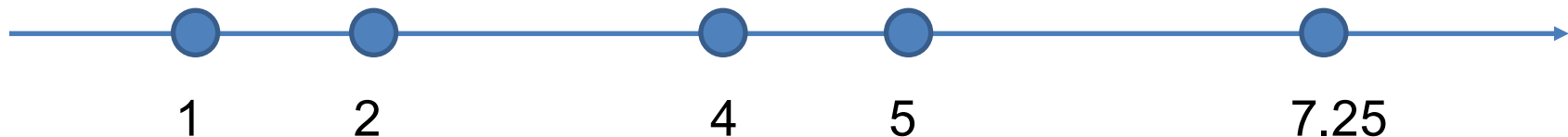
Single-linkage Example



Complete-linkage Example

We'll merge using complete-linkage

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

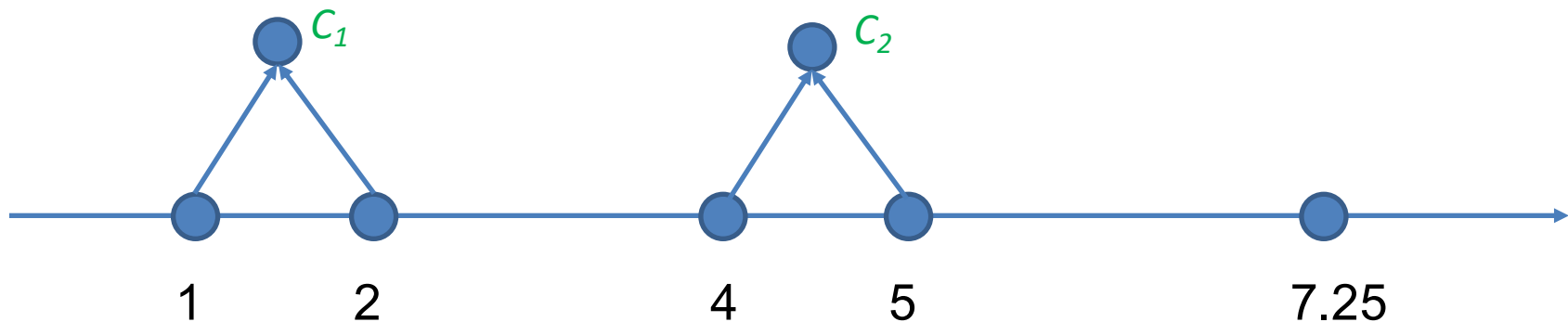


Complete-linkage Example

Beginning is the same...

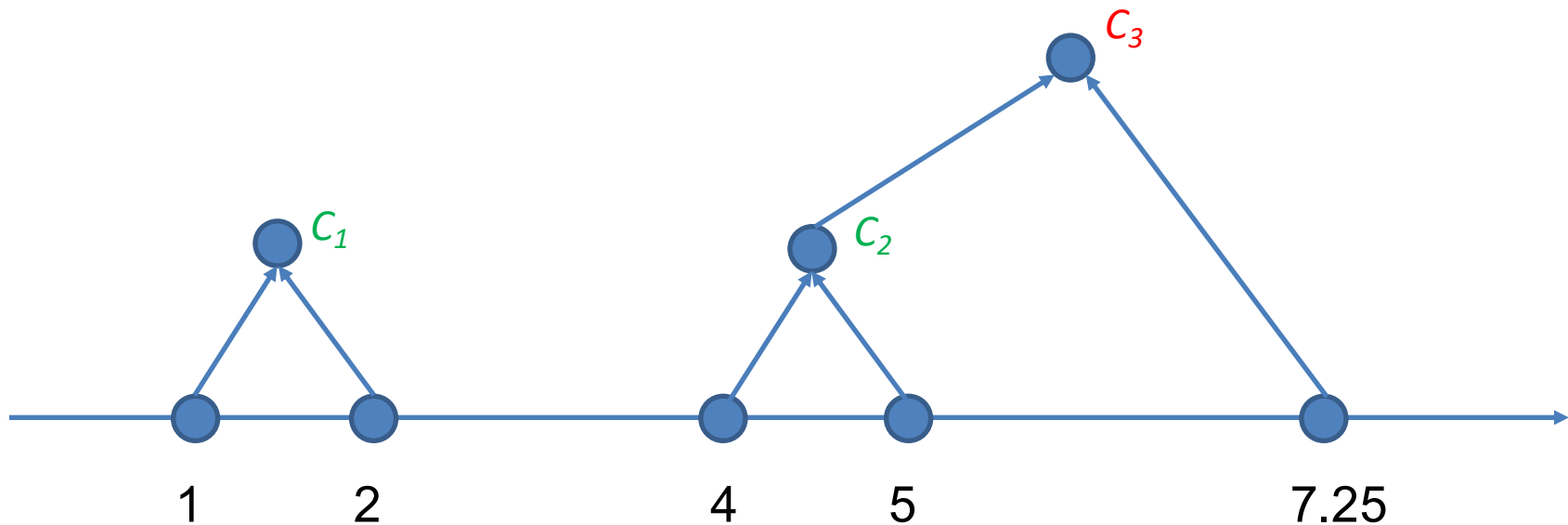
$$d(C_1, C_2) = d(1, 5) = 4$$

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

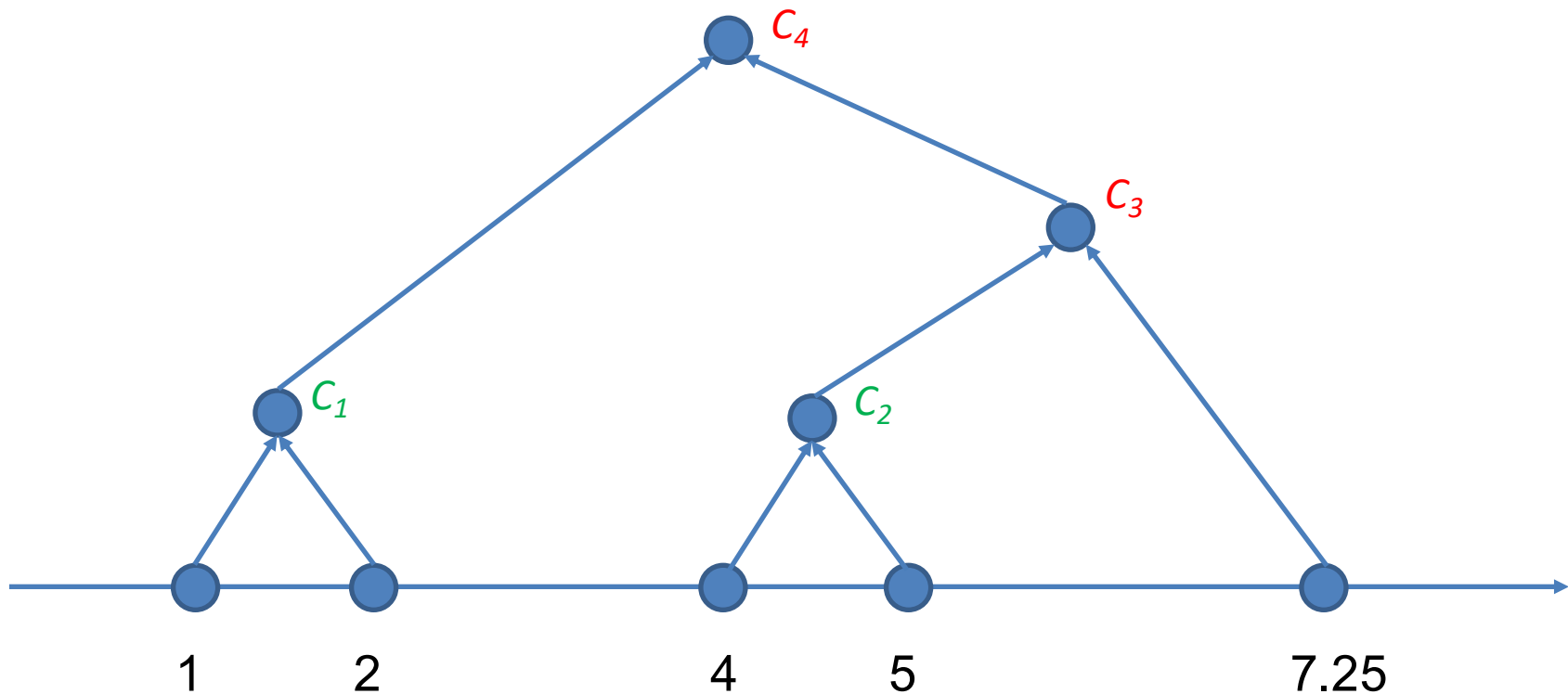


Complete-linkage Example

Now we diverge:



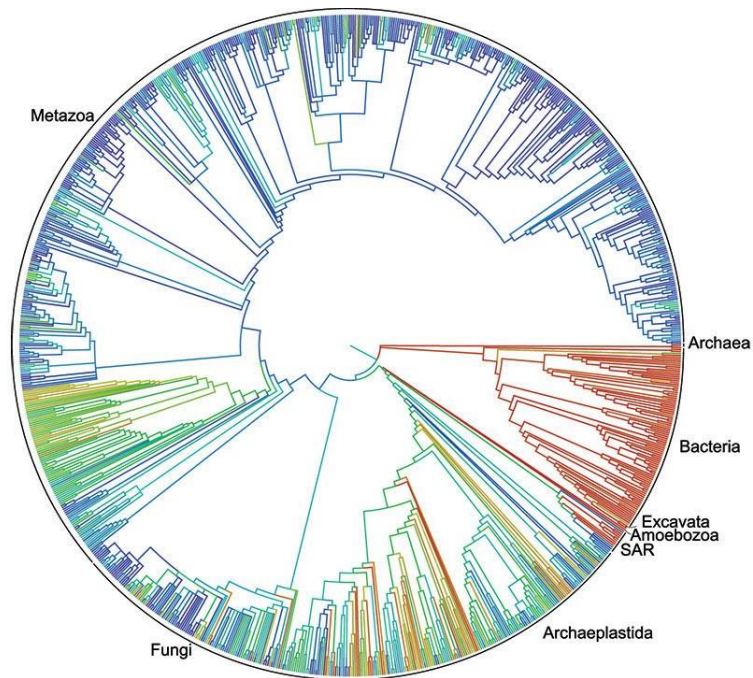
Complete-linkage Example



How to get clusters? i.e. when to stop the tree?

No simple answer:

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



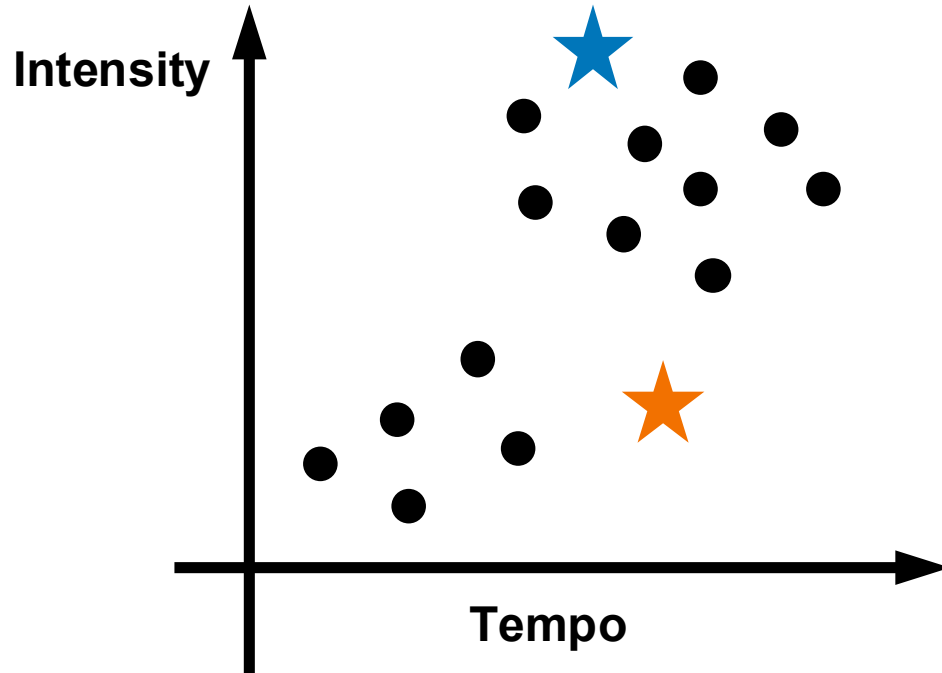
<http://opentreeoflife.org/>

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.

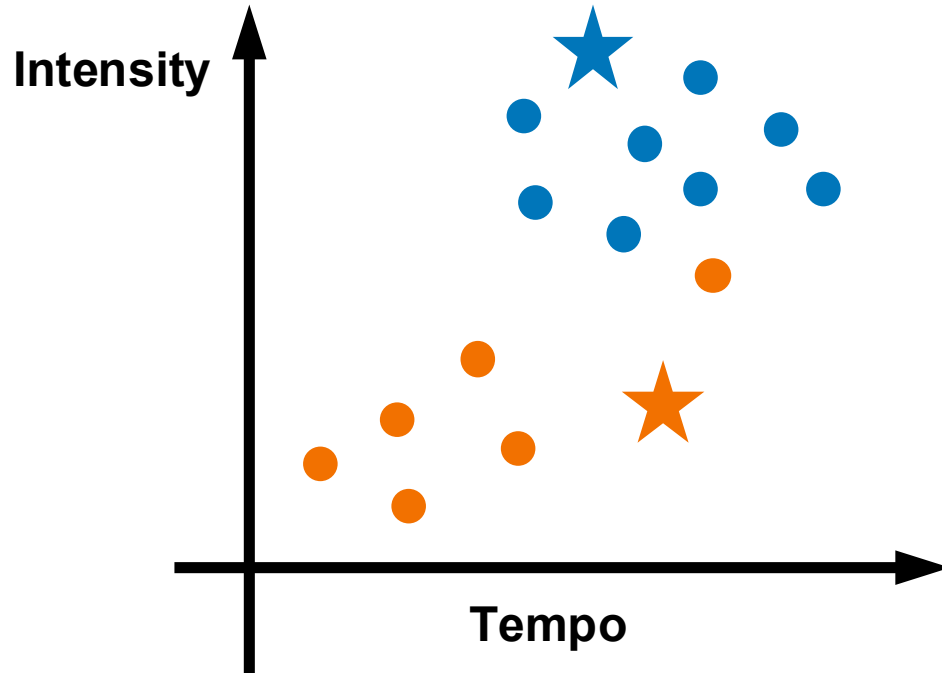
K-means clustering with $K=2$

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



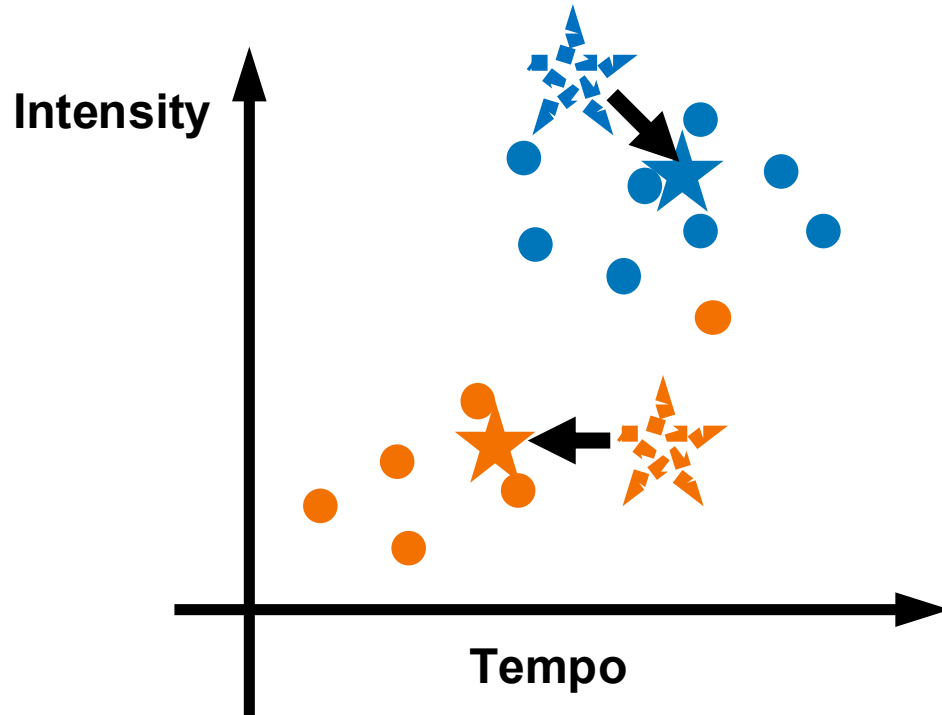
K-means clustering

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



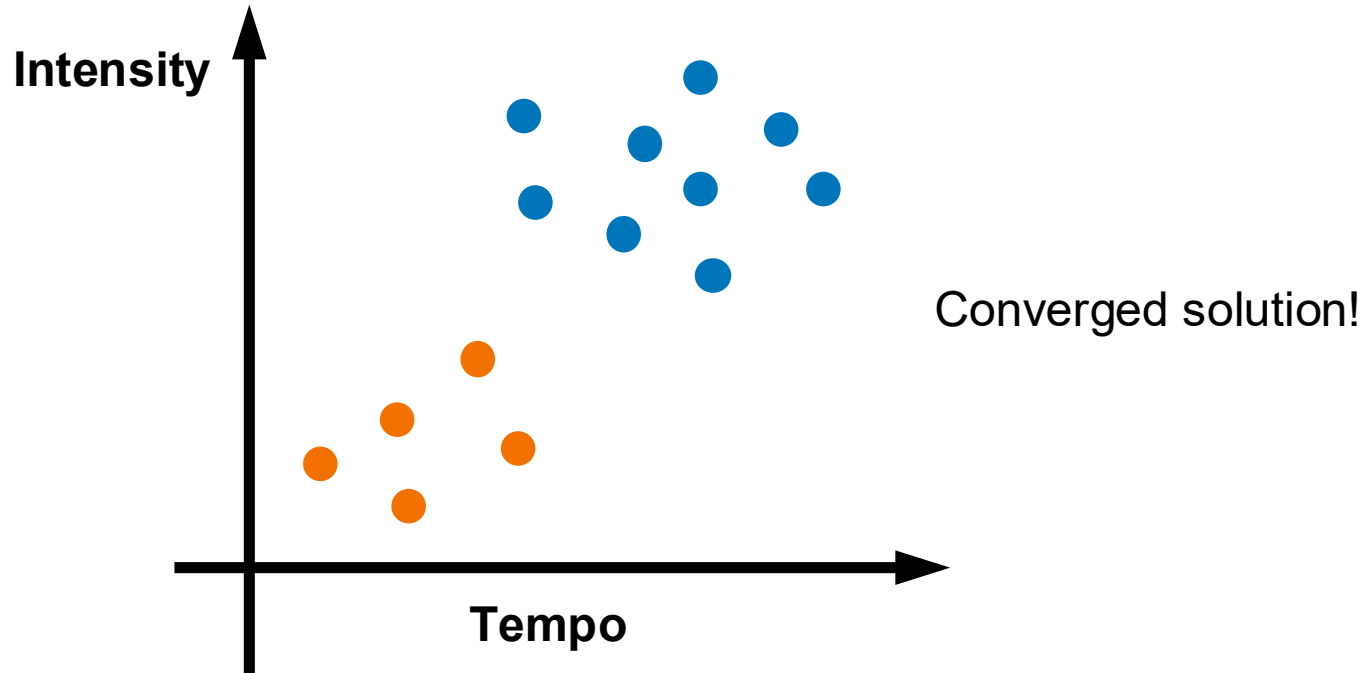
K-means clustering

Step 3: update all cluster centers as the centroids



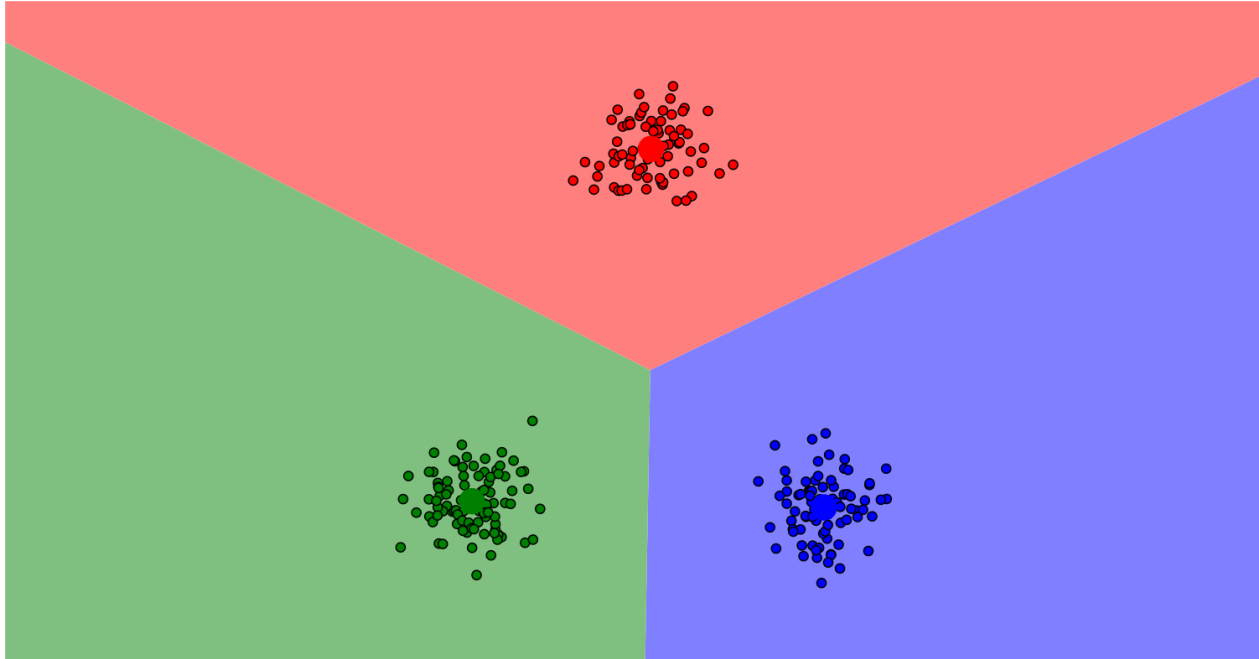
K-means clustering

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.

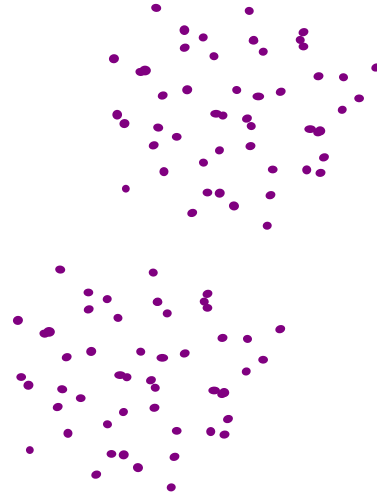
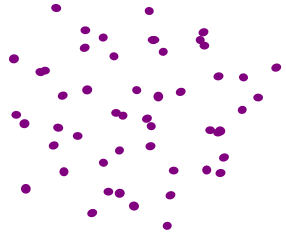
Each iteration consists of:

- 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.

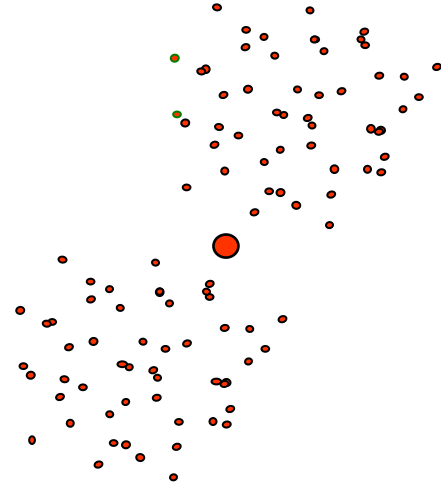
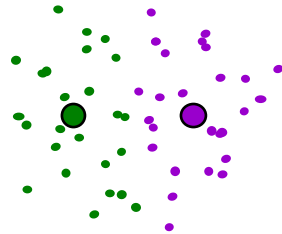
Questions on k-means

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

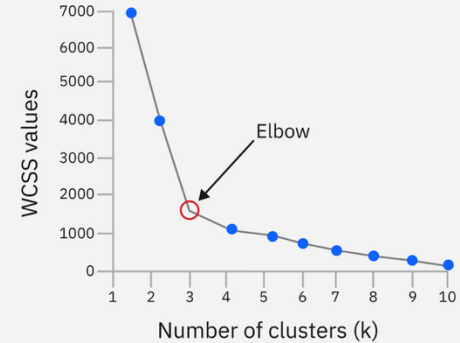
How to pick starting cluster centers?

- Randomly
- Kmeans++ (improving initialization)

Questions on k-means

How many clusters should we use?

- In general, no "right" answer.
- May require 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.



IBM



Thanks!