# Artificial Intelligence: Semi-supervised Learning

# Do you remember this slide? Naïve Bayes Example 3

$$P(H_1 \mid E_2) = \frac{P(H_1) \times P(E_2 \mid H_1)}{P(E_2)} = \frac{.2x.2}{.31} = .129$$

$$P(H_2 \mid E_2) = \frac{P(H_2) \times P(E_2 \mid H_2)}{P(E_2)} = \frac{.5x.3}{.31} = .484$$

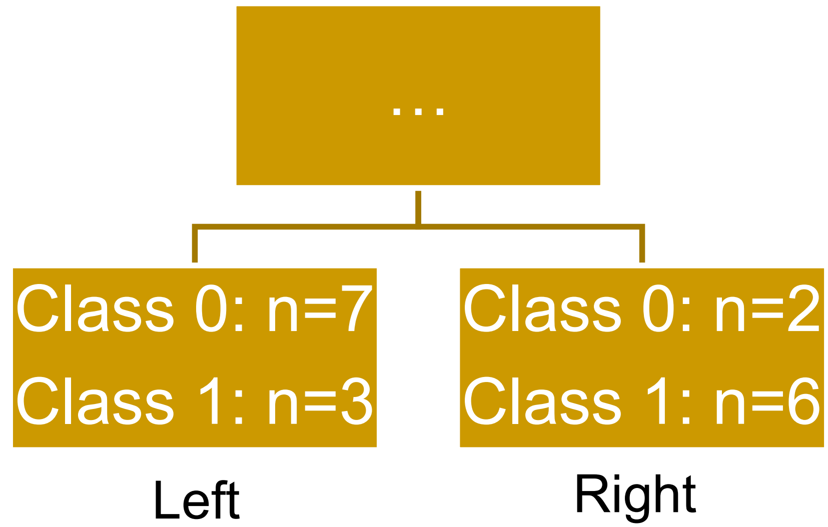
$$P(H_3 \mid E_2) = \frac{P(H_3) \times P(E_2 \mid H_3)}{P(E_2)} = \frac{.3x.4}{.31} = .387$$

® **$H_2$ is the most likely hypothesis, given the evidence**
**$P(H_2 \mid E_2)$** is the highest
Tomorrow the weather will be bad

$$H_{NB} = \underset{H_i}{\operatorname{argmax}} \frac{P(H_i) \times P(E \mid H_i)}{P(E)}$$

# How about this slide ? (Decision Tree classification error)

- What would be the classification error in these leaves ?




- In the left leaf, the prediction (predicted label) is class 0
- The predicted probability of class 1 is 0.3
- We are 30% confident that label should be class 1 (thus 70% confident of being class 0)
- In the right leaf, we are 75% confident of predicting class 1

# Prediction Probability

- Most machine learning classifiers not only returns (predicts) the class label but also a probability indicating the confidence

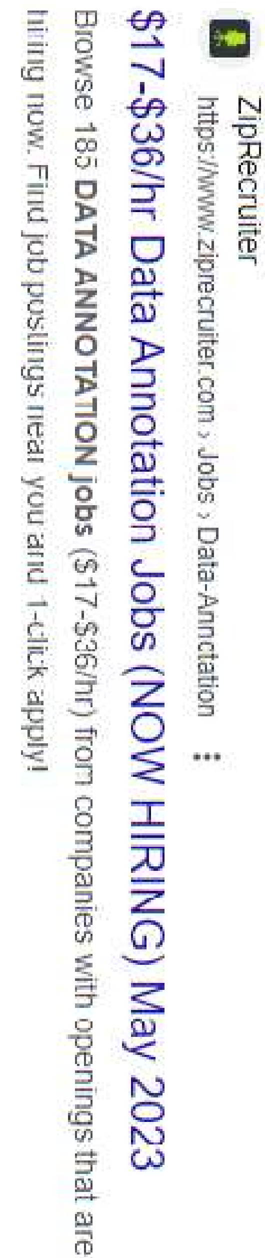
- Usually, the class prediction is based on this probability applying a cut point

- Cut point is by default 0.5 for binary classification, but is adjusted based on the importance of performance metrics (recall, precision, specificity, …)

Probability that this observation belongs to class 1 Ξ p (y=1)
p (y=1) > 0.5 => Classifier predicts class 1
p (y=1) <= 0.5 => Classifier predicts class 0

# Why semi-supervised learning ?

- Unlabeled data is cheap and available
- Labeled data can be hard to get
- Labelling (data annotation) can be very expensive
  - tedious task and time-consuming
  - may need experts' intervention
  - error-prone
- **Motivation**: Using both labeled and unlabeled data for learning



# Semi-supervised learning: Self training

- $n_l$ labelled samples (x, f(x))
- $n_n$ unlabeled samples (x,)
- Usually, $n_l << n_n$
- A classifier (learner) x -> f(x)

**Assumption**: Classifier's high confidence predictions are correct.

- Train on labelled samples
- Predict on unlabeled samples
- Add (x, $\hat{y}$)'s with high confidence to the labelled samples
- Repeat

# Self training semi-supervised

- A simple semi-supervised learning method.
- Iteration using existing classifiers

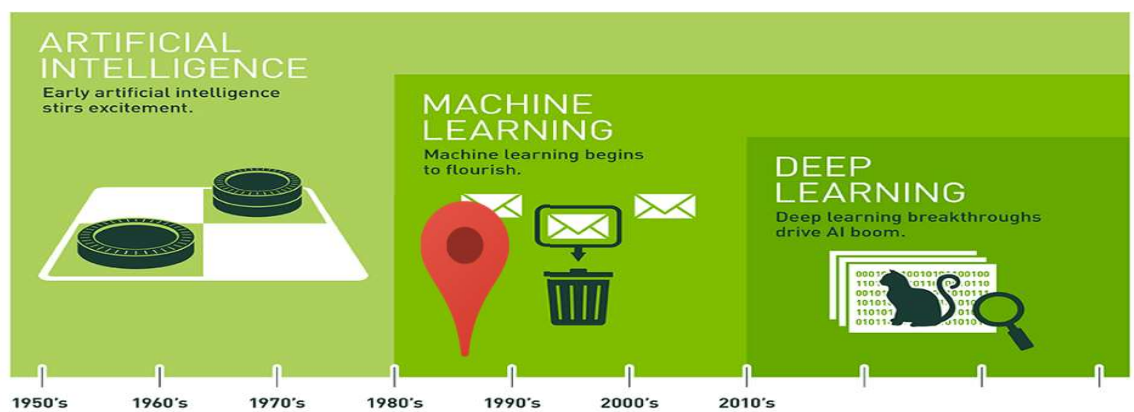
- Mistakes can be costly especially in the first iterations
  - Add only very confident samples to the labelled set
  - Add samples using the confidence as weight
  - Unable a sample if the confidence drops
  - …

# Artificial Intelligence: Unsupervised Learning

- Many slides are courtesy of Prof. René Witte.

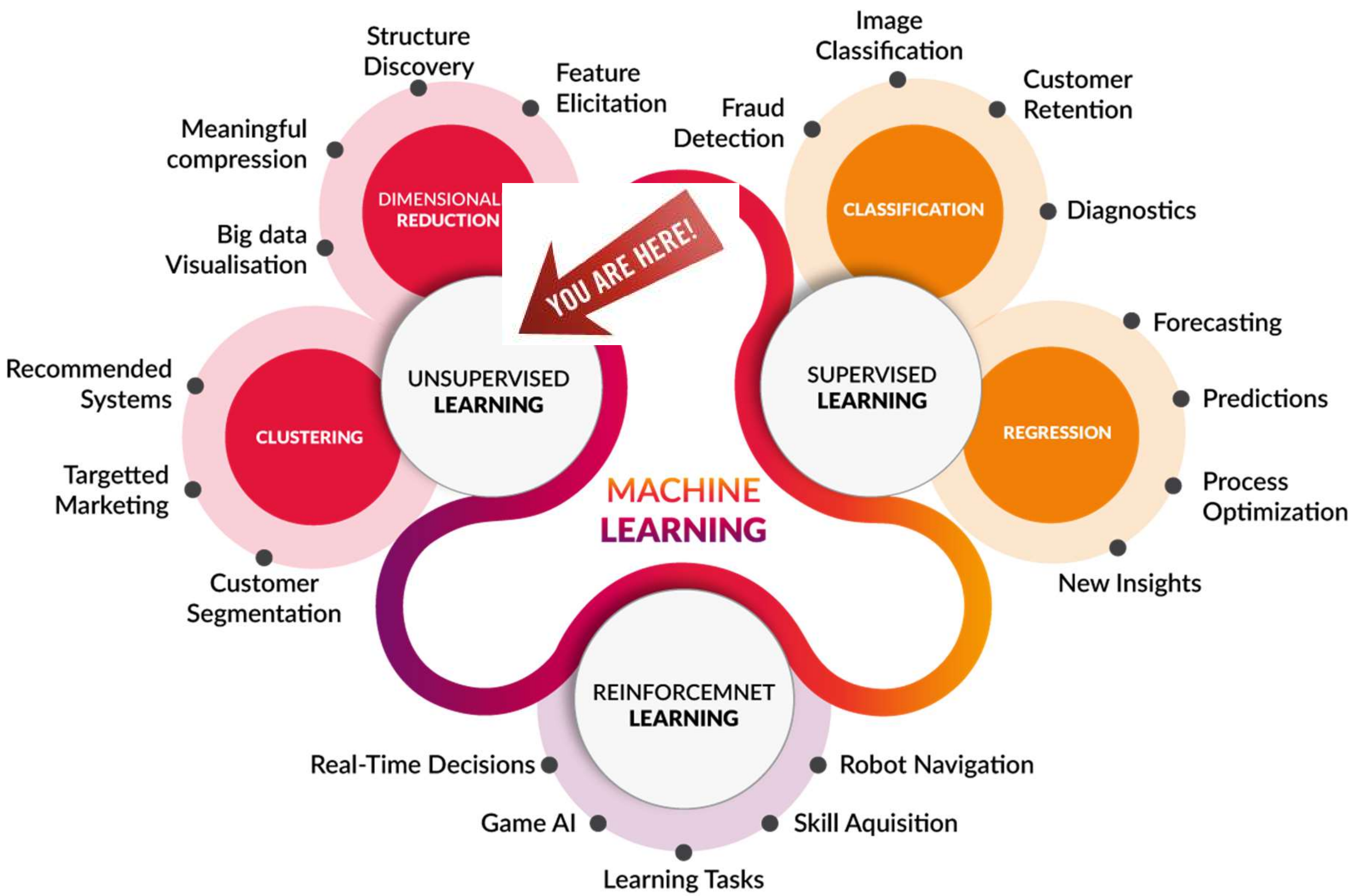
# Today


1. Unsupervised Learning
2. k-means Clustering
3. Hierarchical Clustering



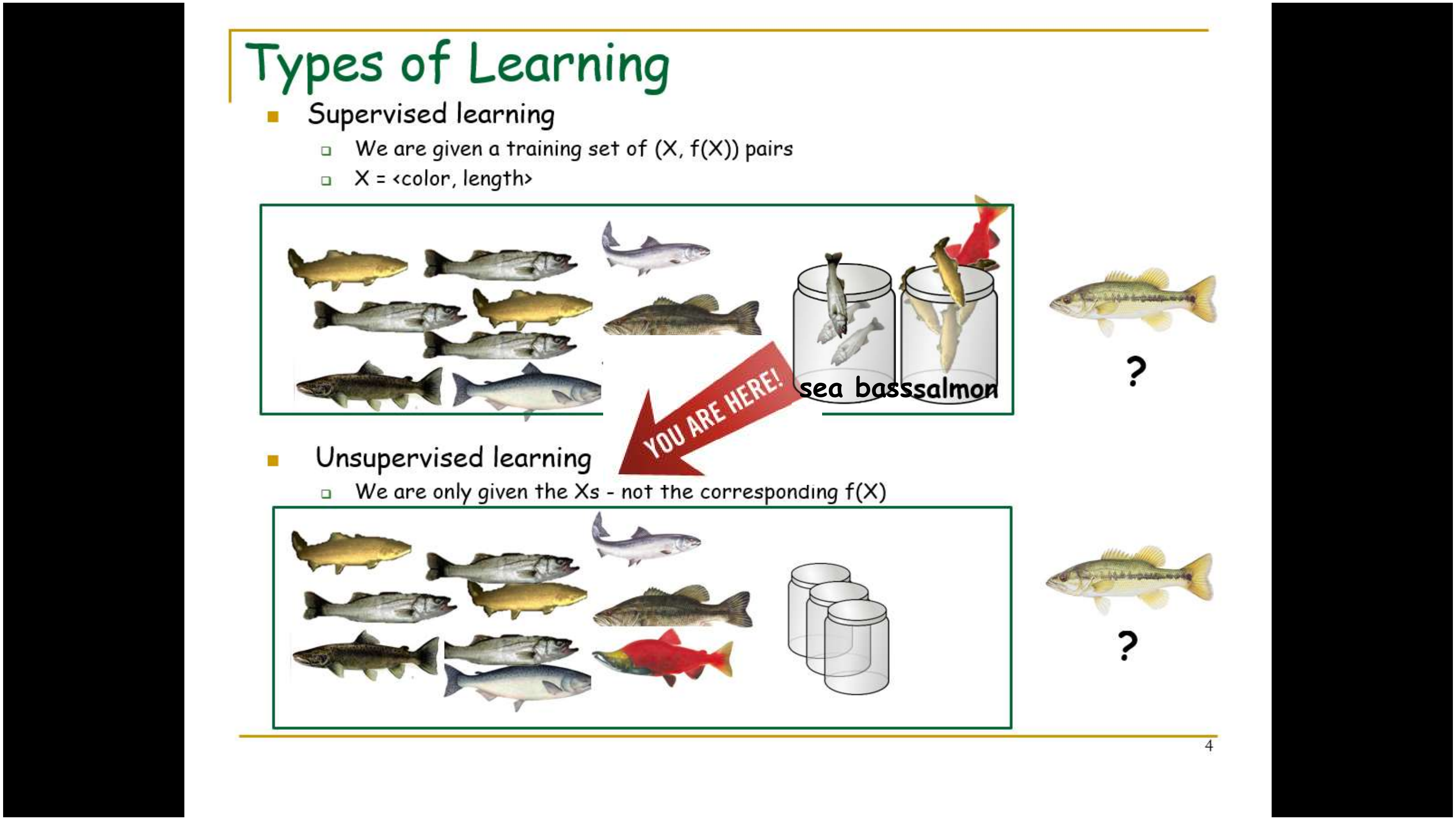

https://medium.com/machine-learning-for-humans/neural-networks-deep-learning-cdad8aeae49b

# Types of Machine Learning




http://www.cognub.com/index.php/cognitive-platform/

# Remember this slide?



# Unsupervised Learning

- Learn without labeled examples
  - i.e. X is given, but not f(X)

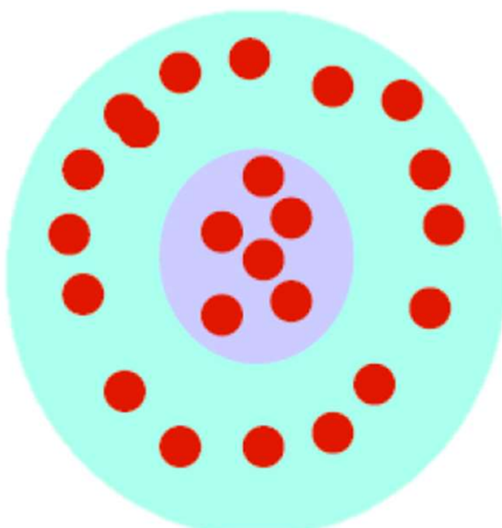
| | | | | |
|---|---|---|---|---|
| small nose | big teeth | small eyes | moustache | f(X) = ? |

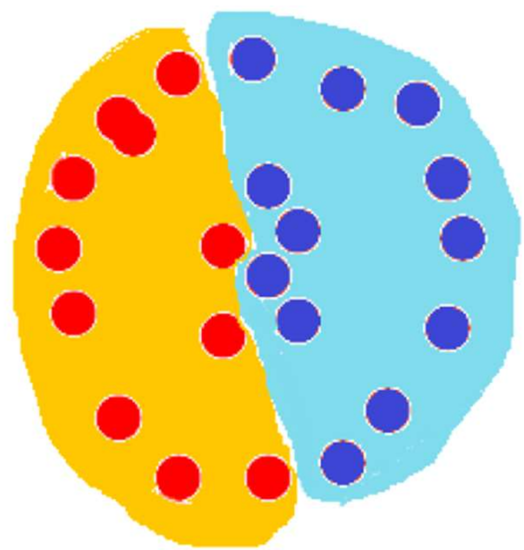
- Without a f(X), you can't really identify/label a test instance

- But you can:
  - Cluster/group the features of the test data into a number of groups
  - Discriminate between these groups without actually labeling them

# What is Clustering

- The organization of unlabeled data into similarity groups called clusters.
- A cluster is a collection of data items which are "similar" between them, and "dissimilar" to data items in other clusters.



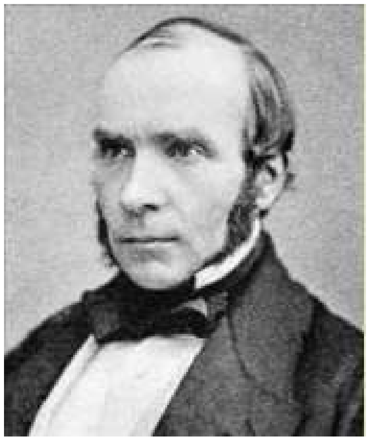


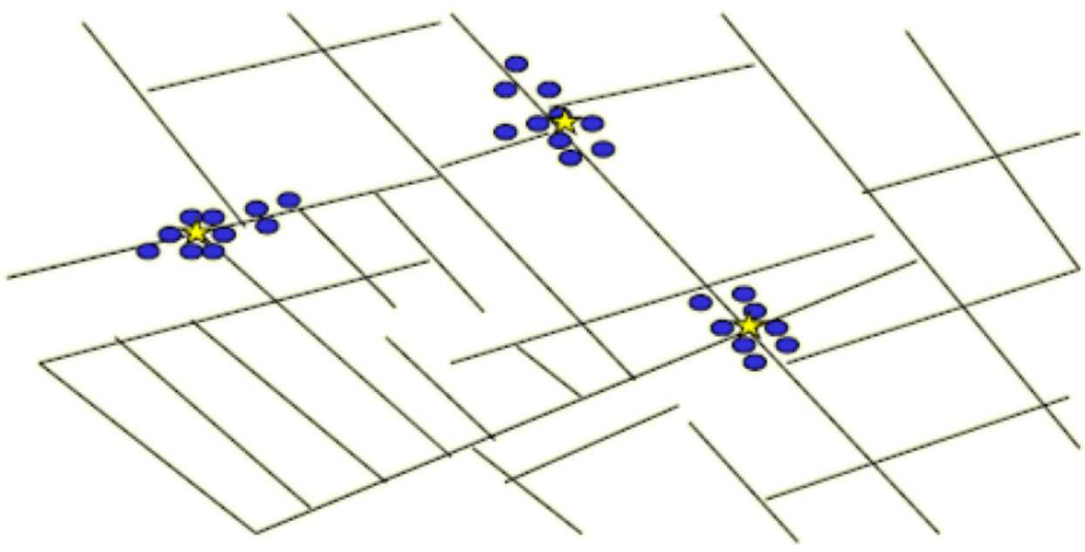
# Applications of Clustering

- Exploratory data analysis (EDA)
- Customer segmentation in marketing to identify similar groups of customers based on their purchase behavior
- Image segmentation in computer vision to group pixels with similar attributes for object recognition
- Document clustering in natural language processing (NLP)
- ...

# Historic Application of Clustering

- John Snow, a London physician plotted the location of cholera on a map during an outbreak in the 1850s.
- The locations indicated that cases were clustered arounds certain intersections where there were polluted wells – thus exposing both the problem and the solution.

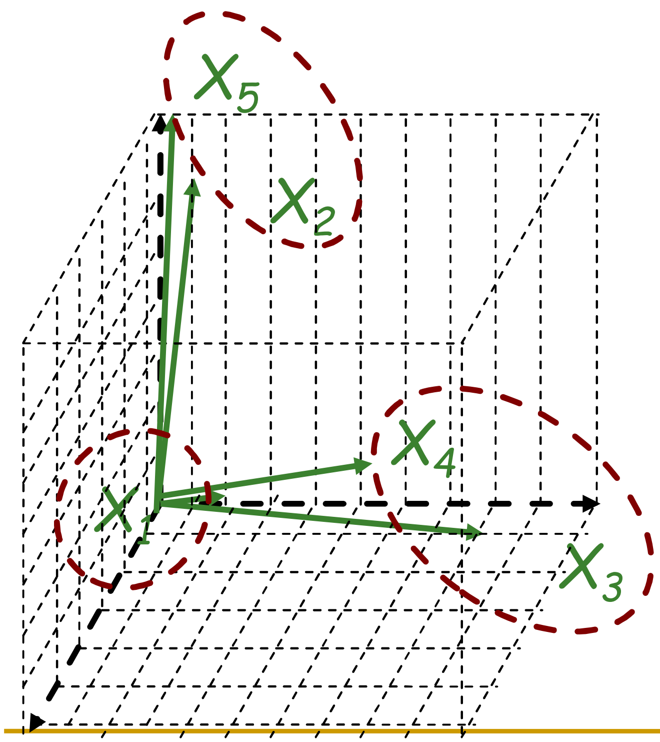




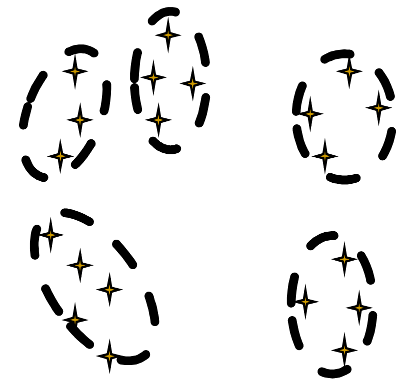
# Clustering

- Represent each instance as a vector $\langle a_1, a_2, a_3, \ldots, a_n \rangle$
- Each vector can be visually represented in an *n*-dimensional space

| | $a_1$ | $a_2$ | $a_3$ | Output |
|---|---|---|---|---|
| $X_1$ | 1 | 0 | 0 | ? |
| $X_2$ | 1 | 6 | 0 | ? |
| $X_3$ | 8 | 0 | 1 | ? |
| $X_4$ | 6 | 1 | 0 | ? |
| $X_5$ | 1 | 7 | 1 | ? |

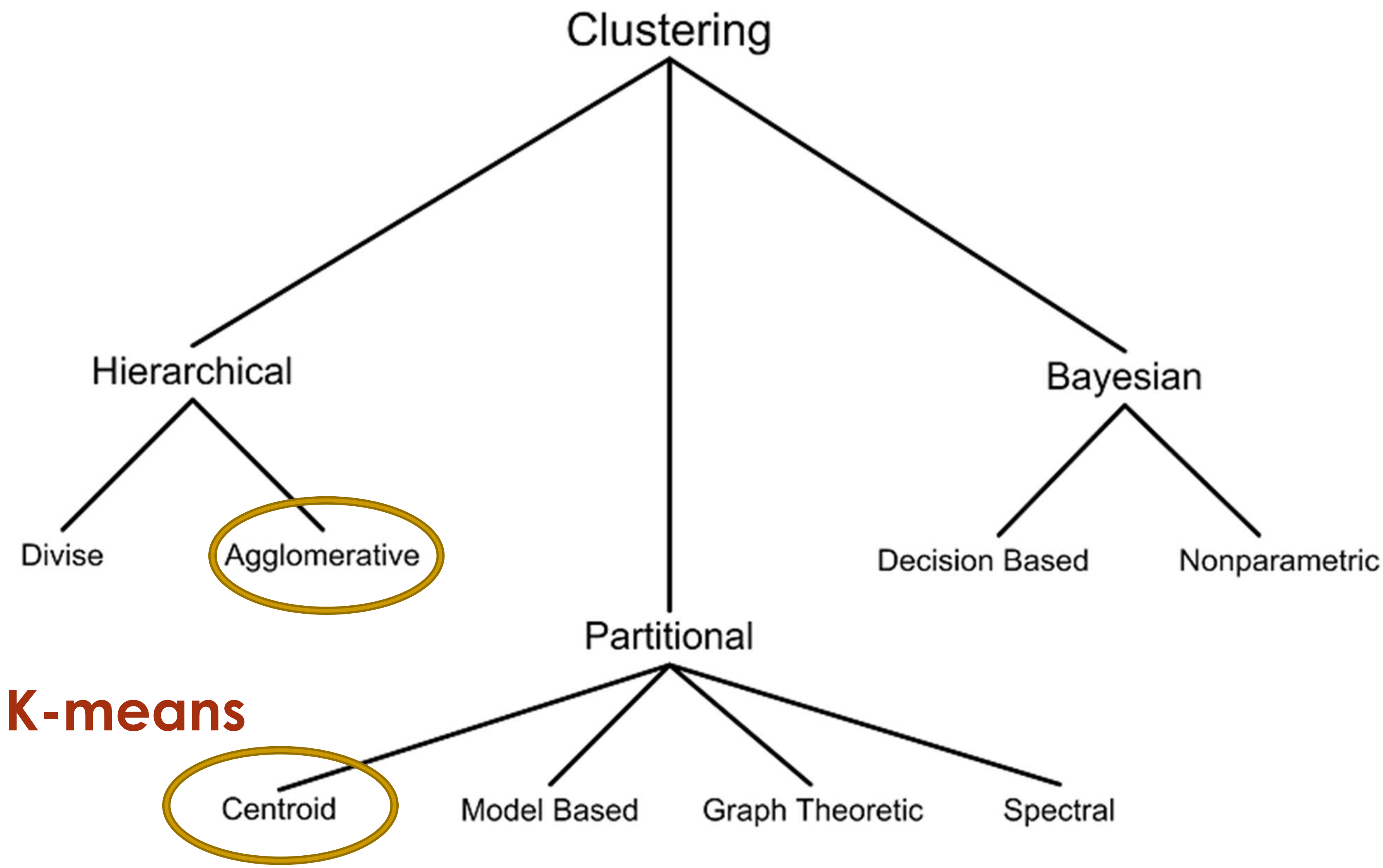


# Clustering



- Clustering algorithm
  - Represent test instances on a n dimensional space
  - Partition them into regions of high density
    - How?  … many algorithms (ex. k-means)
  - Compute the centroïd of each region as the average of data points in the cluster

# Clustering Techniques



# k-means Clustering

- User selects how many clusters they want… (the value of k)

1. Place k points into the space (ex. at random). These points represent initial group centroïds.

2. Assign each data point $x_n$ to the nearest centroïd.

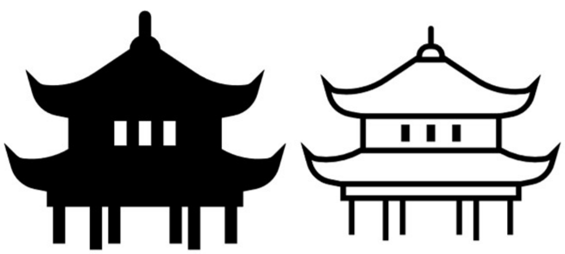
3. When all data points have been assigned, recalculate the positions of the K centroïds as the average of the cluster (new centroïds)

4. Repeat Steps 2 and 3 until none of the data instances change group.

# Nearest centroïd !!
## But how to define similarity (distance) ?

1. For two objects $x^1$, $x^2$ distance $d(x^1, x^2)$ is a numerical representation of their dissimilarity.

$d(x^1, x^2) = 0.1$



$d(x^1, x^2) = 0.5$

$d(x^1, x^2) = 0.8$

2. Several ways to define it in Machine Learning.
   - Euclidean distance
   - Manhattan distance

# Euclidean Distance

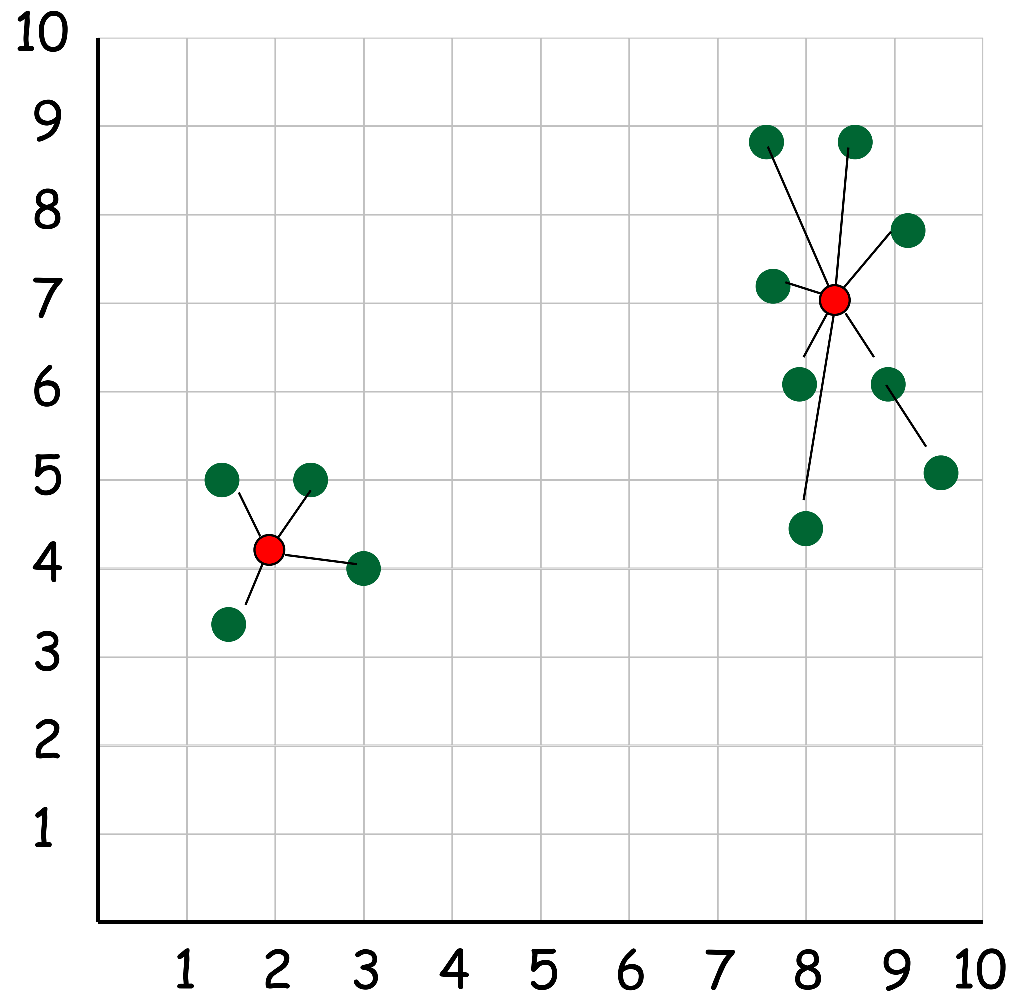
- To find the nearest centroïd…
- a possible metric is the Euclidean distance
- distance between 2 pts

  $p = (p_1, p_2, \ldots., p_n)$

  $q = (q_1, q_2, \ldots., q_n)$

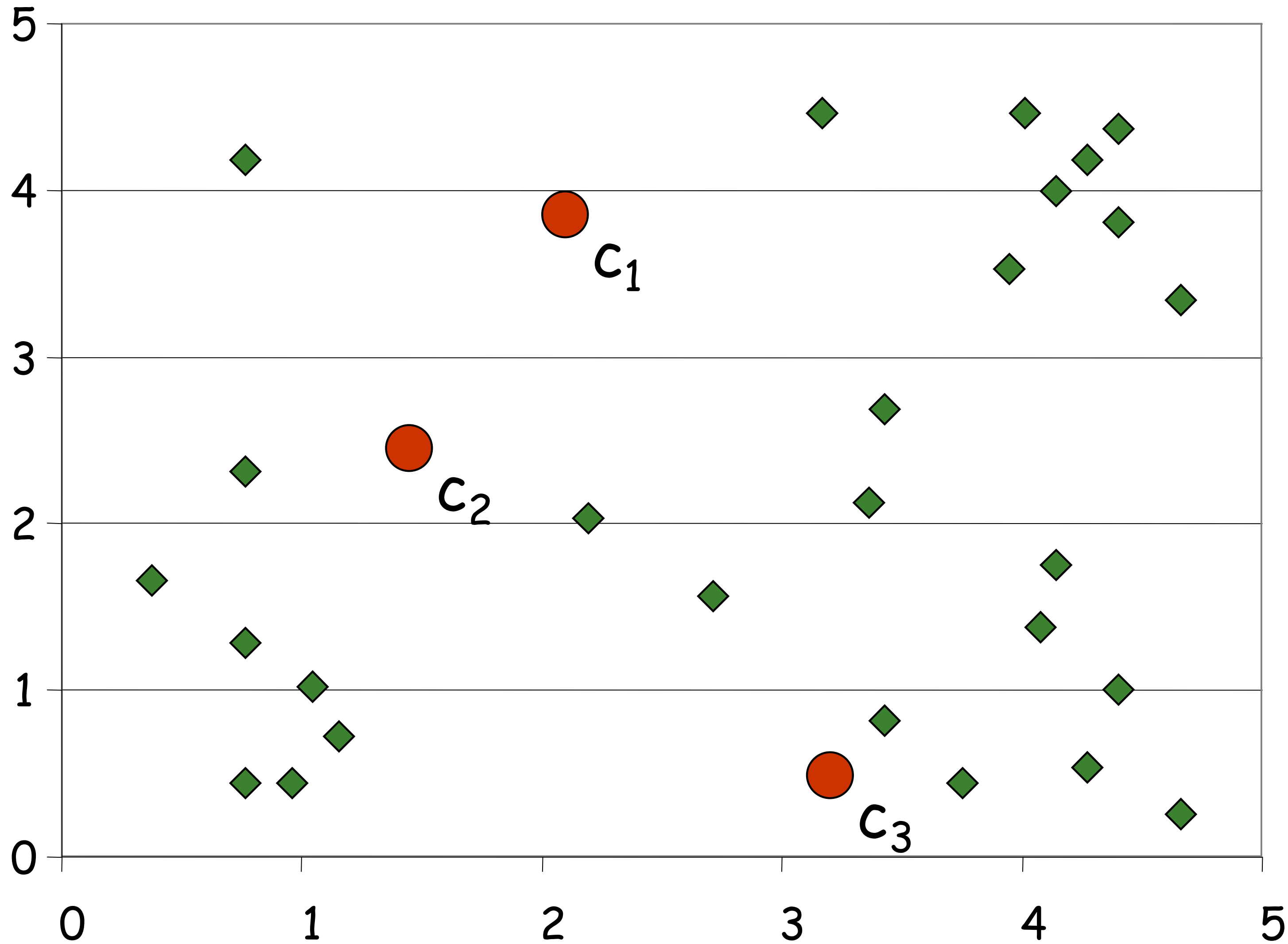
$$d = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

- where to assign a data point x?
- For all k clusters, chose the one where x has the smallest distance



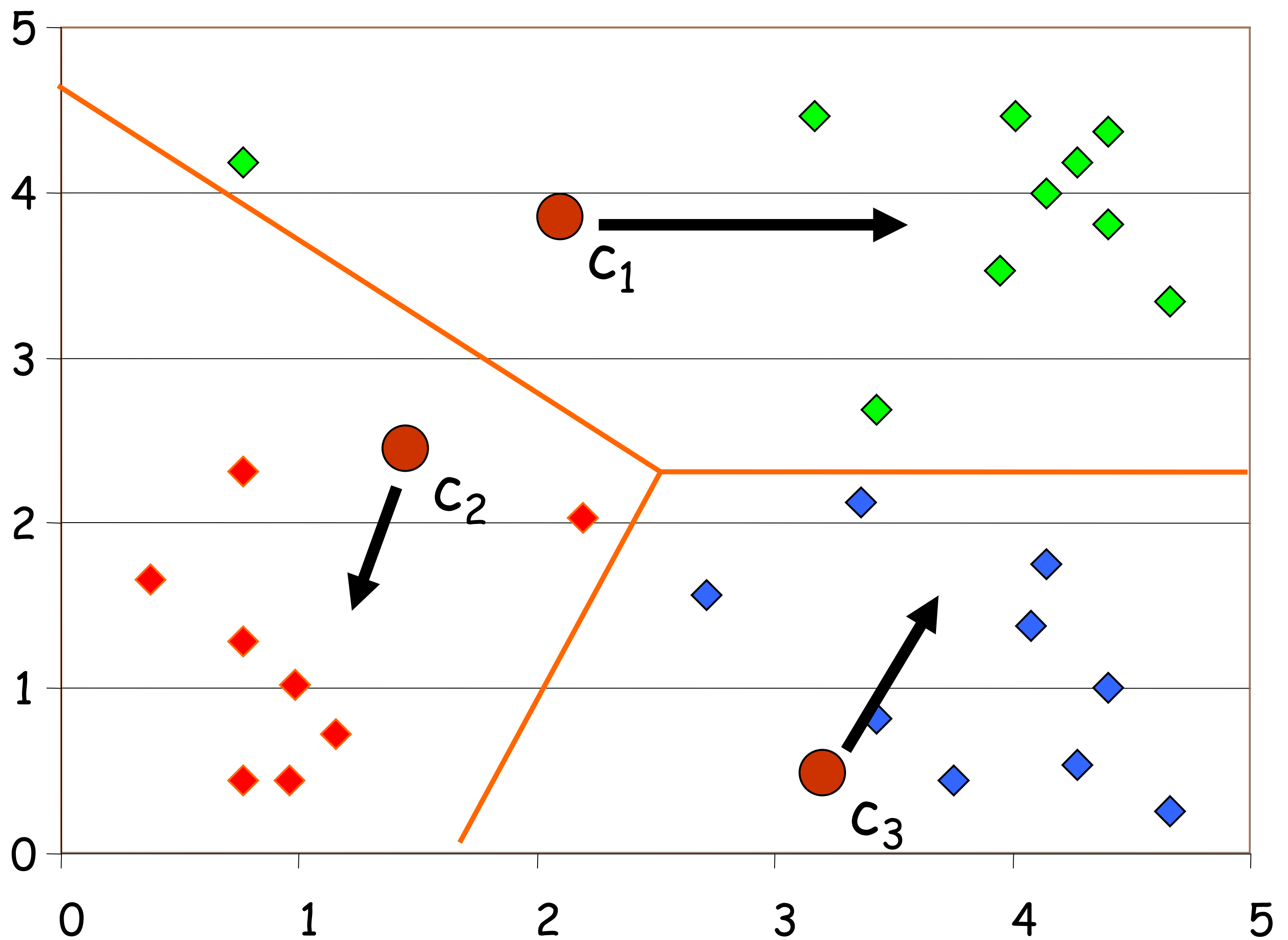
# Example (in 2-D... i.e. 2 features)

initial 3 centroïds (ex. at random)



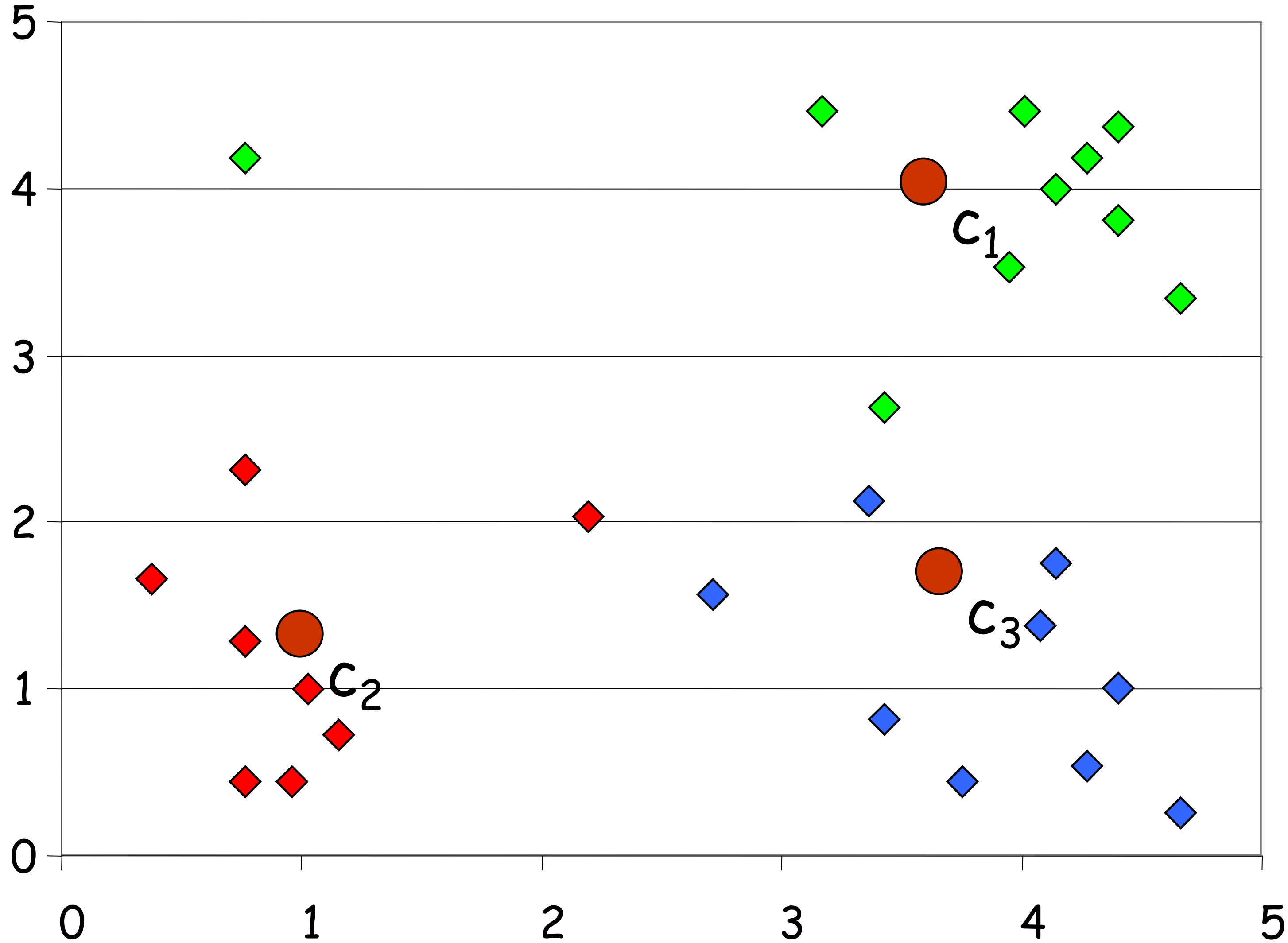
# Example

partition data points to closest centroïd



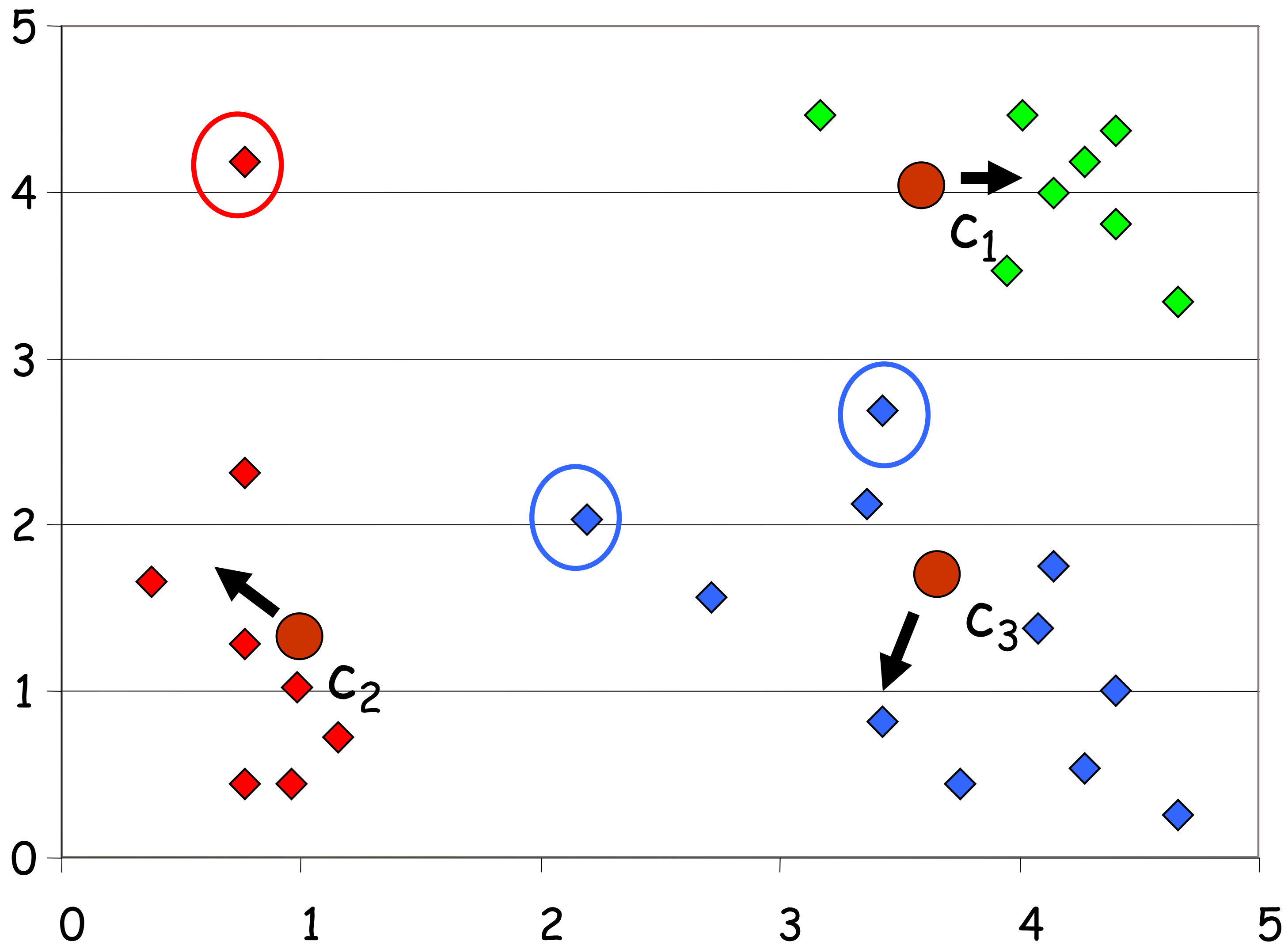
# Example

re-compute new centroïds

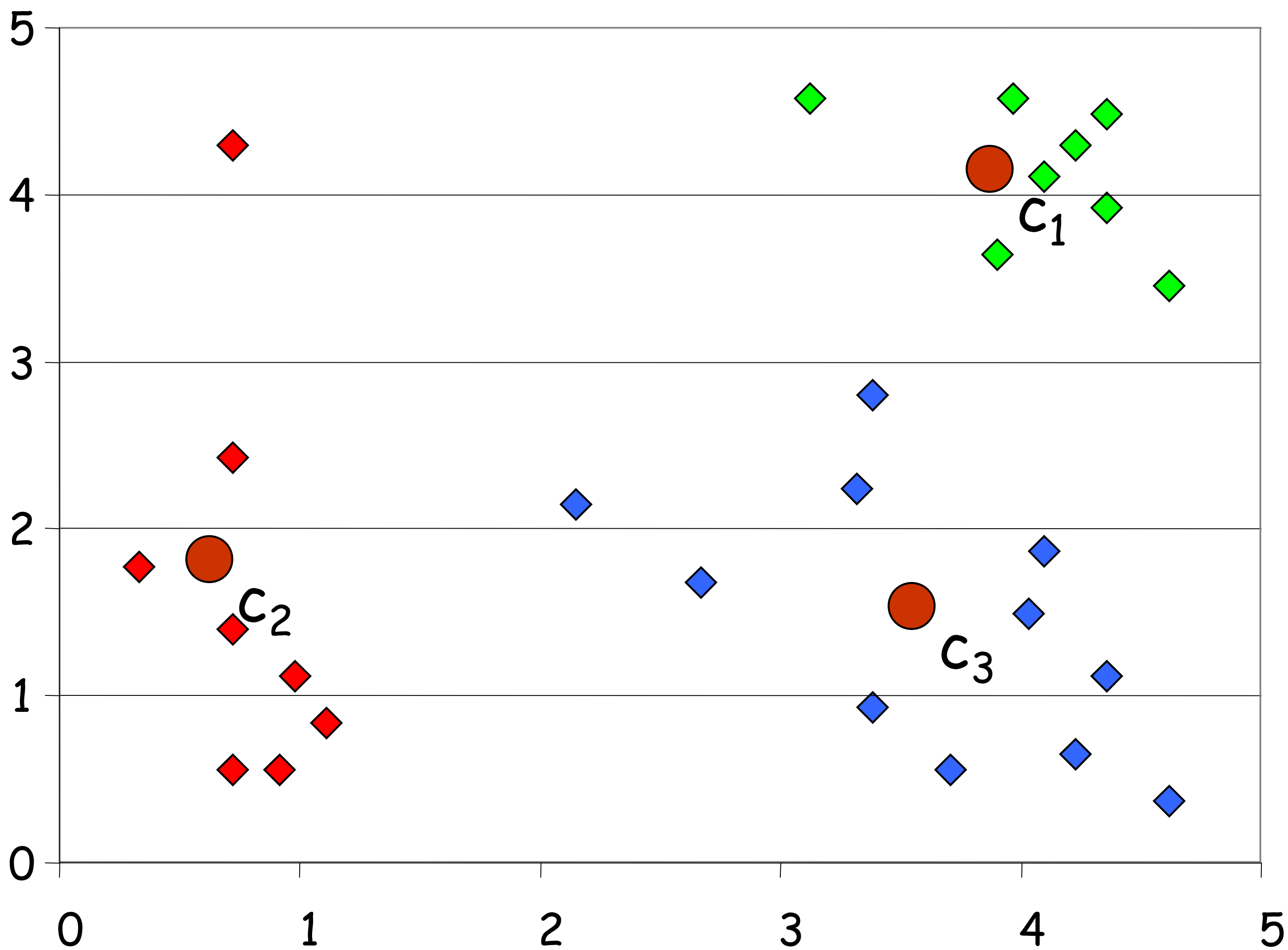


# Example

re-assign data points to new closest centroïds



# Example



# Notes on k-means

- converges very fast!
- BUT:
  - very sensitive to initial choice of centroids
    - many find useless clusters…
  - user must set initial k
    - not easy to do…

- many other clustering algorithms…

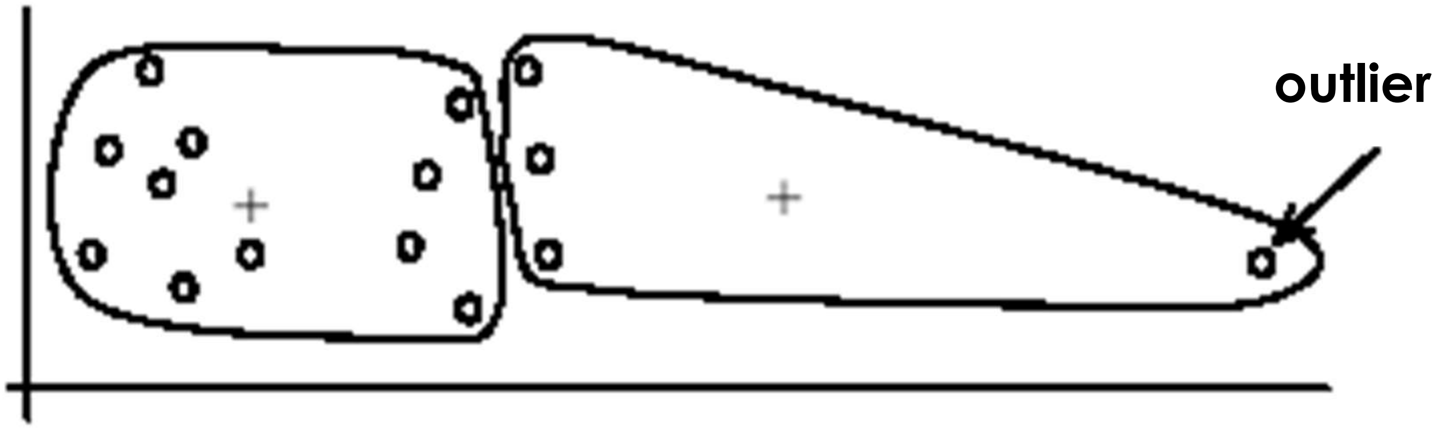
# Why use k-means?

- Strengths:
  - Simple
    - Easy to understand and implement
  - Efficient: Time complexity $O(t \cdot k \cdot n)$
    - n number of data points
    - k number of clusters
    - t number of iterations
  - With small k and t, linear performance on practical problems

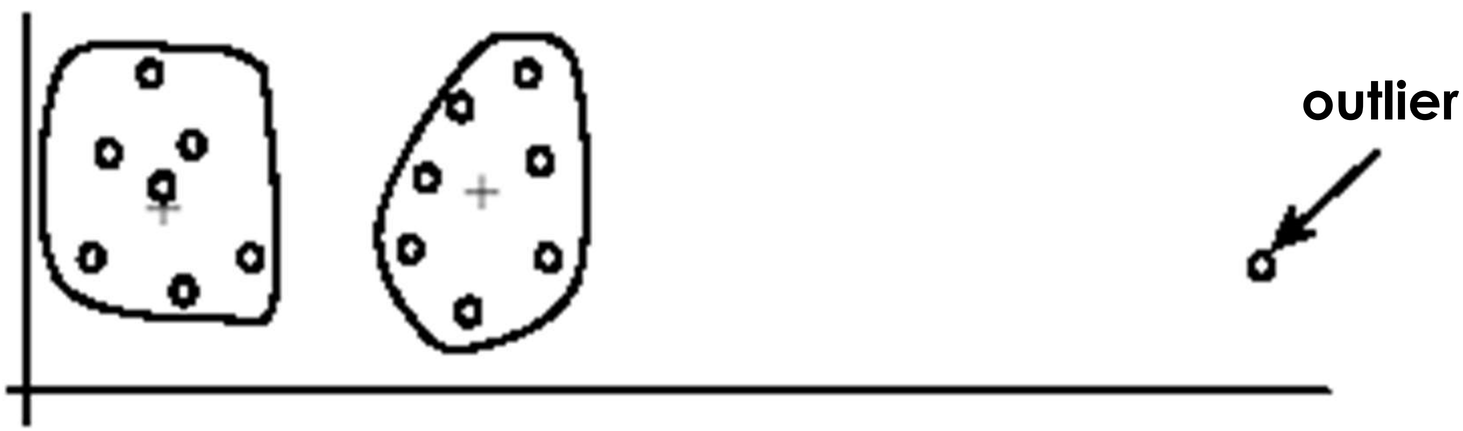
# Weakness of k-means

- User needs to specify k
- Algorithm is sensitive to outliers
  - i.e., data points that are far away from others
  - Could be errors in the data or special data points with very different characteristics

# Outliers

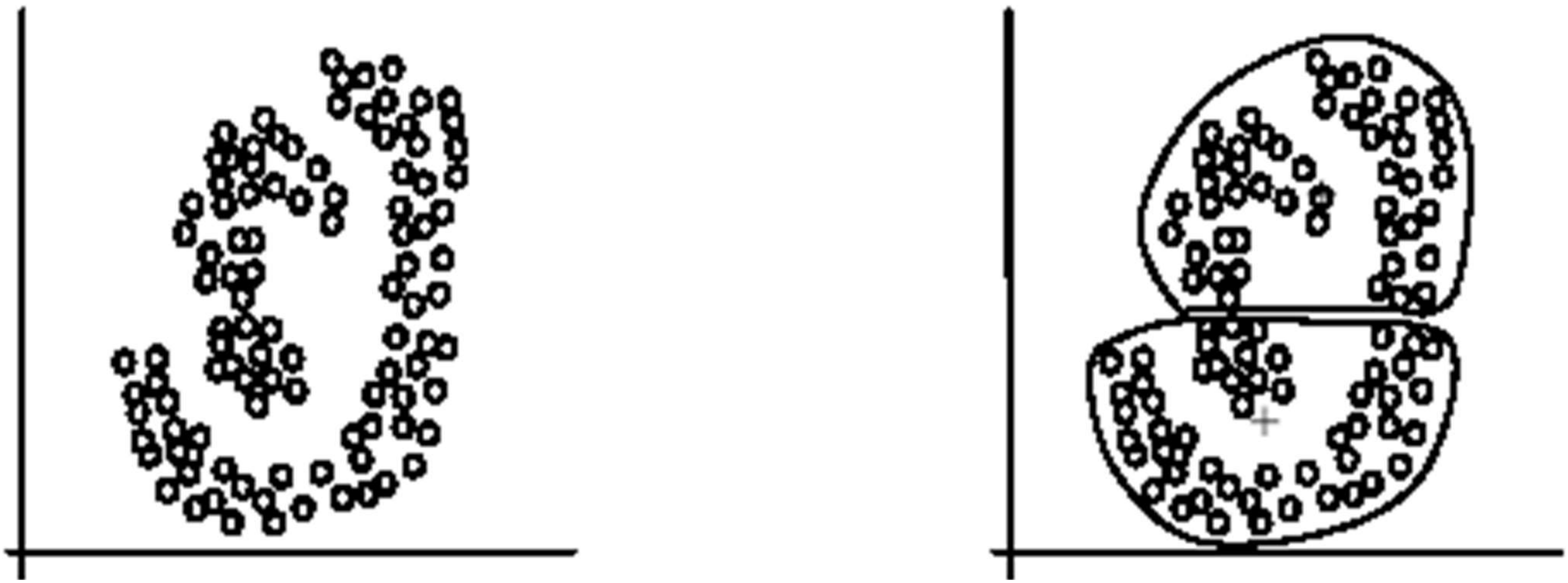



**(A) Undesirable clusters**




**(B) Ideal clusters**

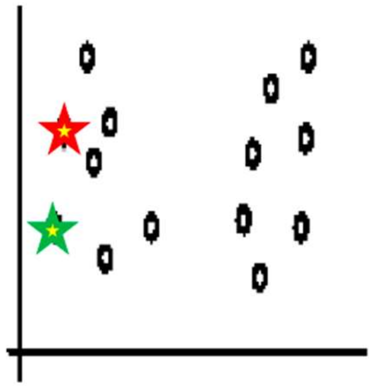
# Special data structures



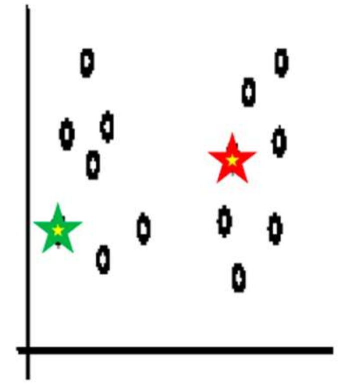
(A) Two natural clusters

(B) k-means clusters

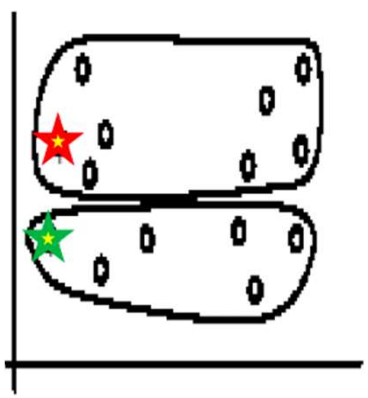
# Sensitivity to initial seeds



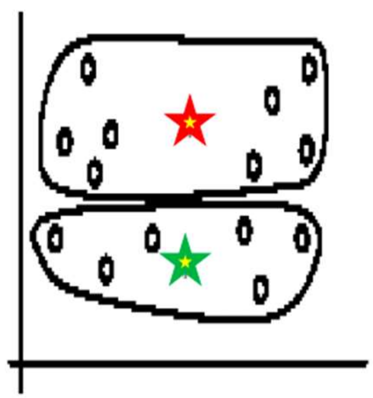
Random selection of seeds (centroids)



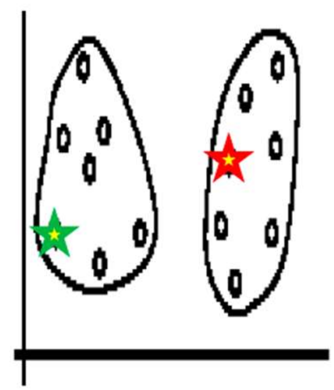
Random selection of seeds (centroids)



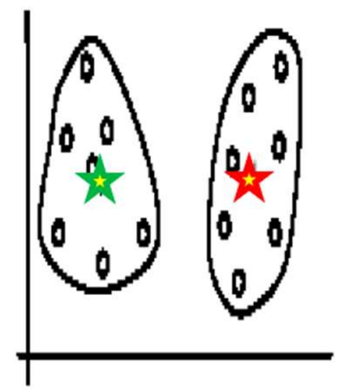
Iteration 1



Iteration 2



Iteration 1



Iteration 2

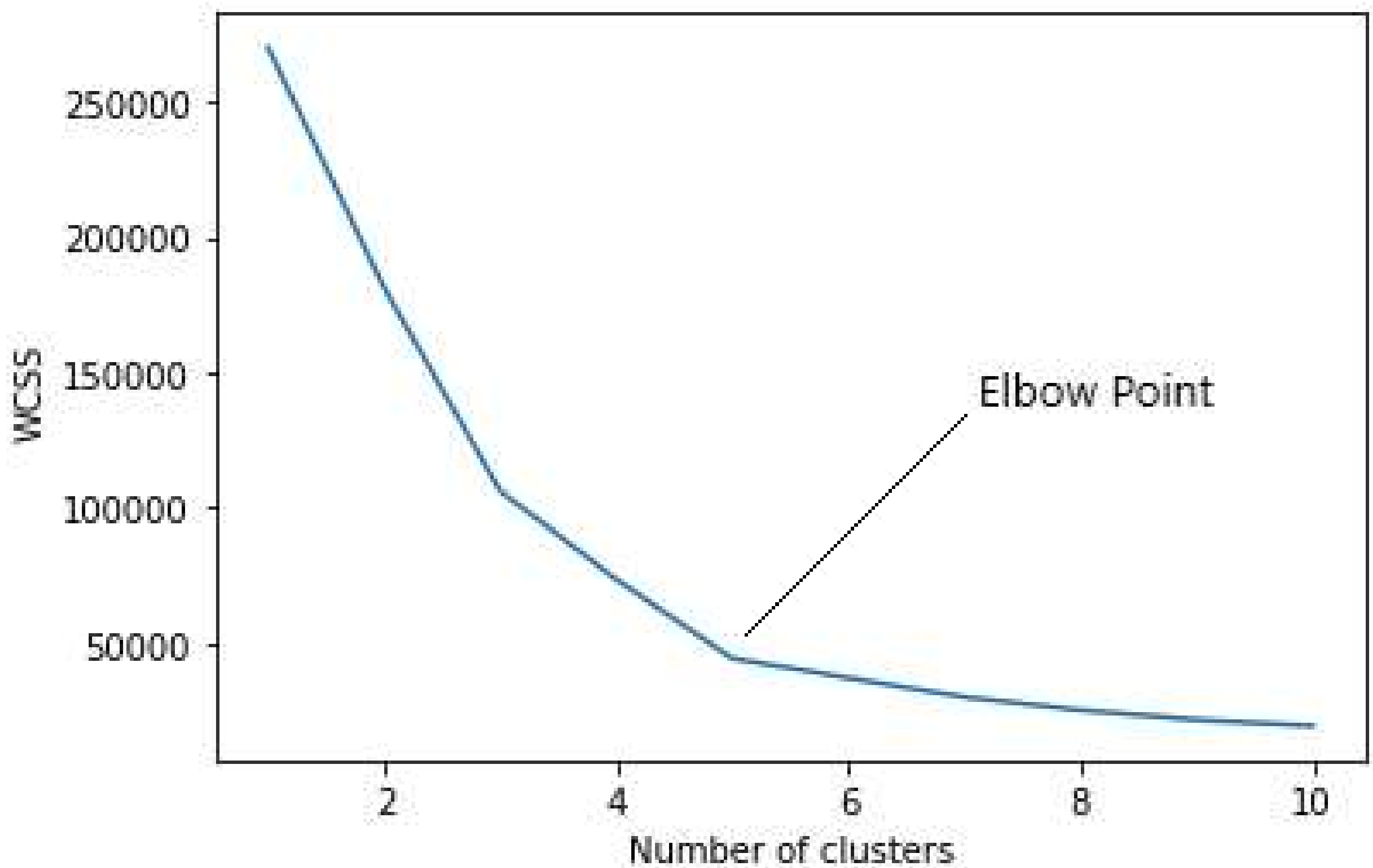
# How to decide the number of clusters (K)

- In some contexts, it may be given
  - Classification of clients into Platinum, Gold, and Silver
- Finding the “Optimal” K
  - Trying a few plausible values
  - Using the elbow method

# Elbow Method

- Define a clustering performance metric
  - Within cluster sum of square distances (WCSS)
- Calculate the performance for a few K
  - Most performance metrics are expected to decrease when K increasing
  - Take the K at the elbow

# Elbow Method

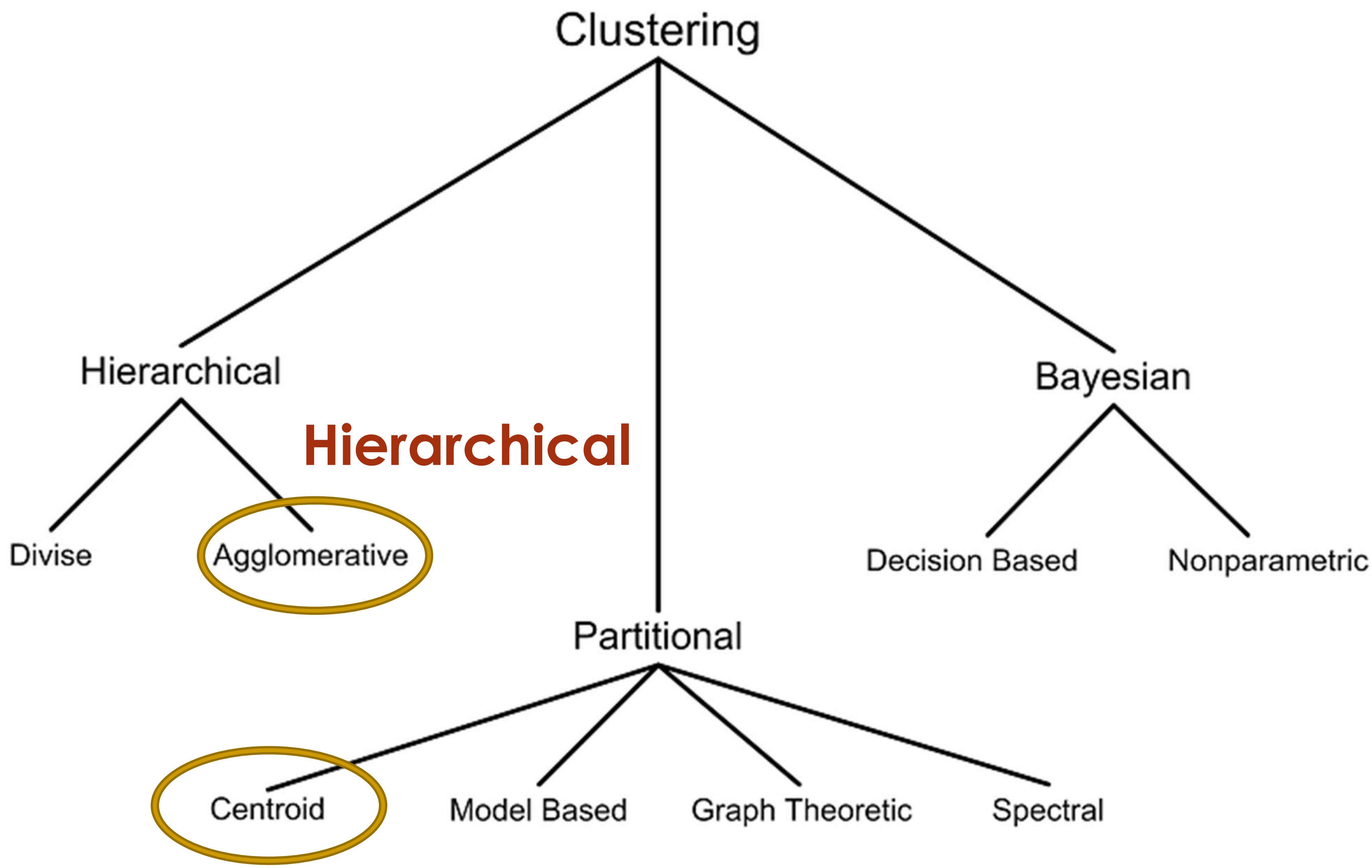



Source: https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning.

# K-means: Summary

- Despite weaknesses, k-means is still one of the most popular algorithms, due to its simplicity and efficiency
- No clear evidence that any other clustering algorithm performs better in general
- Comparing different clustering algorithms is a difficult task.
  No one knows the correct clusters!

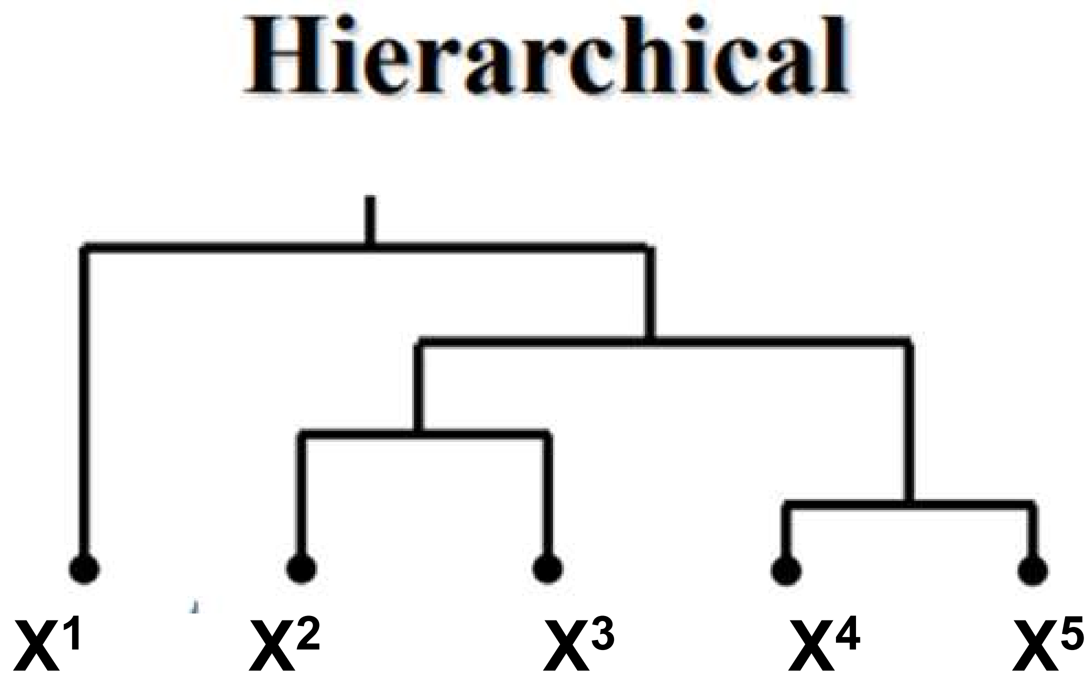
# Clustering Techniques



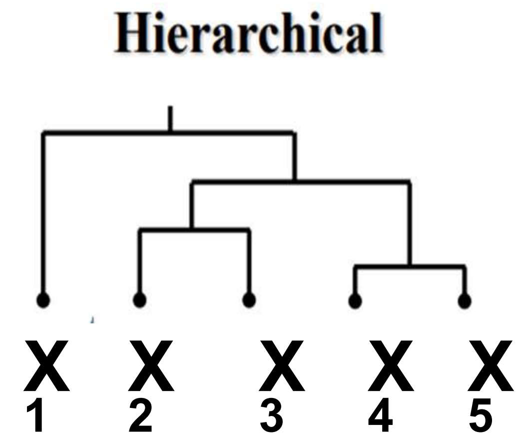
# Hierarchical Clustering

- A hierarchical decomposition of the observations using distance-based criteria

- Bottom-Up (Agglomerative) Clustering is the most popular

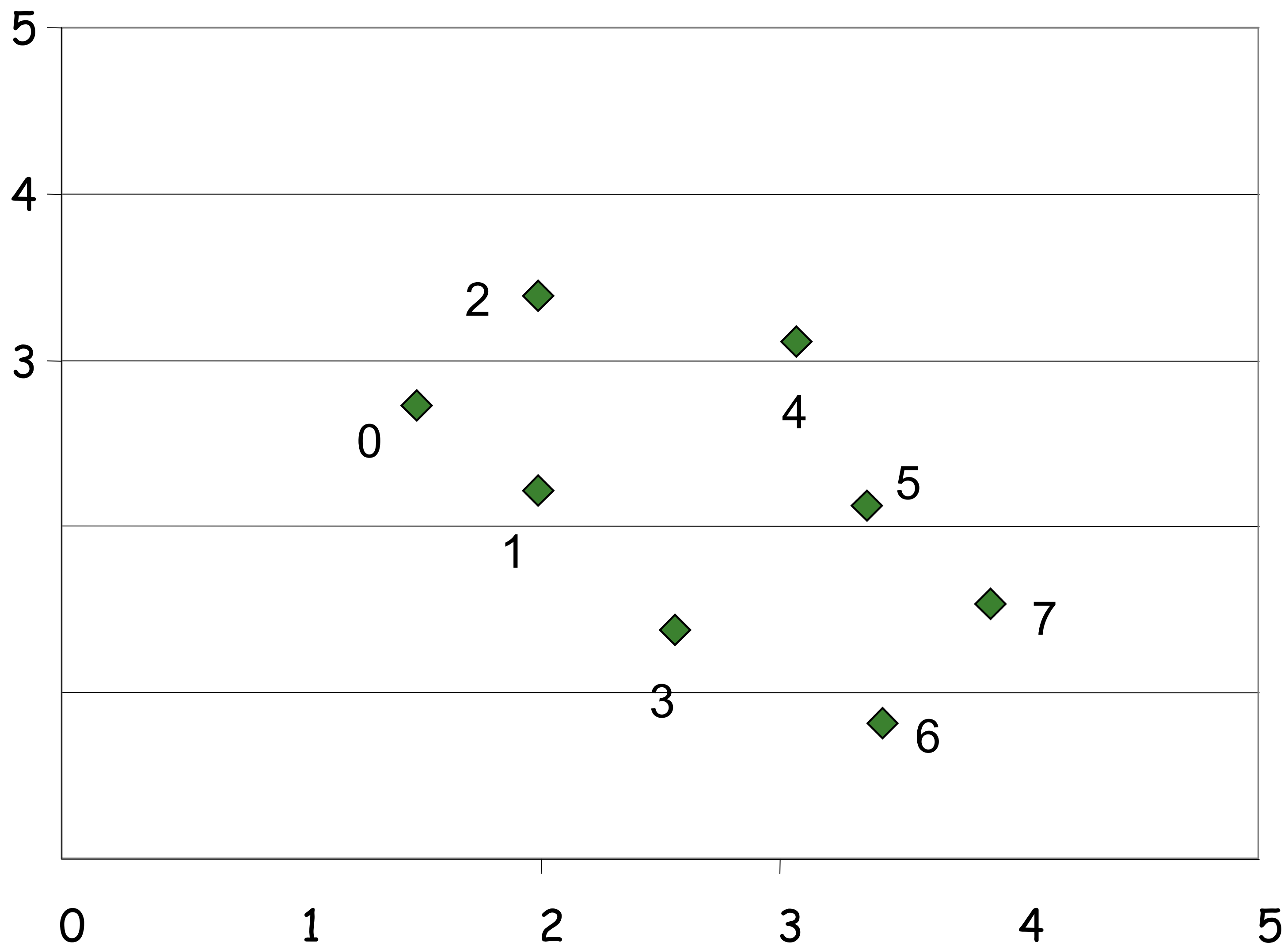


# Bottom-Up (Agglomerative) Clustering

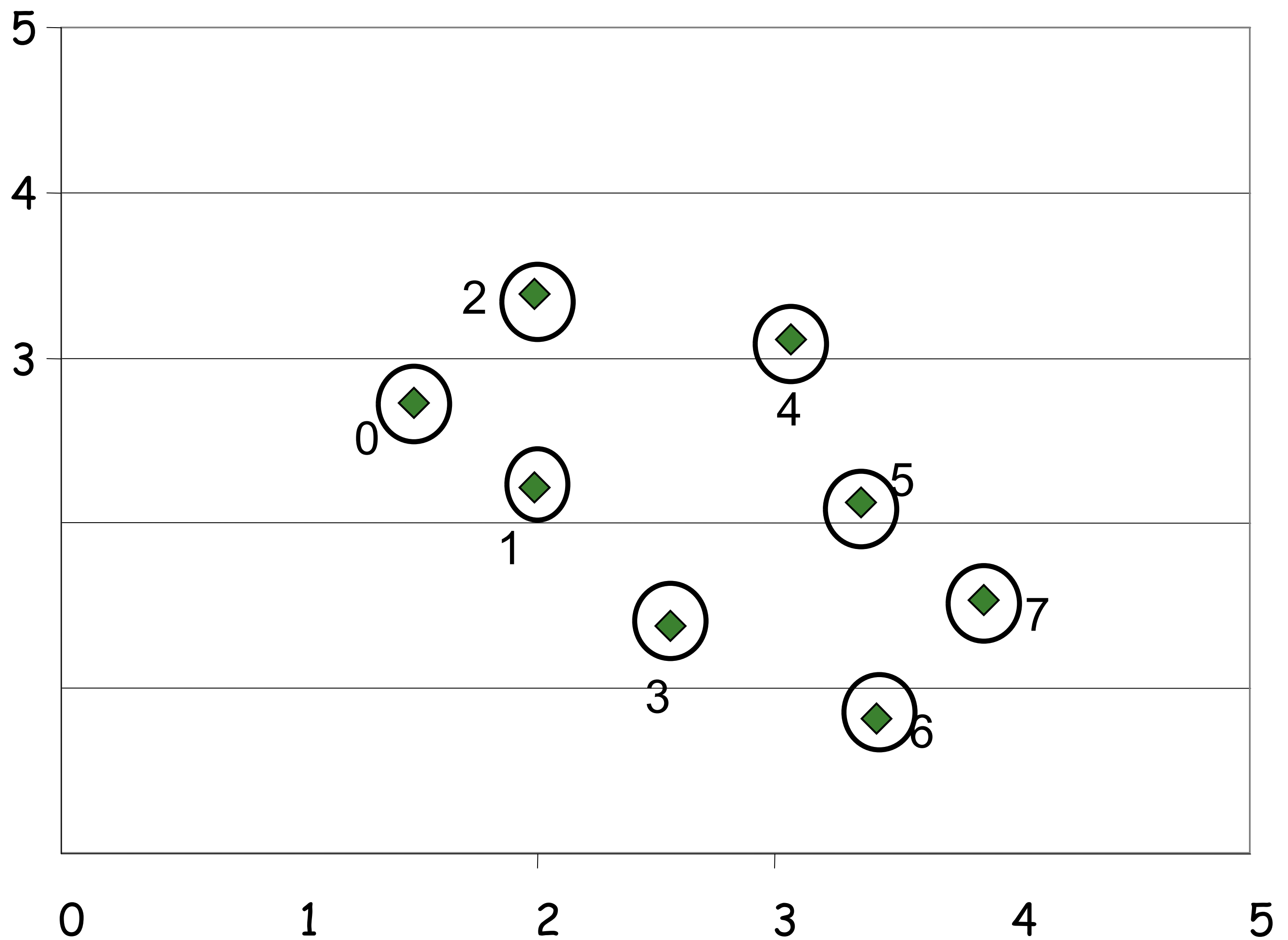



- 1) Start with every observation as a cluster
- 2) Find the best two clusters to merge to become a new cluster
- Repeat Step 2 until all clusters are merged as a single cluster with all n observation.
- Analyze the cluster formation (or the dendrograms) and select the optimal number of clusters

# Agglomerative Clustering, Example



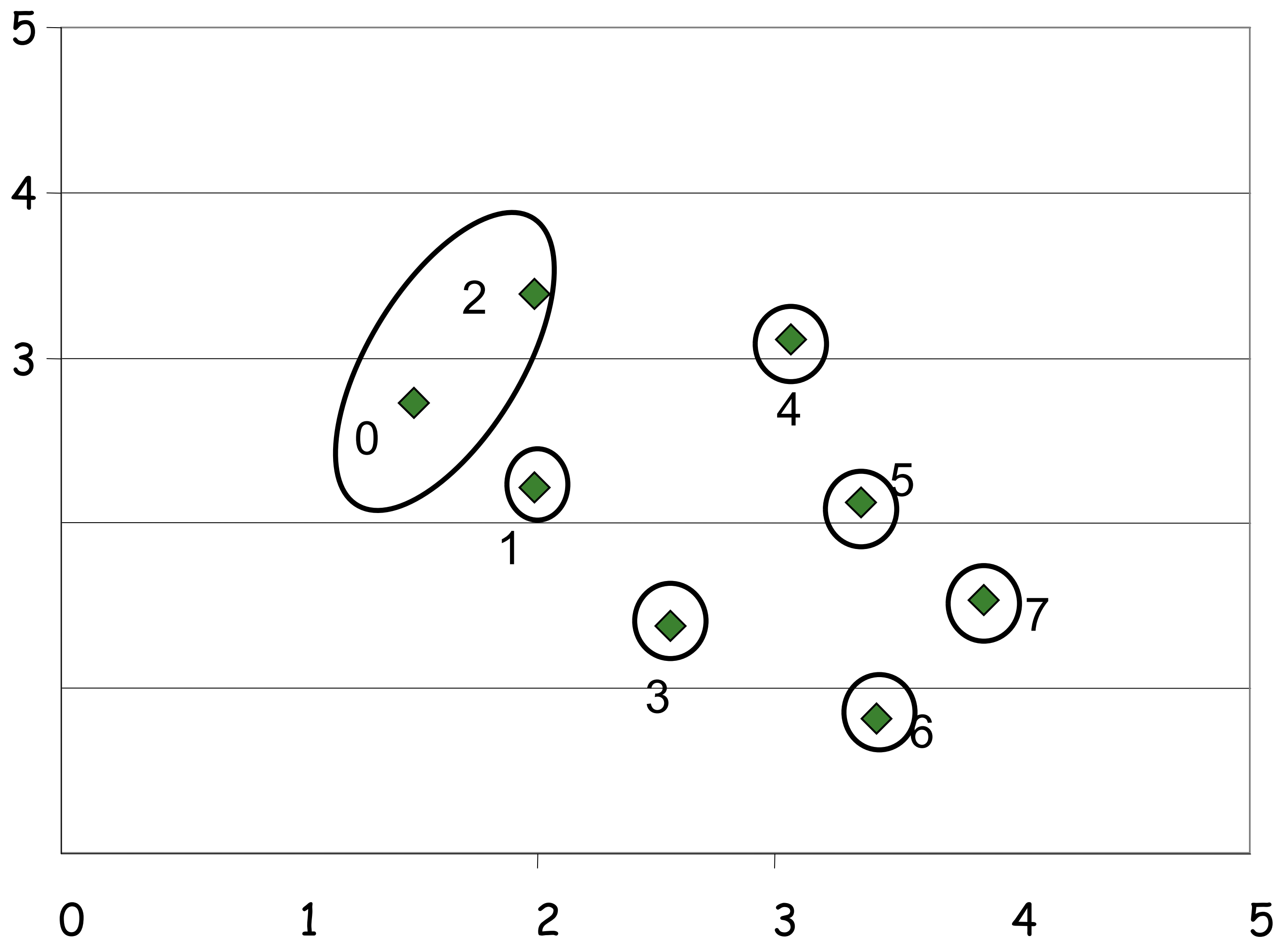
# Agglomerative Clustering, Example



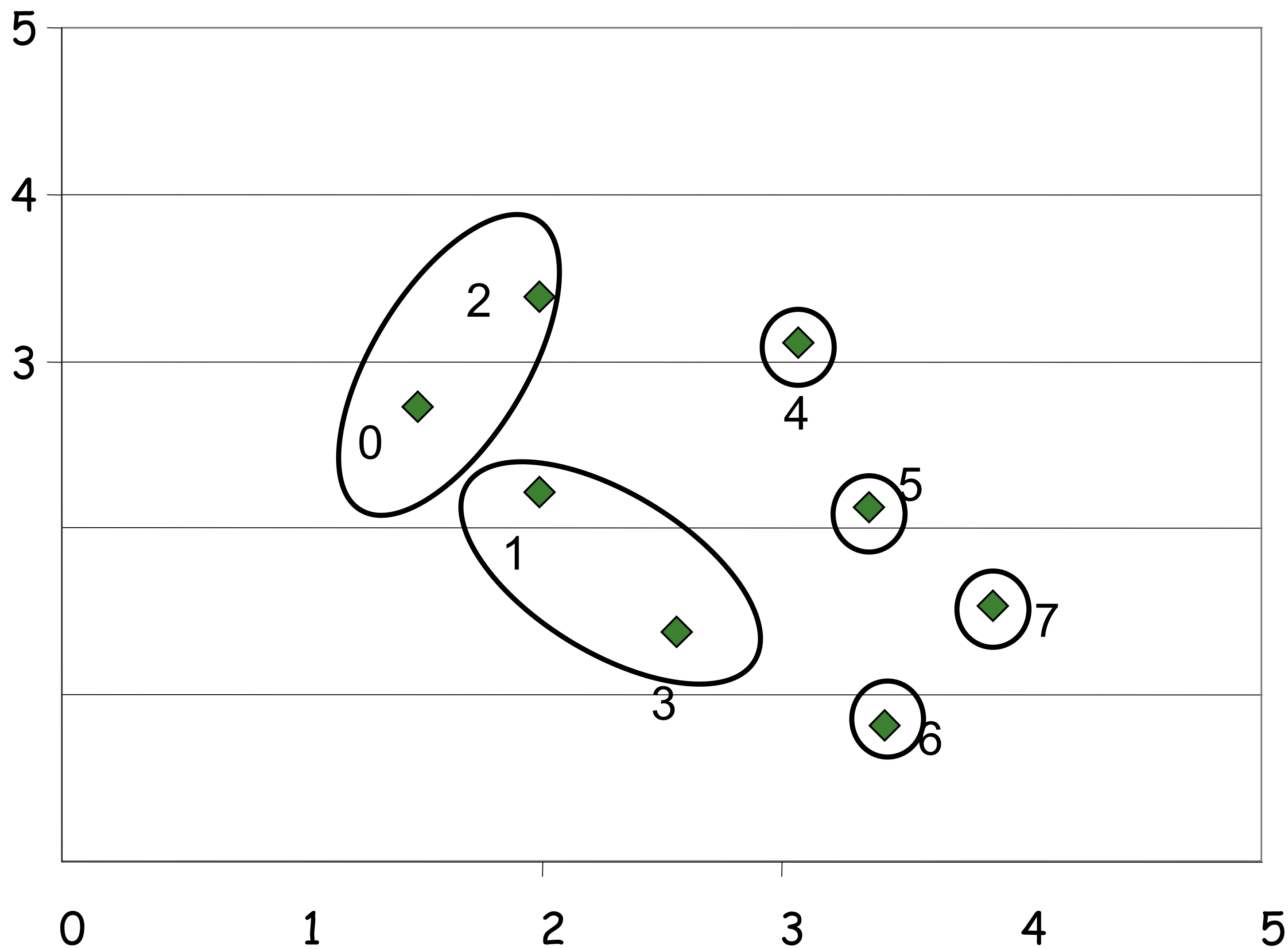

Every observation as a cluster

Nc = m = 8

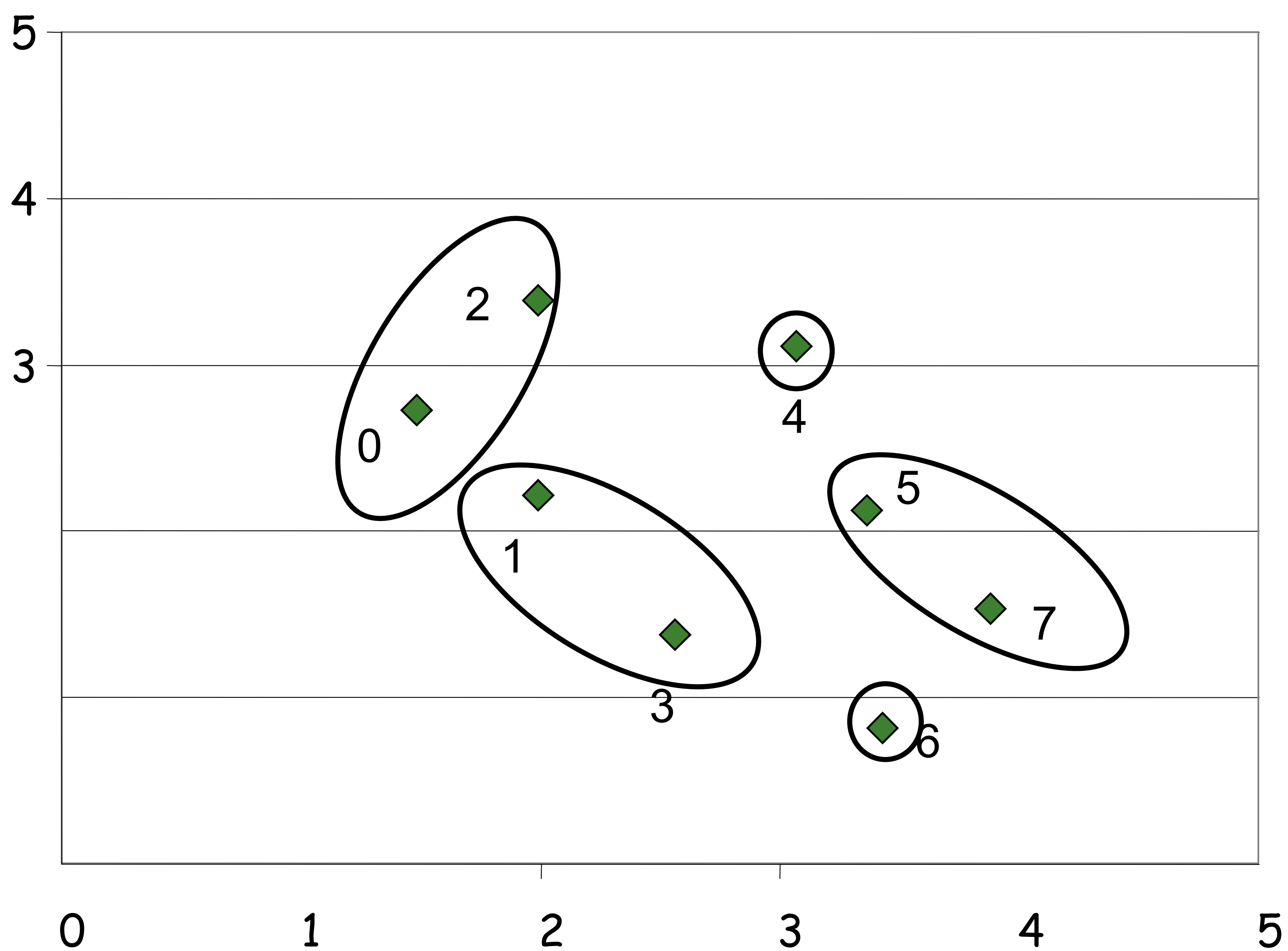
# Agglomerative Clustering, Example



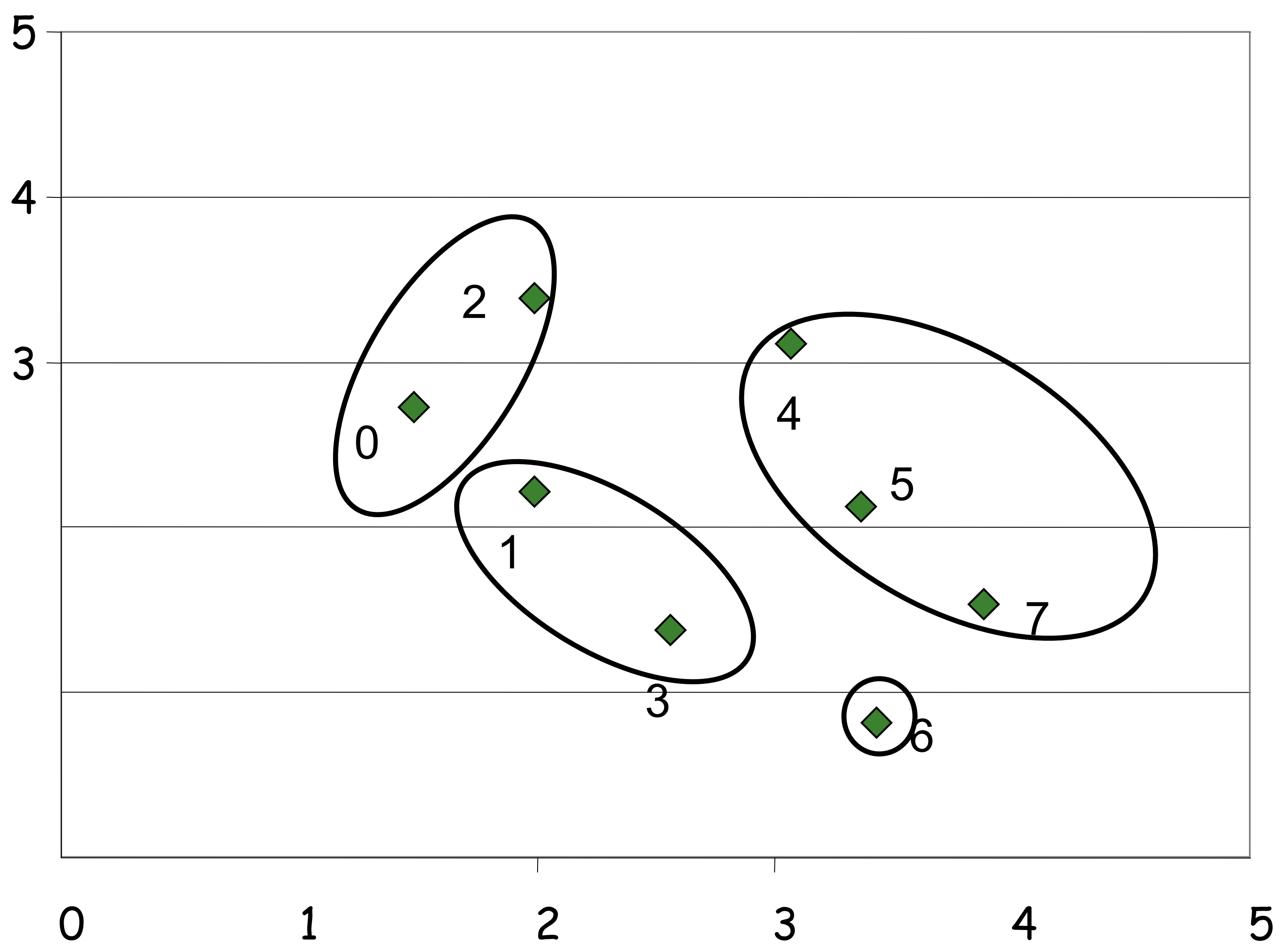
# Agglomerative Clustering, Example



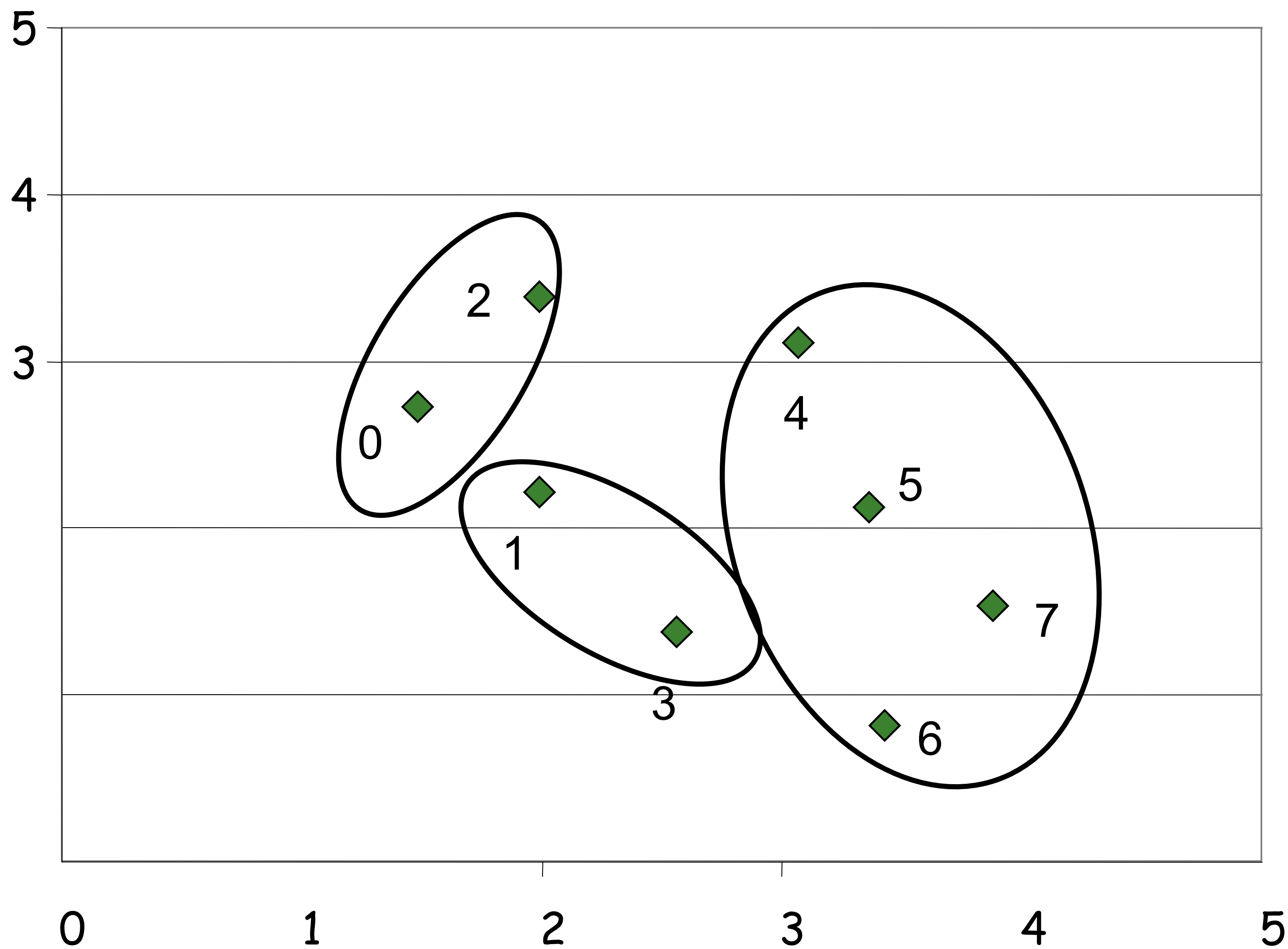
# Agglomerative Clustering, Example



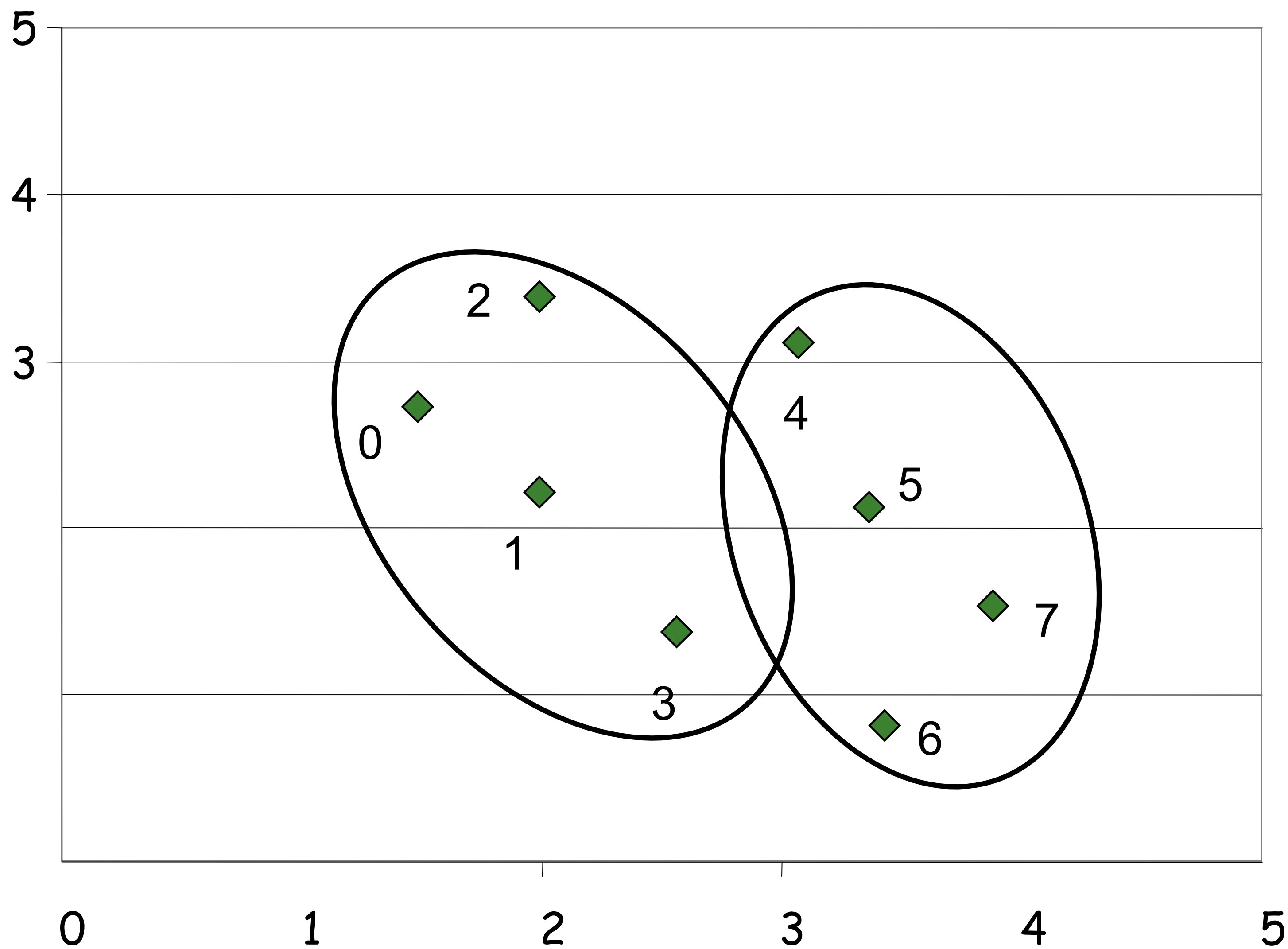
# Agglomerative Clustering, Example



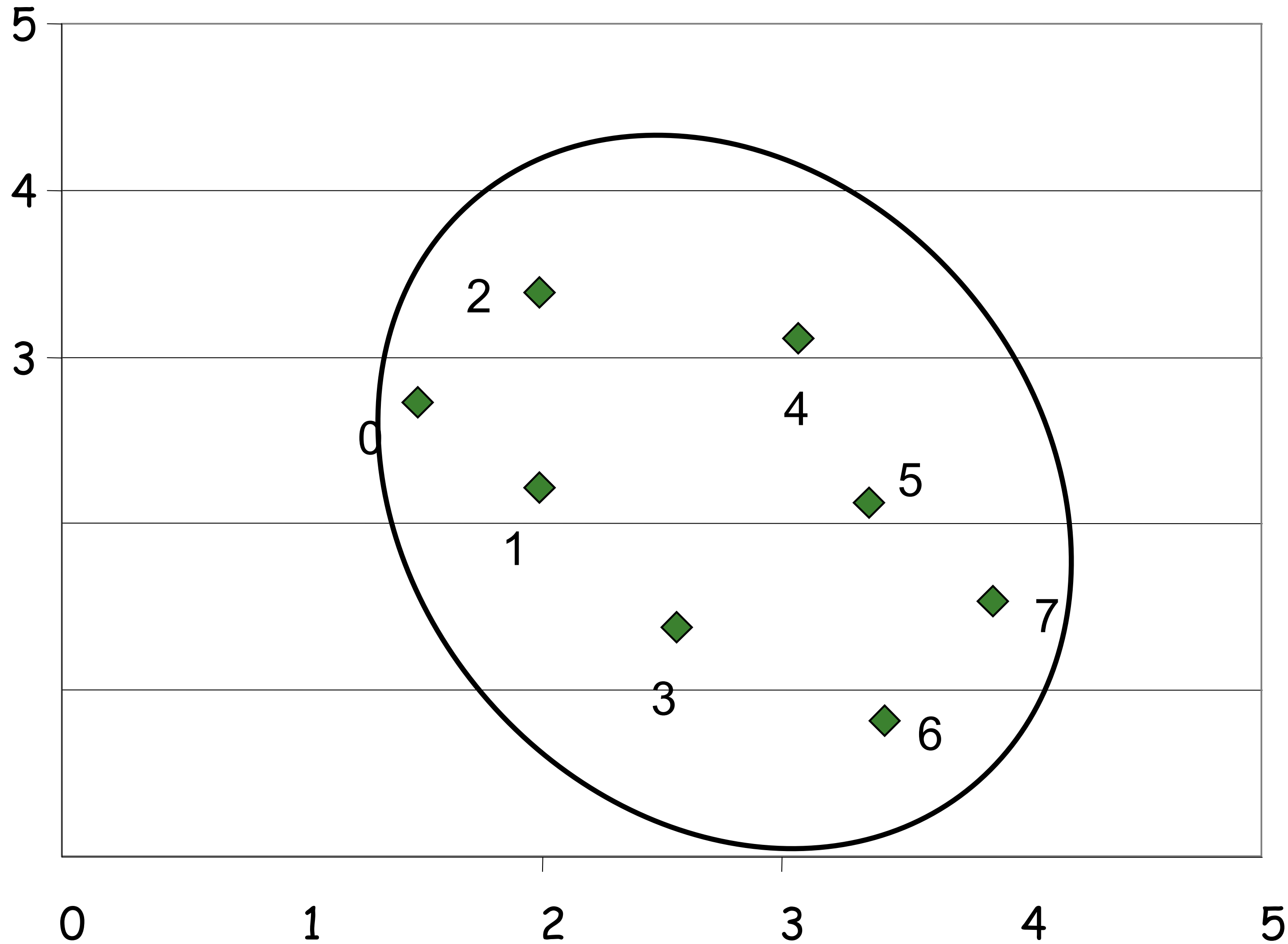
# Agglomerative Clustering, Example



# Agglomerative Clustering, Example



# Agglomerative Clustering, Example

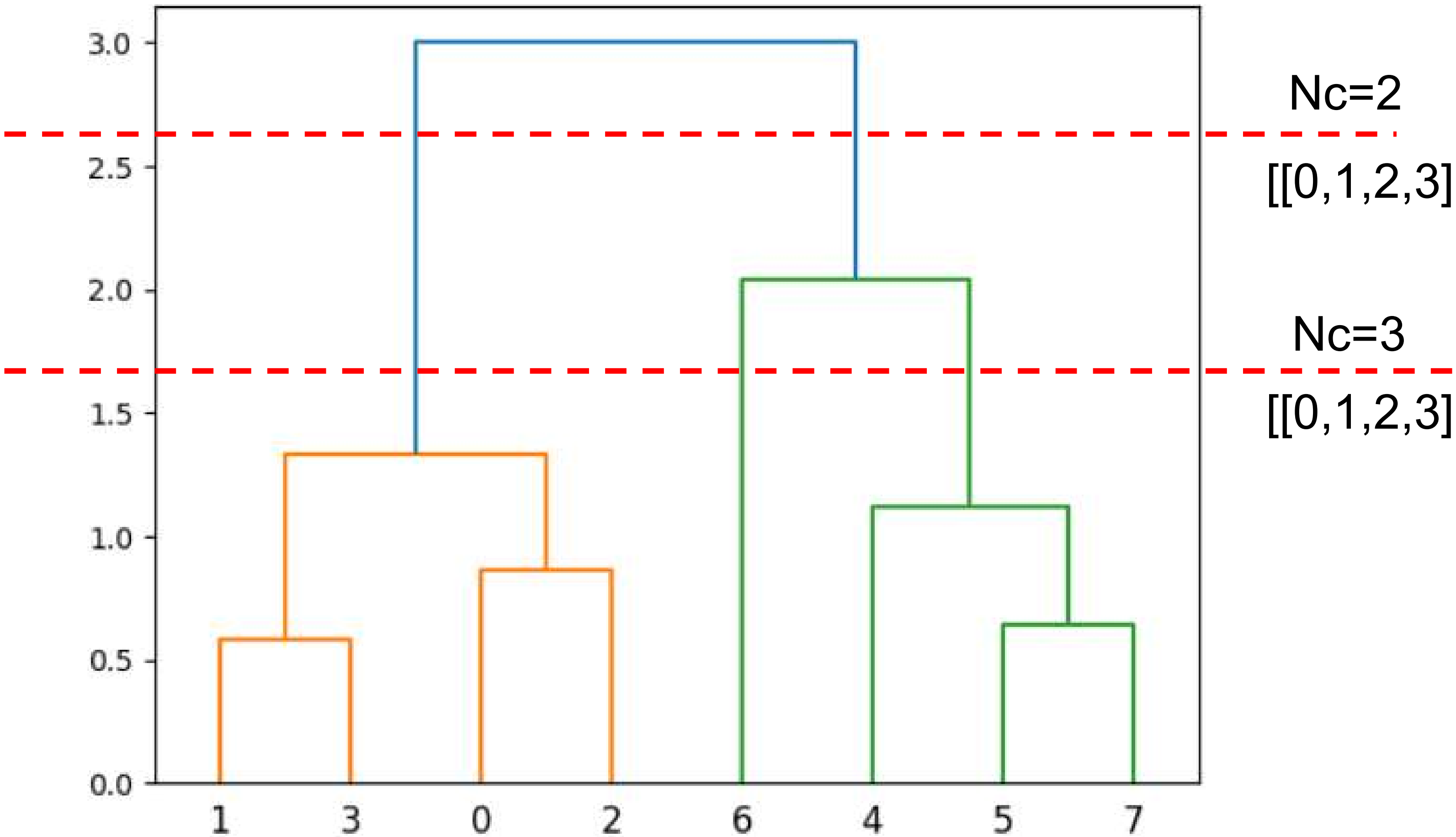



One cluster with all *m*= *8* observation

# Optimal number of clusters

A big jump in distance (**A high cost to merge two clusters**) indicates that these two clusters are not similar, and the merge is not a proper decision.

A horizontal line cutting such "improper" moves gives the optimal number of clusters




Nc=2

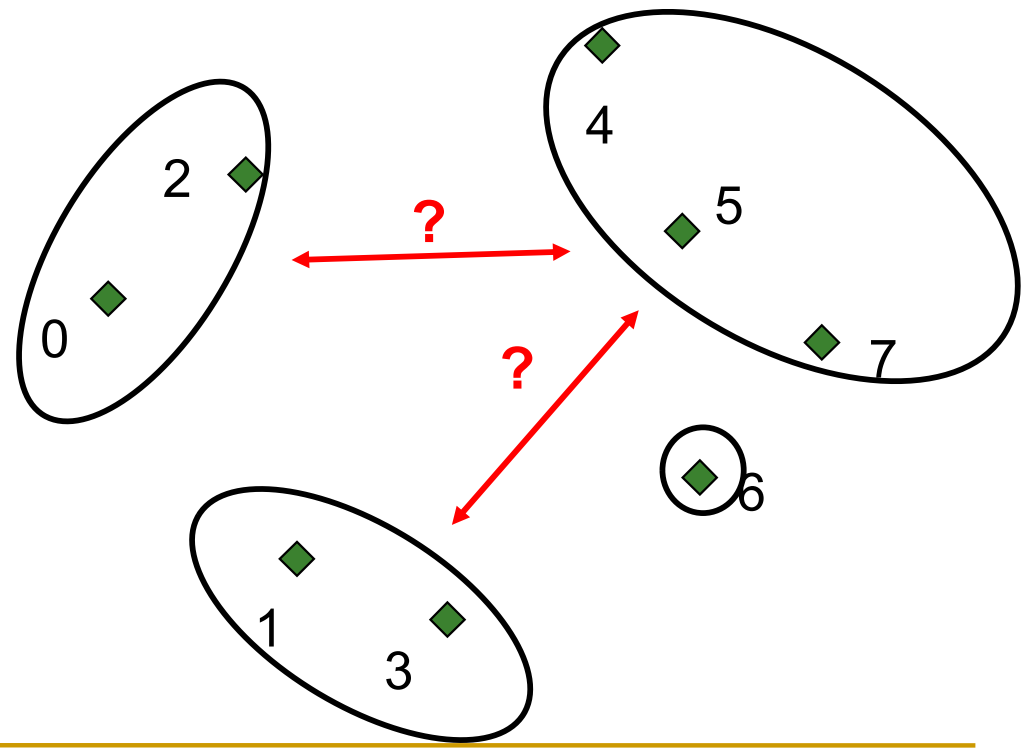
[[0,1,2,3], [4,5,6,7]]

Nc=3

[[0,1,2,3], [6], [4,5,7]]

# Distance of two clusters ?

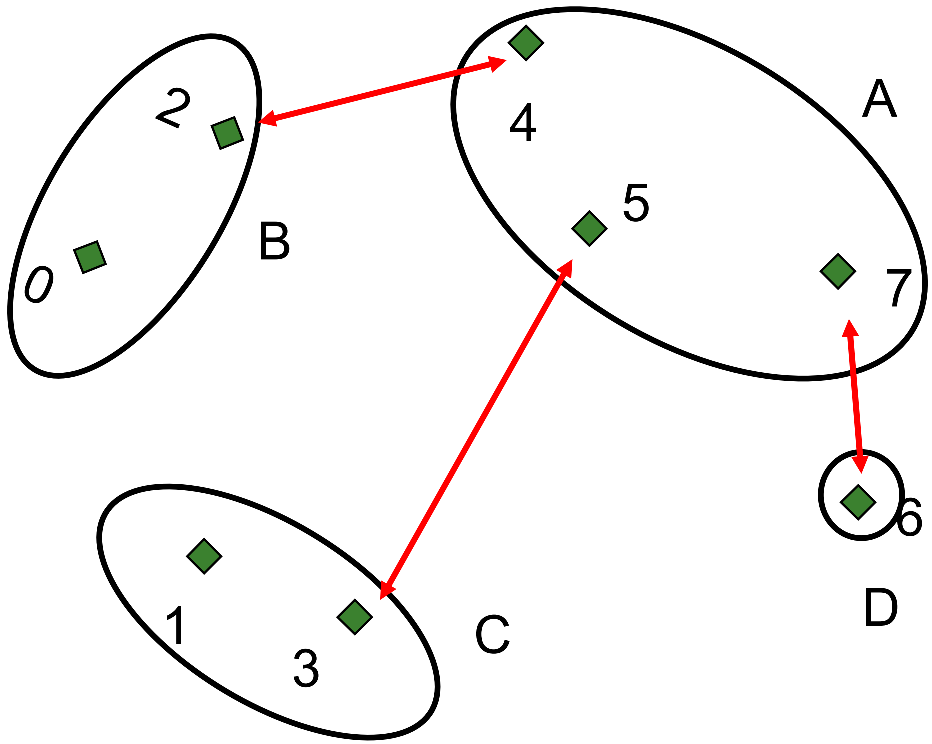
- We know how to find the distance of two observations. But, what about two cluster ?
- Different approaches
  - Closest members
  - Farthest members
  - Average of all members
  - Distance of centroids
  - …



# Distance of two clusters: Single Link

- Distance of **two closest members**
- Potentially results in skinny long clusters

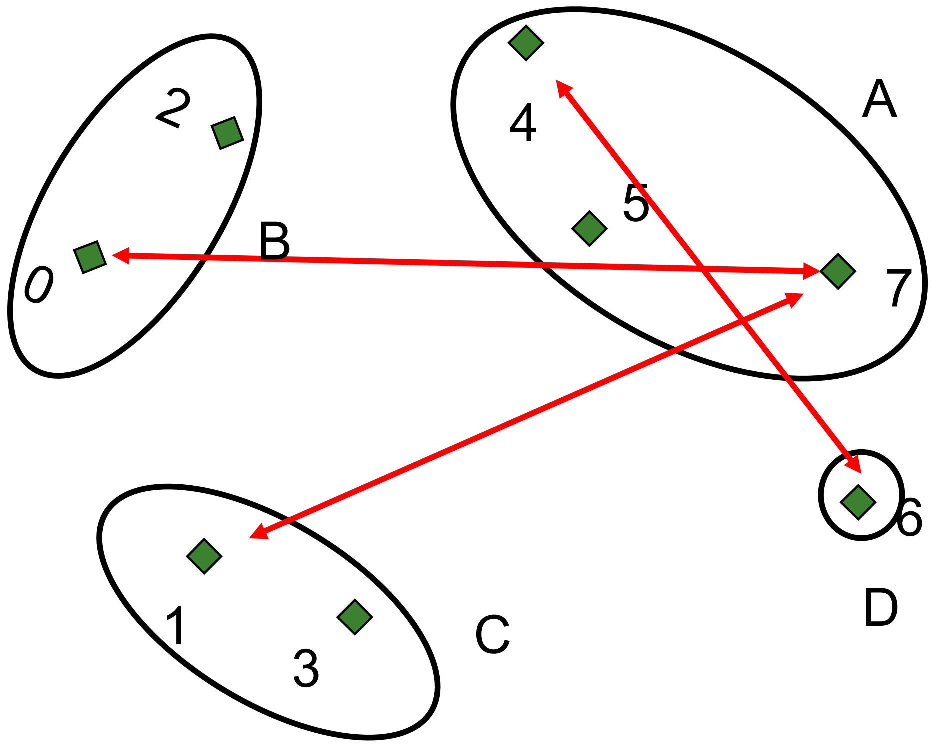
| d (ci, cj) | Given by |
|---|---|
| A, B | d (2,4) |
| A, C | d (5,3) |
| A, D | d (7,6) |
| … | … |



# Distance of two clusters: Complete Link

- Distance of **two farthest members**
- Prone to noise and outliers

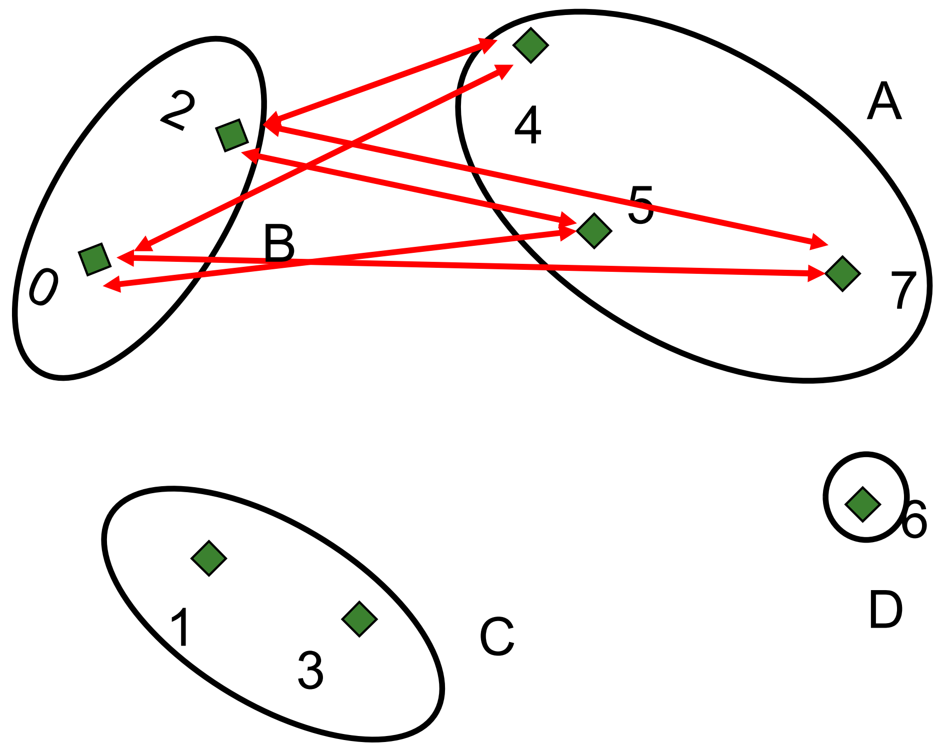
| d (ci, cj) | Given by |
|---|---|
| A, B | d (7,0) |
| A, C | d (7,1) |
| A, D | d (4,6) |
| … | … |



# Distance of two clusters: Average Link

- Average distance of **all pairs between two clusters**
- More robust against noise and outliers

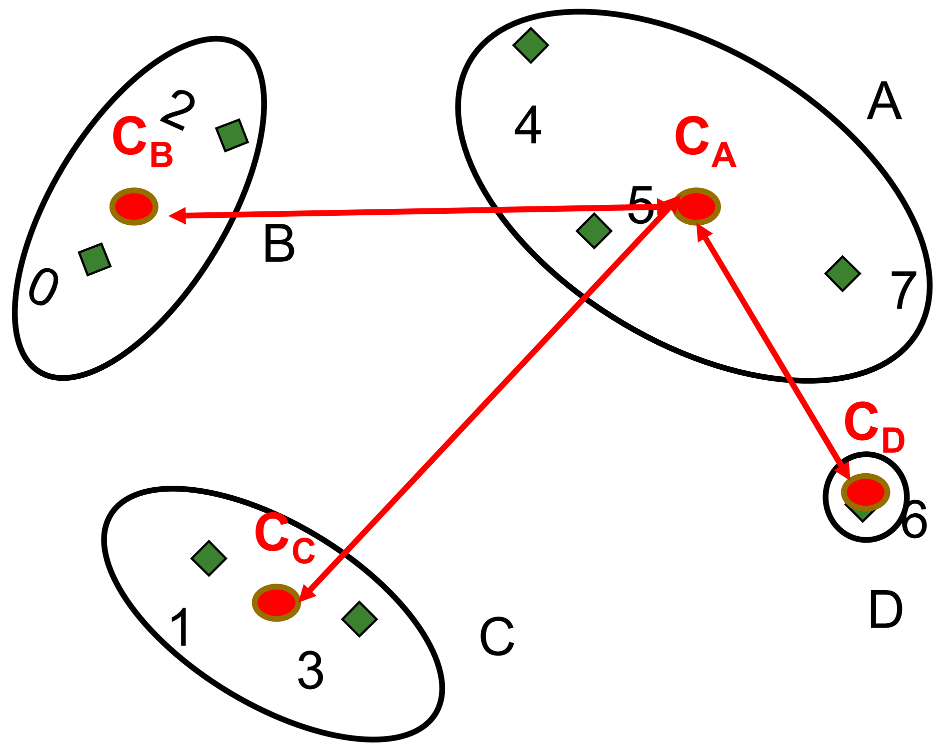
| d (ci, cj) | Given by |
|---|---|
| A, B | Average of all 6 links |
| A, C | Average of all 6 links |
| A, D | Average of d(4,6), d(5,6), and d(7,6) |
| … | … |



# Distance of two clusters: Centroid Link

- Distance **between the centroids of two clusters**
- More robust against noise and outliers

| d (ci, cj) | Given by |
|---|---|
| A, B | d ($C_A$, $C_B$) |
| A, C | d ($C_A$, $C_C$) |
| A, D | d ($C_A$, $C_D$) |
| … | … |

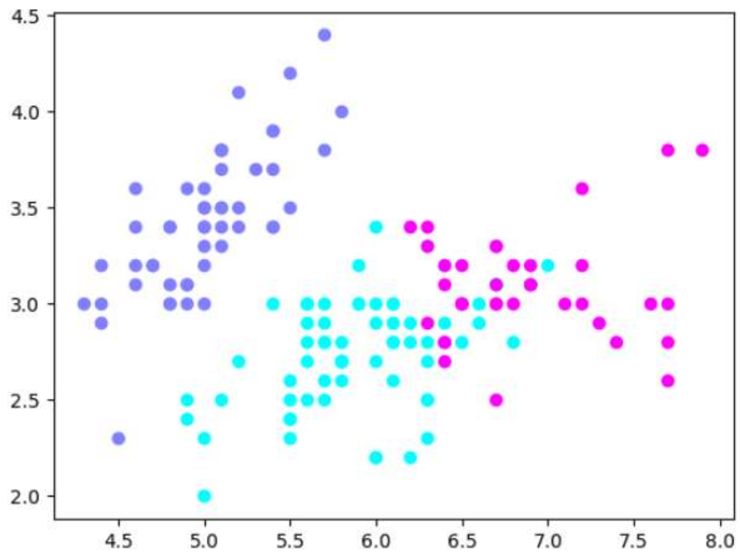


## Distance of two clusters: Ward Minimum Variance Link

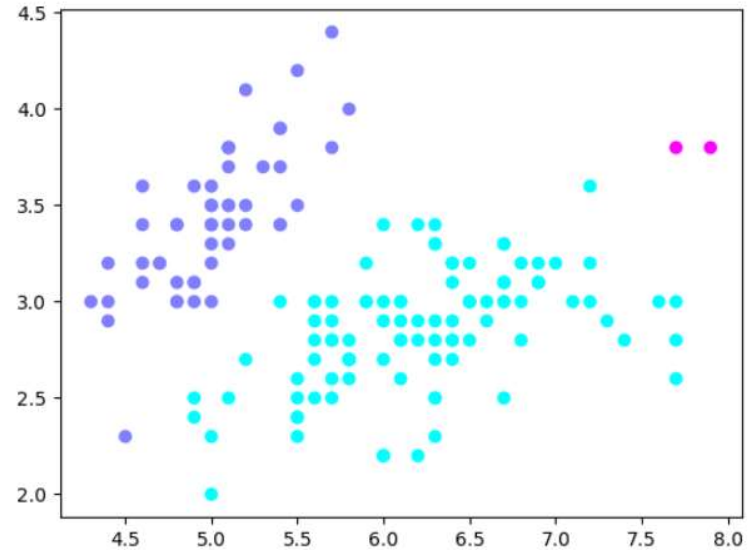
- Minimizes the total within-cluster variance
- Merges two clusters that results in a minimum increase of within cluster variance
- Tends to merge smaller clusters together

$$d(A, B) = \frac{||CA - CB||^2}{\frac{1}{n_A} + \frac{1}{n_B}}$$

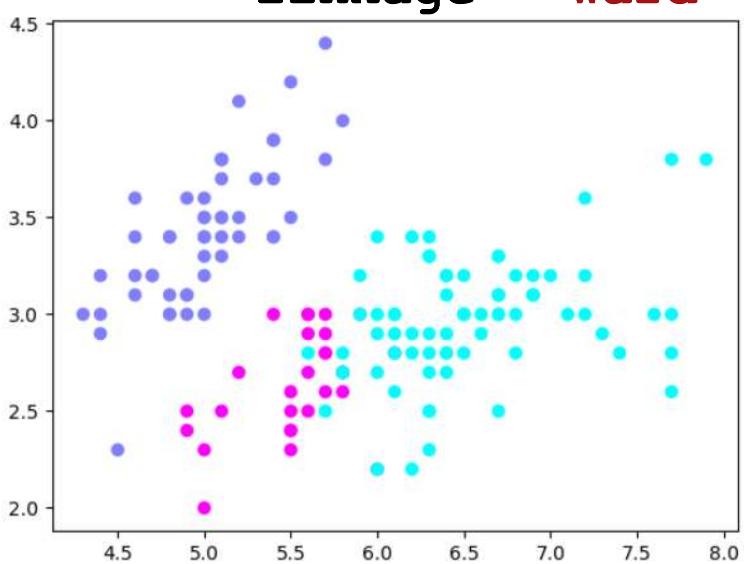
# Compare linkage options



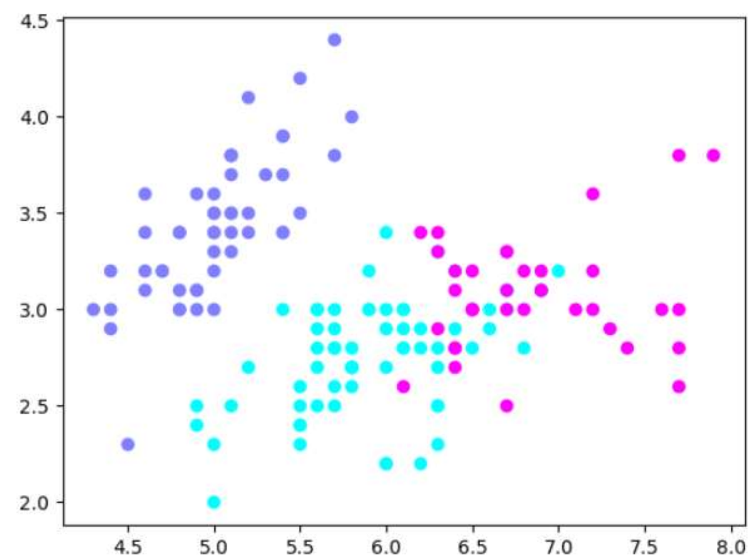

`linkage= 'ward'`




`linkage='single'`




`linkage='Complete'`




`linkage='Average'`

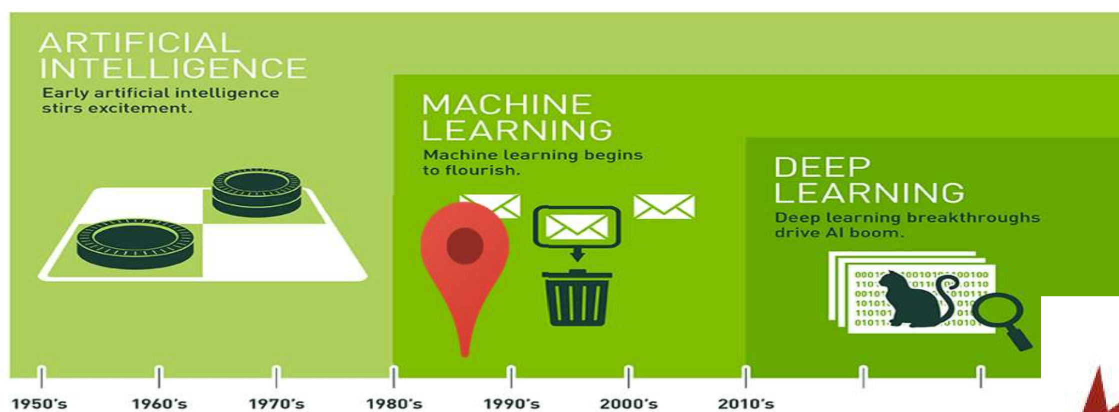
Iris dataset from `sklearn`, using Euclidean distance

# Agglomerative Clustering : Summary

- Simple and easy to understand
- No need to specify the number of clusters in advance.
- Complexity is usually higher than K-Means
  Number of optimal clusters is subjective.
  No one knows the correct clusters!

# Today

1. Hierarchical Clustering
2. K-means Clustering
3. Hierarchical Clustering





https://medium.com/machine-learning-for-humans/neural-networks-deep-learning-cdad8aeae49b