# CS559 – Unsupervised Learning I: Clustering

# Announcement

- Assignments and Exams are still in the process of grading.
- Assignment 3 on decision trees, ensemble methods, and clusters will be announced after the lecture.
- Project will be discussed in the next slide.

# Outline

- Project Discussion
- Intro to Unsupervised Learning
- Clustering
  - K-means Clustering
  - Hierarchical Clustering
- Principal Component Analysis (Will be discussed next week)

# Project

Project Topic: Novel Corona Virus 2019 Dataset (Day level information on covid-19 affected cases) from Kaggle

Objectives:

- To predict the spreading of coronavirus – Can we help mitigate the secondary effects of covid-19 by predicting its spread?
- To predict how the epidemic will end – If your model suggests it will end…
- To correlate between growth rate and types of mitigation across countries.
- To correlation weather conditions and corona virus spread through data.
- Your own objective.

Data: Available on Canvas

Details of data and further information: https://www.kaggle.com/sudalairajkumar/novel-corona-virus-2019-dataset

# Project

- Individual Work – No discussions and sharing work.

- Resources – Any resources are possible but copying codes from kernels in Kaggle is prohibited. You may look on discussion forums.

- Due Date: 4/30/2020 Thursday 11:59 PM

- Submission Requirements:
  - A complete report
  - Codes
  - 10-15 minutes Recorded Presentation

# Recap Supervised Learning

- Task: to predict the values of one or more response variables Y from a given set of predictor variables X.

- Predictions are based on the training data of previously solved cases.

- Performance can be estimated by some loss function using the training-test splitting or cross-validation

- Regression – simple/multiple linear regression, regression trees, etc.
- Classification – logistic classification, support vector machines, classification trees, etc.

# Unsupervised Learning

- Only a set of N observations with p features, *no response variables*.
- Goal: to infer the properties directly without knowing the "correct" answers or the error for each observation
- "learning without a teacher" – No direct measure of success
- Two methods:
  - Principal Components Analysis (PCA) – often used for data visualization or data preprocessing for supervised learning
  - Clustering – a broad class of methods for grouping or segmenting a collection of objects into distinct subjets known as "clusters".

# Unsupervised Learning

- Unlabeled data is easier to obtain than labeled data.
- No specific prediction goals, therefore more subjective.
- We are usually interested in discovering the hidden pattern of the data.
- Examples:
  - Groups of online shoppers characterized by their browsing and purchase histories.
  - Movies grouped by the comments given by movie viewers.

# Cluster Analysis

We have been concerned with building models that **perform predictions**:

- Regression systems that attempts to predict a numeric outputs
- Classifiers that attempts to predict class membership.

In contrast, *cluster analysis* is an unsupervised task that:

- Does **not** aim to specifically predict a numeric output or class label.
- Does aim to uncover **underlying structure** of the data and see what pattern exists in the data.
    - We aim to group together observations that are similar while separating observations that are dissimilar.

# Clustering Analysis

Cluster analysis attempts to explore possible subpopulations that exist within your data.

Typical questions that cluster analysis attempts to answer are:

- Approximately how many subgroups exist in the data?
- Approximately what are the sizes of the subgroups in the data?
- What commonalities exist among members in similar subgroups?
- Are there smaller subgroups that can further segment current subgroups?
- Are there any outlying observations?
    - Notice that these questions are largely exploratory in nature.

# K-Means Clustering

With the K-means clustering algorithm, we aim to split up our observations into a predetermined number of clusters.

- The number of clusters *K* must be specified in advance.
- These cluster memberships are distinct and non-overlapping.

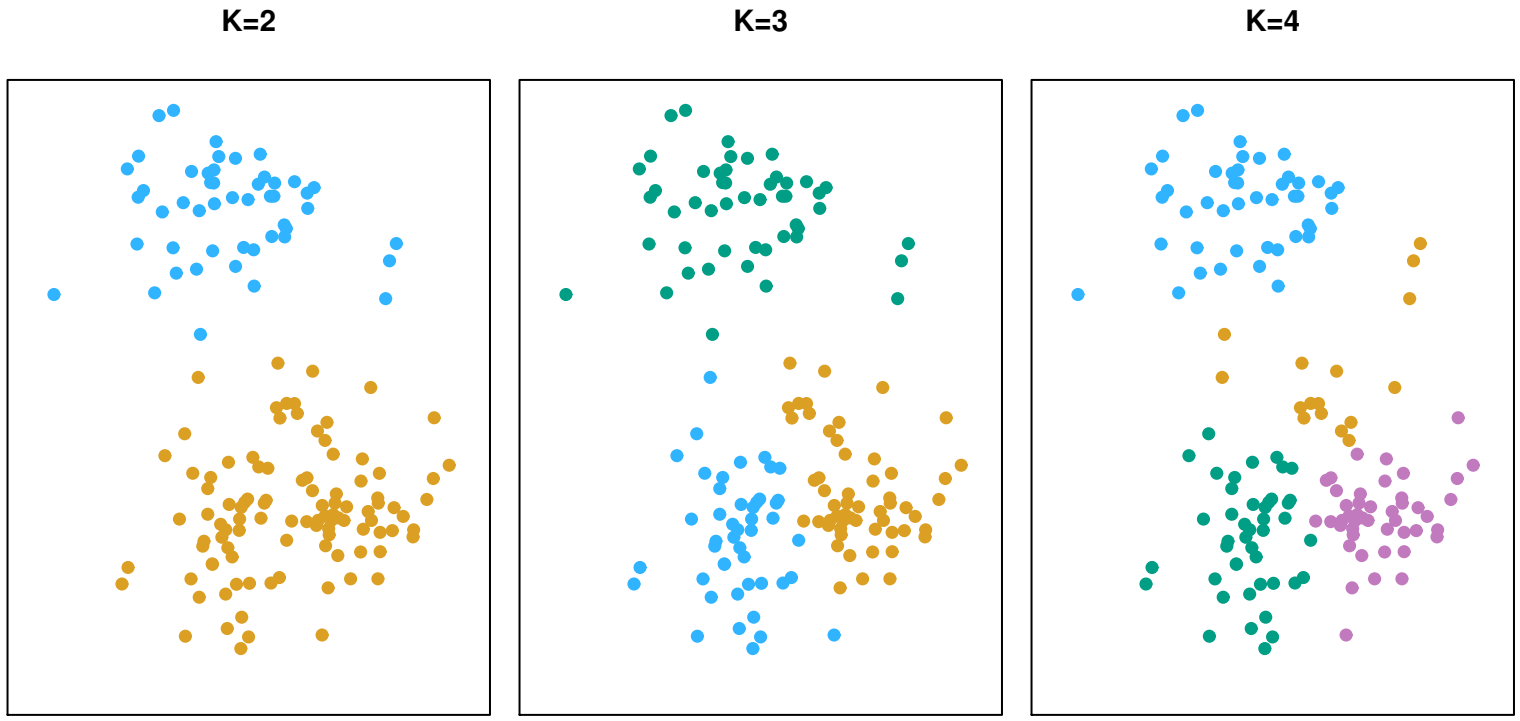
The data points in each of the clusters are determined to be mostly similar to a specific centroid value:

- The centroid of a cluster represents the average of the observations within a given cluster; it is a single theoretical center that represents the prototypical member that exists within the given cluster.

- Each observation will be assigned to exactly one of the K clusters depending on where the observation falls in the feature space relative to the cluster centroid locations.

# K-Means Clustering

A simulated data set with 150 observation in 2-dimensional space.




The color labels the cluster to which it has been assigned. Note that the cluster coloring is arbitrary since there is no absolute ordering of the clusters.

# K-Means Clustering

Now the main question: what technique does k-means algorithm use to create these clusters?
Suppose we use the *Euclidean* distance. Then the within-cluster variation is defined as:

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{P} \left(x_{ij} - x_{i'j}\right)^2$$

The number of variables/features in our dataset.

The total number of observations in cluster *k*.

Indices of observations in cluster $C_k$

# K-Means Clustering

Since the within-cluster variation is a quantitative gauge of the amount by which the observations in a specific cluster differ from one another, we want to ***minimize*** the sum of this quantity $W(C_k)$ over all clusters:

$$\min_{C_1,\ldots,C_k} \left\{ \sum_{k=1}^{K} W(C_k) \right\}$$

In other words, we would like to partition the observations into $K$ clusters such that the total within-cluster variation aggregated across all $K$ clusters is as *small as possible*; the optimization problem for $K$-means is as follows:

$$\min_{C_1,\ldots,C_k} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{P} \left(x_{ij} - x_{i'j}\right)^2 \right\}$$

# K-Means Clustering

To find the global minimum of the above function is very difficult.

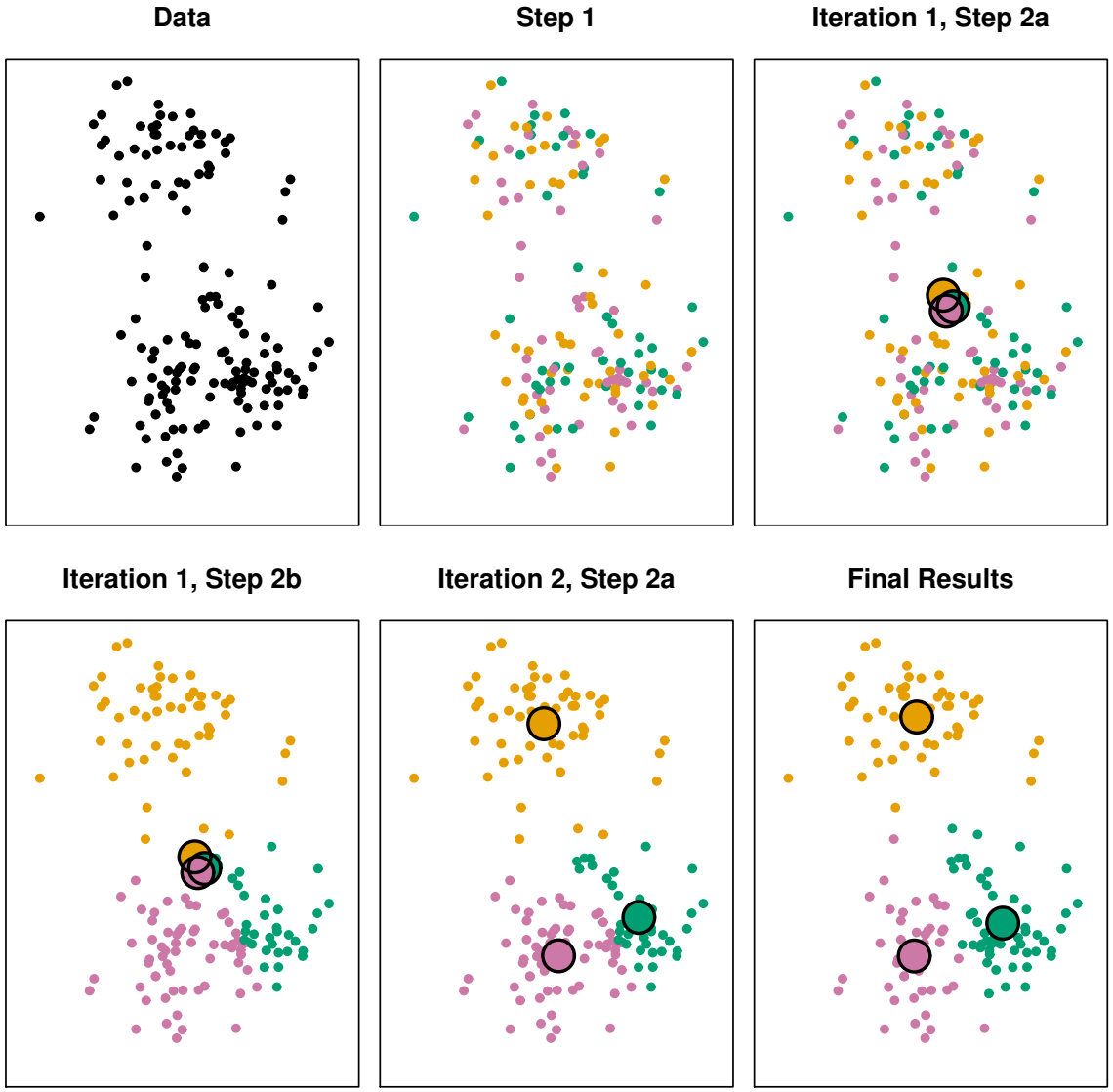
In practice, most K-means packages perform the following greedy algorithm, also known as Lloyd algorithm in the computer science circle:

1. Randomly assign an integer label, from 1 to $K$ (where $K$ is the number of clusters), to each of the observations. These serve as initial cluster assignments for the observations.

2. Iterate until the cluster assignments stop changing:
   1) For each of the K clusters, compute the cluster's new centroid.
   2) Assign each observation to the cluster whose centroid is closest (closest is measured using Euclidean distance).

# K-Means Clustering

The algorithm in action (cited from ISLR):



# K-Means Clustering

The K-means procedure always ***converges***:

- If you run the algorithm from a fixed initial assignment, it will reach a stable endpoint where the clustering solution will no longer change through the iterations.

Unfortunately, the guaranteed convergence is to a ***local minimum***.

- Thus, if we begin the K-means algorithm with a different initial configuration, it is possible that convergence will find different centroids and therefore ultimately assigning different cluster memberships.

What can we do to get around this?

- Run the K-means procedure *several times* and pick the clustering solution that yields the ***smallest aggregate within-cluster variance***.

# K-Means Clustering

The Kmeans++ (2007) improves the random seeding of the original KMeans.

- The initialization step runs inductively.

- Firstly, pick a data point randomly as the first centroid.

- Suppose that k of the seed centroid have been chosen, compute for each data point x the distance D(x) to the closest centroid among these k seed centroids. Select the **(k+1)th** centroid randomly, according to a probability distribution with probability proportional to $D(x)^2$.

- In each inductive step, the newly found seed centroid trends to keep a far distance from the existing ones.

The default initialization scheme of Scikit-Learn's KMeans uses KMeans++.

# K-Means Clustering

The algorithm will stop when it has found the least/best (local optimum) value.

# Hierarchical Clustering (Agglomerative Clustering).

K-means clustering algorithms require: (1) the choice of the number of classes to be clustered, (2) a starting cluster configuration assignment.

- In most cases, this is hard to determine.

*Hierarchical clustering* method is another popular clustering method which seeks to build a *hierarchy* of clusters.

- It does not require the user to specify the number of clusters. Instead, it requires to measure the **dissimilarity** between the pairs of clusters.

- The clusters at a higher level are created by merging clusters at the lower level:
  - At the lowest level, each cluster contains a single observations.
  - At the highest level, all of the data points from a single cluster.

# Hierarchical Clustering (Agglomerative Clustering).

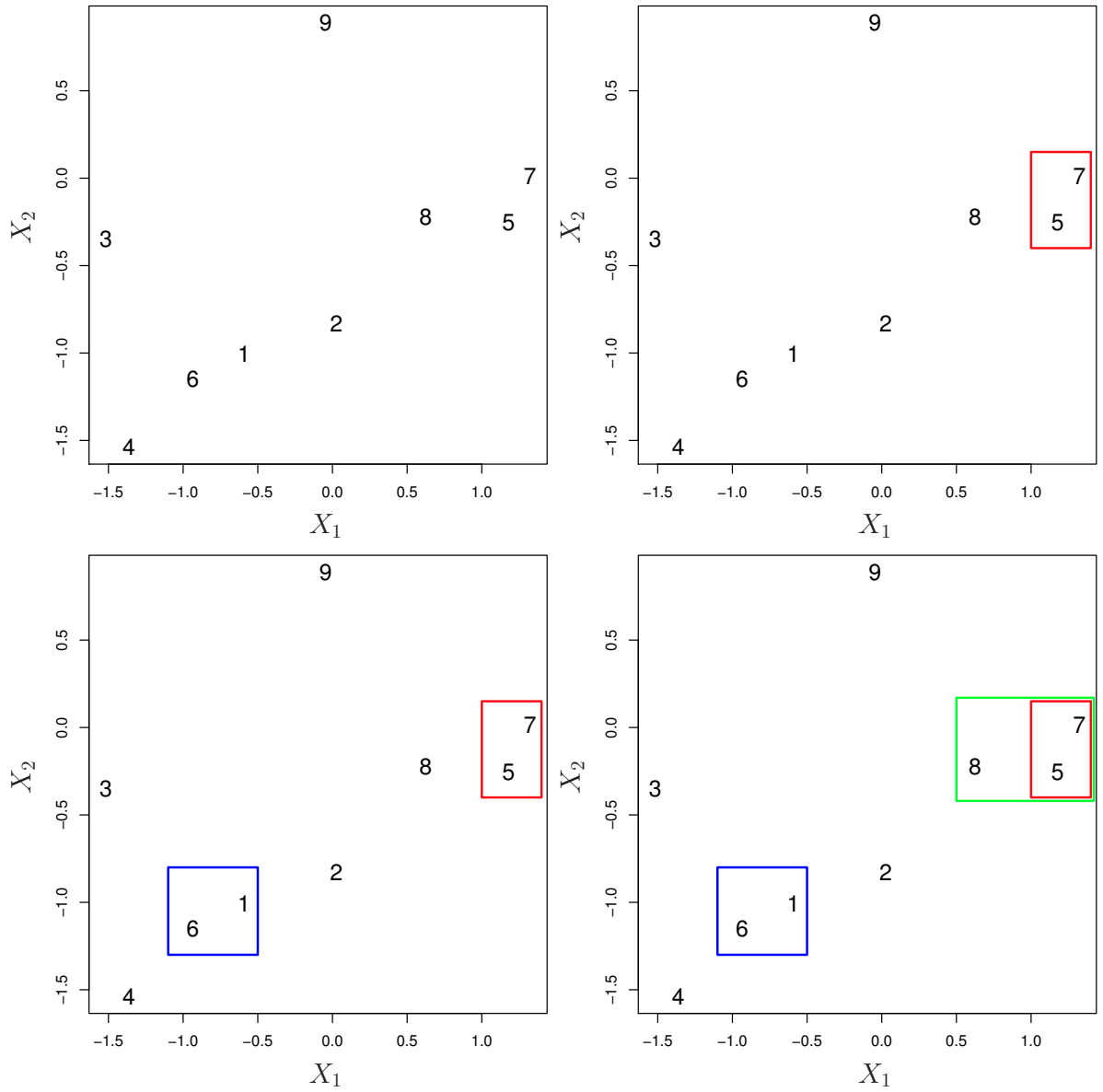
Two strategies for hierarchical clustering: bottom-up and top-down.

The approach can be summarized as:

- Start with each point in its own cluster.
- Identify the two clusters which are most similar and merge them.
- Repeat step 2.
- Ends when all data points are in a single cluster.

In the next few slides we show how to build a hierarchy in a bottom-up fashion.

# Hierarchical Clustering (Agglomerative Clustering).



# Hierarchical Clustering

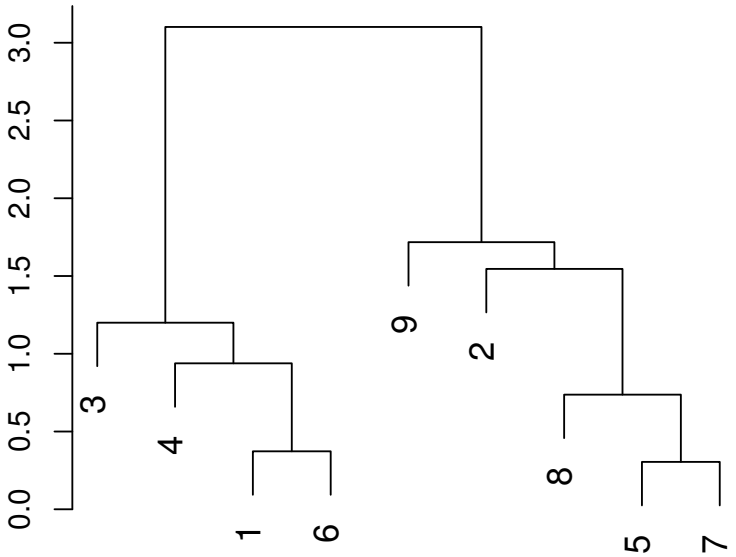
There are some interpretative advantages to visualizing the dendrogram created from hierarchical clustering:

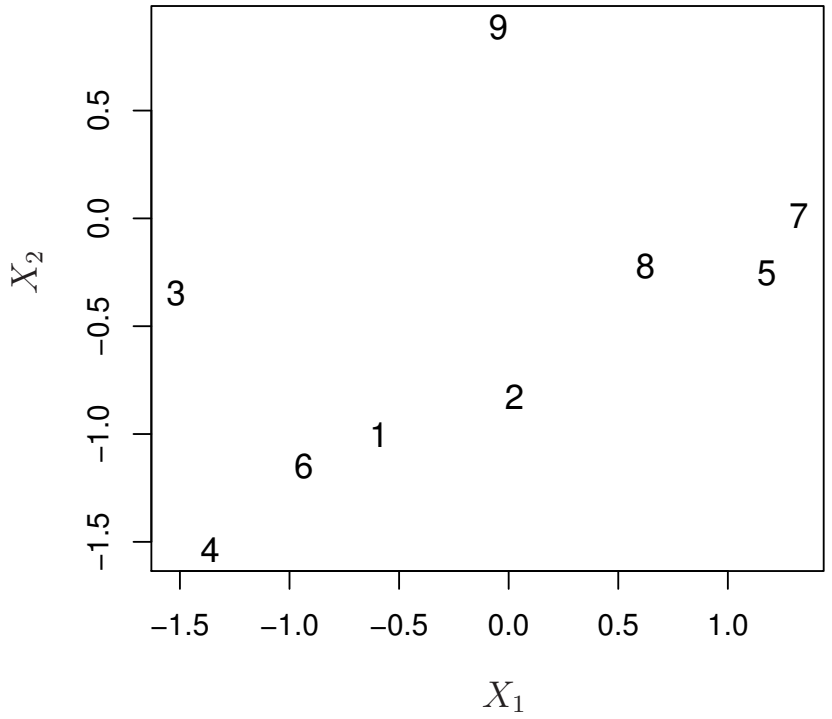
- The lower down in the dendrogram a cluster fusion occurs, the more similar the fused clusters are to each other.
- The higher up in the dendrogram a fusion occurs, the more dissimilar the fused groups are to each other.

In general, for any two observations we can inspect the dendrogram and find the location at which the groups that contain those two observations are fused together to get an idea of their dissimilarity.

- Be careful to consider the groups of points in the fusions within the dendrograms, not just individual points.

# Hierarchical Clustering (Agglomerative Clustering).



# Hierarchical Clustering Algorithm

Begin with $n$ observations and a distance measure of all pairwise dissimilarities. At this step, treat each of the $n$ observations as their own clusters.

For $i = n, (n-1), \ldots, 2$:

a. Evaluate all pairwise inter-cluster dissimilarities among the $i$ clusters and fuse together the pair of clusters that are the least dissimilar.
b. Note the dissimilarity between the recently fused cluster pair and mark that as the associated height in the dendrogram.
c. Repeat the process in step a, calculating the new pairwise inter-cluster dissimilarities among the remaining ($i$ - 1) clusters.

# Hierarchical Clustering Algorithm

While we do not need to specify $K$ a priori, in order to perform hierarchical clustering there are a few choices we need to make. Particularly:

- A dissimilarity measure.
- A linkage method.

We're already familiar with the idea of choosing a dissimilarity measure with the choice of distance metric. In many cases, it is sufficient to use the Euclidean distance.

A linkage is a measure of the dissimilarity between two group of points. So far we only define the distance between two points, but what do we do when we want to assess the similarity among two groups of points?

# Hierarchical Clustering Algorithm: Linkage

The most common types of linkage are described below.

First, compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B. Then:

- *Complete Linkage*: Maximal inter-cluster dissimilarity.
  - Record the largest of the dissimilarities listed between members of A and of B as the overall inter-cluster dissimilarity.

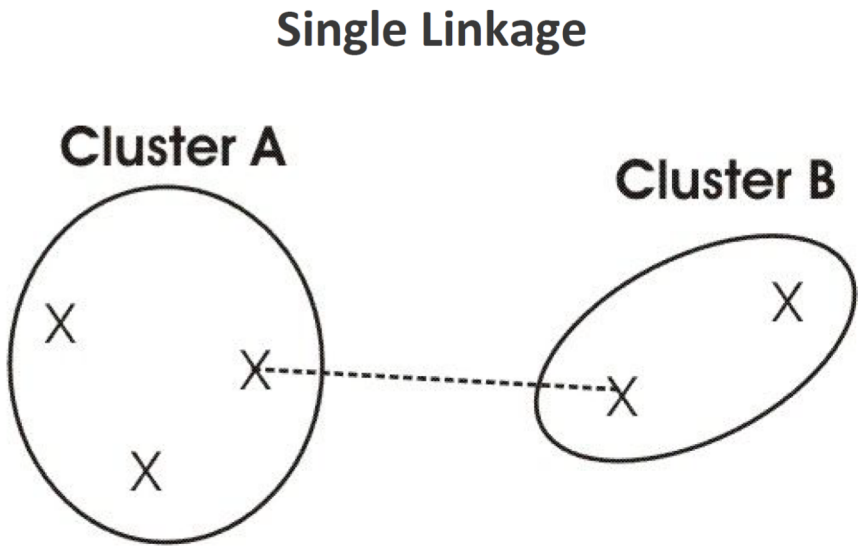
- *Single Linkage*: Minimal inter-cluster dissimilarity.
  - Record the smallest of the dissimilarities listed between members of A and of B as the overall inter-cluster dissimilarity.

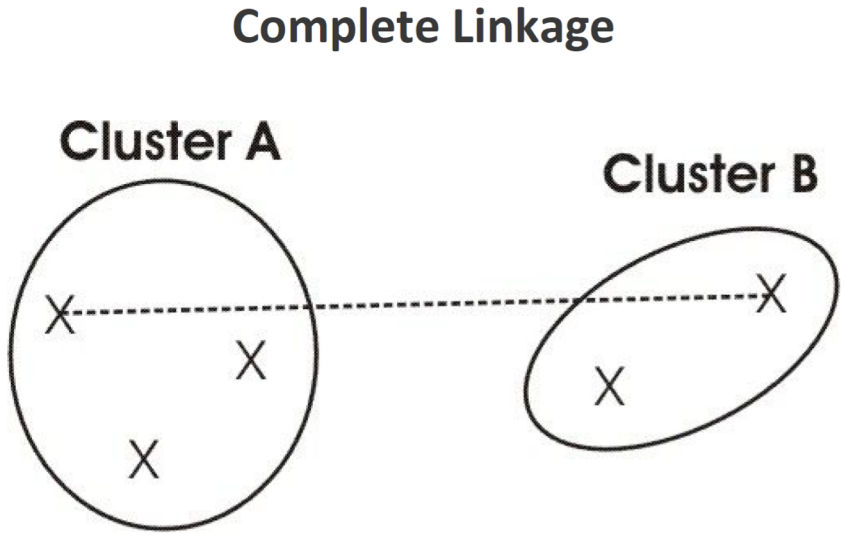
# Hierarchical Clustering Algorithm: Linkage

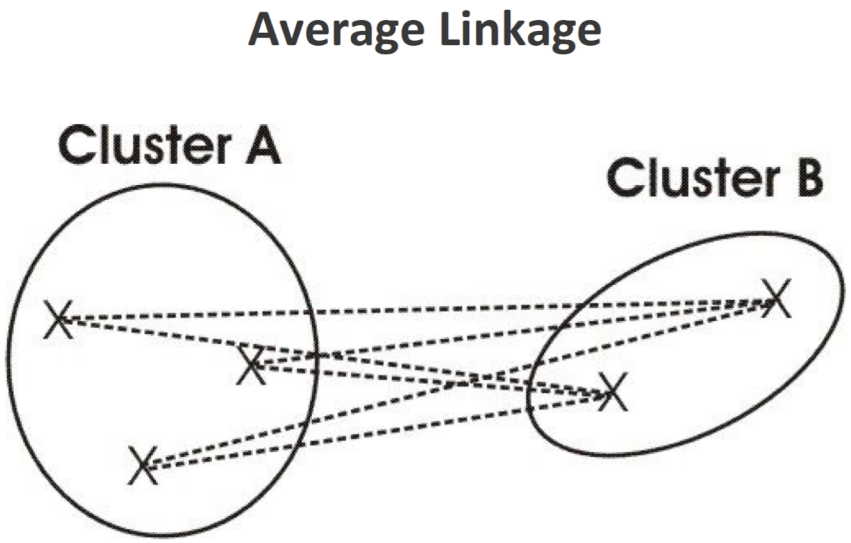
- *Average Linkage*: Mean inter-cluster dissimilarity.
  - Record the average of the dissimilarities listed between the members of A and of B as the overall inter-cluster dissimilarity.

- *Ward's Linkage*: Minimum variance method (for Euclidean distance).
  - Minimize the variance of the clusters being merged.

# Hierarchical Clustering Algorithm: Linkage




- *Complete Linkage*: Maximal inter-cluster dissimilarity.
- *Single Linkage*: Minimal inter-cluster dissimilarity.
- *Average Linkage*: Mean inter-cluster dissimilarity.

# Hierarchical Clustering Algorithm: Linkage

*Complete linkage* is sensitive to outliers, yet it tends to identify clusters that are compact, somewhat spherical objects with relatively similar diameters.

*Single linkage* is not as sensitive to outliers, yet tends to identify clusters that have a chaining effect; these clusters often fail to represent intuitive groupings among our data, and the observations in the same cluster might be quite distant from one another.

*Average linkage* tends to strike a balance between the pros and cons of complete linkage and single linkage.