Machine Learning and Financial Applications

Lecture 9
Clustering and dimension reduction

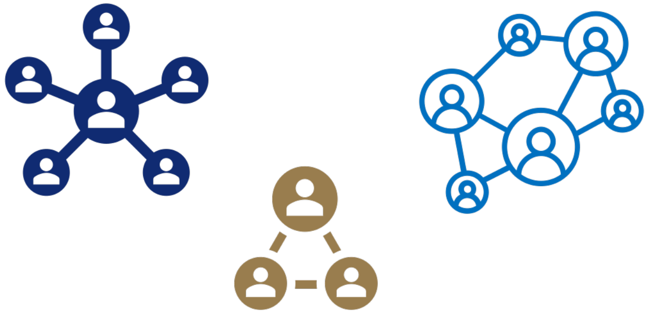
Liu Peng

liupeng@smu.edu.sg

Fundamentals of Clustering

- Clustering discovers natural groupings in data, i.e., learning from **unlabeled data**
- To group **homogenous** (similar) observations into the **same cluster** and **heterogeneous** (dissimilar) observation into **different clusters**
- Observations within a cluster shall be similar, while observations in a cluster shall be dissimilar to the observations in other clusters
- Mathematically, we aim to **minimize within-cluster variance, and maximize between-cluster variance**
- Supervised or unsupervised learning?

Social network analysis



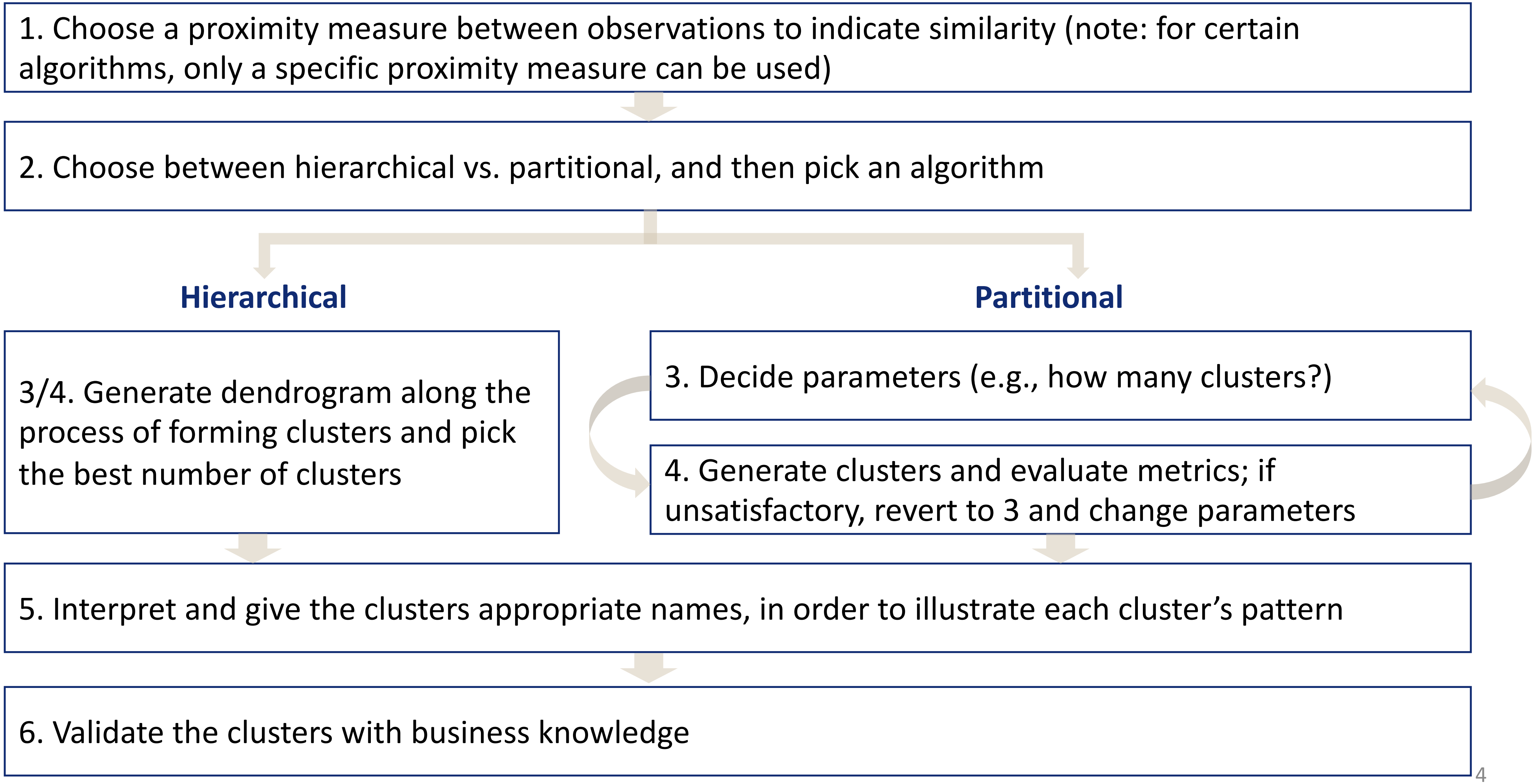
Market segmentation

Organize computing clusters

Clustering vs. Classification

	y label exists for data?	Train + Test data set?	Evaluate model results?
Clustering			
Classification			

Basic steps of cluster analysis



1. Choose a proximity measure between observations to indicate similarity (note: for certain algorithms, only a specific proximity measure can be used)

2. Choose between hierarchical vs. partitional, and then pick an algorithm

Hierarchical

3/4. Generate dendrogram along the process of forming clusters and pick the best number of clusters

Partitional

3. Decide parameters (e.g., how many clusters?)

4. Generate clusters and evaluate metrics; if unsatisfactory, revert to 3 and change parameters

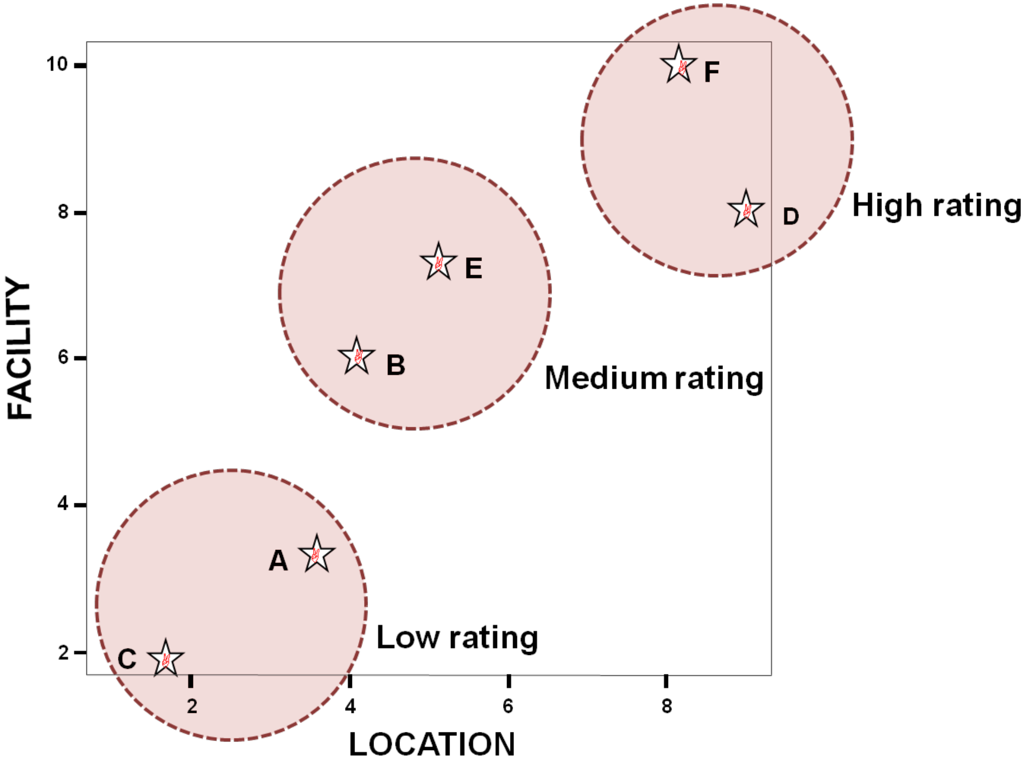
5. Interpret and give the clusters appropriate names, in order to illustrate each cluster's pattern

6. Validate the clusters with business knowledge

A small dataset HOTEL.csv to illustrate clustering algorithms

HOTEL	FACILITY	LOCATION
A	3	3
B	6	4
C	2	1
D	8	9
E	7	5
F	10	8

- n = 6 (6 premier hotels, labelled A to F)
- p = 2 (2 input variables)
  - rankings of FACILITY
  - rankings of LOCATION
- Both rankings are measured from 1 to 10
- 10 indicates very good facility or very good location



A simple scatter plot reveals that there could be 3 clusters

What is a proximity measure?

Purpose

Clustering relies on proximity measures to find similar observations and put them in the same cluster

Different flavors

- Euclidean Distance
- Mahalanobis Distance
- Minkowski Distance (not covered)
- Manhattan Distance (not covered)
- Chebyshev Distance (not covered)

Definitions

- With *n* observations, matrix X is the data set
- Let x_i as the i-th row, where *i* =1, 2, …, n
- v_j indicates the j-th input variable (column)

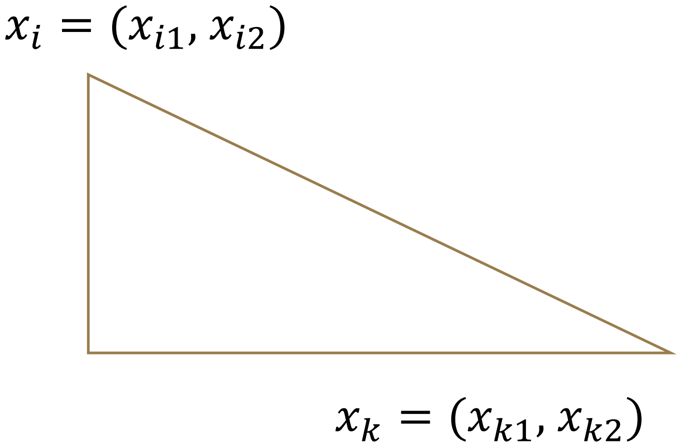
	v_1	v_2	v_3		v_p
x_1					
x_2					
…					
x_n					

- Let there be k clusters: C_1, C_2, …, C_k
- Each cluster is a subset of X
- The union of all clusters is equivalent to X
- The intersection of any two clusters must be empty

The most common proximity measure: Euclidean Distance

$$\mathrm{d}(x_i,x_k) = \sqrt{\sum\nolimits_{j=1}^{p} (x_{ij} - x_{kj})^2}$$

When *p* = 2: Pythagoras Theorem



Student ID	Height	Score	Age
1	64	580	19
2	66	570	21
3	68	590	18
4	69	660	24
5	73	600	23

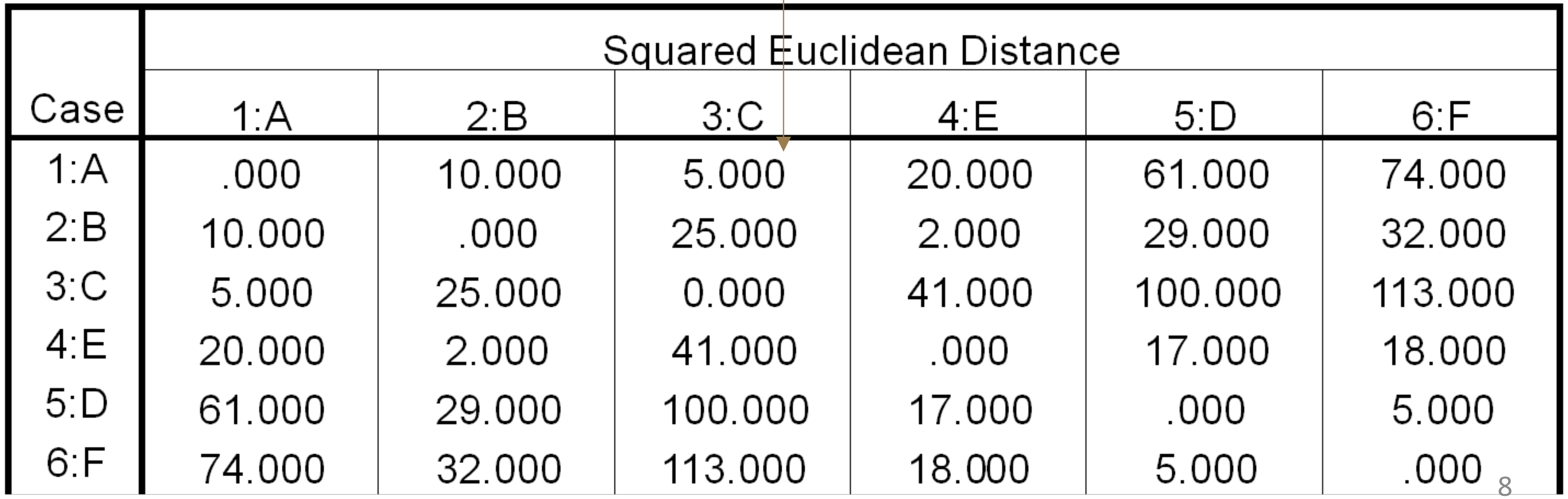
- d(x_1, x_2) = $\sqrt{(64-66)^2 + (580-570)^2 + (19-21)^2}$ = 10.39
- d(x_1, x_4) = $\sqrt{(64-69)^2 + (580-660)^2 + (19-24)^2}$ = 80.31

- The distance is dominated by the input variable ‘score’
- May need to perform normalisation on all inputs
- Assumption of Euclidean Distance: all input variables are equally weighted and independent of each other

n*n proximity matrix with **Squared** Euclidean distance for hotel data set

HOTEL	FACILITY	LOCATION
A	3	3
B	6	4
C	2	1
D	8	9
E	7	5
F	10	8

$d^2(A, C) = 1 + 4 = 5$



	Squared Euclidean Distance					
Case	1:A	2:B	3:C	4:E	5:D	6:F
1:A	.000	10.000	5.000	20.000	61.000	74.000
2:B	10.000	.000	25.000	2.000	29.000	32.000
3:C	5.000	25.000	0.000	41.000	100.000	113.000
4:E	20.000	2.000	41.000	.000	17.000	18.000
5:D	61.000	29.000	100.000	17.000	.000	5.000
6:F	74.000	32.000	113.000	18.000	5.000	.000

Mahalanobis Distance

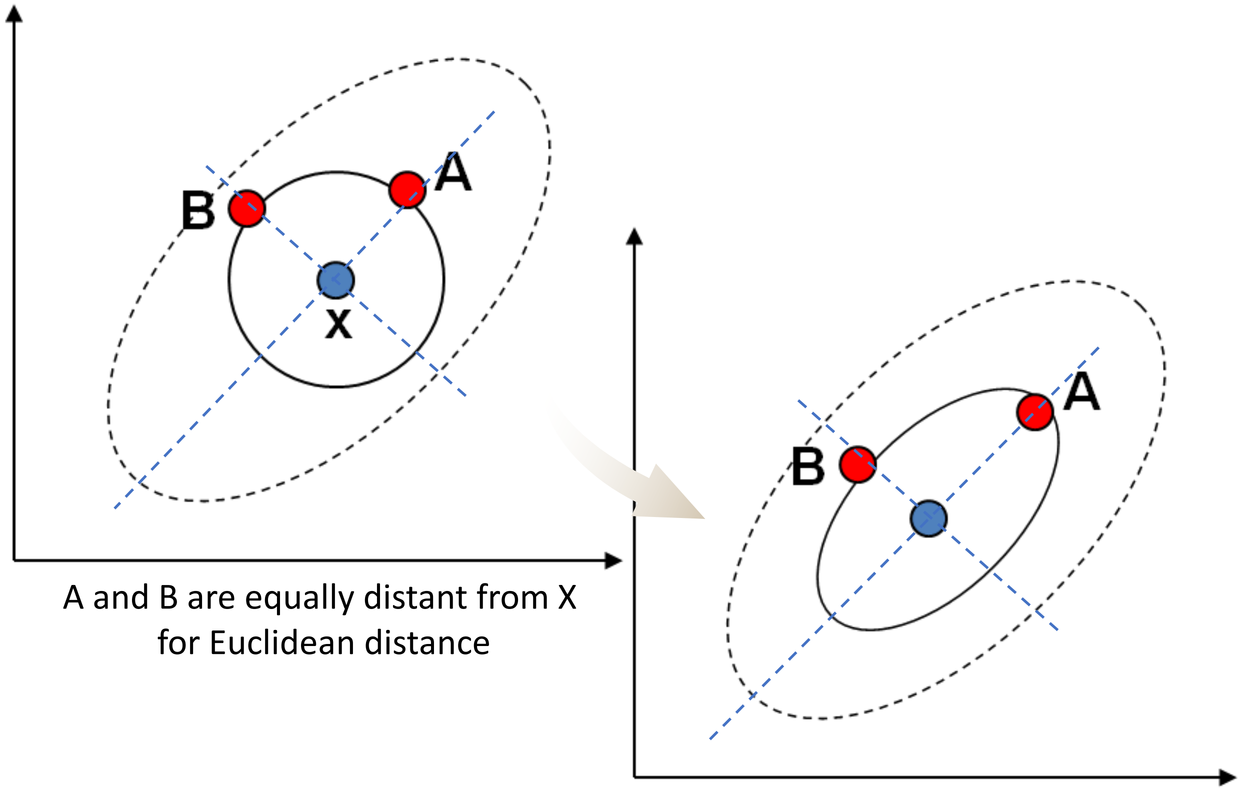
Key facts

- It accounts for the fact that the variances in each direction are different
- Distance between two points as a form of **standardized Euclidean distance**, i.e., distance between a data point and a distribution

Equation

$$\mathrm{d}(x_i, x_k) = \sqrt{(x_i - x_k) S^{-1} (x_i - x_k)'}$$

- $(x_i - x_k)'$ is the transpose of $(x_i - x_k)$
- S is the sample covariance matrix



A and B are equally distant from X for Euclidean distance

A and B are equally distant from X for Mahalanobis distance

Proximity measures work for **numerical input variables** only. What about categorical?

Proximity measure for categorical variables (1/2)

- **Scenario 1**
  - Categorical input variable **Fruit** has 3 values: Apple, Banana, Orange, with **no sequence**
  - How to measure the distance between Apple and Banana?
  - How to measure the distance between two data observations if there are categorical and continuous variables together?
- **Solution**
  - Use **one hot encoding** to create 2 dummy variables, is_apple (1 or 0), is_banana (1 or 0); if both are 0, the row has value ‘Orange’
- **Potential issue**
  - If all numerical input variables are all normalized to the range of [0,1], will the **dummy variables** derived from categorical variables **overshadow** the numerical variables?
  - What if the categorical input variable has too many values hence too many dummy variables are created?

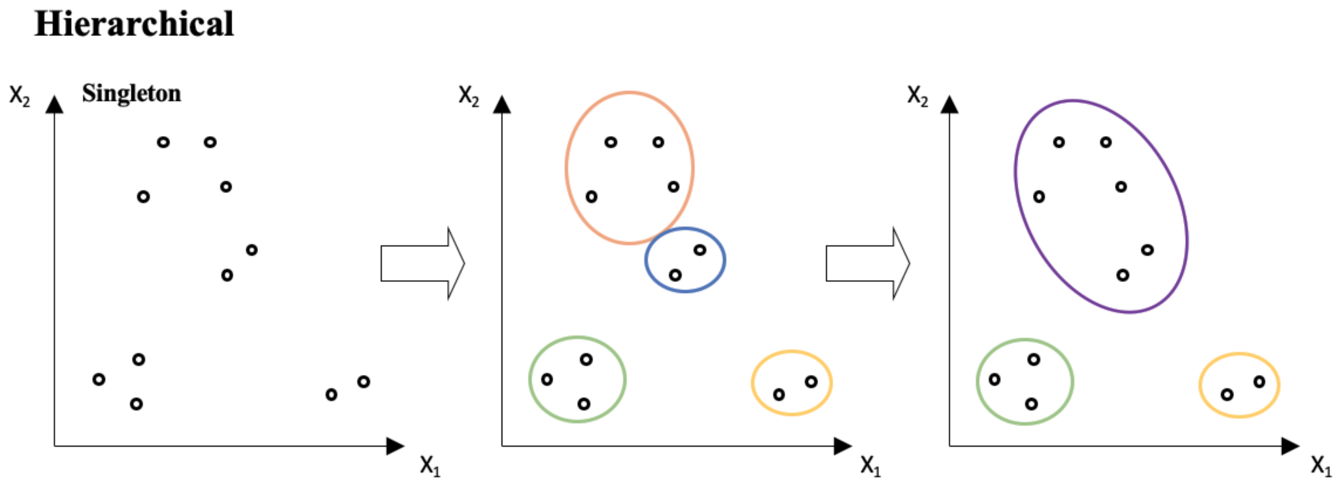
Proximity measure for categorical variables (2/2)

- **Scenario 2**
  - Categorical input variable **Satisfaction** has three values: Bad, Average, Good, with **an order,** known as an **ordinal** variable
- **Solution**:
  - Integer encoding assign integer scores to each value, and **Satisfaction** becomes 0, 1, 2
- **Potential issue**
  - If all numerical input variables are all normalized to the range of [0, 2], will **Satisfaction variable overshadow** the numerical variables?

- **Alternative algorithms (not covered; revisit after introducing K-Means later)**
  - K-modes: extension of K-means (K-Means can only deal with numerical input variables) to solve categorical variables, based on modes to select centroid of each cluster
  - K-prototypes: combine K-means and K-modes
  - Both implemented by Nelis J de Vos in an open-source library **kmodes**

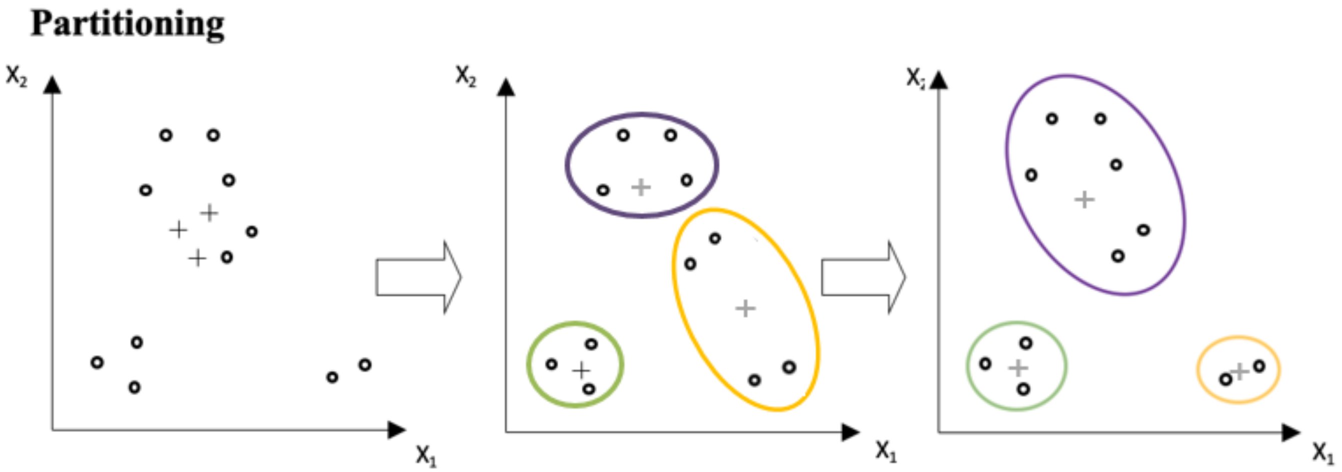
Source: https://pypi.org/project/kmodes/

Choice between Hierarchical vs. Partitioning



In this course we only discuss **Agglomerative clustering**

- We begin with all observations identified as single clusters (singletons)
- Observations are then merged in an agglomerative manner, based on the distances between candidate clusters



In this course we only discuss **K-means**

- Number of clusters is fixed in advance; randomly set centroids to start
- Clustering process facilitates membership movements with the number of clusters staying the same

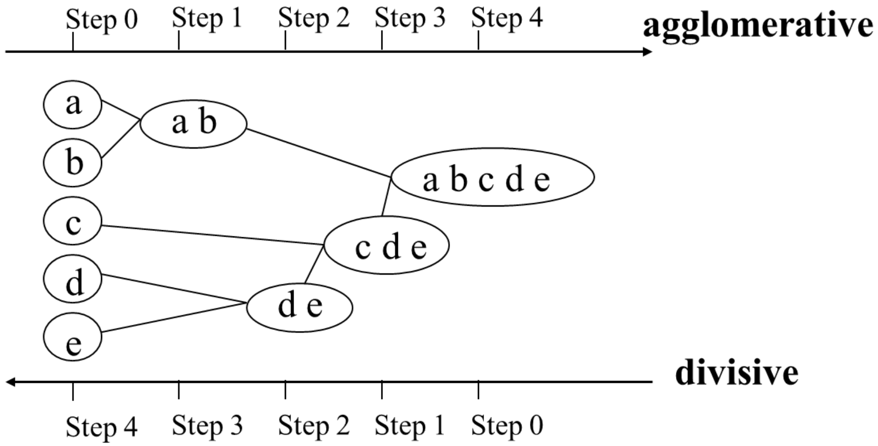
Hierarchical Clustering

Agglomerative (bottom-up)

- Starting with each observation as a single cluster (singleton)
- Merge a pair of closest clusters based on proximity between each pair of clusters
- Stop when one cluster is formed for all observations

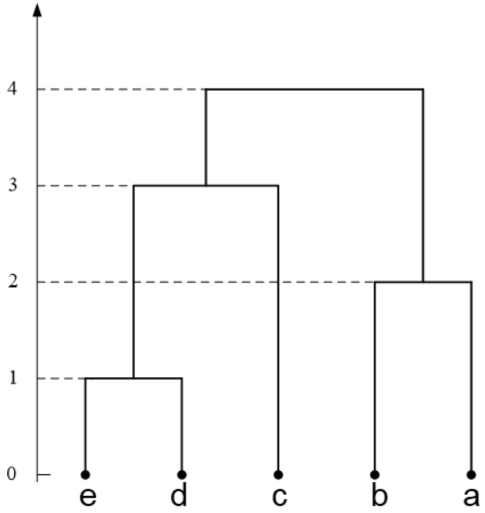
Divisive (top-down), not covered

- Reverse of Agglomerative

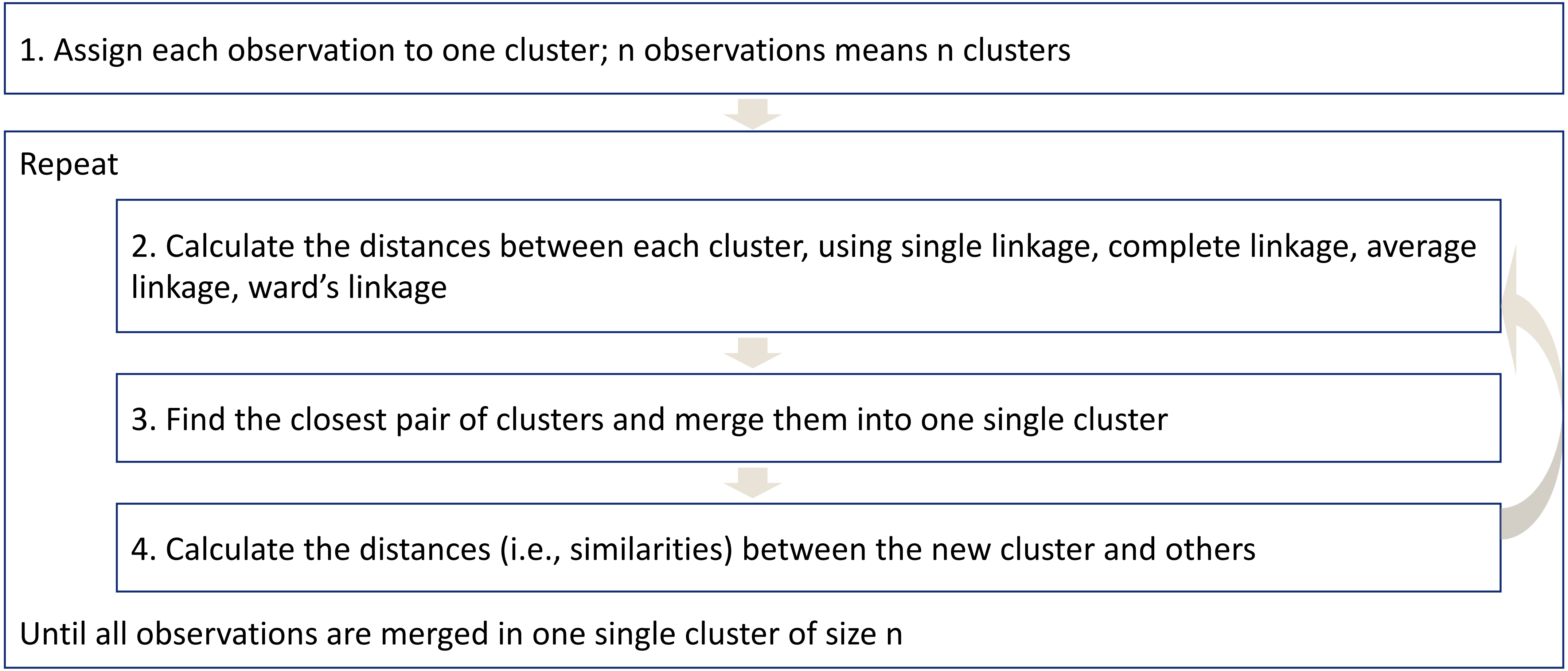


What a dendrogram shows

- How the observations are merged into clusters hierarchically
- x-axis: all observations
- Horizontal line representing clusters being merged
- y-axis: distances between clusters before merging



Details of agglomerative clustering



1. Assign each observation to one cluster; n observations means n clusters

Repeat

2. Calculate the distances between each cluster, using single linkage, complete linkage, average linkage, ward’s linkage

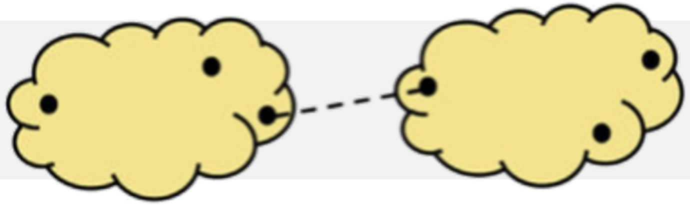
3. Find the closest pair of clusters and merge them into one single cluster

4. Calculate the distances (i.e., similarities) between the new cluster and others

Until all observations are merged in one single cluster of size n

Hierarchical clustering with **single linkage** (1/2)

The proximity of two clusters is the **minimum** distance of all possible combination of observations in the two clusters



6 by 6 proximity matrix (Squared Euclidean distance)

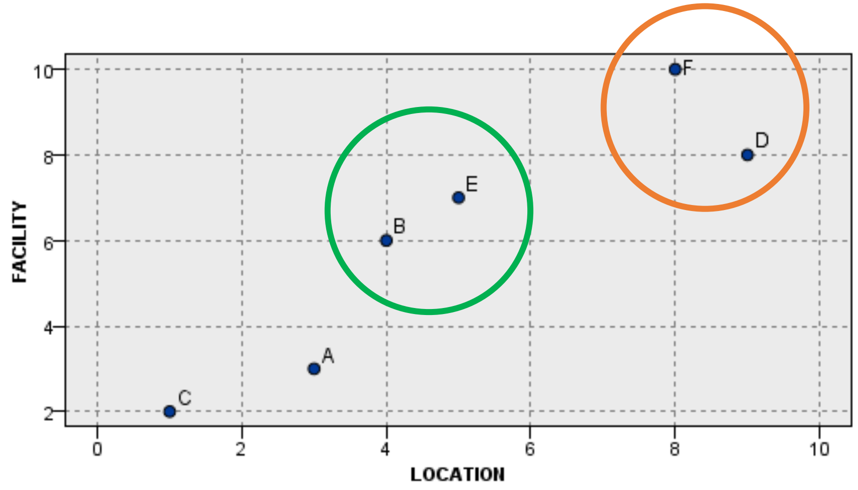
	A	**B**	**C**	**D**	**E**	**F**
A	0					
B	10	0				
C	5	25	0			
D	61	29	100	0		
E	20	2	41	17	0	
F	74	32	113	5	18	0

Closest clusters:
B and E

	A	**B,E**	**C**	**D**	**F**
A	0				
B,E	10	0			
C	5	25	0		
D	61	17	100	0	
F	74	18	113	5	0

$\min(d^2(A,B), d^2(A,E)) = \min(10, 20) = 10$

Closest clusters:
D and F



Hierarchical clustering with **single linkage** (2/2)

	A	B,E	C	D,F
A	0			
B,E	10	0		
C	5	25	0	
D,F	61	17	100	0

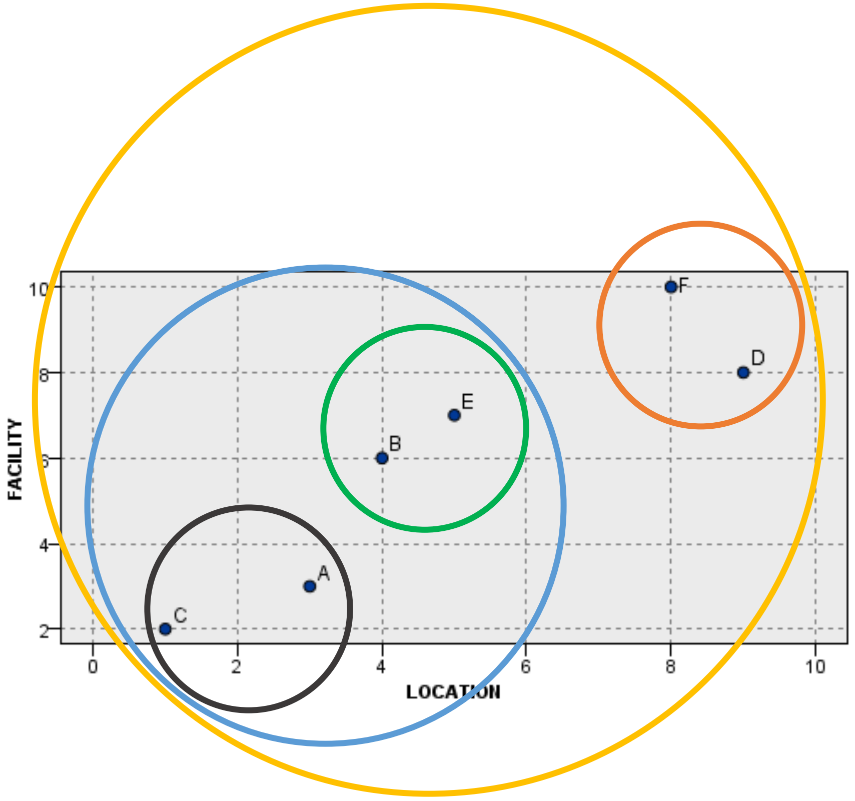
Closest clusters: A and C

	B,E	D,F	A,C
B,E	0		
D,F	17	0	
A,C	10	61	0

Closest clusters: {A,C} and {B,E}

	A,B,C,E	D,F
A,B,C,E	0	
D,F	17	0

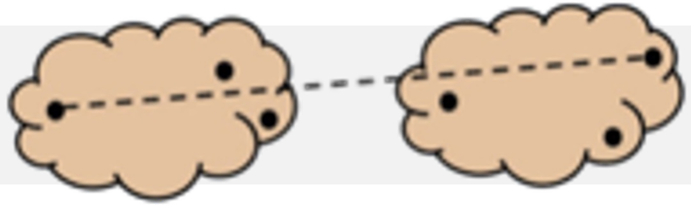
Closest clusters: {A,B,C,E} and {D,F}



Exercise: Implement Agglomerative Clustering with Single Linkage

Hierarchical clustering with **complete linkage** (1/2)

The proximity of two clusters is the **maximum** distance of all possible combination of observations in the two clusters



6 by 6 proximity matrix (Squared Euclidean distance)

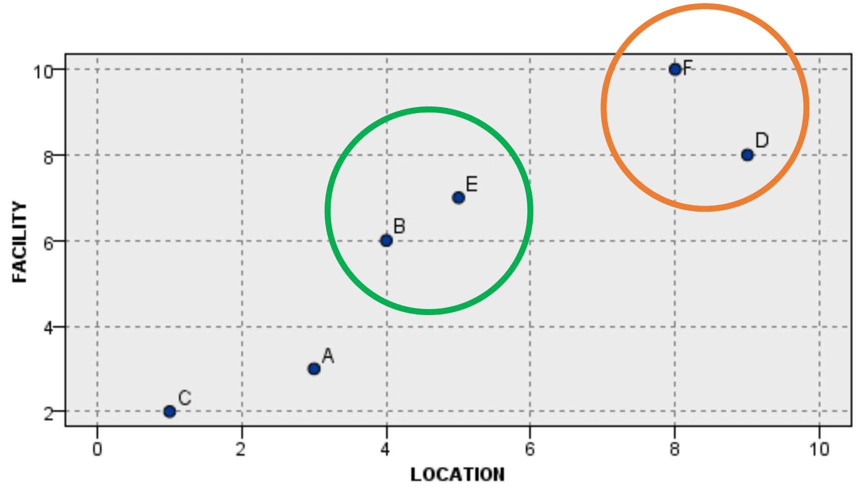
	A	**B**	**C**	**D**	**E**	**F**
A	0					
B	10	0				
C	5	25	0			
D	61	29	100	0		
E	20	2	41	17	0	
F	74	32	113	5	18	0

Closest clusters: B and E

	A	**B,E**	**C**	**D**	**F**
A	0				
B,E	20	0			
C	5	41	0		
D	61	29	100	0	
F	74	32	113	5	0

$\max(d^2(A,B), d^2(A,E)) = \max(10, 20) = 20$

Closest clusters: D and F



Hierarchical clustering with **complete linkage** (2/2)

	A	B,E	C	D,F
A	0			
B,E	20	0		
C	5	41	0	
D,F	74	32	113	0

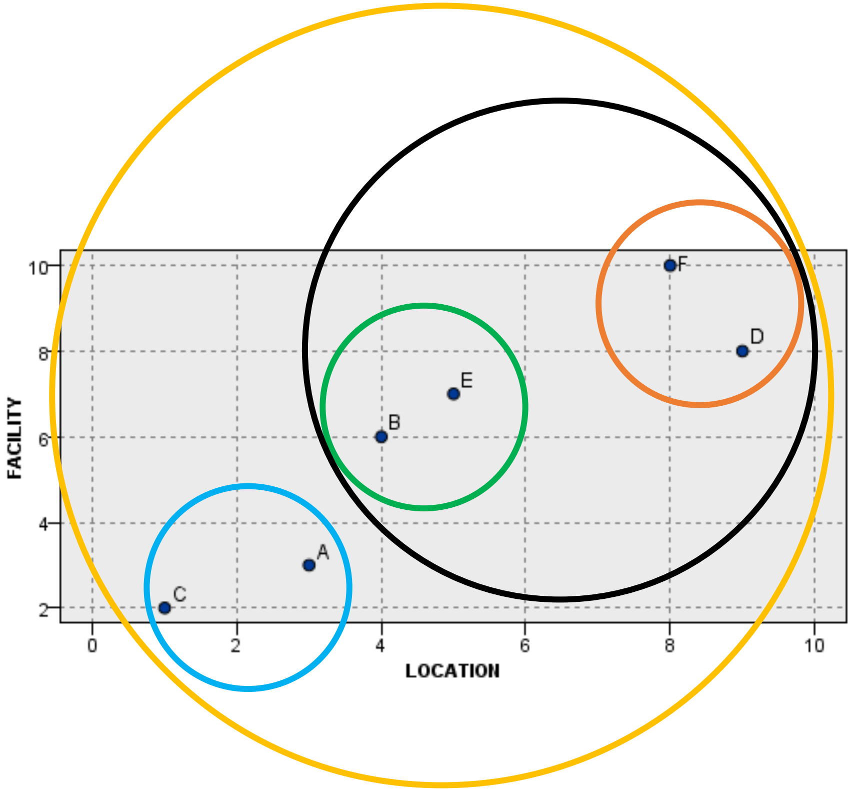
Closest clusters: A and C

	B,E	D,F	A,C
B,E	0		
D,F	32	0	
A,C	41	113	0

Closest clusters: {D,F} and {B,E}

	B,D,E,F	A,C
B,D,E,F	0	
A,C	113	0

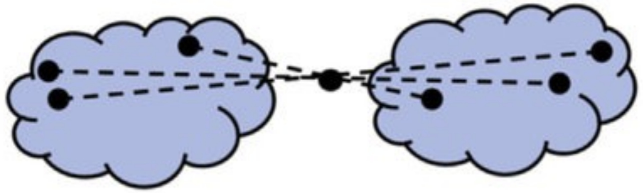
Closest clusters: {A,C} and {B,D,E,F}



Other linkage metrics

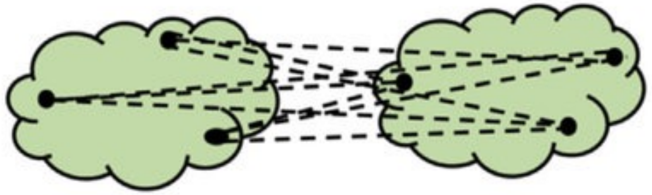
Ward’s linkage

Error sum of squares (ESS): the distances from all observations in the two clusters, to the future centroid if the two cluster were to be merged



Average linkage

Uses the average pair-wise proximity among all pairs of observations in different clusters



Which linkage to use?

Single linkage

- Good for detecting arbitrarily-shaped clusters
- Cannot detect overlapping clusters
- Likely to bring bias

Complete linkage

- Good for detect overlapping clusters
- Not for detecting arbitrarily-shaped clusters
- Likely to bring bias

Average linkage and ward's linkage

- Somewhere in between Single-linkage and Complete-linkage
- Generally, quite useful

Business knowledge, e.g., bank customer's characteristics

Explore the data set to learn more about the properties

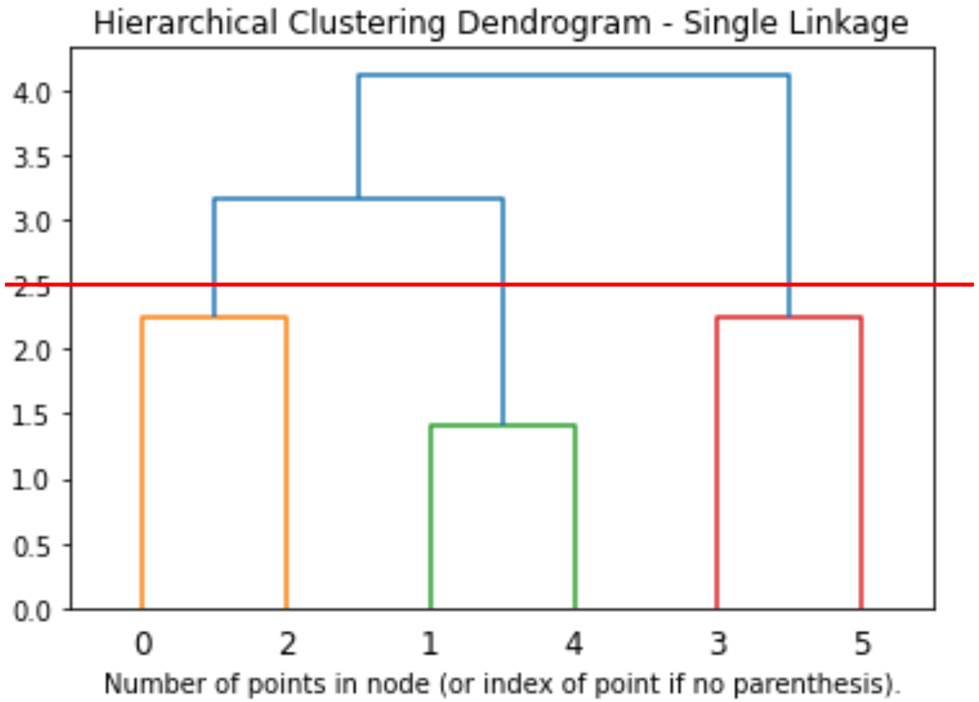
Determine the number of clusters

Rule of thumb

- Look for horizontal lines with significant height
- The higher the horizontal lines are, the more dissimilar the merged cluster is to the two child clusters below
- Set a distance threshold in such a way that it cuts some vertical lines; effectively we draw a horizontal line in the dendrogram; the mergers above the threshold should not happen

Keep in mind

- The optimal number of clusters depends on business knowledge and operational needs, e.g., can we have 1000 clusters and generate 1000 different marketing campaigns for each cluster?



Exercise: Implement Agglomerative Clustering with distance threshold 2.5

Performance issues of hierarchical clustering

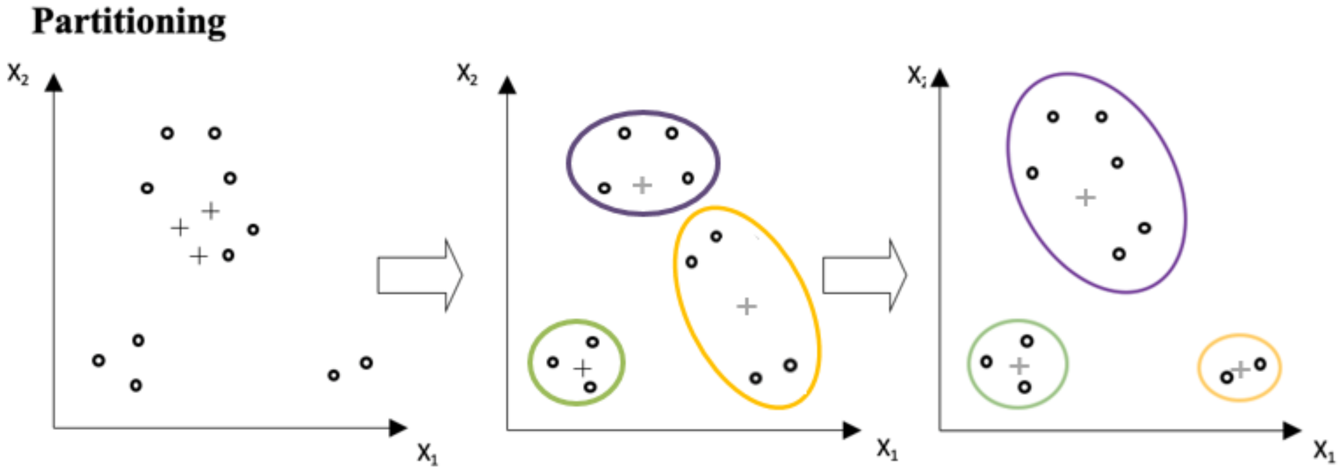
- To obtain n by n proximity matrix, if the data set has 1000 records
  - how many time units for computation?
  - how many memory units to store the proximity values? 1

- After the initial proximity matrix, we need to constantly update the distances between newly formed clusters

- Any alternative solutions?

Partitional Clustering: K-means illustration

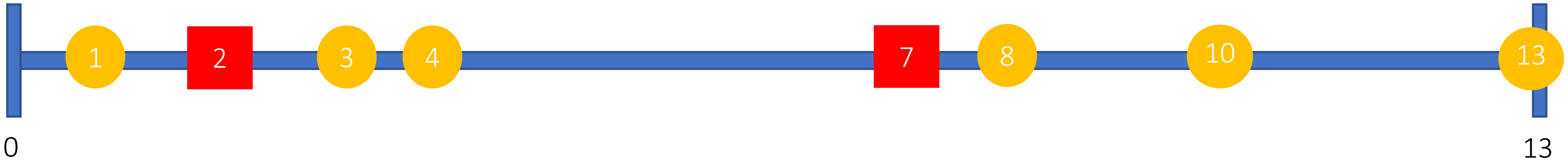
```
Select K seed points as initial centroids
Repeat:
    Form K clusters by assigning observations to its closest centroid (default Euclidean Distance)
    Update the centroid of each cluster
Until no more change in the membership and no change in centroids
```



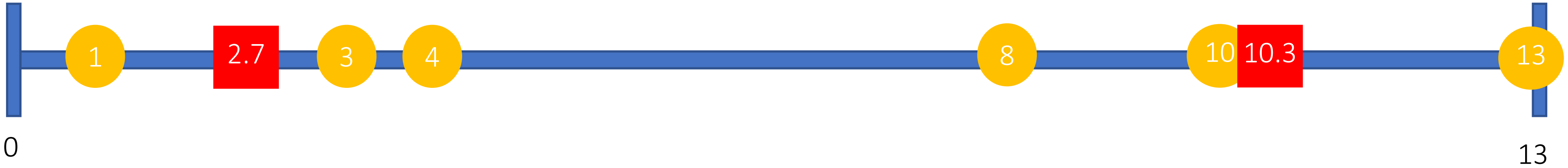
Partitional Clustering: K-Means 1D example (1/2)

1D data set: {1,3,4,8,10,13}

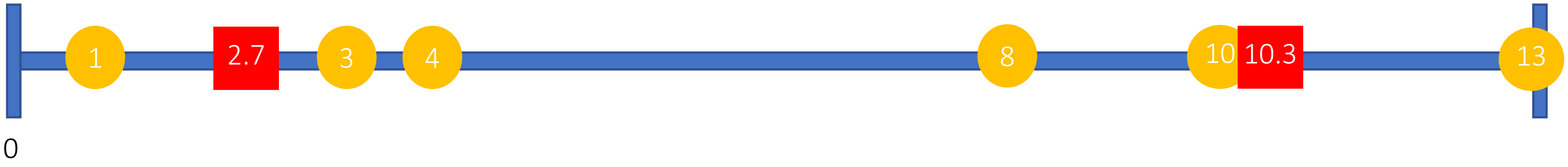
- Use K-means to create two clusters, K=2
- Randomly select the below centroids as a start
  - Cluster 1's centroid is : {2}
  - Cluster 2's centroid is : {7}



- Round 1: Assign membership of all points based on the nearest centroid
  - {1,3,4} are nearest to cluster 1's centroid {2}; cluster 1's new centroid is : {2.7} or (1+3+4)/3
  - {8,10,13} are nearest to cluster 2's centroid {7}; cluster 2's new centroid is : {10.3} or (8+10+13)/3



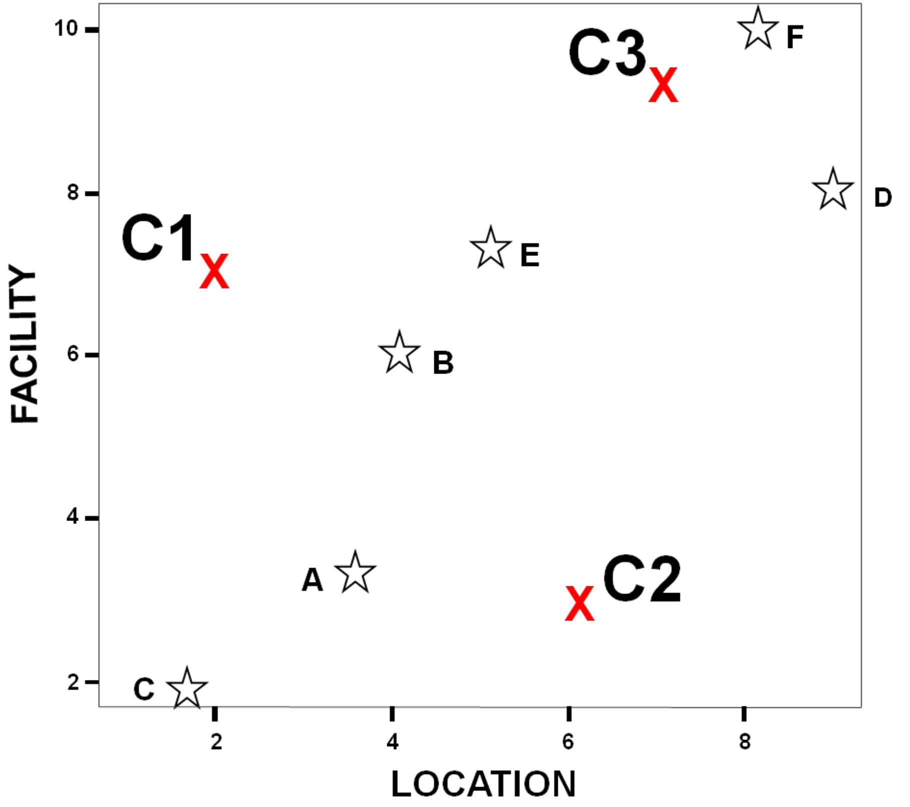
Partitional Clustering: K-Means 1D example (2/2)



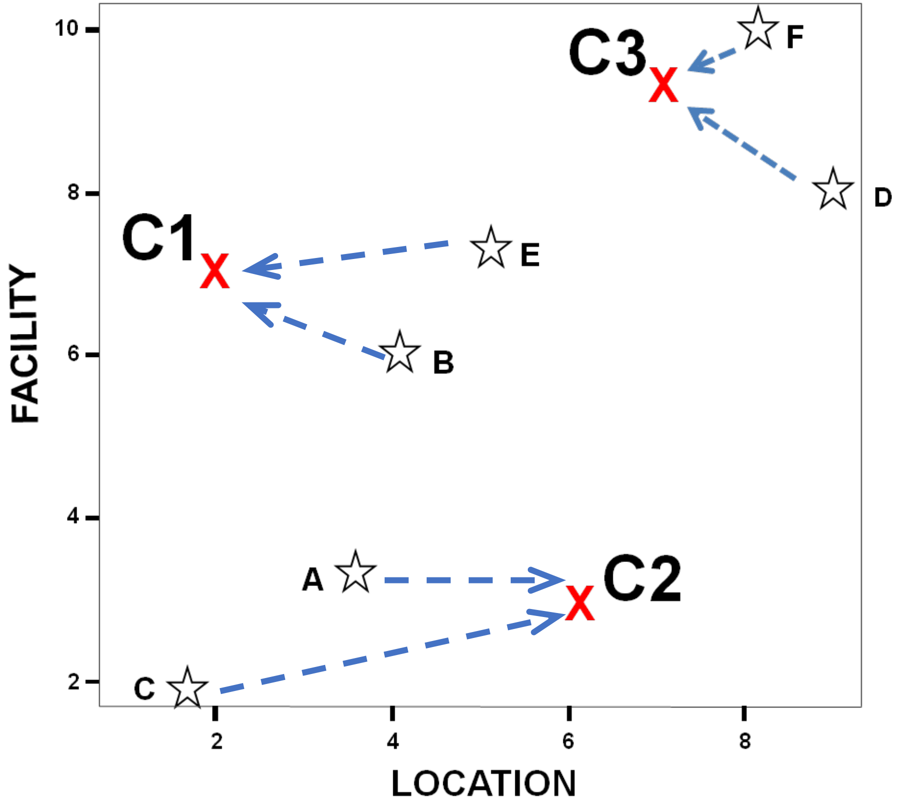
- Round 2: Check again the membership of all points based on nearest centroids
  - Do {1,3,4} still belong to cluster 1 with centroid {2.7}? □
  - Do {8,10,13} still belong to cluster 2 with centroid {10.3}? □
- No change in membership and no change in centroids
- Stop the algorithm

Other stopping criteria can be when the algorithm reaches max number of iterations

Partitional Clustering: K-Means 2D example (1/2)

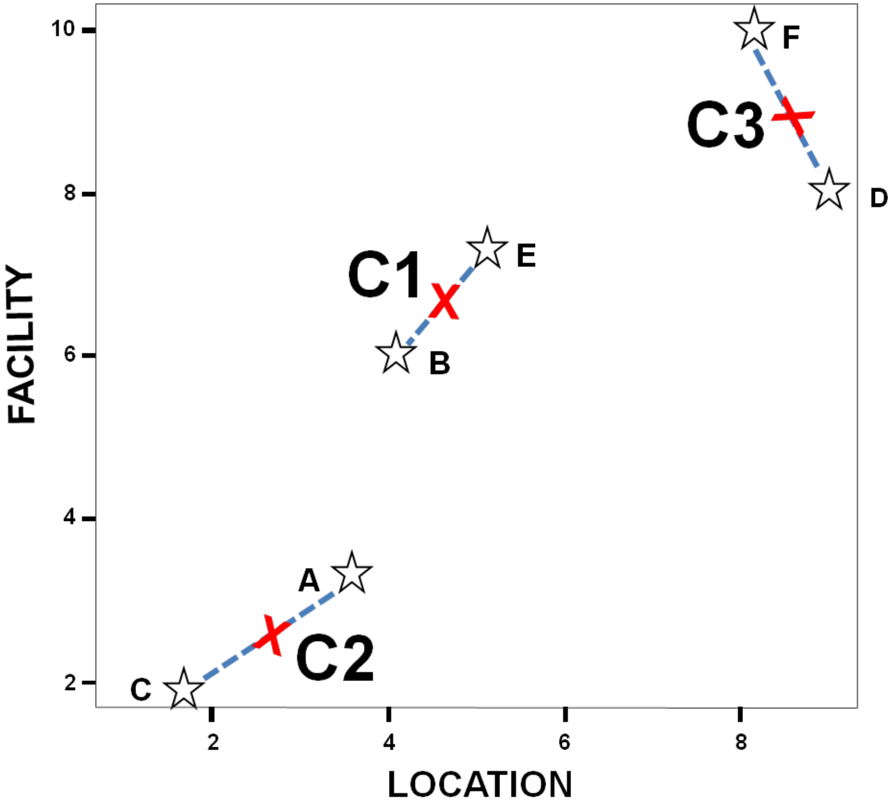


- Set k = 3
- Randomly set 3 centroids

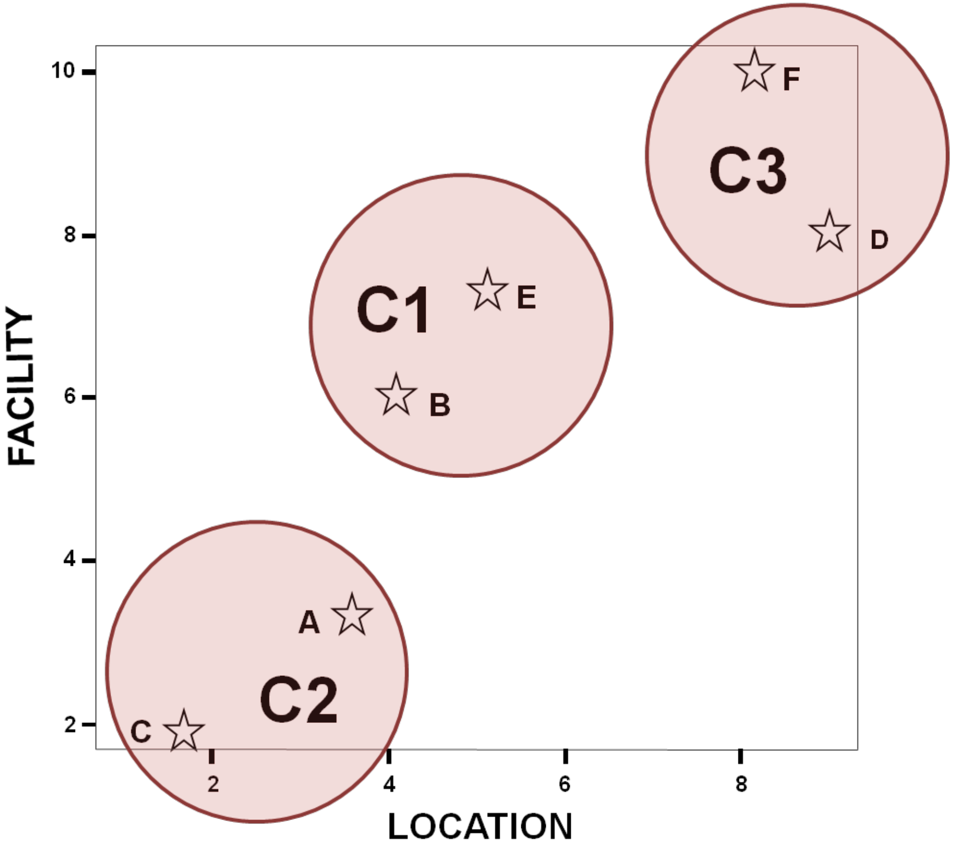


- A and C nearest to C2’s centroid
- D and F nearest to C3’s centroid
- B and E nearest to C1’s centroid

Partitional Clustering: K-Means 2D example (2/2)



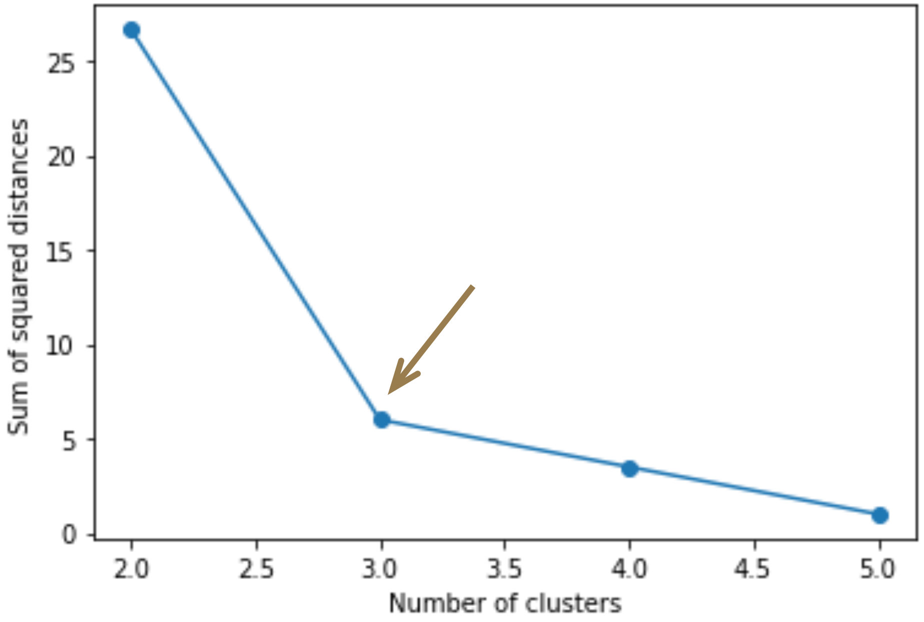
Update centroids of C1 C2 C3



One more check of membership shows no change in membership; 3 clusters formed

Determine the number of clusters – elbow method

- Plot sum of squared distances within all clusters, against a number of values for K
- Find the elbow point where the sum of squared distances goes up drastically
- Pick this value of K and rerun K-means model



Exercise: Understand the ‘for loop’ to plot elbow line; rerun K-Means with K=3 for hotel.csv data set

Final evaluation with Average Silhouette score

- Objective measures for evaluating the quality of clustering results
  - **Cohesion:** How close are the observations in a cluster
  - **Separation**: How far are the clusters from each other
  - **Parsimony**: Minimum number of clusters to capture the variations in the data set
- **Average Silhouette score:** range is [−1, +1]

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

a(i): average distance between row i and other observations in the same cluster

b(i): minimum average distance from i to all clusters where i does not belong

- s(i) measures how similar row i is to its own cluster (cohesion) as compared to other clusters (separation); **average of s(i)** is the Average Silhouette score

Other criteria

- Akaike's information Criterion (not covered)
- Bayesian Information Criterion(not covered)

Exercise: generate Average Silhouette score to assess the clustering quality

Issues with K-Means clustering

- **Sensitive to initial centroids**: selection of different initial centroids may give different results
- **Sensitive to outliers**: a small number of outliers can substantially influence the mean value of a cluster. It is advisable to remove outliers before performing k-means
- Very small clusters may **not** be detected
- Mainly generates **spherical clusters**, i.e., clusters that are elongated may be broken down into smaller round clusters

Closing notes on cluster analysis

- Largely an **exploratory** process
- **Different clustering results** may be obtained with different parameters and stopping rules
- Usually, more than one competing models are evaluated before the clustering solution is determined
- Business knowledge is essential to name the clusters in a meaningful way; the **interpretation and naming** of a cluster is a subjective or even creative task
- To ease visualization and analysis of each cluster, clustering results should **not have too many clusters**
- Number of **criteria to form any cluster** should **not be excessive**, so that the clustering results can be interpreted relatively easily

Exercise: Implement Agglomerative Clustering and K-means with FARM_CREDIT.csv

The emergence of Dimension Reduction

Curse of high dimensionality

- When dimension (no. of input variables) increases, data usually become **sparse** because the distribution of the data points is spread over a larger (vector) space

- Sparsity in high dimensional data makes the observations **appear dissimilar** in many ways, which prevents data understanding from being efficient

- High dimension also requires **heavy computation**

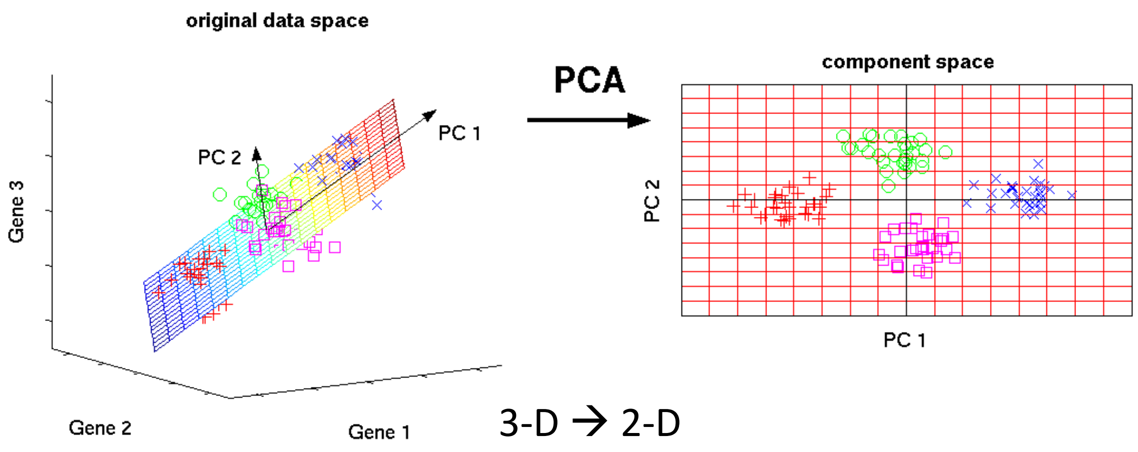
What is Dimension Reduction?

- Transformation from a high-dimensional space to a **low-dimensional** representation

- Must **preserve** the **essential characteristics** or information of the original high-dimensional data set

Principal Component Analysis (PCA)

- Unsupervised technique to perform Dimension Reduction
- Converts the possibly correlated high-dimensional input variables into a set of linearly uncorrelated variables, named **Principal Components**



What are Principal Components?

- New variables that are constructed as **linear combinations** of the original input variables, which shall be **uncorrelated** or orthogonal from each other
- They are **ranked** based on how much of original variance they can explain
- Usually, we only keep the first several principal components that keep at least **80% of the original information**
- **Limitation**: the resulted principal components likely do not have clear real-life meanings and **cannot be interpreted**

Exercise: Implement PCA

Recap and next lecture

Recap

- Clustering vs. Classification
- Proximity measure
- Hierarchical vs. Partitioning
- Agglomerative clustering (Single, Complete, Average, Ward’s)
- K-Means
- Determine the number of clusters
- Average Silhouette score
- Dimension Reduction
- Principal Component Analysis (PCA)

Next lecture

- Portfolio optimization using reinforcement learning