


Cluster Analysis: Basic Concepts and Methods

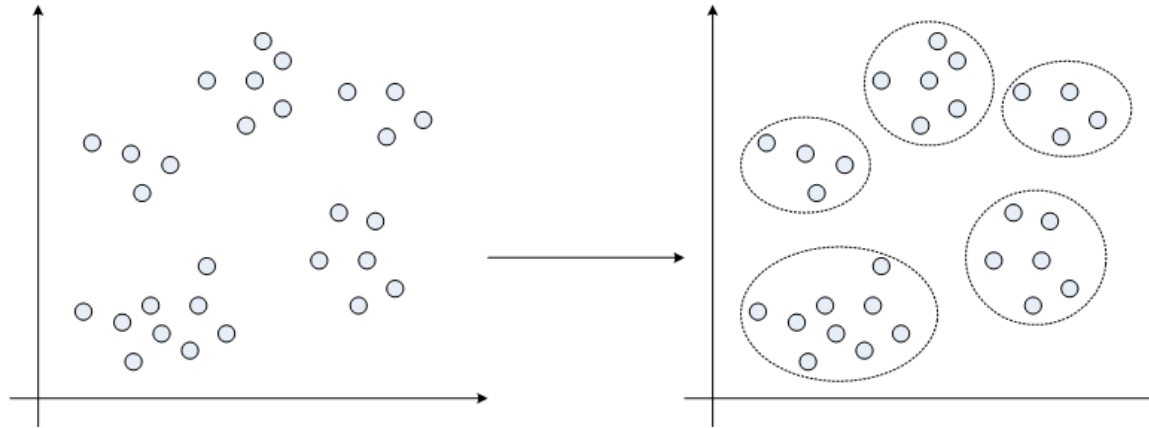
- Cluster Analysis: Basic Concepts 
- Partitioning Methods
- Hierarchical Methods

Clustering (introduction)

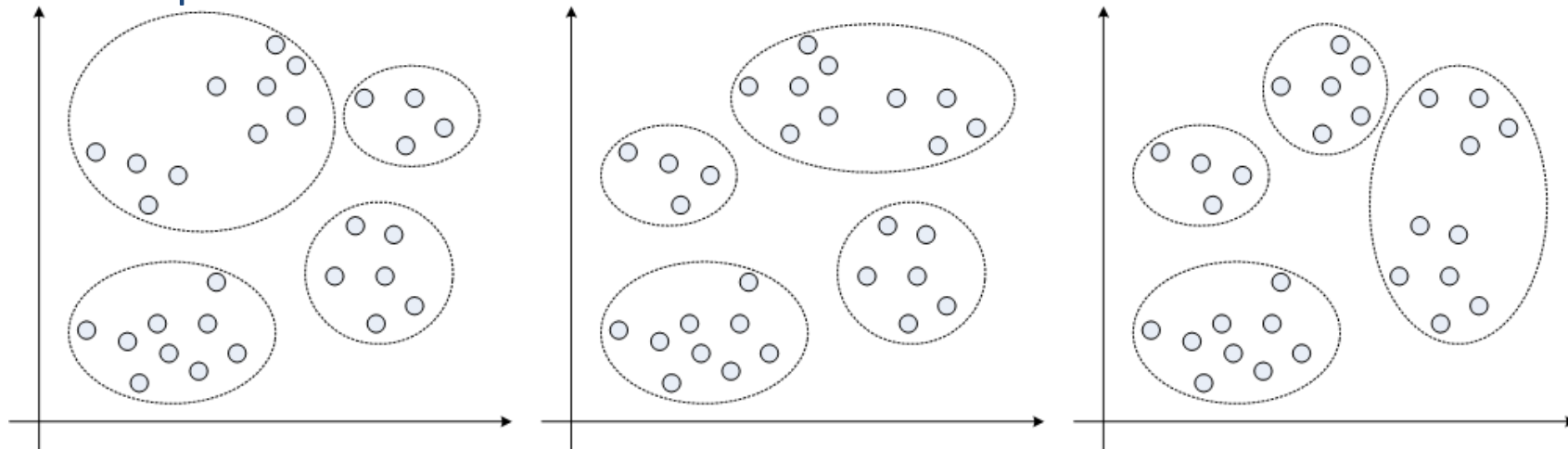
- Clustering is a type of unsupervised machine learning
- It is distinguished from supervised learning by the fact that there is not a priori output (i.e. no labels)
 - The task is to learn the classification/grouping from the data
- A cluster is a collection of objects which are *similar* in some way
- Clustering is the process of grouping *similar* objects into groups
- Eg: a group of people clustered based on their height and weight
- Normally, clusters are created using distance measures
 - Two or more objects belong to the same cluster if they are “close” according to a given distance (in this case geometrical distance like Euclidean or Manhattan)
- Another measure is conceptual
 - Two or more objects belong to the same cluster if this one defines a concept *common* to all that objects
 - In other words, objects are grouped according to their fit to descriptive concepts, not according to simple similarity measures

Clustering (introduction)

- Example: using distance based clustering



- This was easy but how if you had to create 4 clusters?
- Some possibilities are shown below but which is correct?



Clustering (introduction – ctd)

- So, the goal of clustering is to determine the intrinsic grouping in a set of unlabeled data
- But how to decide what constitutes a good clustering?
- It can be shown that there is no absolute “best” criterion which would be independent of the final aim of the clustering
- Consequently, it is the user which must supply this criterion, to suit the application
- Some possible applications of clustering
 - data reduction – reduce data that are homogeneous (similar)
 - find “natural clusters” and describe their unknown properties
 - find useful and suitable groupings
 - find unusual data objects (i.e. outlier detection)

Clustering: Application Examples

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Information retrieval: document clustering
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults
- Climate: understanding earth climate, find patterns of atmospheric and ocean
- Economic Science: market research

Clustering – Major approaches

- Exclusive (partitioning)
 - Data are grouped in an exclusive way, one data can only belong to one cluster
 - Eg: *K-means*
- Agglomerative
 - Every data is a cluster initially and iterative unions between the two nearest clusters reduces the number of clusters
 - Eg: Hierarchical clustering
- Overlapping
 - Uses fuzzy sets to cluster data, so that each point may belong to two or more clusters with different degrees of membership
 - In this case, data will be associated to an appropriate membership value
 - Eg: Fuzzy C-Means
- Probabilistic
 - Uses probability distribution measures to create the clusters
 - Eg: Gaussian mixture model clustering, which is a variant of *K-means*

Exclusive (partitioning) clustering

- Aim: Construct a partition of a database D of N objects into a set of K clusters
- Method: Given a K , find a partition of K clusters that optimises the chosen partitioning criterion
- K -means (MacQueen'67) is one of the commonly used clustering algorithm
- It is a heuristic method where each cluster is represented by the centre of the cluster (i.e. the centroid)
- *Note: One and two dimensional (i.e. with one and two features) data are used in this lecture for simplicity of explanation*
- *In general, clustering algorithms are used with much higher dimensions*

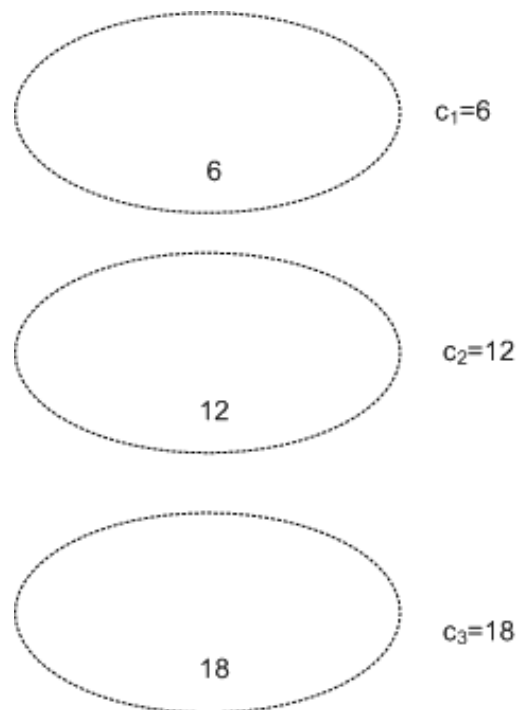
***K*-means clustering algorithm**

Given K , the *K-means* algorithm is implemented in four steps:

1. Choose K points at random as cluster centres (centroids)
2. Assign each instance to its closest cluster centre using certain distance measure (usually Euclidean or Manhattan)
3. Calculate the centroid of each cluster, use it as the new cluster centre (one measure of centroid is mean)
4. Go back to Step 2, stop when cluster centres do not change any more

K-means – an example

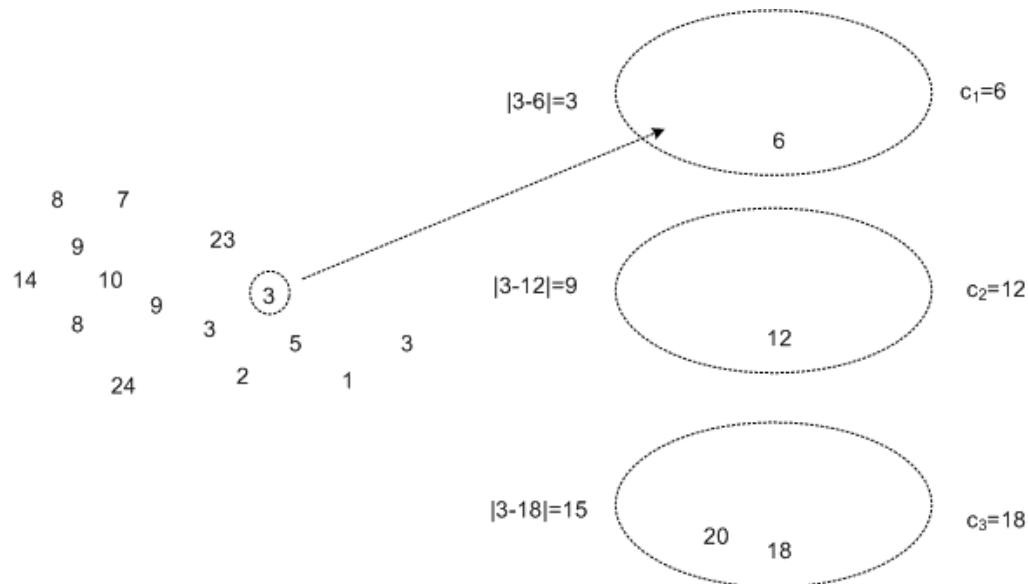
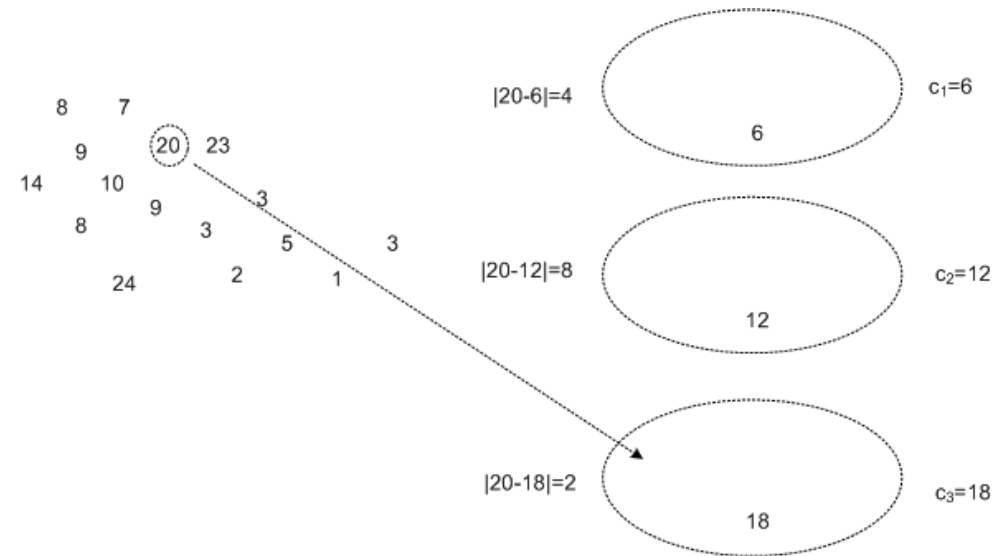
- Say, we have the data: {20, 3, 9, 10, 9, 3, 1, 8, 5, 3, 24, 2, 14, 7, 8, 23, 6, 12, 18} and we are asked to use K-means to cluster these data into 3 groups
- Assume we use Manhattan distance*
- Step one: Choose K points at random to be cluster centres
- Say 6, 12, 18 are chosen



*note for one dimensional data, Manhattan distance=Euclidean distance

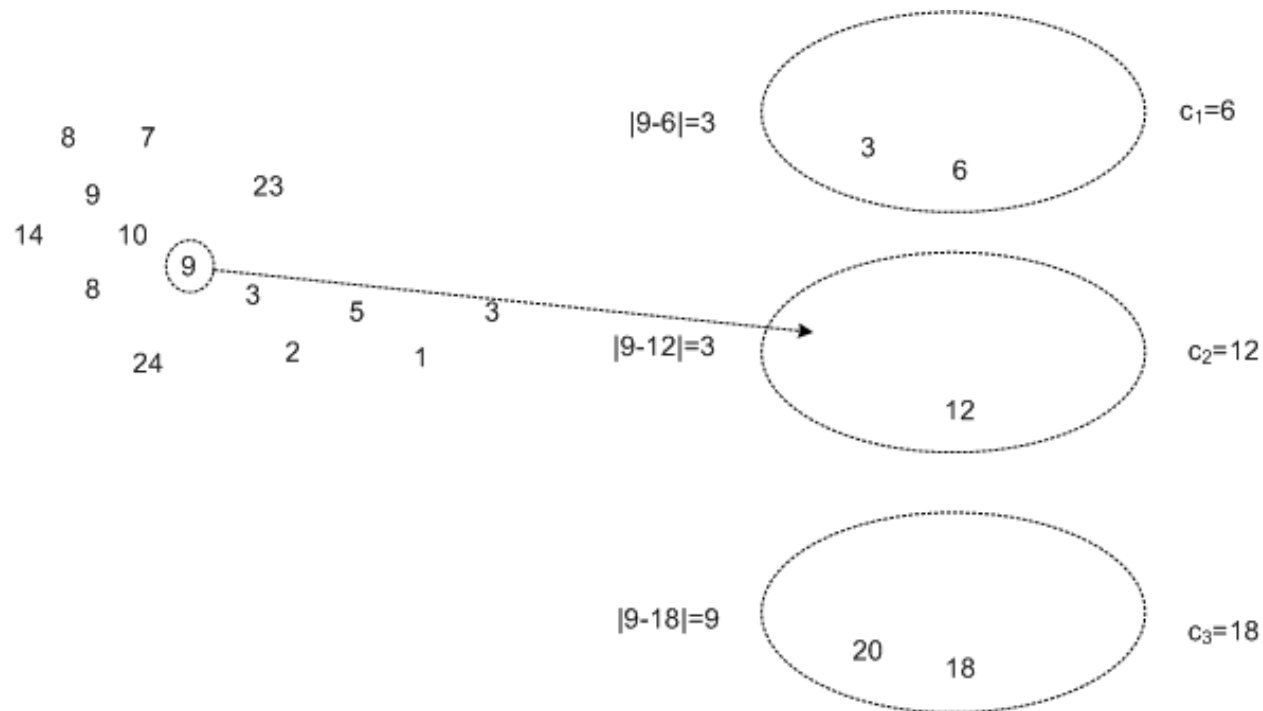
K-means – an example (ctd)

- Step two: Assign each instance to its closest cluster centre using Manhattan distance
- For instance:
 - 20 is assigned to cluster 3
 - 3 is assigned to cluster 1



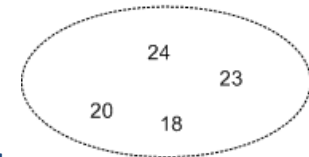
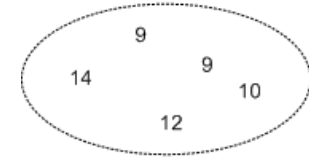
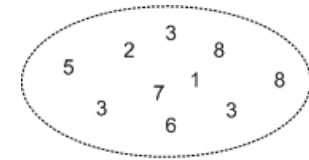
K-means – Example (ctd)

- Step two continued: 9 can be assigned to cluster 1, 2 but let us say that it is arbitrarily assigned to cluster 2
- Repeat for all the rest of the instances

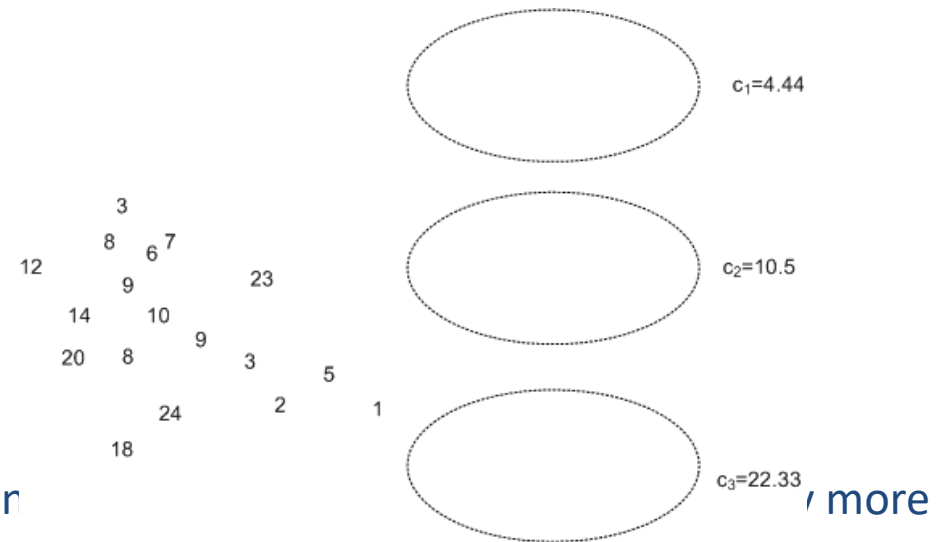


K-Means – Example (ctd)

- And after exhausting all instances...



- Step three: Calculate the centroid (i.e. mean) of each cluster, use it as the new cluster centre



- End of iteration 1
- Step four: Iterate (repeat steps 2 and 3) until convergence

K - means

- Strengths

- Relatively efficient: where N is no. objects, K is no. clusters, and T is no. iterations. Normally, $K, T \ll N$.
- Procedure always terminates successfully (but see below)

- Weaknesses

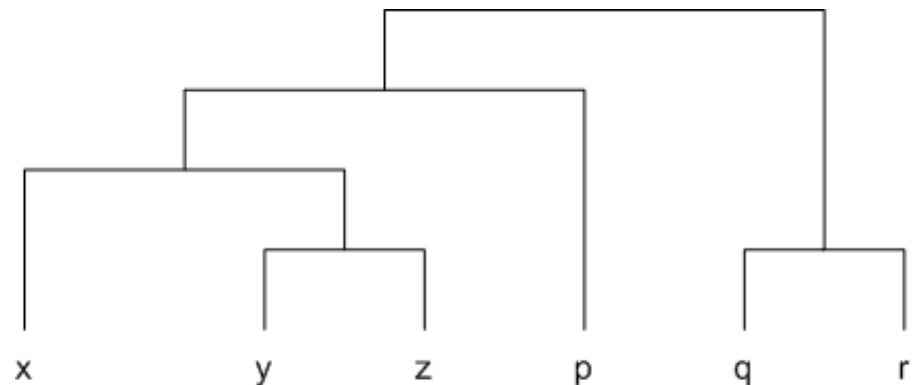
- Does not necessarily find the most optimal configuration
- Significantly sensitive to the initial randomly selected cluster centres
- Applicable only when mean is defined (i.e. can be computed)
- Need to specify K , the number of clusters, in advance

Agglomerative clustering

- K-means approach starts out with a fixed number of clusters and allocates all data into the exactly number of clusters
- But agglomeration does not require the number of clusters K as an input
- Agglomeration starts out by forming each data as one cluster
 - So, data of N object will have N clusters
- Next by using some distance (or similarity) measure, reduces the number of clusters (one in each iteration) by merging process
- Finally, we have one big cluster than contains all the objects
- But then what is the point of having one big cluster in the end?

Dendrogram (ctd)

- While merging cluster one by one, we can draw a tree diagram known as dendrogram
- **Dendrograms** are used to represent agglomerative clustering
- From dendrograms, we can get any number of clusters
- Eg: say we wish to have 2 clusters, then *cut the top one link*
 - Cluster 1: q, r
 - Cluster 2: x, y, z, p
- Similarly for 3 clusters, cut 2 top links
 - Cluster 1: q, r
 - Cluster 2: x, y, z
 - Cluster 3: p



A dendrogram example

Hierarchical clustering - algorithm

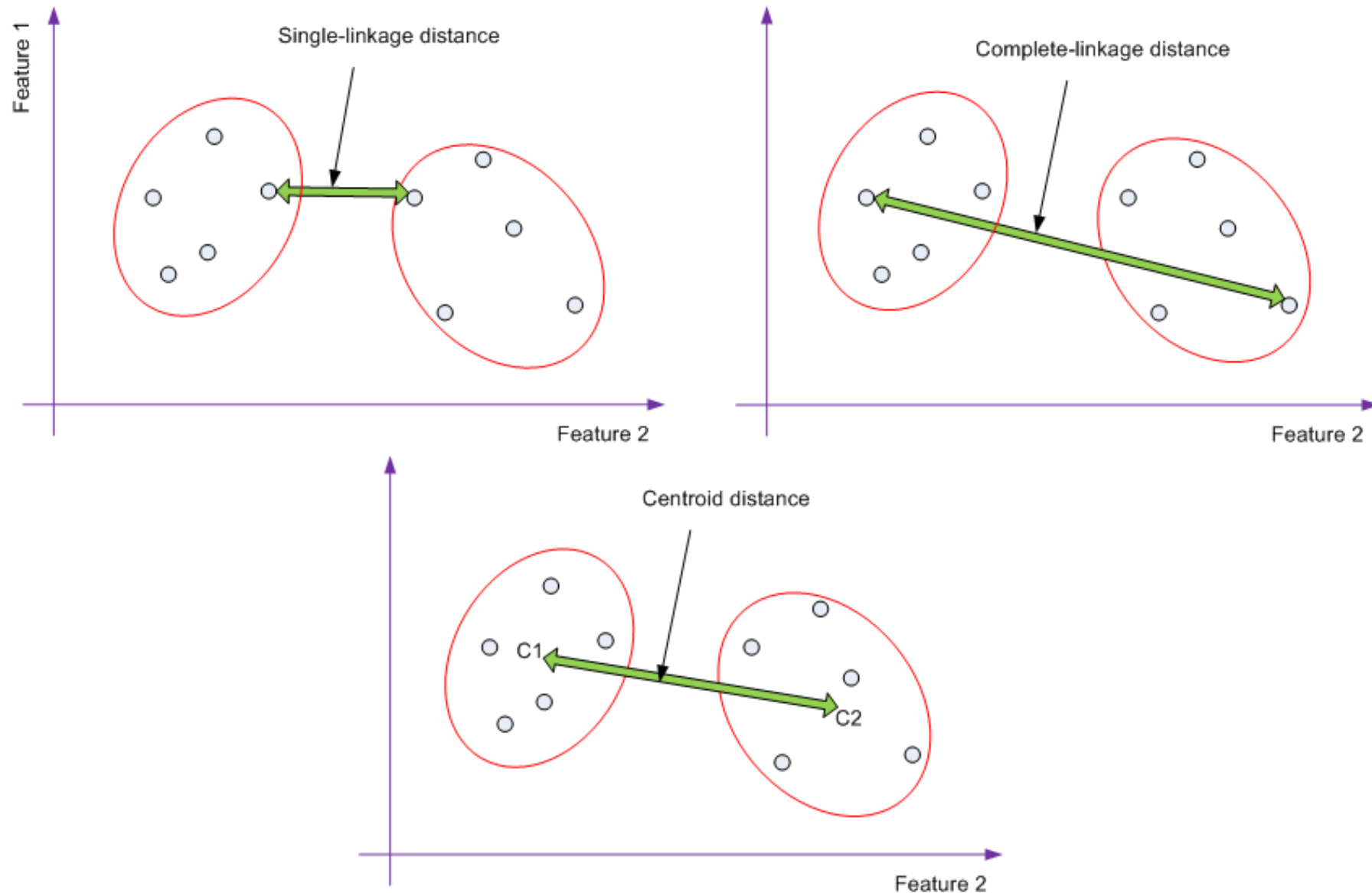
- Hierarchical clustering algorithm is a type of agglomerative clustering
- Given a set of N items to be clustered, hierarchical clustering algorithm:
 1. Start by assigning each item to its own cluster, so that if you have N items, you now have N clusters, each containing just one item
 2. Find the closest distance (most similar) pair of clusters and merge them into a single cluster, so that now you have one less cluster
 3. Compute pairwise distances between the new cluster and each of the old clusters
 4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N
 5. Draw the dendrogram, and with the complete hierarchical tree, if you want K clusters you just have to cut the $K-1$ top links

Note any distance measure can be used: Euclidean, Manhattan, etc

Hierarchical clustering algorithm– step 3

- *Computing distances between clusters for Step 3 can be implemented in different ways:*
 - **Single-linkage clustering**
 - The distance between one cluster and another cluster is computed as the shortest distance from any member of one cluster to any member of the other cluster
 - **Complete-linkage clustering**
 - The distance between one cluster and another cluster is computed as the greatest distance from any member of one cluster to any member of the other cluster
 - **Centroid clustering**
 - The distance between one cluster and another cluster is computed as the distance from one cluster centroid to the other cluster centroid

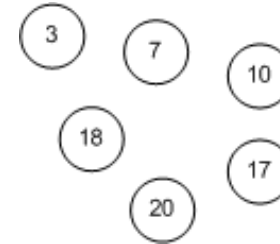
Hierarchical clustering algorithm– step 3



Hierarchical clustering – an example

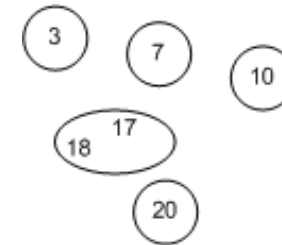
- Assume $X=[3\ 7\ 10\ 17\ 18\ 20]$

1. There are 6 items, so create 6 clusters initially



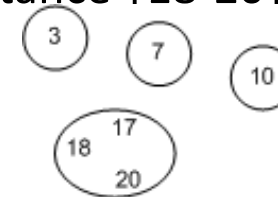
2. Compute pairwise distances of clusters (assume Manhattan distance)

The closest clusters are 17 and 18 (with distance=1), so merge these two clusters together



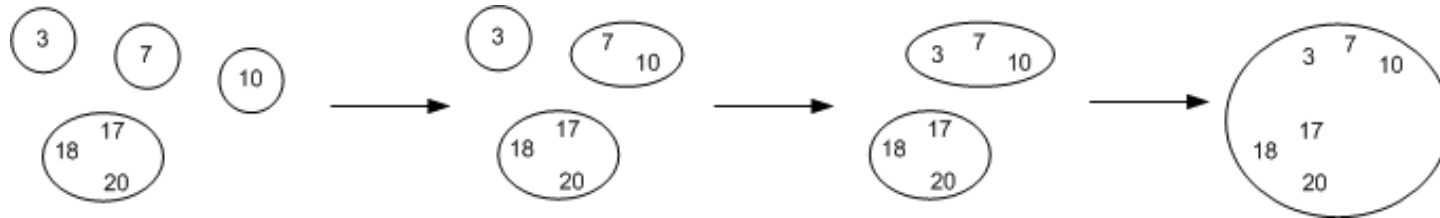
3. Repeat step 2 (assume single-linkage):

The closest clusters are cluster_{17,18} to cluster₂₀ (with distance $|18-20|=2$), so merge these two clusters together

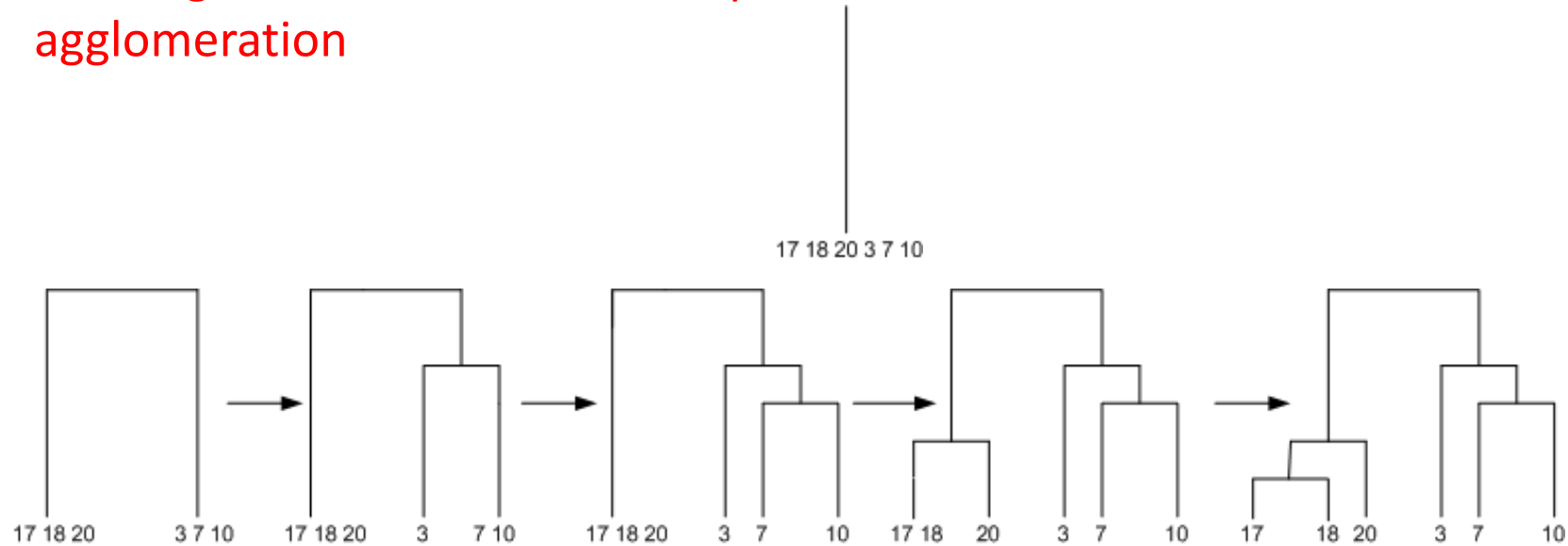


Hierarchical clustering – an example (ctd)

- Go on repeat cluster mergers until one big cluster remains

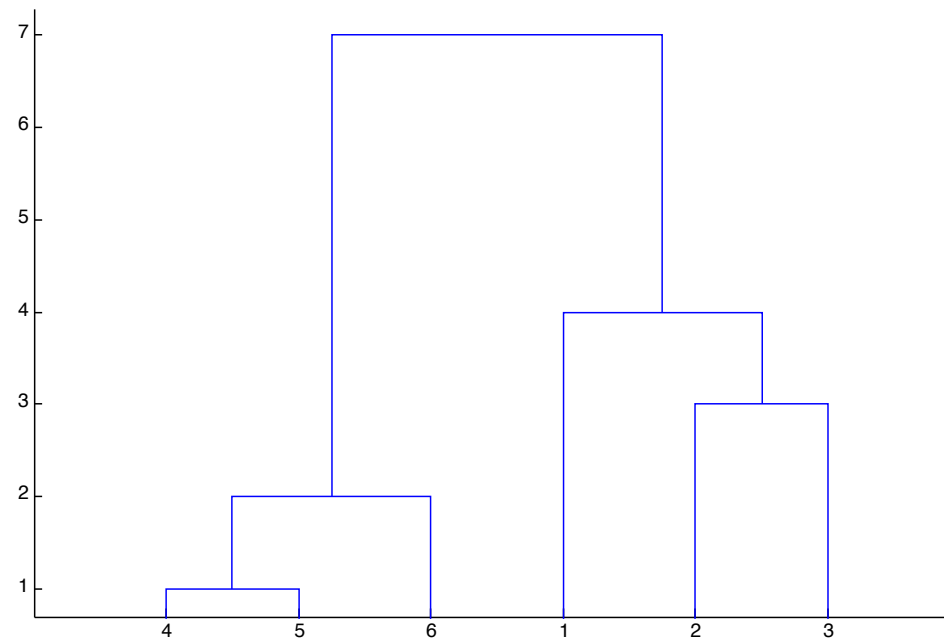


- Draw dendrogram (draw it in reverse of the cluster mergers) – **remember the height of each cluster corresponds to the manner of cluster agglomeration**



Hierarchical clustering – an example (ctd) –using MATLAB

- `%Hierarchical clustering example`
- `X=[3 7 10 17 18 20]; %data`
- `Y=pdist(X', 'cityblock'); %compute pairwise Manhattan distances`
- `Z=linkage(Y, 'single'); %do clustering using single-linkage method`
- `dendrogram(Z); %draw dendrogram –note only indices are shown`



Comparing agglomerative vs exclusive clustering

- Agglomerative - advantages
 - Preferable for detailed data analysis
 - Provides more information than exclusive clustering
 - We can decide on any number of clusters without the need to redo the algorithm –in exclusive clustering, K has to be decided first, if a different K is used, then need to redo the whole exclusive clustering algorithm
 - One unique answer
- Disadvantages
 - Less efficient than exclusive clustering
 - No backtracking, i.e. can never undo previous steps

Clustering validity problem

- *Problem 1*
- A problem we face in clustering is to decide the optimal number of clusters that fits a data set
- *Problem 2*
- The various clustering algorithms behave in a different way depending on
 - the features of the data set the features of the data set (geometry and density distribution of clusters)
 - the input parameters values (eg: for K -means, initial cluster choices influence the result)
- So, how do we know which clustering method is better/suitable?
- *We need a clustering quality criteria*

Clustering validity problem

- In general, good clusters, should have
 - High intra-cluster similarity, i.e. low variance among intra-cluster memberswhere variance for x is defined by

with \bar{x} as the mean of x

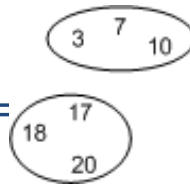
$$\text{var}(x) = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2$$

- For eg: if $x=[2 \ 4 \ 6 \ 8]$, then $\bar{x}=5$ so $\text{var}(x)=6.67$

$$\bar{x} = 5$$

- Computing intra-cluster similarity is simple
- For eg: for the clusters shown

- $\text{var}(\text{cluster1})=2.33$ while $\text{var}(\text{cluster2})=10.67$
- So, cluster 1 is better than cluster 2
- *Note: use 'var' function in MATLAB to compute variance*



Clustering Quality Criteria

- But this does not tell us anything about how good is the overall clustering or on the suitable number of clusters needed!
- To solve this, we also need to compute inter-cluster variance
- Good clusters will also have low inter-cluster similarity, i.e. high variance among inter-cluster members in addition to high intra-cluster similarity, i.e. low variance among intra-cluster members
- One good measure of clustering quality is Davies-Bouldin index
- The others are
 - Dunn's Validity Index
 - Silhouette method
 - C-index
 - Goodman-Kruskal index
- So, we compute DB index for different number of clusters, K and the best value of DB index tells us on the appropriate K value or on how good is the clustering method

Davies-Bouldin index

- It is a function of the ratio of the sum of within-cluster (i.e. intra-cluster) scatter to between cluster (i.e. inter-cluster) separation
- Because a low scatter and a high distance between clusters lead to low values of R_{ij} , *a minimization of DB index is desired*
- Let $C=\{C_1, \dots, C_k\}$ be a clustering of a set of N objects:

$$DB = \frac{1}{k} \cdot \sum_{i=1}^k R_i$$

with

and

$$R_i = \max_{j=1, \dots, k, i \neq j} R_{ij}$$

$$R_{ij} = \frac{\text{var}(C_i) + \text{var}(C_j)}{\|c_i - c_j\|}$$

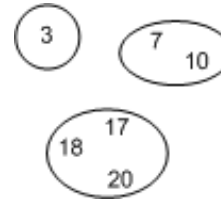
where C_i is the i^{th} cluster and c_i is the centroid for cluster i

Numerator of R_{ij} is a measure of intra-cluster similarity while the denominator is a measure of inter-cluster separation

Note, $R_{ij}=R_{ji}$

Davies-Bouldin index example

- For eg: for the clusters shown



- Compute $R_{ij} = \frac{\text{var}(C_i) + \text{var}(C_j)}{\|c_i - c_j\|}$

Note, variance of one element is zero and centroid is simply the element itself

- $\text{var}(C_1)=0, \text{var}(C_2)=4.5, \text{var}(C_3)=2.33$
- Centroid is simply the mean here, so $c_1=3, c_2=8.5, c_3=18.33$
- So, $R_{12}=1, R_{13}=0.152, R_{23}=0.797$

- Now, compute $R_i = \max_{j=1,..k, i \neq j} R_{ij}$

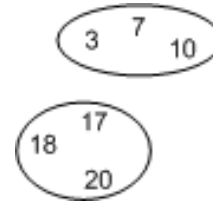
- $R_1=1$ (max of R_{12} and R_{13}); $R_2=1$ (max of R_{21} and R_{23}); $R_3=0.797$ (max of R_{31} and R_{32})

- Finally, compute $DB = \frac{1}{k} \cdot \sum_{i=1}^k R_i$

- DB=0.932

Davies-Bouldin index example (ctd)

- For eg: for the clusters shown



- Compute $R_{ij} = \frac{\text{var}(C_i) + \text{var}(C_j)}{\|c_i - c_j\|}$

- Only 2 clusters here
- $\text{var}(C_1)=12.33$ while $\text{var}(C_2)=2.33$; $c_1=6.67$ while $c_2=18.33$

- $R_{12}=1.26$

- Now compute $R_i = \max_{j=1,..k, i \neq j} R_{ij}$

- Since we have only 2 clusters here, $R_1=R_{12}=1.26$; $R_2=R_{21}=1.26$

- Finally, compute $DB = \frac{1}{k} \cdot \sum_{i=1}^k R_i$

- DB=1.26

Davies-Bouldin index example (ctd)

- DB with 2 clusters=1.26, with 3 clusters=0.932
- *So, $K=3$ is better than $K=2$ (since DB smaller, better clusters)*
- In general, we will repeat DB index computation for all cluster sizes from 2 to $N-1$
- So, if we have 10 data items, we will do clustering with $K=2, \dots, 9$ and then compute DB for each value of K
 - $K=10$ is not done since each item is its own cluster
- Then, we decide the best clustering size (and the best set of clusters) would be the one with minimum values of DB index