# Unsupervised Learning:

# Clustering Algorithms

CSIP5403 – Research Methods and AI Applications

Dr. Nathanael L. Baisa

# Lecture Content

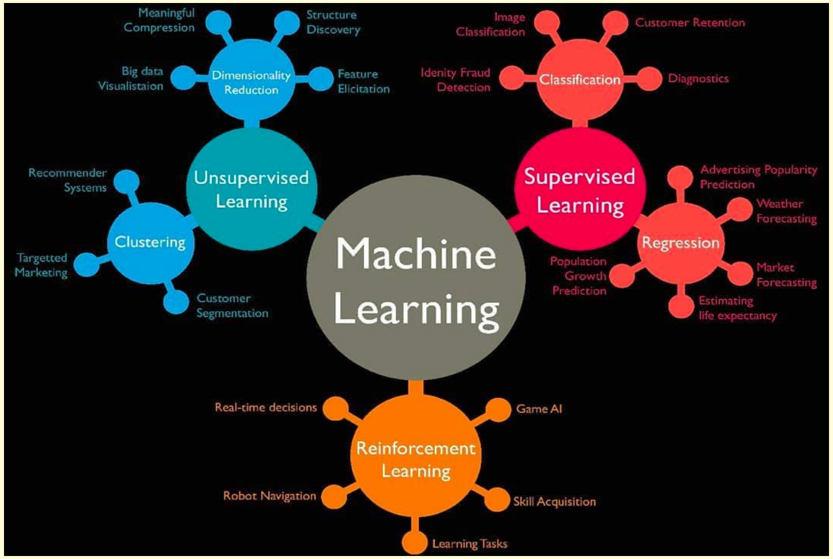
- Introduction
- Clustering Algorithms: What are they?
- Clustering Algorithms: k-means Clustering?
- Clustering Algorithms: Hierarchical Clustering?

# Session outcomes

- Understand what unsupervised learning is in detail.
- Acquire knowledge of k-means clustering.
- Gain knowledge of hierarchical clustering.
- Understand how to apply clustering algorithms for different applications.

DE MONTFORT UNIVERSITY LEICESTER

# Introduction



- **Unsupervised Learning Methods:**
  - No labelling of dataset is required.
  - Imitates learning by exploring.
  - These methods include:
    - **Clustering** - is viewed in the context of knowledge discovery. e.g. k-means.
    - **Dimensionality reduction** - is useful for big data visualization. e.g. PCA
- In this lecture, we focus on clustering algorithms.

# Clustering Algorithms

# What are they?

# Clustering Algorithms: What are they?

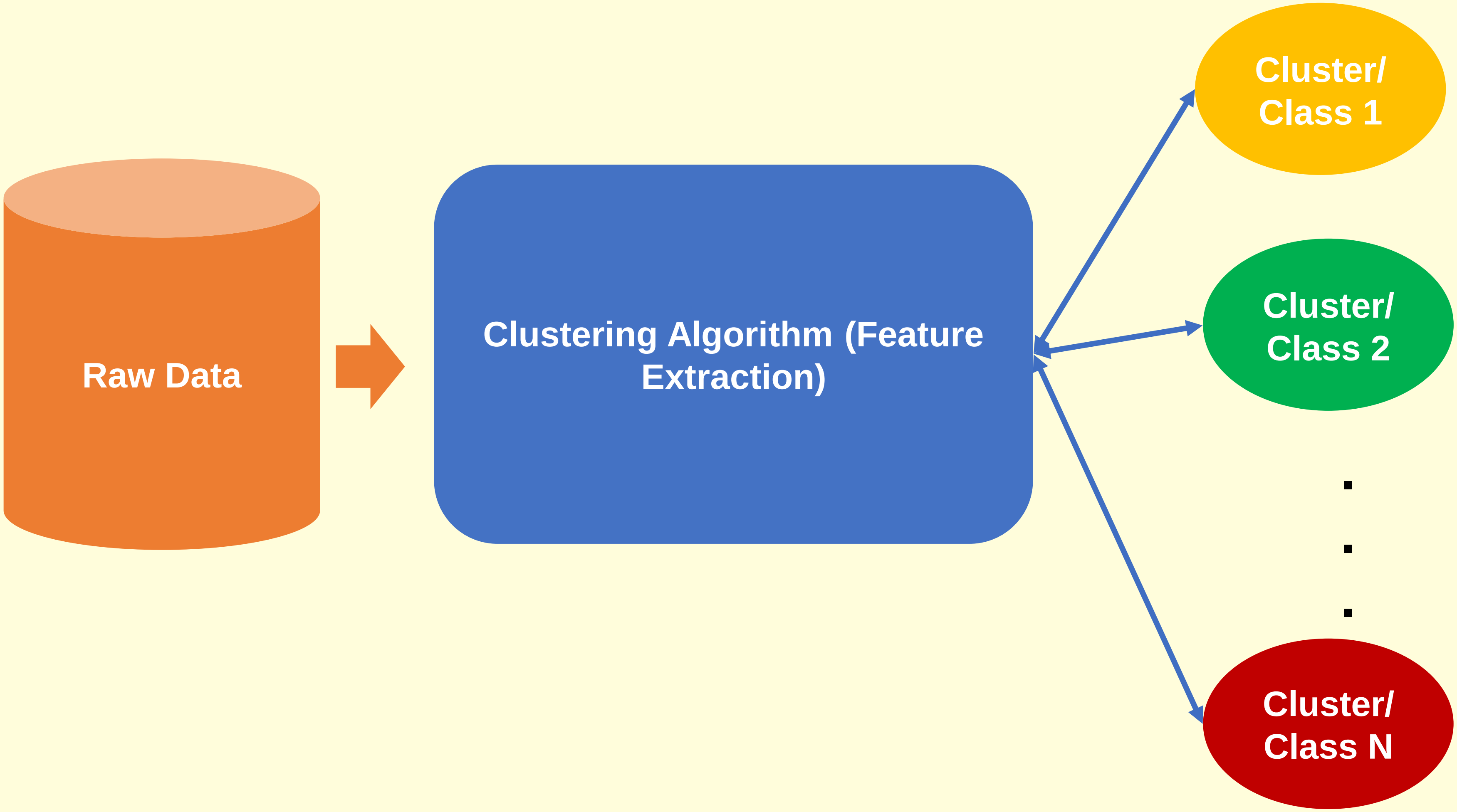
- ❑ **Clustering algorithms** group the data in an **unlabeled dataset** into ***class*** **or** ***cluster*** based on the underlying **hidden features (patterns)** in the data. Because there are **no labels**, there's **no way to evaluate the result** (a key difference from **supervised learning** algorithms). By grouping data through clustering algorithms, you **learn (discover)** something about the **raw data** that likely would not be visible otherwise. In highly dimensional or large datasets, the usefulness of such unsupervised learning algorithms become more prominent.

- ❑ In addition to **clustering data into groups**, the algorithm makes it possible to use these groups to understand the hidden features and exploit them in different applications.

- ❑ In theory, **data points** that are in the **same group** **should have similar properties and/or features,** while data points in **different groups** **should have highly dissimilar properties and/or features.**

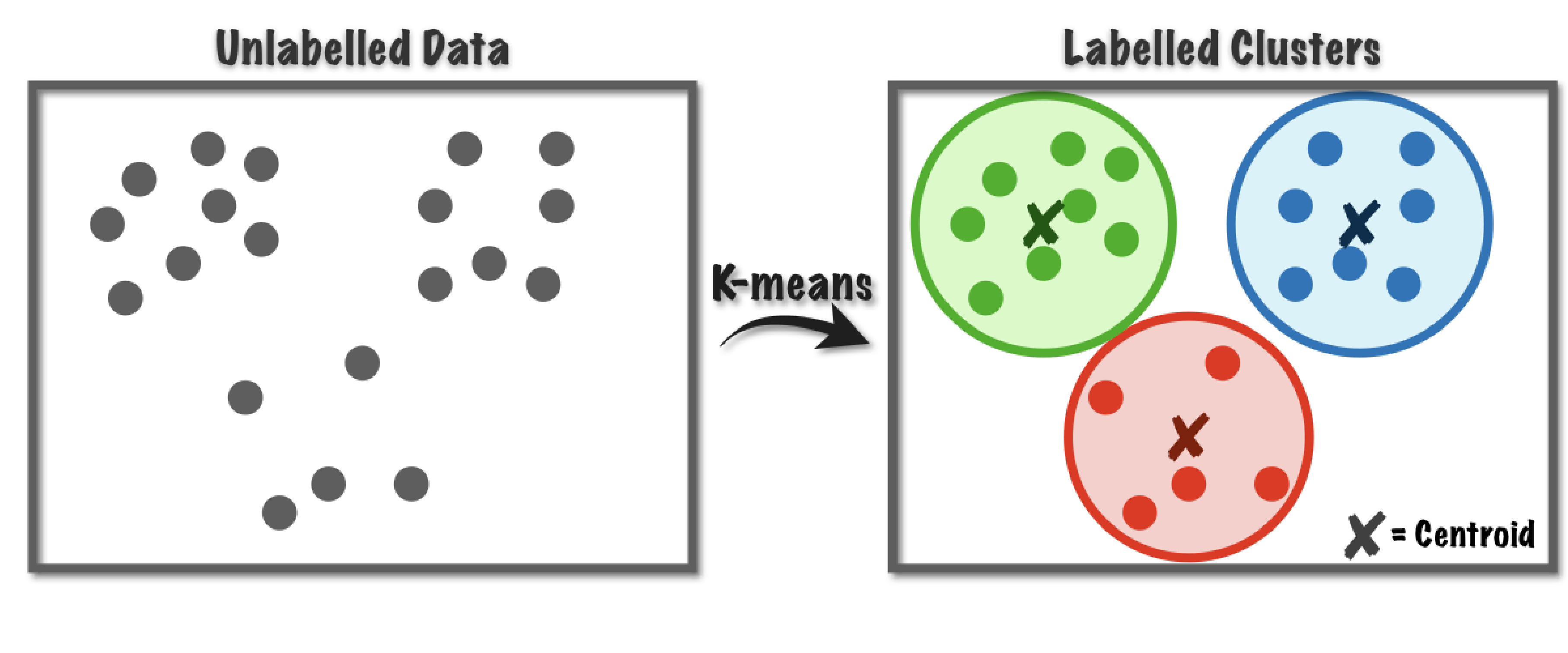
# Clustering Algorithms: What are they?

❑ General architecture of clustering algorithms:



# Clustering Algorithms: What are they?

- General architecture of clustering algorithms:



- https://medium.com/mlearning-ai/ml-k-means-clustering-5c11c1d2577b

# Clustering Algorithms

# A . k-means Clustering

DE MONTFORT UNIVERSITY LEICESTER

# k-means Clustering

- **k-means** is a partitional clustering algorithm.

- Let the set of data points *D* be

  $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$,

  where $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ is a vector in a real-valued space

  $x \subseteq R^r$, and $r$ is the number of attributes (dimensions) in the data.

- **K-means algorithm** partitions the given data into **k clusters**.
  - Each cluster has a cluster center, called **centroid**.
  - **k** is specified by the **user.**

# k-means Clustering
## Algorithm Steps

❑ Assume **k value is given**, then the k-means algorithm works as follows:

1. **Randomly choose k data points (*seeds*)** to be the initial centroids (cluster centers).
2. **Assign each data point** to the ***closest centroid***.
3. **Re-compute the centroids** using the current cluster memberships.
4. If a **convergence criterion** is ***not* met**, go to **step 2**.

# k-means Clustering
## Algorithm Steps Pseudocode

❑ **Pseudocode for the k-means algorithm (k, D) can be defined as:**

```
Select k data points as the initial centroids (cluster centres).
repeat
    for each data point $x \in D$ do
        compute the distance from x to each centroid;
        assign x to the closest centroid  /* a centroid represents a cluster */
    end
    compute the centroids using the current cluster memberships.
until stop criterion is true
```

# k-means Clustering

## Algorithm Steps Pseudocode Stop Criteria

❑ **Stop (*convergence*) Criteria:**

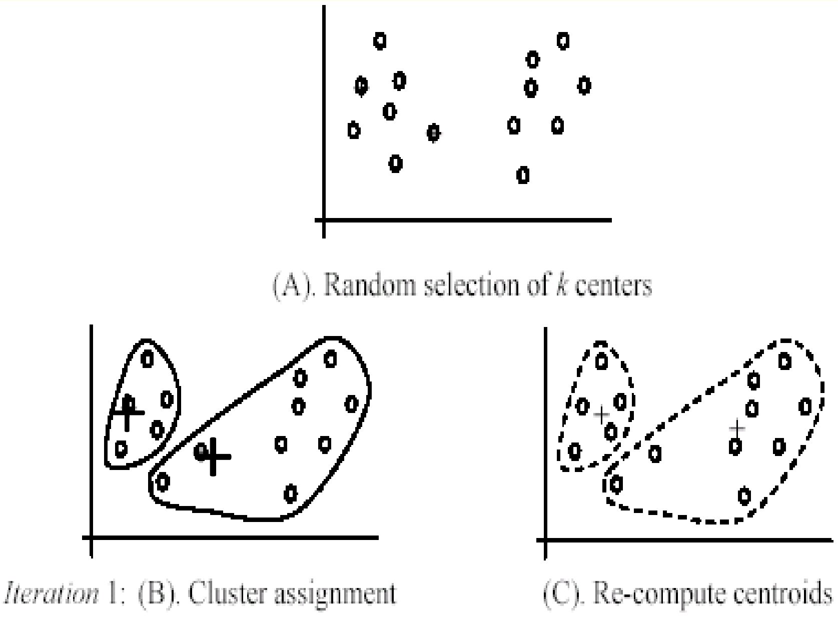
1. None or minimum re-assignments of data points to different clusters.
2. None or minimum change of centroids.
3. Minimum decrease in the **sum of squared error** (SSE),

$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} dist(\mathbf{x}, \mathbf{m}_j)^2 \qquad (1)$$

*where* $C_i$ is the *jth* cluster, $\mathbf{m}_j$ is the centroid of cluster $C_j$ (the mean vector of all the data points in $C_j$), and *dist*($\mathbf{x}$, $\mathbf{m}_j$) is the distance between data point $\mathbf{x}$ and centroid $\mathbf{m}_j$.

# k-means Clustering

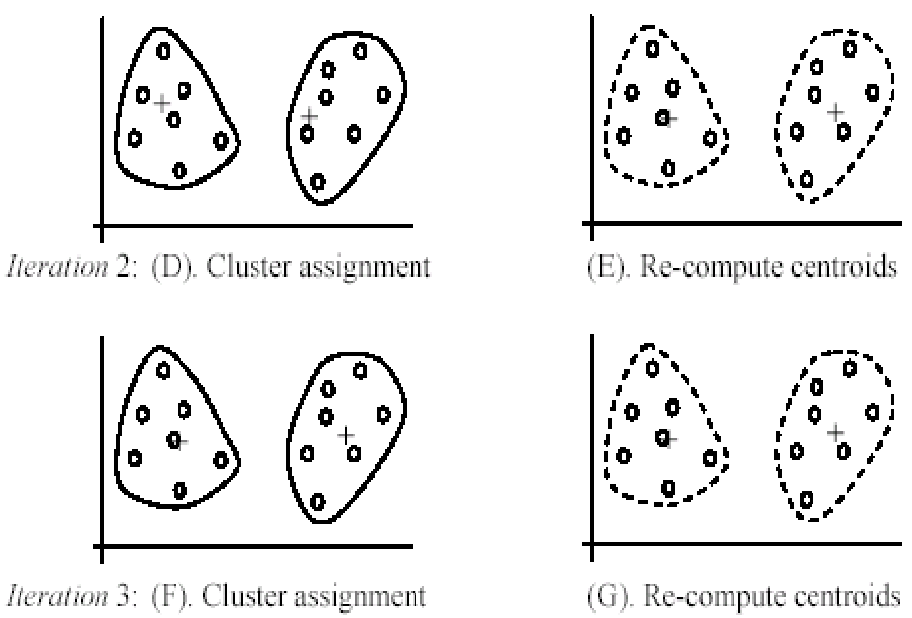
## Algorithm Steps Example



(A). Random selection of *k* centers

*Iteration* 1: (B). Cluster assignment

(C). Re-compute centroids



*Iteration* 2: (D). Cluster assignment

(E). Re-compute centroids

*Iteration* 3: (F). Cluster assignment

(G). Re-compute centroids

# k-means Clustering
## Distance Function

- **k-means algorithm** can be used for any application dataset where the mean can be defined and computed. In the **Euclidean space**, the mean of a cluster is computed with:

$$m_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i \qquad (2)$$

where $|C_j|$ is the number of data points in cluster $C_j$. The distance from one data point $x_i$ to a mean (centroid) $m_j$ is computed with:

$$dist(x_i, m_j) = \|x_i - m_j\| \qquad (3)$$

$$= \sqrt{(x_{i1} - m_{j1})^2 + (x_{i2} - m_{j2})^2 + \cdots + (x_{in} - m_{jn})^2}$$

# Clustering Algorithms

# k-means Pros and Cons

# k-means Clustering
## Pros

- ❑ **Simple**: easy to understand and to implement.
- ❑ **Efficient**: computational time complexity is ***O*(tkn)**

  where n is the number of data points,

  k is the number of clusters, and

  t is the number of iterations.

  both k and t are small.

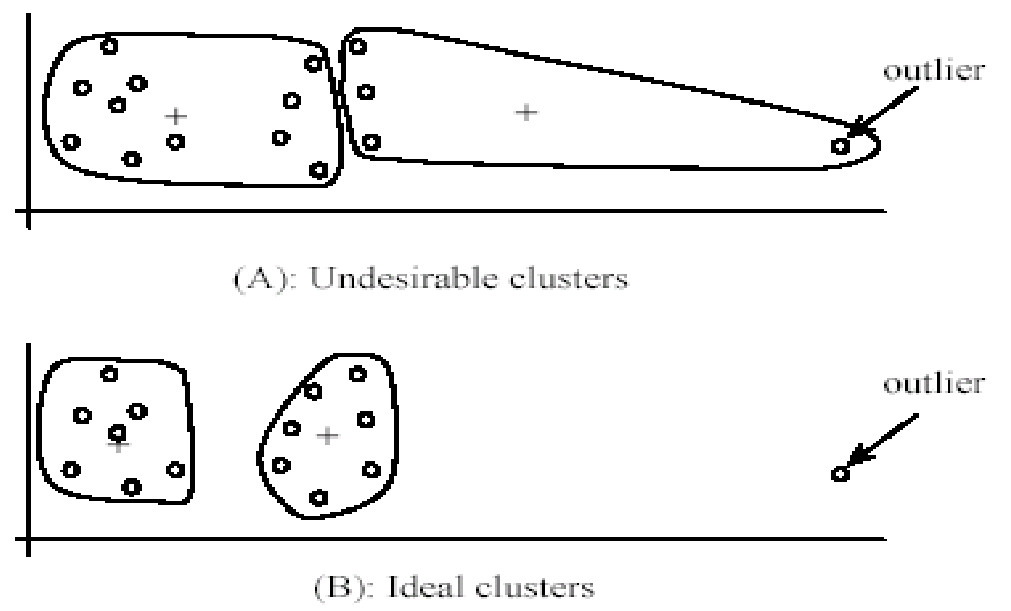
- ❑ **k-means is the most popular clustering algorithm.**

➢ ***NB:*** *k-means may fail to produce an optimal solution if it stacks at a local optimum when **SSE** is used rather than converge at the global optimum.*

# k-means Clustering
## Cons

- **k-means algorithm** is only applicable if the **mean** is defined.
- **User needs to specify k**.
- The algorithm is **sensitive to outliers**
    - **Outliers** are data points that are **very far away** from other data points.
    - **Outliers** could be **errors in the data recording** or **some special data points with very different values.**

# k-means Clustering
## Cons



(A): Undesirable clusters

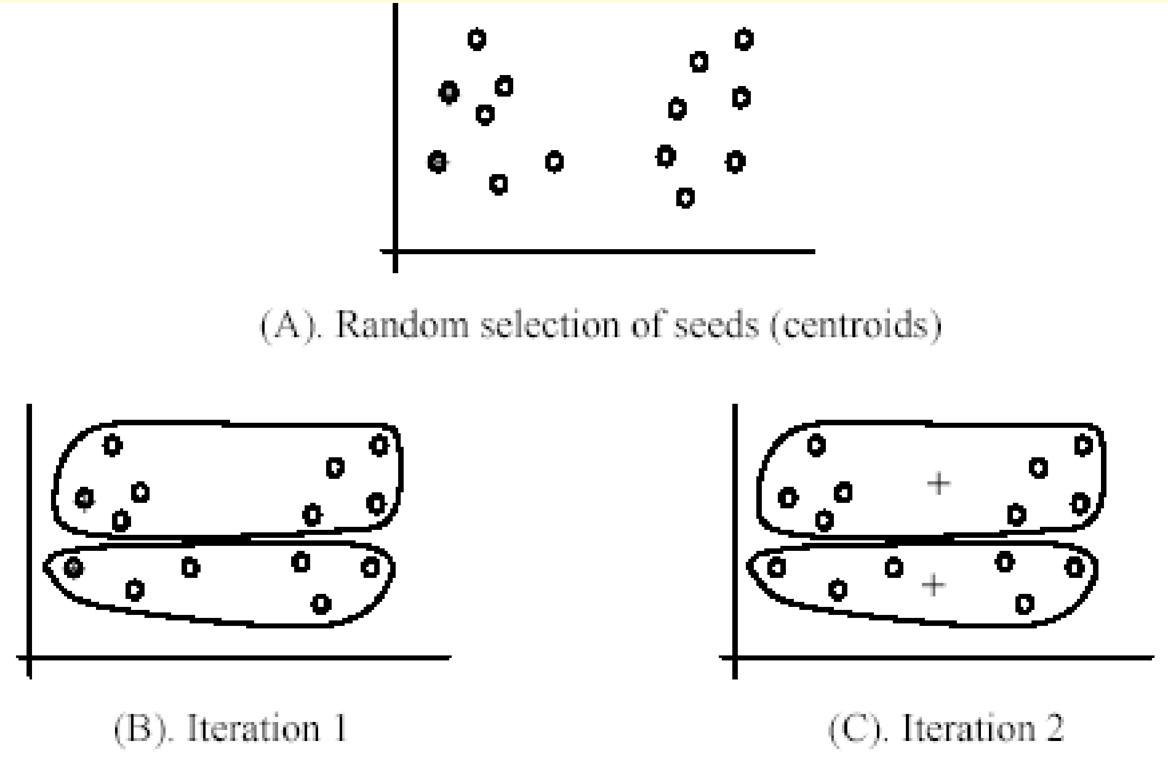
(B): Ideal clusters

# k-means Clustering
## Cons

- **To deal with Outliers:**
  - One method is to **remove some data points** in the clustering process that are **much farther away from the centroids** than other data points.
    - We may want to **monitor these possible outliers over a few iterations** and then decide to remove them.
  - Another method is to perform **random sampling**. Since in sampling we only choose a small subset of the data points, the chance of selecting an **outlier** is very small.
    - Assign the rest of the data points to the clusters by **distance** or **similarity comparison**, or **classification**.

# k-means Clustering
## Cons

- ❑ **k-means algorithm is sensitive to *initial seeds*.**



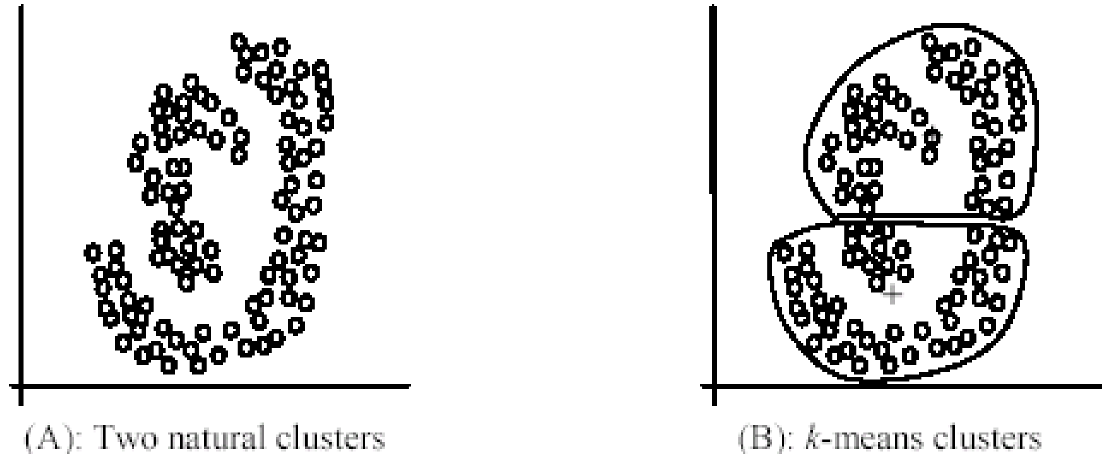
(A). Random selection of seeds (centroids)

(B). Iteration 1

(C). Iteration 2

# k-means Clustering
## Cons

- ❑ **k-means algorithm** is **not suitable** for discovering clusters that are **not hyper-ellipsoids** or in higher-dimension problems that are **not hyper-spheres**.
- ❑ In effect, clusters that have **convex shapes** can be easily classified using k-means than clusters which have **non-convex shapes**.



# Clustering Algorithms

# k-means: summary

# Clustering Algorithms
## k-means summary

- ❑ Despite weaknesses, **k-means remains the most popular algorithm** due to its **simplicity**, **efficiency** and since other clustering algorithms have also their own lists of cons.

- ❑ **No clear evidence** that any **other clustering algorithm performs better,** in general, although they may be more suitable for some specific types of data or applications.

- ❑ **Ground truthing** for comparing different clustering algorithms is a **difficult task.** Can you be certain what are the correct clusters?

DE MONTFORT
UNIVERSITY
LEICESTER

# Clustering Algorithms

# Clusters Representation

# Clustering Algorithms
## Clusters Representation

- ❑ **Hyper-elliptical** (or ellipsoid of dimension n-1 in Euclidean space of dimension n) and **hyper-spherical** clusters are usually **easy** to represent, using their **centroids** together with **spreads**.

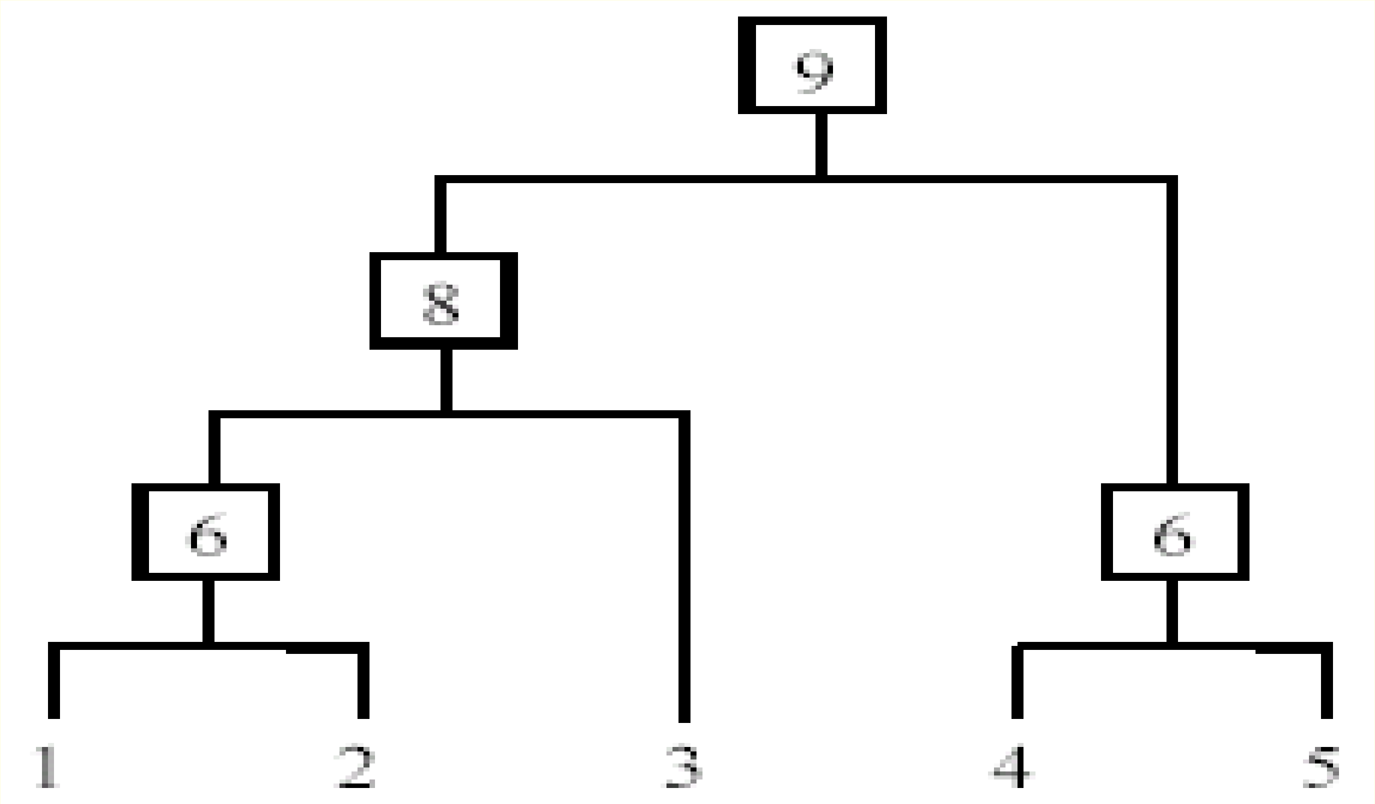
- ❑ **Irregular shape** clusters are **hard** to represent.

# Clustering Algorithms

# B. Hierarchical Clustering

# Hierarchical Clustering

- **Hierarchical clustering** produces a **nested sequence** of **clusters**, a tree, also called ***Dendrogram***.



# Hierarchical Clustering

## ❑What are the different types?

➢**Agglomerative (bottom up) clustering**: It builds the dendrogram (tree) from the **bottom level:**

- It merges the most similar (or nearest) pair of clusters.
- It stops when all the data points are merged into a single cluster i.e. the root cluster;

➢**Divisive (top down) clustering:** It starts with **all data points** in **one cluster**, the ***root:***

- **Splits** the **root** into **a set of *child clusters***. Each child cluster is *recursively divided* further;
- **Stops** when **only *singleton* clusters** of individual data points remain. ***NB: Singleton cluster is a cluster with only a single point.***

# Hierarchical Clustering
## Agglomerative

- **Agglomerative** hierarchical clustering is **more popular** than **divisive methods.**

- **Agglomerative hierarchical clustering algorithm works as follows:**
  - At the beginning, **each data point** forms a **cluster** (also called a **node**).
  - **Merge** nodes/clusters that have the **least distance**.
  - Go on merging
  - Eventually **all nodes** belong to **one cluster**

# Hierarchical Clustering
## Measuring the distance of two clusters

- ❑ A few ways to measure **distances of two clusters** in **agglomerative hierarchical clustering**. Each way results in different variations of the algorithm.

  - ➢ **Single link -** two clusters with the minimum distance are merged.
  - ➢ **Complete link -** two clusters with the maximum distance are merged. This method is also known as farthest neighbour clustering.
  - ➢ **Average link -** this method uses the average pair-wise proximity among all pairs of objects in different clusters. Clusters are merged based on their lowest average distances.
  - ➢ **Centroids -** two clusters with the lowest centroid distance are merged.

- ❑ For **more details**, look at:
  - ➢ https://levelup.gitconnected.com/distance-measures-and-linkage-methods-in-hierarchical-clustering-8b7d488d7ebc

# Clustering Algorithms

## Hierarchical Clustering: Complexity

# Hierarchical Clustering Complexity (distance functions)

- ❑**All the algorithms** are at least **O($n^2$)** where **n** is the number of data points.
- ❑***Single link*** can be done in **O($n^2$)**.
- ❑***Complete*** and ***average links*** can be done in ***O($n^2$logn)***.
- ❑**Due to the complexity**, it is **hard to use for large data sets**.
  - ➢ Sampling
  - ➢ Scale-up methods e.g. **BIRCH** (Balanced Iterative Reducing and Clustering using Hierarchies) algorithm.

# Conclusion

- Clustering Algorithms: What are they?
- Clustering Algorithms: k-means Clustering?
- Clustering Algorithms: Hierarchical Clustering?

# APPENDIX

- Distance Functions
- Cluster Evaluation

# Distance functions

- Key to clustering. “similarity” and “dissimilarity” are the commonly used terms.
- There are numerous distance functions for
  - Different types of data
    - Numeric data
    - Nominal data
  - Different specific applications

# Distance functions for numeric attributes

- Most commonly used functions are
  - Euclidean distance and
  - Manhattan (city block) distance
- We denote distance with: *dist*(**x**$_i$, **x**$_j$), where **x**$_i$ and **x**$_j$ are data points (vectors)
- They are special cases of Minkowski distance. h is positive integer.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = ((x_{i1} - x_{j1})^h + (x_{i2} - x_{j2})^h + \ldots + (x_{ir} - x_{jr})^h)^{\frac{1}{h}}$$

# Euclidean distance & Manhattan distance

- If *h* = 2, it is the Euclidean distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{ir} - x_{jr})^2}$$

- If *h* = 1, it is the Manhattan distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = | x_{i1} - x_{j1} | + | x_{i2} - x_{j2} | + ... + | x_{ir} - x_{jr} |$$

- Weighted Euclidean distance

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{w_1(x_{i1} - x_{j1})^2 + w_2(x_{i2} - x_{j2})^2 + ... + w_r(x_{ir} - x_{jr})^2}$$

# Squared distance & Chebychev distance

- Squared Euclidean distance: to place progressively greater weight on data points that are farther apart.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = (x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \ldots + (x_{ir} - x_{jr})^2$$

- Chebychev distance: one wants to define two data points as "different" if they are different on any one of the attributes.

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \max(| x_{i1} - x_{j1} |, | x_{i2} - x_{j2} |, \ldots, | x_{ir} - x_{jr} |)$$

# Cluster Evaluation: hard problem

- The quality of a clustering is very hard to evaluate because
  - We do not know the correct clusters.
- Some methods are used:
  - User inspection
    - Study centroids, and spreads
    - Rules from a decision tree.
    - For text documents, one can read some documents in clusters.

# Cluster evaluation: ground truth

- We use some labeled data (for classification)
- Assumption: Each class is a cluster.
- After clustering, a confusion matrix is constructed. From the matrix, we compute various measurements, entropy, purity, precision, recall and F-score.
  - Let the classes in the data *D* be

    $C = (c_1, c_2, \ldots, c_k)$.

    The clustering method produces *k* clusters, which divides *D*

    into *k* disjoint subsets, $D_1, D_2, \ldots, D_k$.