UNIVERSITY OF BIRMINGHAM

Week 10: K-Means and Hierarchical Clustering

Lecturer: Dr. Sharu Theresa Jose

Learning Outcomes

- Understand principles of K-means and Hierarchical Clustering algorithms
- Learn to apply the algorithms to clustering problems
- Understand the challenges

Overview of Lecture

- Recap: Partitional Clustering as Optimization Problem
- K-Means Algorithm
  - Introduction and Examples
  - Challenges and Solutions
  - Application – Vector Quantization
- Hierarchical Clustering
  - Agglomerative Hierarchical Clustering
  - Inter-Cluster Dissimilarity Metrics
  - Characteristics of Hierarchical Clustering

Partitional Clustering

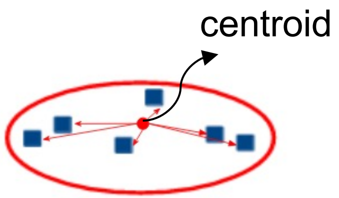
- Goal: assign N observations into K (K<N) clusters to ensure high intra-cluster similarity and low inter-cluster similarity
- Can be formulated as a combinatorial optimization problem.
- Notation:
    - $\boldsymbol{C}$ denotes a clustering structure with K clusters
    - $\mathrm{C} \in \boldsymbol{C}$ denotes a component cluster
    - n_C denotes the number of examples in cluster C
    - $e \in C$ denotes an example in cluster
    - $c_k \in C_k$ denotes the centroid of the kth cluster

UNIVERSITY OF BIRMINGHAM

Measure of intra-cluster similarity

Variability (or Inertia) of a cluster C:

$$variability(C) = \sum_{e \in C} d\big(e, centroid(C)\big).$$



- Commonly used distance measure: squared Euclidean distance, i.e., $d(\boldsymbol{a}, \boldsymbol{b}) = d_{Euc}(\boldsymbol{a}, \boldsymbol{b})^2$.
- Centroid of a cluster is usually taken as the average of all examples in the cluster i.e.,

$$centroid(C) = \frac{attribute\text{−}wise\ sum\ of\ examples\ in\ the\ cluster}{number\ of\ examples\ in\ the\ cluster}$$

  Example: If (a, b) and (c, d) are two examples in a cluster, the cluster centroid is ((a+c)/2, (b+d)/2).

- Variability determines how compact the cluster is.

UNIVERSITY OF BIRMINGHAM

- **Dissimilarity or Within Cluster Sum of Squares** (WCSS) of a clustering structure $\boldsymbol{C}$:

$$dissimilarity(\boldsymbol{C}) = \sum_{C \in \boldsymbol{C}} variability(C)$$

- Optimization problem: Find a clustering structure $\boldsymbol{C}$ of K<N clusters that minimizes the following objective:

$$\min_{\boldsymbol{C}} dissimilarity\ (\boldsymbol{C})$$

  - Larger clusters with high variability are penalized more than smaller clusters with high variability.
  - Under squared Euclidean distance, minimizing $dissimilarity(\boldsymbol{C})$ is equivalent to maximizing overall inter-cluster dissimilarity.

Minimizing WCSS or dissimilarity of a clustering structure is equivalent to maximizing the inter-cluster dissimilarity.

$$\sum_{e} d_{Euc}(e, centroid(data))^2 =$$

Total Sum of Squares (TSS)

$$\underbrace{\sum_{C \in \boldsymbol{C}} \sum_{e \in C} d_{Euc}(e, centroid(C))^2}_{\text{WCSS or Dissimilarity}} + \underbrace{\sum_{C \in \boldsymbol{C}} n_C d_{Euc}(centroid(C), centroid(data))^2}_{\text{Between Cluster Sum of Squares (BCSS)}}$$

- TSS does not depend on the clustering structure, and is thus a constant.
- BCSS: accounts for inter-cluster dissimilarity
- WCSS and BCSS depend on the clustering structure.
- Since WCSS+BCSS = a constant, minimizing WCSS is equivalent to maximizing BCSS.

- Finding exact solution of the optimization problem is prohibitively hard.
    - Infeasible when large number of examples present
- Solution: Iterative Greedy Algorithms
    - Provide a sub-optimal approximate solution
    - Includes K-means, K-medoids

K-means

- Iterative greedy descent algorithm that finds a sub-optimal solution to

$$\min_{\boldsymbol{C}} dissimilarity\,(\boldsymbol{C}) = \min_{\boldsymbol{C}} \sum_{C \in \boldsymbol{C}} \sum_{e \in C} d_{Euc}\big(e, centroid(C)\big)^2$$

- K-means iteratively alternates between the following two steps:
  - **Assignment step**: For given set of K cluster centroids, K-means assigns each example to the cluster with closest centroid.
    - fix centroids and optimize cluster assignments (optimizes the red highlighted part).
  - **Refitting step**: Re-evaluate and update the cluster centroids, i.e., for fixed cluster assignment, optimize the centroids

UNIVERSITY OF BIRMINGHAM

K-Means Algorithm

Input: Number K of clusters and N examples $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(N)}$

1. Select K examples as centroids $c_1, \dots, c_K$
2. Repeat until cluster centroids do not change:

   3. (assignment step) Form K clusters by assigning each observation to its closest cluster centroid, i.e.,

$$Cluster(i) = \arg\min_{k=1,\dots,K} d_{Euc}\left(\boldsymbol{x}^{(i)}, c_k\right)^2 \quad for\ i = 1, \dots, N$$

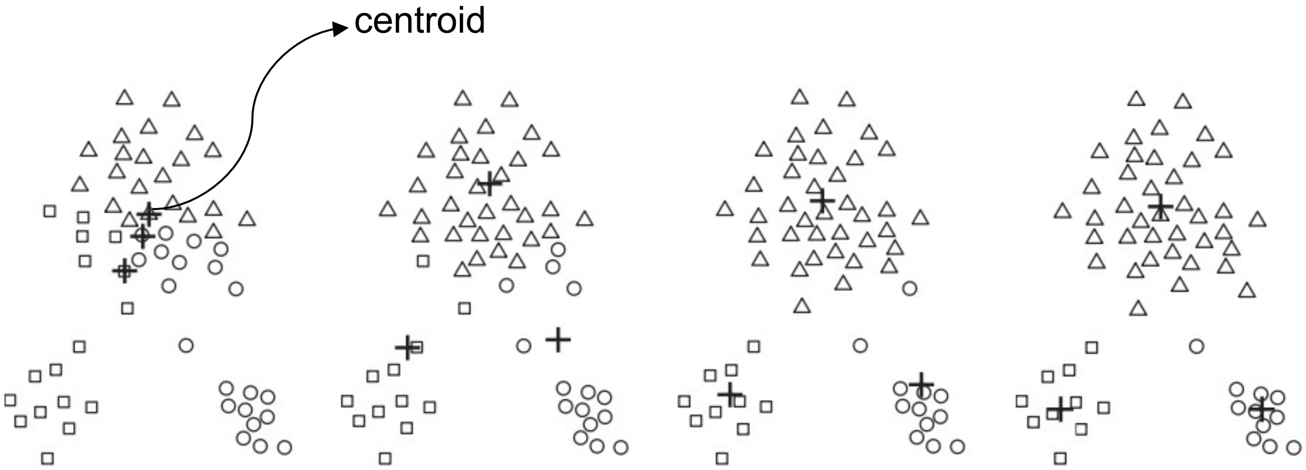
   4. (refitting step) Compute the centroid of the obtained K clusters as

$$c_k = \frac{1}{n_k} \sum_{i:Cluster(i)=k} \boldsymbol{x}^{(i)}, \quad for\ k = 1, \dots K$$

where n_k is the total number of examples in the k^{th} cluster.

UNIVERSITY OF BIRMINGHAM

Illustration



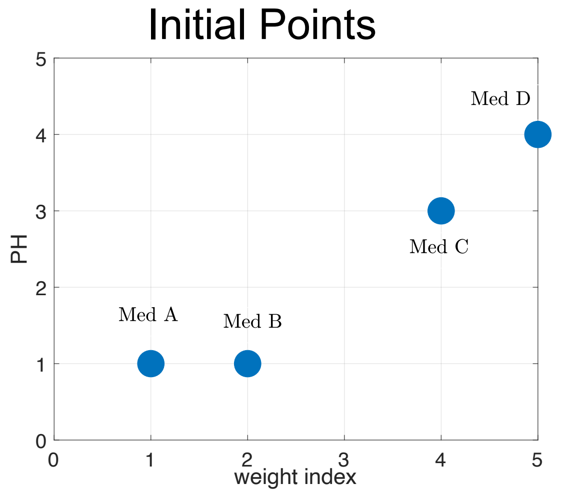
(a) Iteration 1. (b) Iteration 2. (c) Iteration 3. (d) Iteration 4.

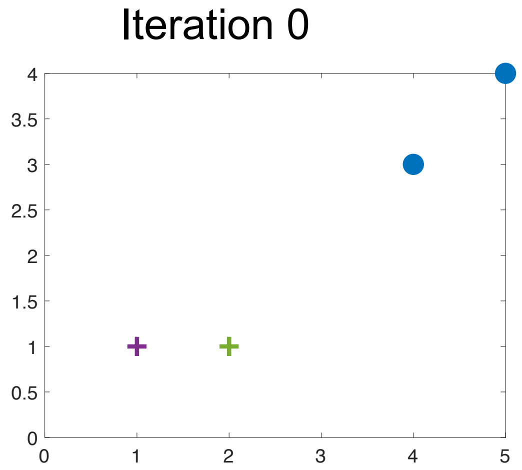
Observations:

1. Cluster centroids need not be examples in later iterations.

Example 1: Clustering of Medicines (K=2)

	Weight index	PH
Med A	1	1
Med B	2	1
Med C	4	3
Med D	5	4





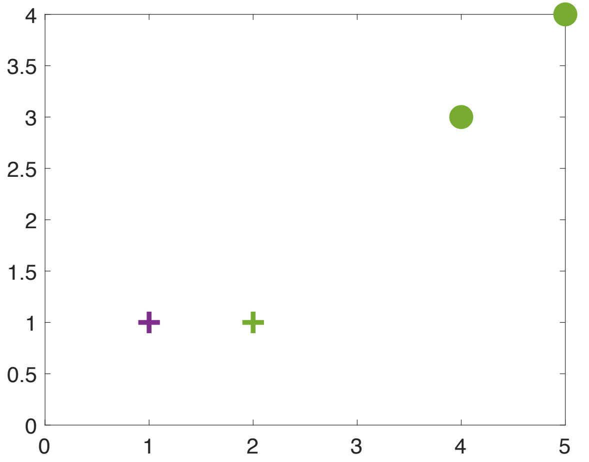
Iteration 0: Initial centroids be Med A and Med B i.e, $c_1 = (1,1)$ and $c_2 = (2,1)$

Iteration 1:

1. Calculate (Euclidean) distance of each point to cluster centroids to form an Object-Centroid Distance Matrix:

	Med A	Med B	Med C	Med D
c_1	0		13	25
c_2		0	8	18

$$d_{Euc}(Med\ C, c_1)^2 = (4-1)^2 + (3-1)^2 = 13$$
$$d_{Euc}(Med\ C, c_2)^2 = (4-2)^2 + (3-1)^2 = 8$$
$$d_{Euc}(Med\ D, c_1)^2 = (5-1)^2 + (4-1)^2 = 25$$
$$d_{Euc}(Med\ D, c_2)^2 = (5-2)^2 + (4-1)^2 = 18$$

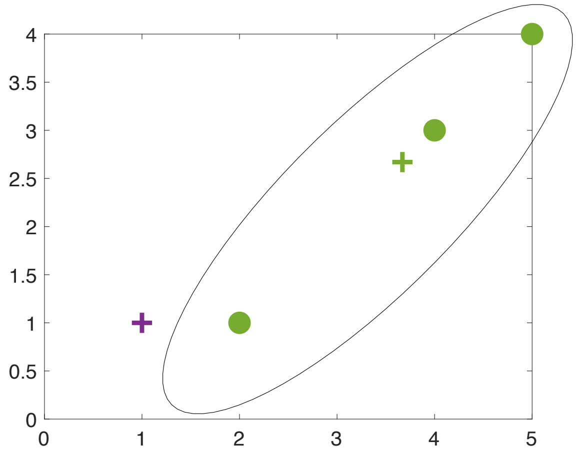


Thus, Medicines B, C and D assigned to Cluster 2.

- 2. Update the centroids of the cluster.

$$c_1 = c_1, \quad c_2 = \frac{Med\ B + Med\ C + Med\ D}{3}$$

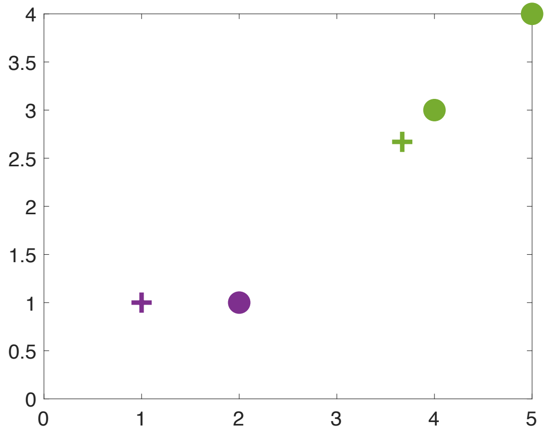
$$= \left(\frac{2+4+5}{3}, \frac{1+3+4}{3}\right) = (3.67, 2.67)$$



Iteration 2:

1. Calculate distance of each point to new cluster centroids.

	Med A	Med B	Med C	Med D
c_1	0	1	13	25
c_2	9.92	5.56	0.22	3.53

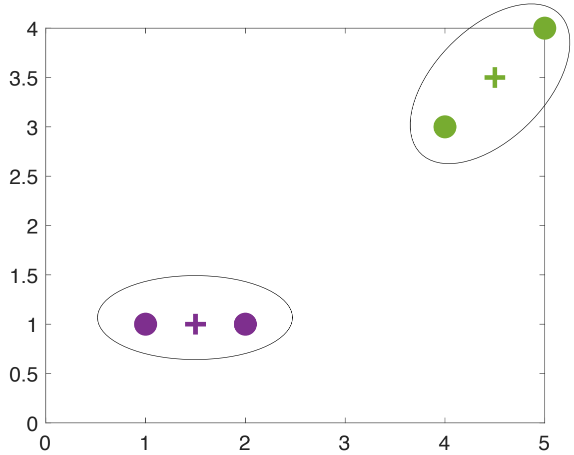


Med B is thus moved to cluster 1.

2. Update the centroids of the cluster.

$$c_1 = \frac{Med\ A + Med\ B}{2} = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = (1.5,1)$$

$$c_2 = \frac{Med\ C + Med\ D}{2} = \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = (4.5,3.5)$$



UNIVERSITY OF BIRMINGHAM

- Repeat the same steps in iteration 3
- Note that cluster assignments do not change
- Algorithm converge.

Space and Time Complexity of K-Means

- Space requirement for K-means is modest because only data observations and centroids are stored
- Storage complexity is of the order $O((N+K)m)$, where m is the number of feature attributes
- Time complexity of K-means: $O(I*K*N*m)$ where I is the number of iterations required for convergence
- Importantly, time complexity of K-means is linear in N.

UNIVERSITY OF BIRMINGHAM

Challenges and Issues in K-Means

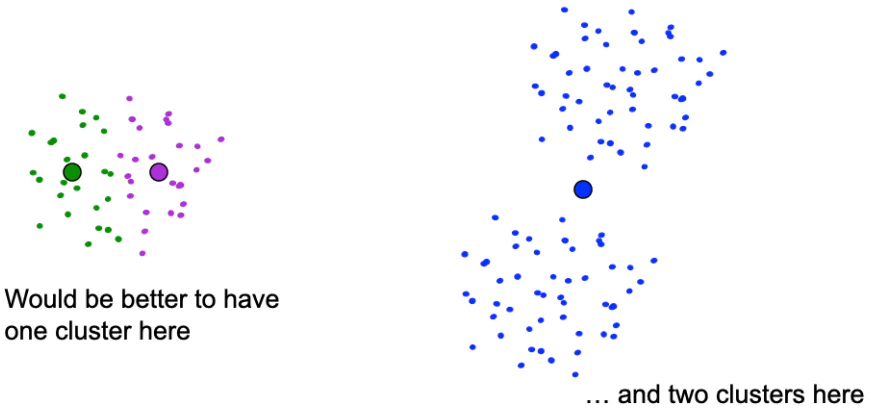
K-Means Questions?

- Does the K-means algorithm always converge?
- Can it always find optimal clustering?
- How should we start the algorithm?
- How should we choose the number of clusters?

Convergence of K-Means

- At each iteration, the assignment and refitting steps ensure that the objective function (1) monotonically decreases.
- Also, K-means work with finite partitions of the data.
- The above two conditions ensure that the K-Means algorithm always converge (i.e., cluster assignments do not change)
- However, the objective function (1) is non-convex. As such, K-Means algorithm may converge to a local minimum and not global minimum.

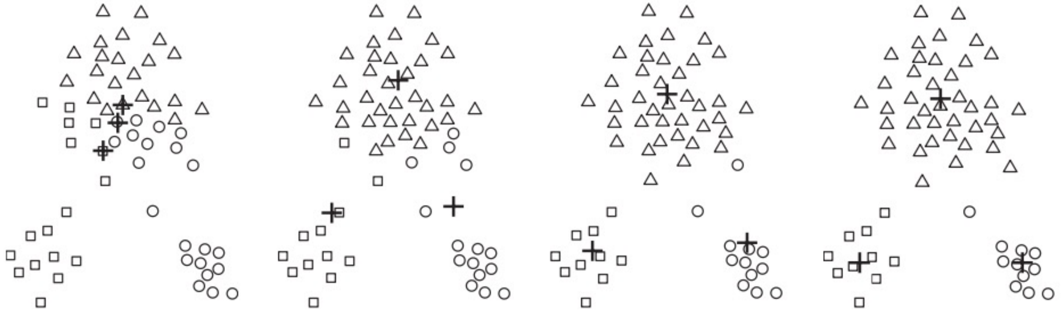
A local optimum:



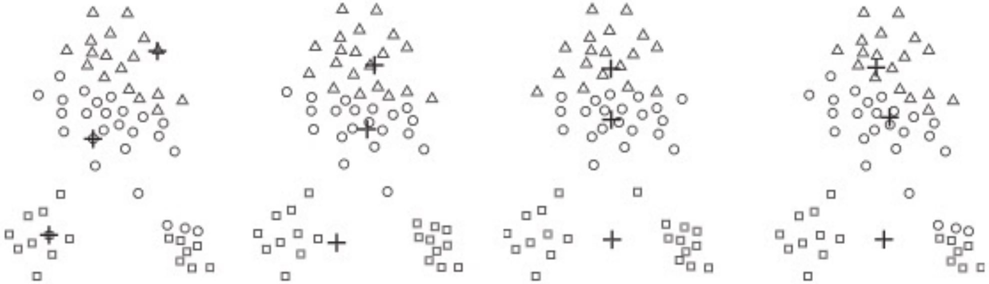
Escaping local minima: multiple random restarts and choose the best clustering result (i.e., the clustering that yields lowest dissimilarity)

Choice of Initial Cluster Centroids

- Choosing initial cluster centroids is crucial for K-means algorithm.
- Different initializations may lead to convergence to different local optima.
- K-means is a non-deterministic algorithm.



(a) Iteration 1. (b) Iteration 2. (c) Iteration 3. (d) Iteration 4.



(a) Iteration 1. (b) Iteration 2. (c) Iteration 3. (d) Iteration 4.

UNIVERSITY OF BIRMINGHAM

Solutions:

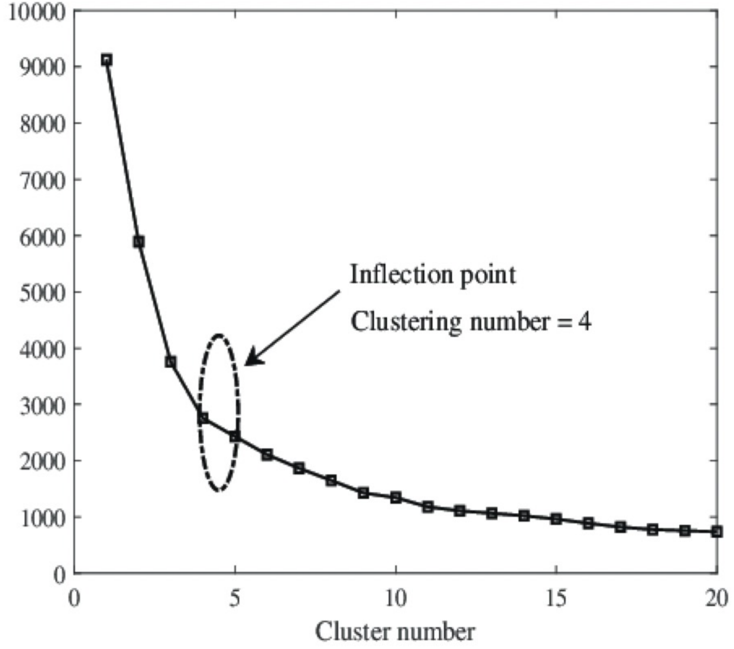
- Run multiple K-means algorithm starting from randomly chosen cluster centroids. Choose the cluster assignment that has the minimum dissimilarity.
- Specialized initialization strategies: K-means++
  - Choose first centroid at random.
  - For each data point $\boldsymbol{x}$, compute its distance $\text{dist}(\boldsymbol{x})$ from the nearest centroid.
  - Choose a data point $\boldsymbol{x}$ randomly with probability proportional to $\text{dist}(\boldsymbol{x})^2$ as the next centroid.
  - Continue until K cluster centroids are obtained.
  - Use the obtained K centroids as initial centroids for the K-means algorithm

Choice of the Number of Clusters K

- Conventional approach: use prior domain knowledge

  Example: data segmentation – a company determines the number of clusters into which its employees must be segmented

- A data-based approach for estimating the optimal number K* of clusters: **Elbow method**
  - Apply K-means algorithm multiple times with different number of clusters.
  - Evaluate the quality of the obtained clustering structure in each run of the algorithm using the metric $dissimilarity(\boldsymbol{C})$.
  - As the number of clusters increases, $dissimilarity(\boldsymbol{C})$ tends to decrease.
  - Plot $dissimilarity(\boldsymbol{C})$ as a function of the number K of clusters.
  - Optimal K* lies at the elbow of the plot.



Elbow criterion:

- Marginal gain in the objective may decrease at true/natural value of K
- Not always ambiguously defined.

UNIVERSITY OF BIRMINGHAM

Application in Image Compression

Vector Quantization

(Left Photo): 1024 x 1024 pixels

each pixel is a greyscale value ranging from 0 to 255

Storage: 8bits per pixel, 1 megabyte of storage

Vector quantization: break image into small blocks of 2x2 to get 512 x 512 blocks of 4 numbers in R^4

UNIVERSITY OF BIRMINGHAM

- K-means clustering algorithm is run on the space of 4-dimensional real numbers. The algorithm returns the collection of cluster centroids called the **codebook.** The clustering process is called encoding.
- Now, each of the 512 x 512 pixel blocks is approximated by its closest cluster centroid.
- The process of constructing an approximate image from the centroids = decoding
- Center figure: K=200 and Right figure: K=4
- Storage for compressed images= $\log_2(K)/4$ bits per pixel

Summary of K-means

Properties:

- Optimizes a global objective function
- Squared Euclidean distance based
- Non-deterministic

Challenges:

- Requires as input: number of clusters and an initial choice of centroids
- Convergence to local minima implies multiple restarts

Hierarchical Clustering

Introduction

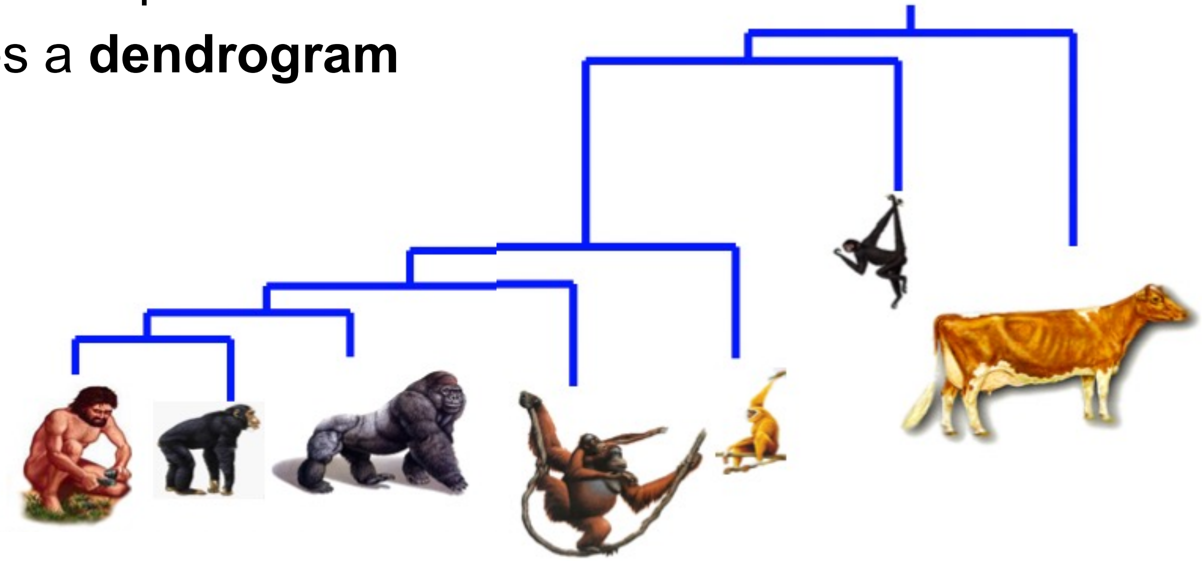
- Input to K-means algorithm: Number K of clusters and an initial choice of cluster centroids

- Hierarchical clustering requires no such specifications

- Instead, user specifies a measure of similarity (or dissimilarity) between a pair of clusters

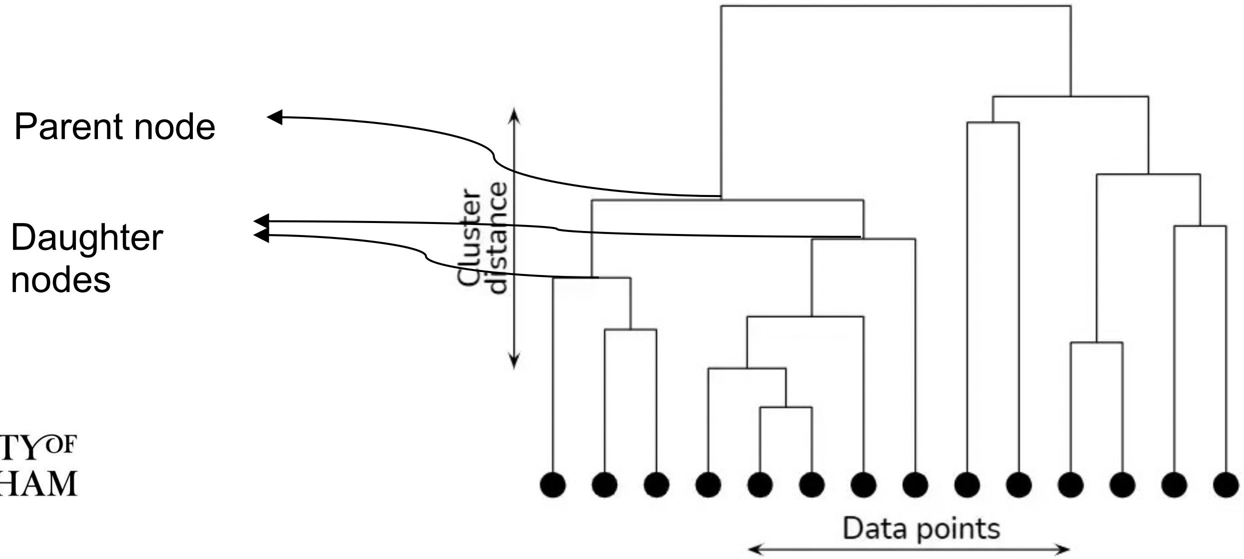
What is hierarchical clustering?

- Create a hierarchical decomposition of the set of examples using a user-specified criterion
- Produces a **dendrogram**



Dendrogram

- Highly interpretable complete description of the hierarchical clustering in a graphical format
- Representation of hierarchical clustering as a rooted binary tree
- Nodes of the trees represent clusters



Pic:Prasad pai

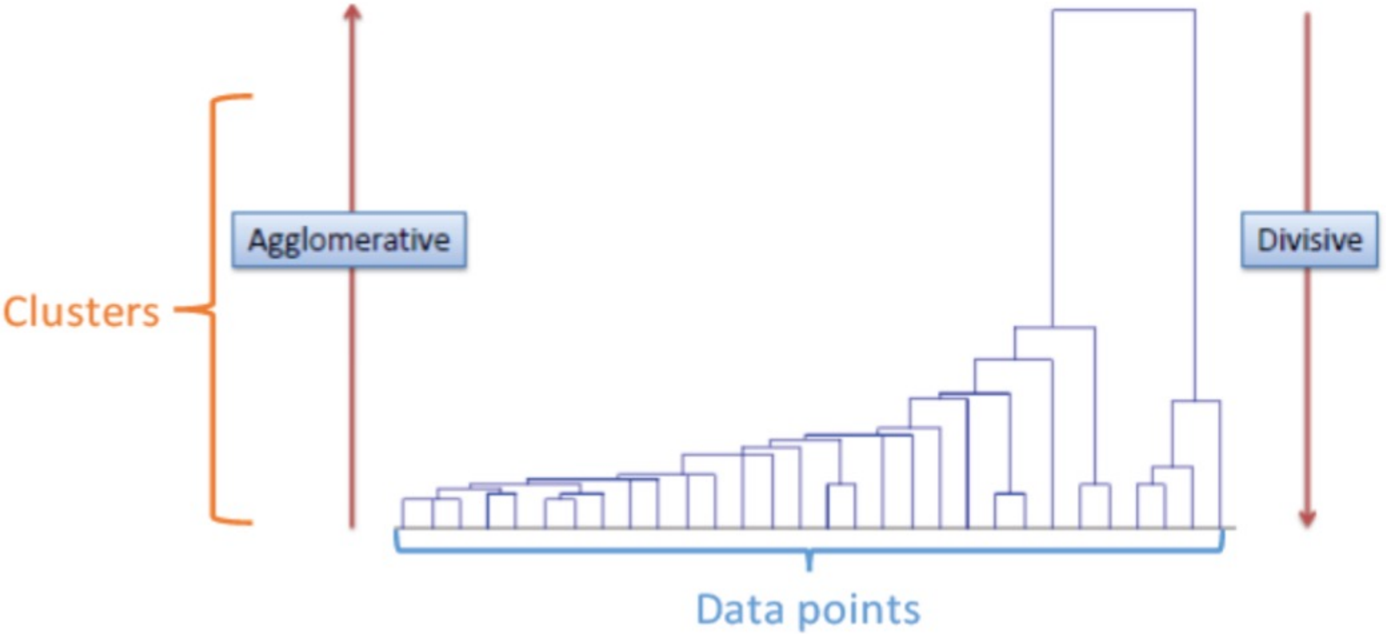
Strategies for Hierarchical Clustering

Agglomerative Clustering

- Bottom-up approach
- Starts at the bottom with each cluster containing a single observation
- At each level up, recursively merge pair of clusters with the smallest inter-cluster dissimilarity into a single cluster.
- A single cluster at the top level

Divisive Clustering

- Top-down approach
- Starts at the top with a single cluster of all observations
- At each level down, recursively split one of the existing clusters into two new clusters with the largest inter-cluster dissimilarity.
- At the bottom, each cluster contains single observation


Agglomerative
Divisive
Clusters
Data points

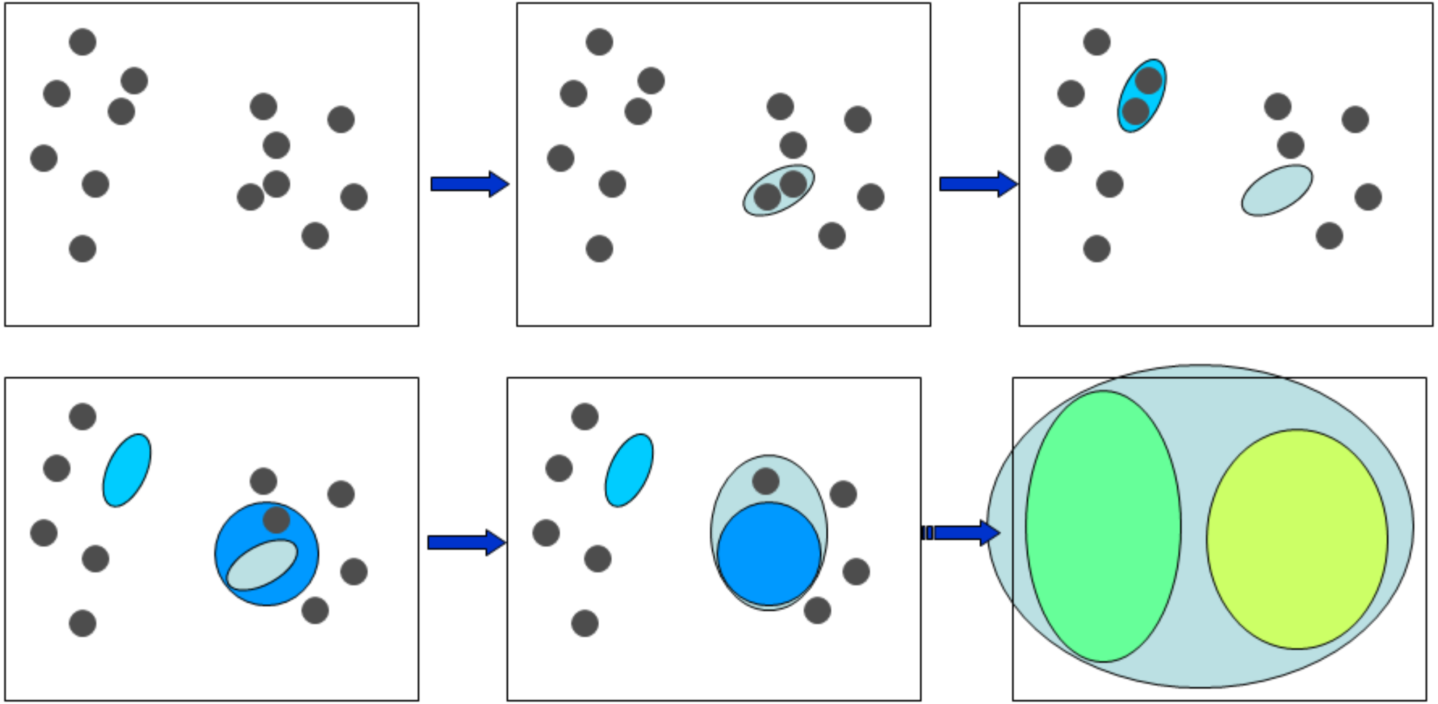
Agglomerative Hierarchical Clustering

Agglomerative Clustering Algorithm

1. Start with all data points in their own clusters.
2. Repeat until only one cluster remains:
   - Find 2 clusters C_1, C_2 that are most similar (i.e., that have the smallest inter-cluster dissimilarity $d(C_1, C_2)$)
   - Merge C_1, C_2 into one cluster

Output: a dendrogram
Reply on: an inter-cluster dissimilarity metric

Agglomerative clustering illustration

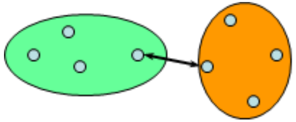


UNIVERSITY OF BIRMINGHAM

Measures of Inter-Cluster Dissimilarity

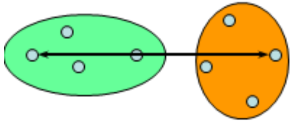
- Single linkage
  - Shortest distance from any member of the cluster to any member of the other cluster

$$d_{SL}(C_1, C_2) = \min_{i \in C_1, j \in C_2} d(i,j)$$



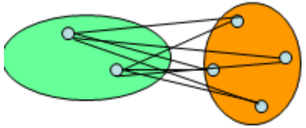
- Complete linkage
  - Largest distance from any member of the cluster to any member of the other cluster

$$d_{CL}(C_1, C_2) = \max_{i \in C_1, j \in C_2} d(i,j)$$



- Group average
  - Average of distances between members of the two clusters

$$d_{GA}(C_1, C_2) = \frac{1}{n_{C_1} n_{C_2}} \sum_{i \in C_1, j \in C_2} d(i,j),$$

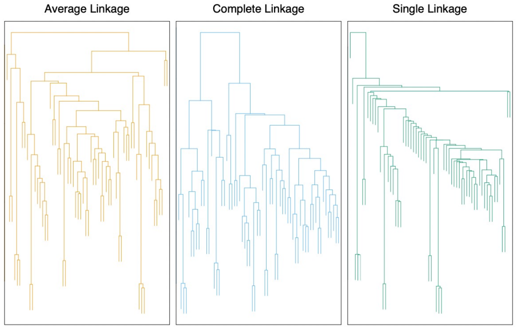


where n_{C_1}, n_{C_2} are the number of examples in cluster C_1, C_2 respectively.

UNIVERSITY OF BIRMINGHAM

More details….

- Does the choice of inter-cluster dissimilarity measure matter?
  - Yes !!!
  - Yields similar results when the (natural) clusters are compact and well-separated



- Single linkage:
  - Determined by the pair of examples in the two clusters that are the closest; other dissimilarities between examples in the groups do not matter
  - Chaining effect = tendency to combine examples linked by a series of close intermediate examples
  - Sensitive to outliers
  - Results in clusters that are not compact: single linkage can produce clusters with large diameter, i.e., $diam(C_1) = \max_{i,j \in C_1} d(i,j)$ is large

UNIVERSITY OF BIRMINGHAM

- Complete Linkage
  - Requires all examples in the two clusters to be relatively similar
  - Produces compact clusters with small diameters
  - Robust to outliers
  - However, members can be closer to other clusters than they are to members of their own clusters
- Group Average Linkage
  - Attempts to produce relatively compact clusters that are relatively far apart
  - Depends on the numerical scale on which the distances are measured

Example 1: Clustering of European cities based on air distance

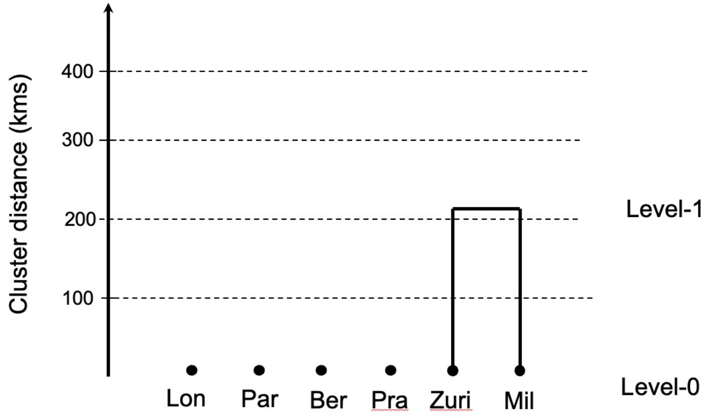
	Lond	Paris	Berlin	Praha	Zurich	Milan
Lond	0	393	932	1027	776	958
Paris		0	878	883	489	641
Berlin			0	279	650	795
Praha				0	528	401
Zurich					0	204
Milan						0

Given the distance matrix, obtain a dendrogram using single-linkage as intra-cluster dissimilarity metric.

- Level-0:
  - Clusters: {(Lond), (Paris), (Ber), (Praha), (Zurich),(Milan)} (6 clusters)

- Level -1:
  - Clusters {Milan} and {Zurich} have the smallest inter-cluster dissimilarity based on single-linkage.
  - Clusters: {(Lond), (Paris), (Ber), (Praha), (Zurich, Milan)}. (5 clusters)

	Lond	Paris	Berlin	Praha	Zurich	Milan
Lond	0	393	932	1027	776	958
Paris		0	878	883	489	641
Berlin			0	279	650	795
Praha				0	528	401
Zurich					0	204
Milan						0



Height at which two clusters merge corresponds to their inter-cluster dissimilarity distance.

- Level-2: Update distance matrix

	Lond	Paris	Berlin	Praha	{Zur, Milan}
Lond	0	393	932	1027	
Paris		0	878	883	
Berlin			0	279	
Praha				0	
{Zur, Milan}					0

$d_{SL}(Lon, \{Zur, Milan\})$
$= \min\{ d(Lon, Zur), d(Lon, Milan)\}$
$= \min\{ 776{,}958\} = 776.$

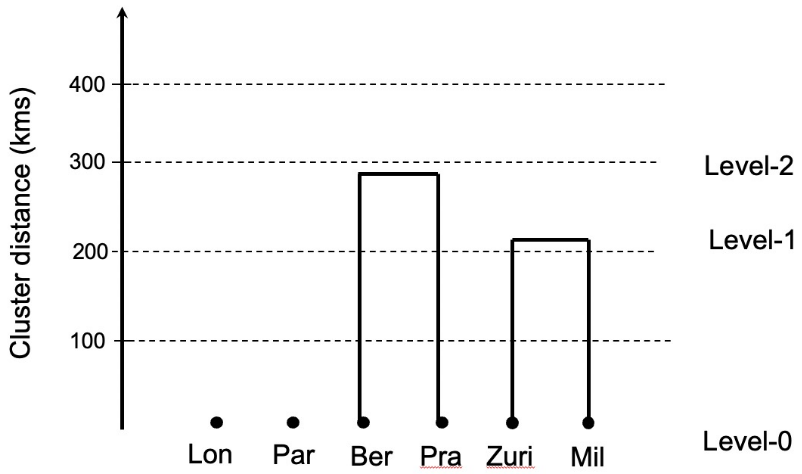
$d_{SL}(Paris, \{Zur, Milan\})$
$= \min\{ d(Paris, Zur), d(Paris, Milan)\}$
$= \min\{ 489{,}641\} = 489.$

$d_{SL}(Berlin, \{Zur, Milan\})$
$= \min\{ d(Berlin, Zur), d(Berlin, Milan)\}$
$= \min\{ 650{,}795\} = 650.$

$d_{SL}(Praha, \{Zur, Milan\})$
$= \min\{ d(Praha, Zur), d(Praha, Milan)\}$
$= \min\{ 528{,}401\} = 401.$

- Level-2: Update distance matrix

	Lond	Paris	Berlin	Praha	{Zur, Milan}
Lond	0	393	932	1027	776
Paris		0	878	883	489
Berlin			0	279	650
Praha				0	401
{Zur, Milan}					0



Clusters: {(Lond), (Paris), (Berlin, Praha), (Zur,Milan)} (4 clusters)

- Level 3: Update distance matrix

	Lond	Paris	{Berlin, Praha}	{Zur, Milan}
Lond	0	393	932	776
Paris		0	878	489
{Berlin, Praha}			0	401
{Zur,Milan}				0

$$d_{SL}(Lon, \{Ber, Praha\}) = 932$$

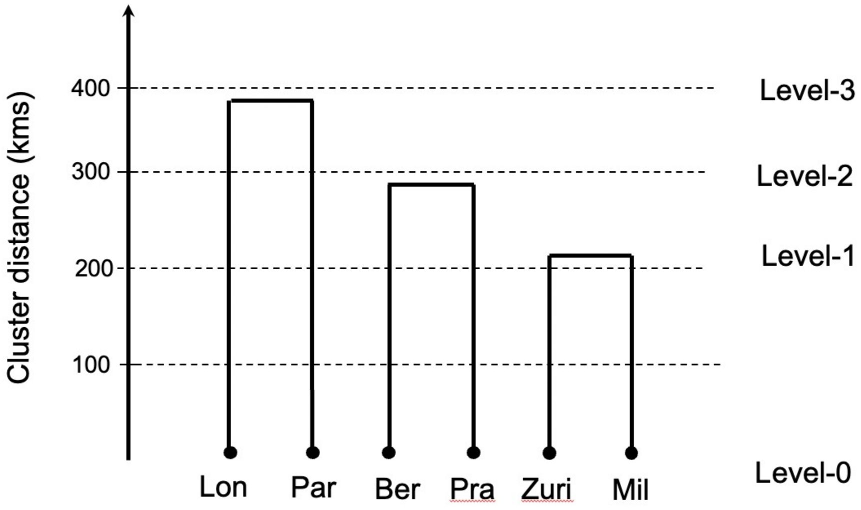
$$d_{SL}(Par, \{Ber, Praha\}) = 878$$

$$d_{SL}(\{Ber, Praha\}, \{Zur, Milan\})$$

$$= \min \begin{Bmatrix} d(Ber, Zur), d(Ber, Milan), \\ d(Praha, Zur), d(Praha, Milan) \end{Bmatrix}$$

$$= \min\{650, 795, 528, 401\} = 401$$

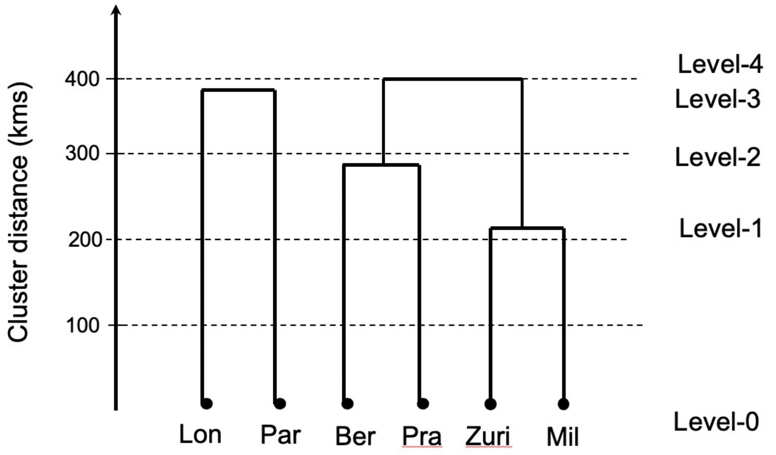
Clusters: {(Lond,Paris), (Berlin, Praha), (Zur,Milan)} (3 clusters)



UNIVERSITY OF BIRMINGHAM

- Level-4 : Update distance matrix

	{Lond,Paris}	{Berlin, Praha}	{Zur, Milan}
{Lond,Paris}	0	878	489
{Berlin, Praha}		0	401
{Zur,Milan}			0



Clusters: {(Lond,Paris), (Berlin, Praha, Zur, Milan)} (2 clusters)

Reading a Dendrogram

- Height at which two clusters merge corresponds to their inter-cluster dissimilarity distance.
- Possesses a monotonicity property, i.e., inter-cluster dissimilarity between merged clusters is monotone increasing with the level of the merger.
- Horizontally cutting dendrogram at a particular height partitions observations into disjoint clusters

Space and Time Complexity

- Storage complexity: $O(N^2)$
  - Computation of distance matrix = requires storage of $\frac{N^2 - N}{2}$ entries
  - Space needed to keep track of the clusters = total number of clusters =N-1
  - Total = $O(N^2)$
- Time complexity: naively $O(N^3)$
  - Depends on the choice of inter-cluster dissimilarity measures adopted
  - By using clever sorting algorithms, complexity can be brought down to $O(N^2 \log N)$
- Space and time complexity severely limits the size of data sets that can be processed

Characteristics of Hierarchical Clustering

- Lack of a global objective function
  - Need not solve hard combinatorial optimization problem as in K-means
  - No issues with local minima or choosing initial points
- Deterministic algorithm
- Merging decisions are final
- May impose a hierarchical structure on an otherwise un-hierarchical data

References:

- Elements of Statistical Learning by Hastie, Trevor and Tibshirani, Robert and Friedman, Jerome - Section 14.3
- Introduction to Data Mining, by Tan, Steinbach and Kumar - Chapter 8
- Algorithms for Clustering Data, Jane and Dubes - Chapter 3
- Introduction to Computation and Programming using Python with Application to Computational Modeling and Understanding Data (3rd edition) by John. V. Guttag - Chapter 25