International Research Journal of Education and Technology
Peer Reviewed Journal, **ISSN 2581-7795**

Exploring Data Patterns: Clustering Approaches in Unsupervised Learning

1.Ms.A. KAMATCHI
Research Scholar, Department of Computer Science,
A.V.V.M.Sri Pushpam College(Autonomous),Poondi,Thanjavur(Dt),Affiliated to Bharathidasan University,Thiruchirappalli,Tamilnadu
2.Dr. V. MANIRAJ
Associate Professor, Research Supervisor, Head of the Department,
Department of Computer Science,
A.V.V.M.Sri Pushpam College(Autonomous),Poondi,Thanjavur(Dt),Affiliated to Bharathidasan University,Thiruchirappalli,Tamilnadu

Abstract

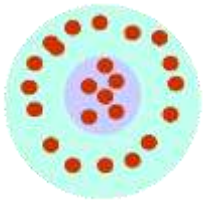
Unsupervised learning plays a critical role in modern data analysis by identifying hidden structures within unlabeled datasets. Among its various techniques, **clustering** is one of the most widely used methods for uncovering natural groupings and patterns in data without prior knowledge of outcomes. This paper explores the fundamental concepts, methodologies, and real-world applications of clustering within the realm of unsupervised learning. Common clustering algorithms such as **K-Means, Hierarchical Clustering, DBSCAN**, and **Gaussian Mixture Models** are reviewed and compared in terms of their strengths, limitations, and suitability for different types of data. The study emphasizes the importance of selecting appropriate clustering methods based on dataset characteristics and desired outcomes. By revealing inherent structures in data, clustering serves as a foundational tool in fields ranging from market segmentation and social network analysis to image recognition and bioinformatics.

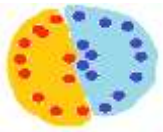
Introduction

In the era of big data, uncovering meaningful insights from vast and unstructured datasets has become a major challenge across various fields. Unsupervised learning, a branch of machine learning, addresses this challenge by identifying hidden patterns in data without the use of labeled outputs. Among the core techniques in unsupervised learning, clustering stands out as a fundamental approach for discovering structure, segmenting data, and revealing relationships between observations. Clustering algorithms work by grouping data points based on similarity, allowing analysts to identify natural divisions and trends that are not immediately apparent. These techniques are widely used in applications such as customer segmentation, anomaly detection, image recognition, and genomic data analysis. This paper explores the principles and methods of clustering in unsupervised learning, focusing on popular algorithms like K-Means, Hierarchical Clustering, DBSCAN, and Gaussian Mixture Models. By analyzing how these algorithms operate and their suitability for different data types, this study aims to provide a comprehensive understanding of how clustering can be used to explore and interpret complex datasets.

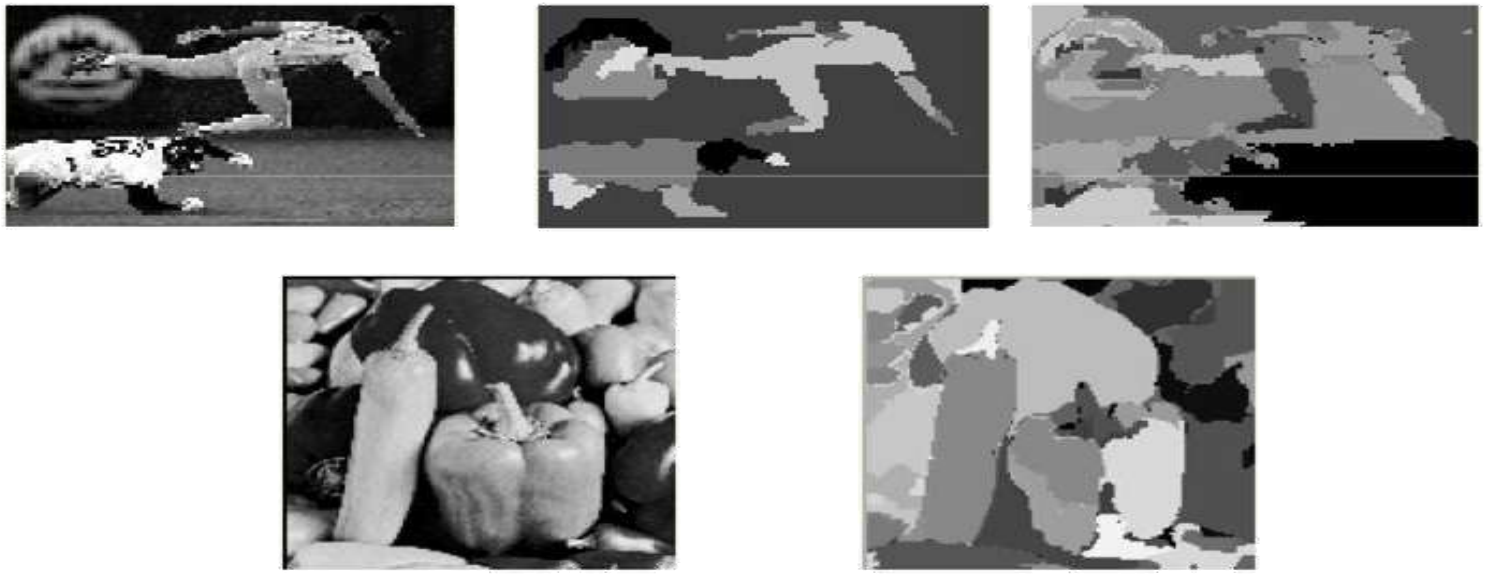
What is clustering?

The organization of unlabeled data into similarity groups called clusters. A cluster is a collection of data items which are “similar” between them, and “dissimilar” to data items in other clusters.





Historic application of clustering Computer vision application:
Image segmentation

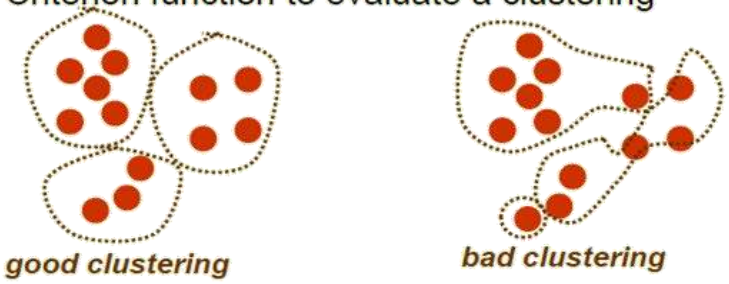


From: Image Segmentation by Nested Cuts, O. Veksler, CVPR2000

What do we need for clustering?

1. Proximity measure, *either*
   - similarity measure $s(x_i,x_k)$: large if x_i,x_k are similar
   - dissimilarity(or distance) measure $d(x_i,x_k)$: small if x_i,x_k are similar

2. Criterion function to evaluate a clustering



3. Algorithm to compute clustering
   - For example, by optimizing the criterion function

Distance (dissimilarity) measures

Intra-cluster cohesion (compactness):

- Cohesion measures how near the data points in a cluster are to the cluster centroid.
- Sum of squared error (SSE) is a commonly used measure.

- **Inter-cluster separation** (isolation):
  - Separation means that different cluster centroids should be far away from one another.

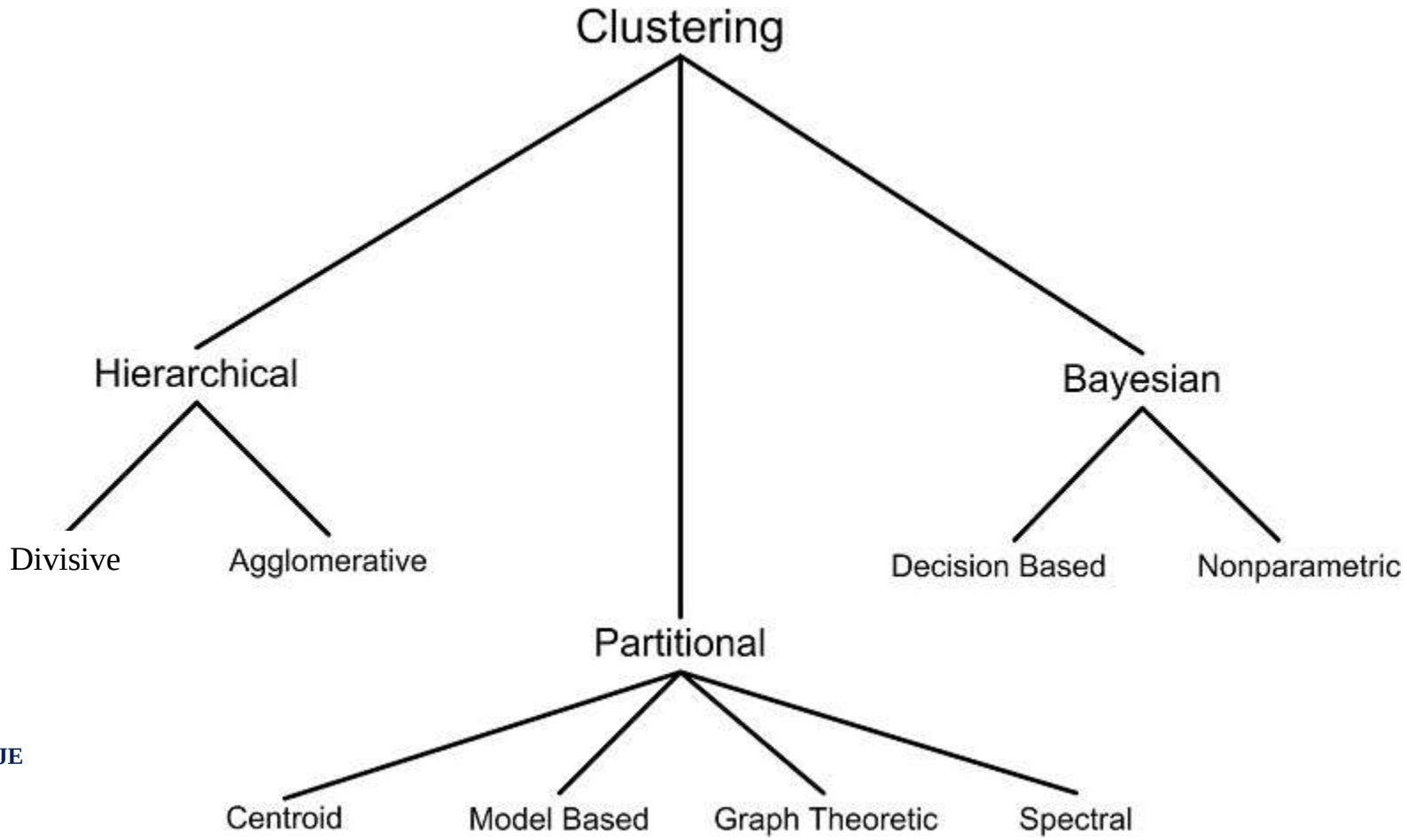
- In most applications, expert judgments are still the key

How many clusters?

3 clusters or 2 clusters?

- Possible approaches
  1. fix the number of clusters to ***k***
  2. find the best clustering according to the criterion function (number of clusters may vary)

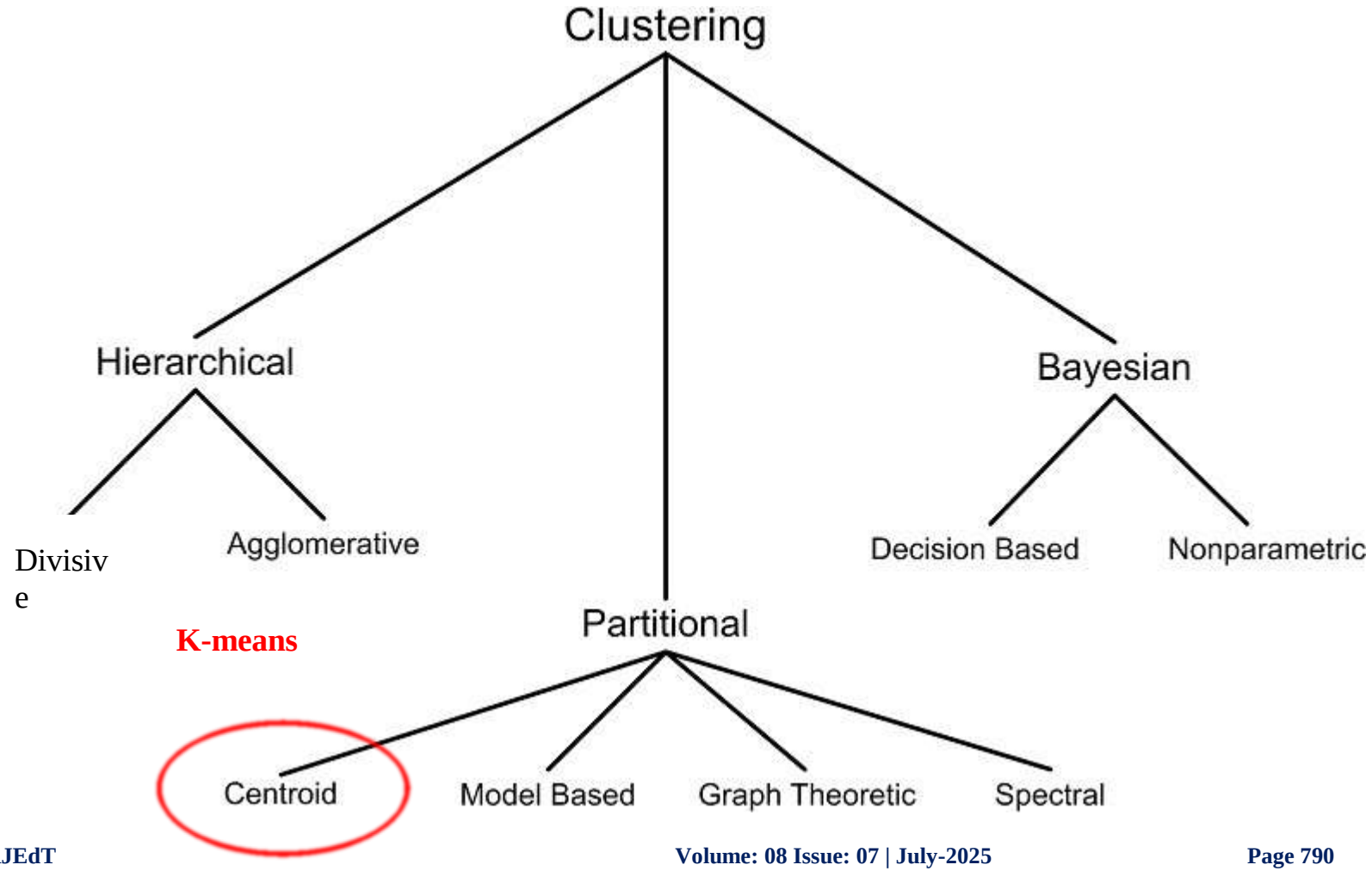
Clustering techniques



Clustering techniques

- **Hierarchical** algorithms find successive clusters using previously established clusters. These algorithms can be either agglomerative ("*bottom-up*") or divisive ("*top-down*"):
  1. Agglomerative algorithms begin with each element as a separate cluster and merge them into successively larger clusters;
  2. Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters.
- **Partitional** algorithms typically determine all clusters at once, but can also be used as divisive algorithms in the hierarchical clustering.
- **Bayesian** algorithms try to generate a *posteriori distribution* over the collection of all partitions of the data.

Clustering techniques



K-Means clustering

K-means (MacQueen, 1967) 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 The 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 algorithm

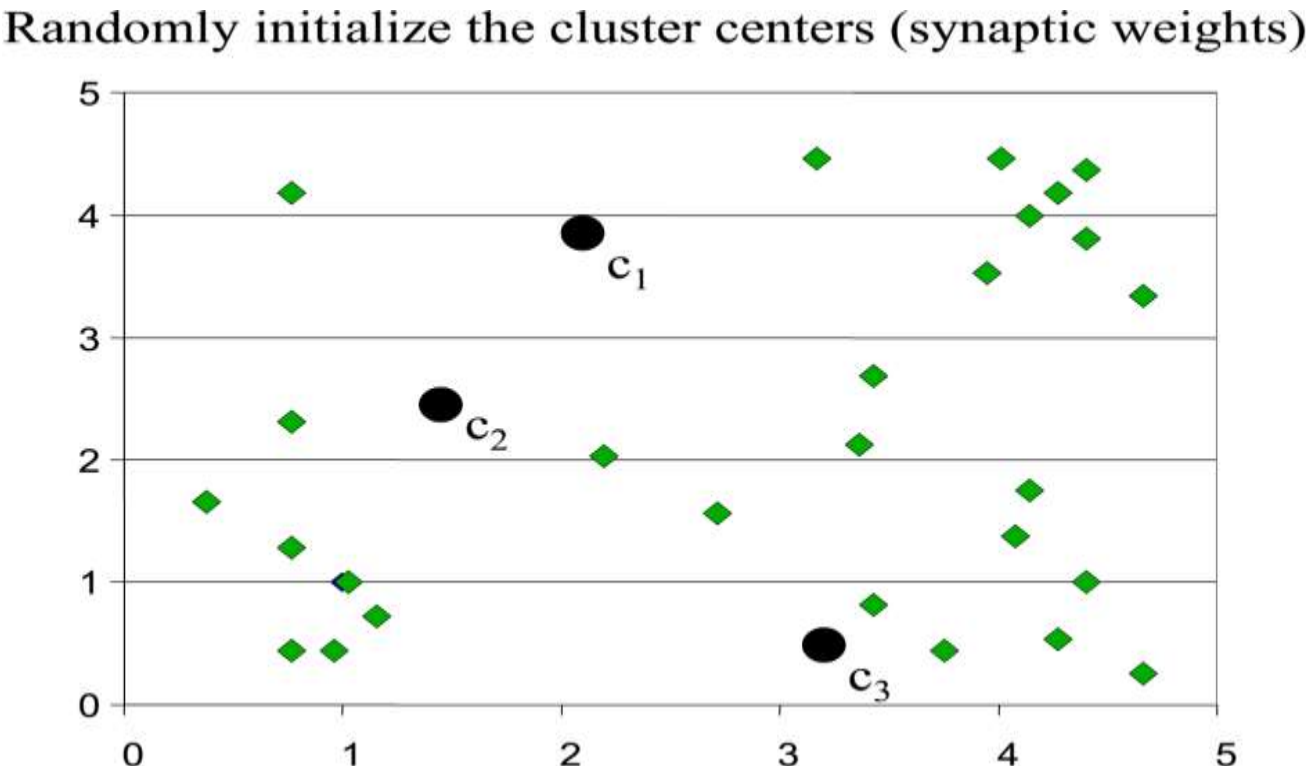
Given k, the *k-means* algorithm works as follows:
1.Choose k (random) data points (seeds) to be the initial centroids, cluster centers
2.Assign each data point to the closest centroid
3.Re-compute the centroids using the current cluster memberships
4.If a convergence criterion is not met, repeat steps 2 and 3

K-means convergence (stopping) criterion

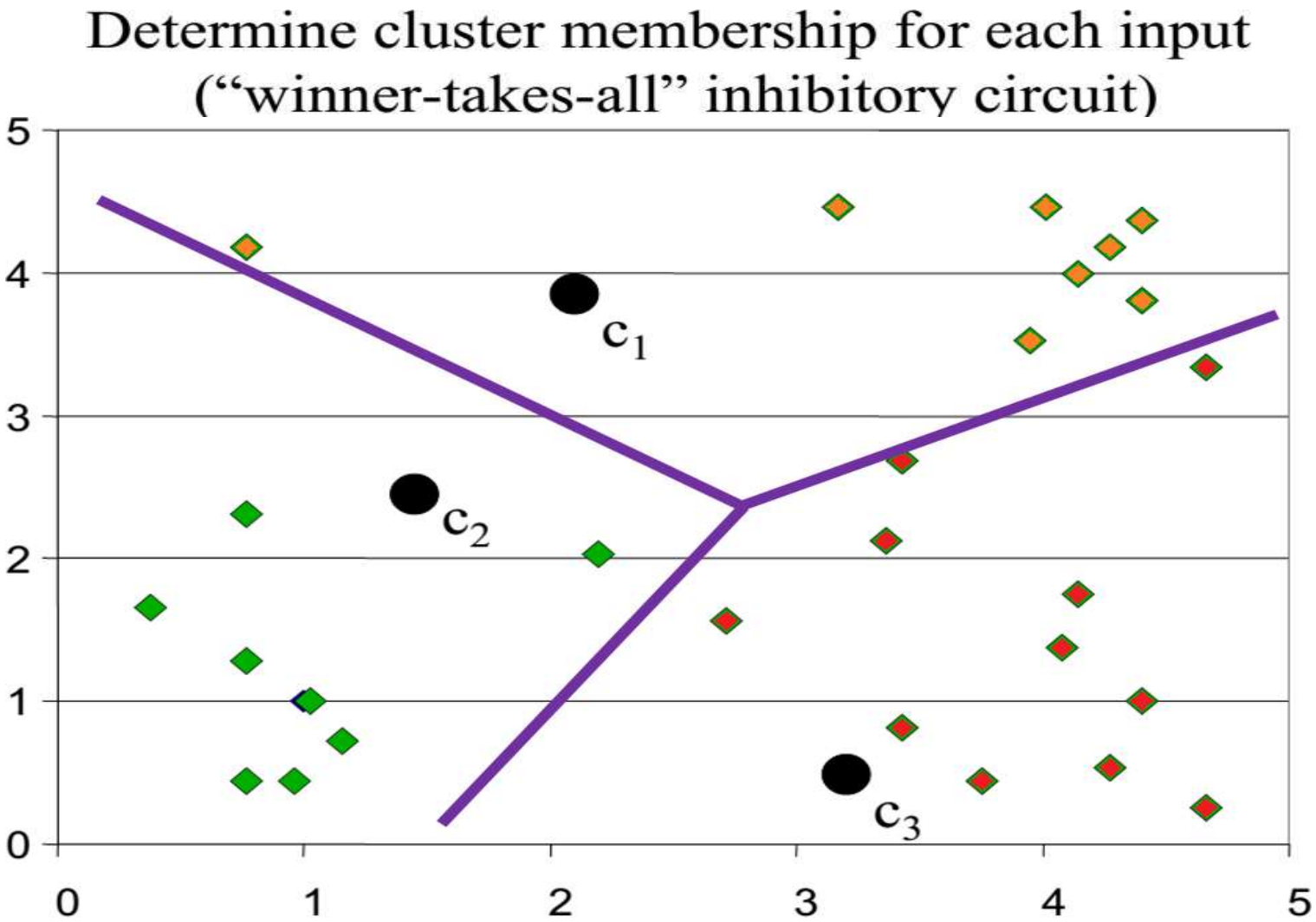
- no (or minimum) re-assignments of data points to different clusters, *or*
- no (or minimum) change of centroids, *or*
- minimum decrease in the **sum of squared error** (SSE),

- $d(\mathbf{x}, \mathbf{m}_j)$ is the (Eucledian) distance between data point $\mathbf{x}$ and centroid $\mathbf{m}_j$.

K-means clustering example:
step 1

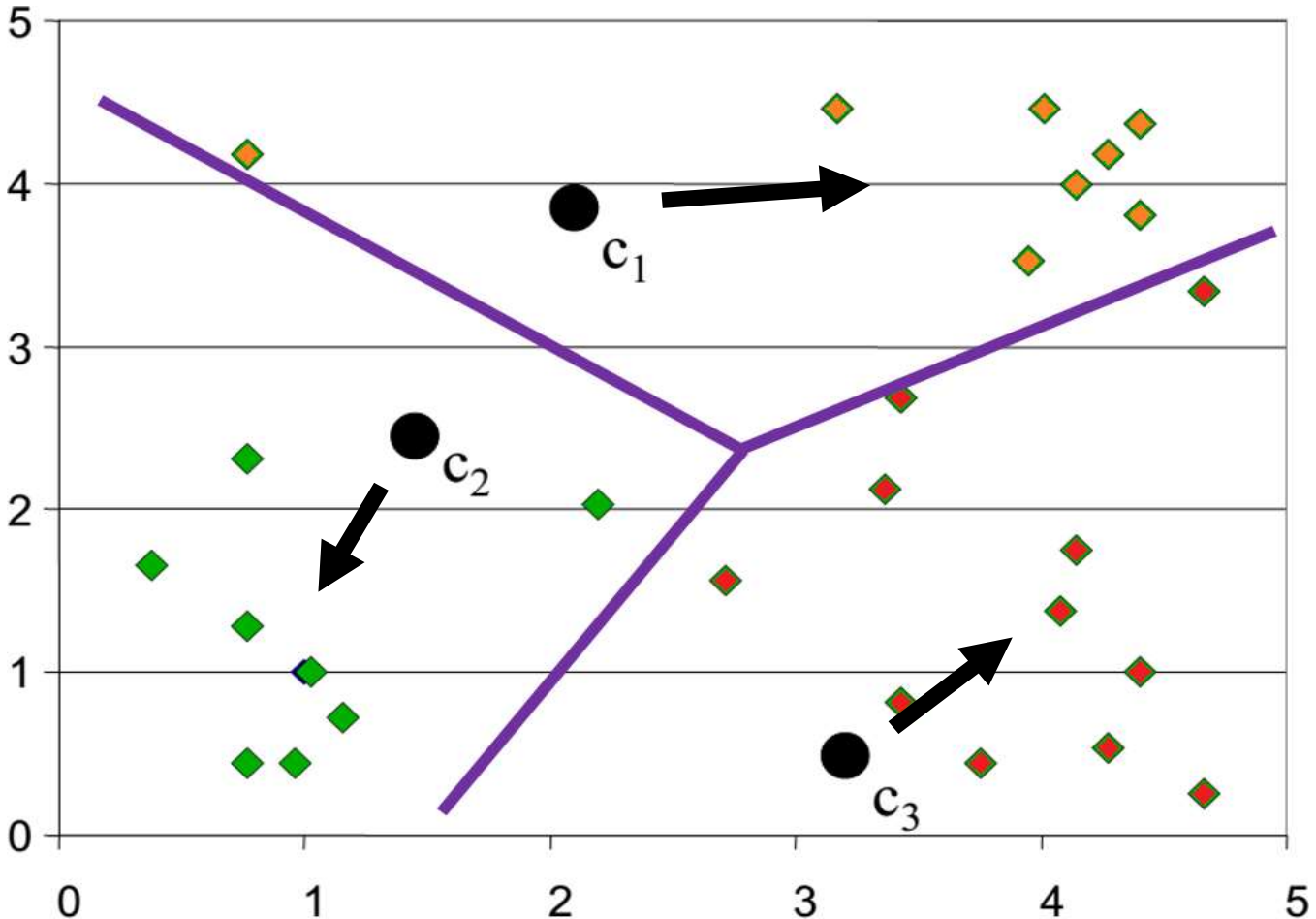


K-means clustering example – step 2



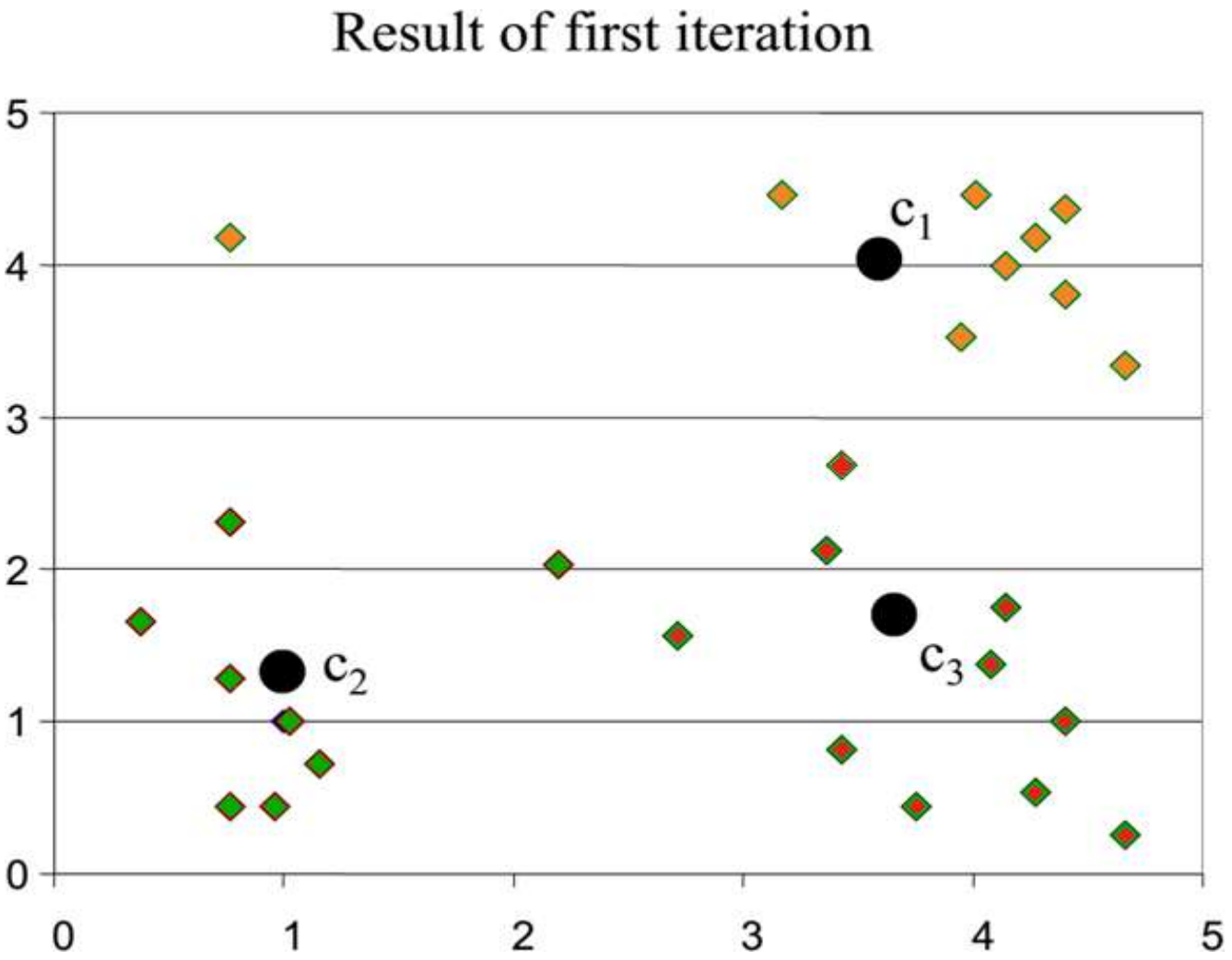
International Research Journal of Education and Technology
Peer Reviewed Journal, **ISSN 2581-7795**

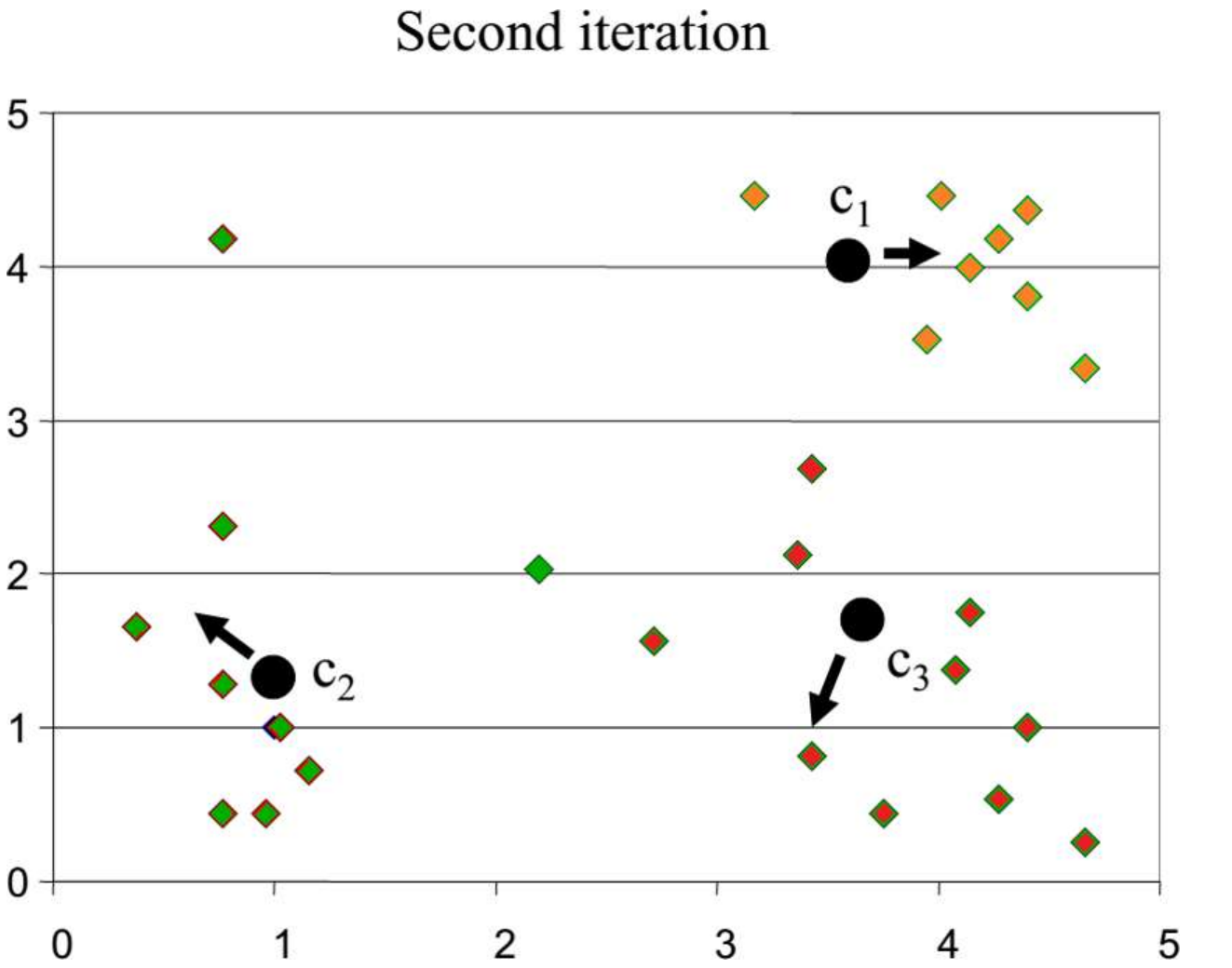
K-means clustering example – step 3



Re-estimate cluster centers (adapt synaptic weights)

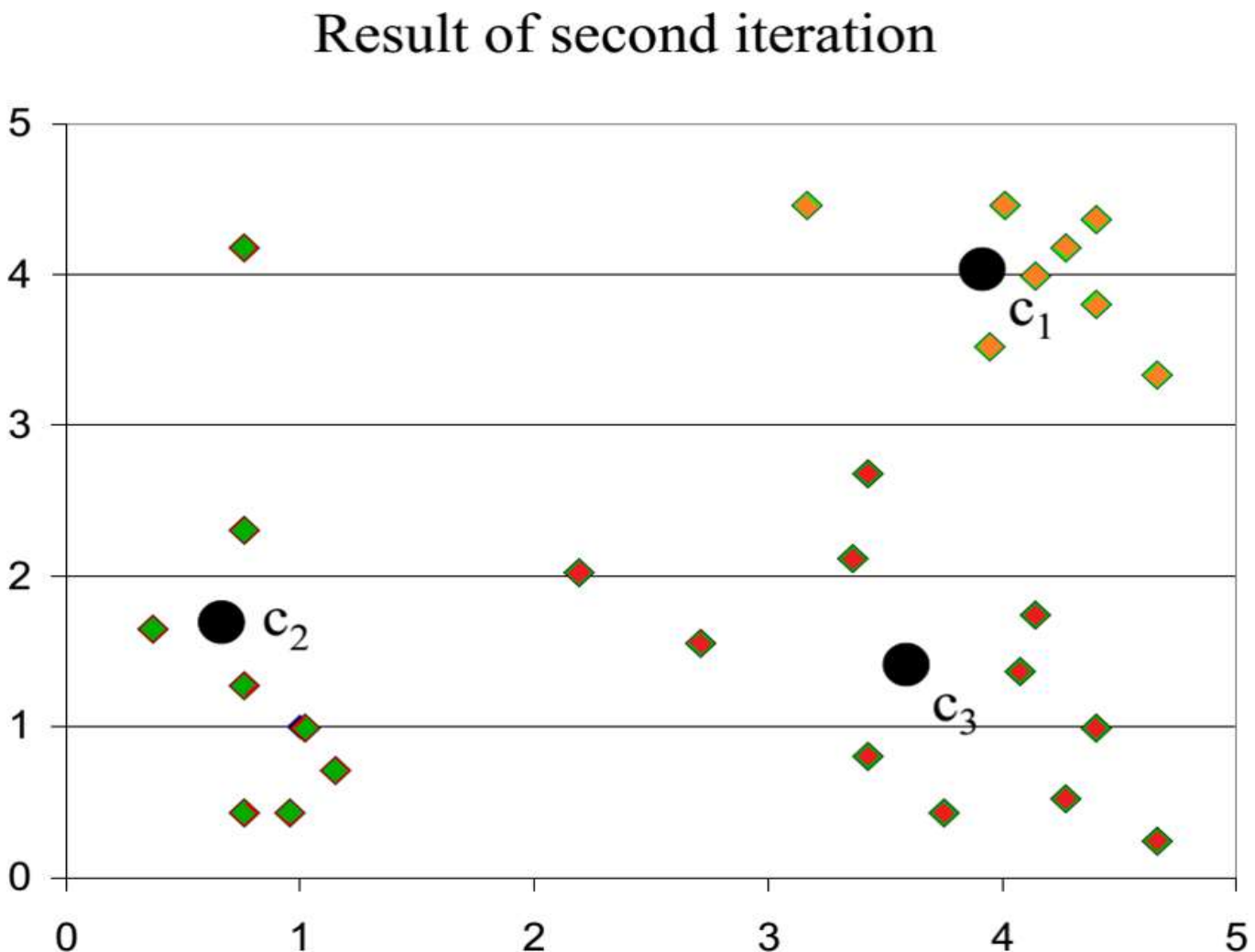
K-means clustering example





K-means clustering example

K-means clustering example



Why use K-means?

Strengths:

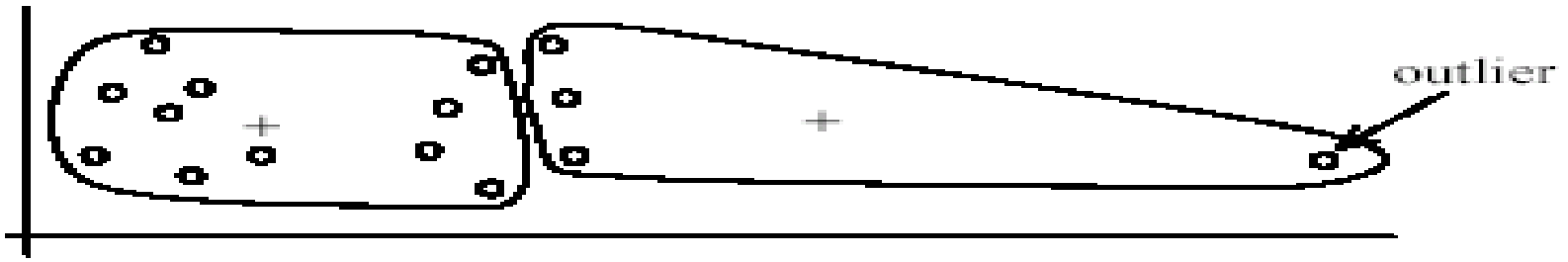
- Simple: easy to understand and to implement
- Efficient: Time complexity: $O(tkn)$, where n is the number of data points, k is the number of clusters, and t is the number of iterations.
- Since both k and t are small. k-means is considered a linear algorithm.

- K-means is the most popular clustering algorithm.
- Note that: it terminates at a local optimum if SSE is used. The global optimum is hard to find due to complexity.

Weaknesses of K-means

- The algorithm is only applicable if the mean is defined.
  - For categorical data, k-mode - the centroid is represented by most frequent values.
- The 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.

Outliers



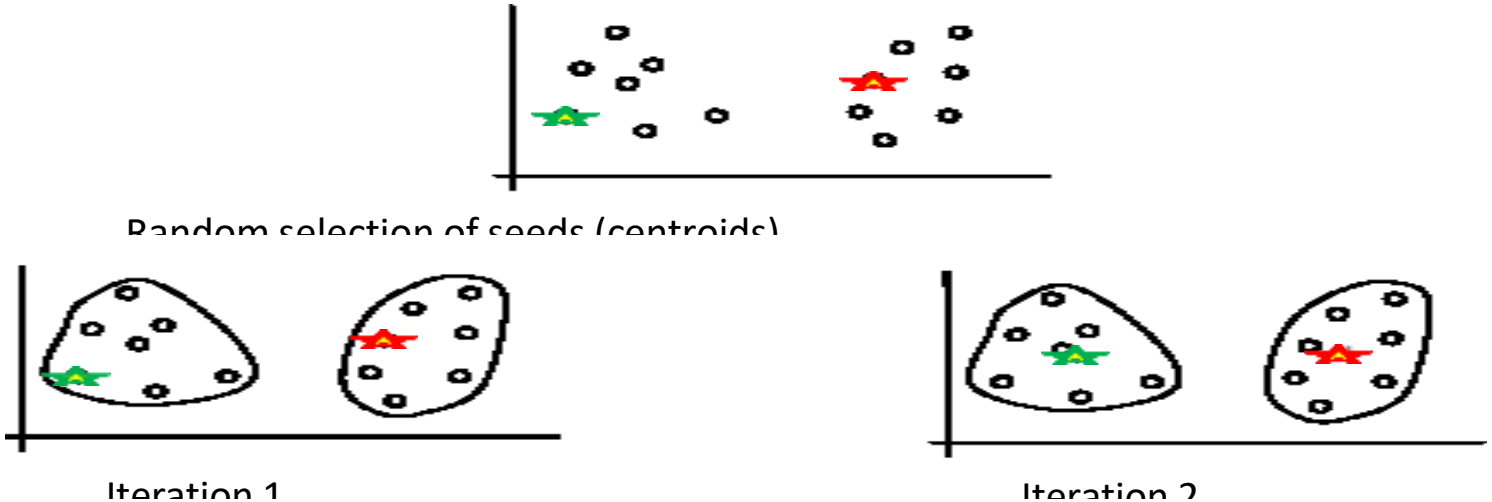
(A): Undesirable clusters

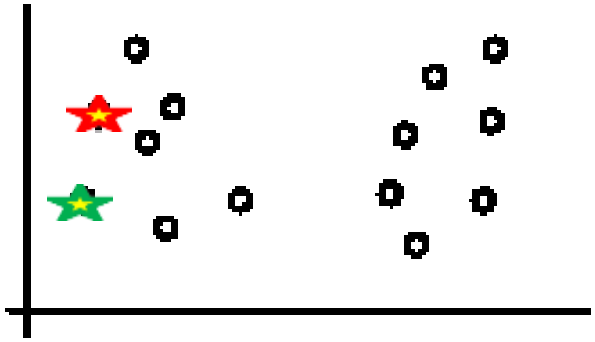
(B): Ideal clusters

Dealing with outliers

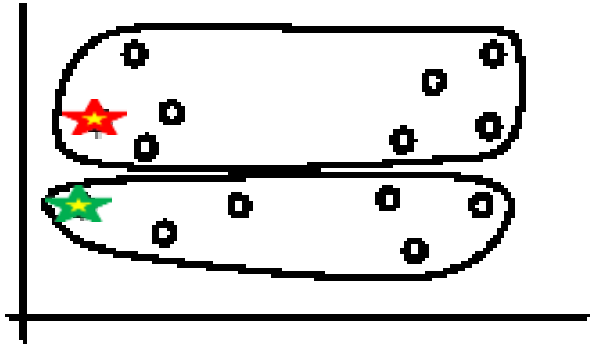
- Remove some data points that are much further away from the centroids than other data points
  - To be safe, we may want to monitor these possible outliers over a few iterations and then decide to remove them.
- Perform random sampling: by choosing a small subset of the data points, the chance of selecting an outlier is much smaller
  - Assign the rest of the data points to the clusters by distance or similarity comparison, or classification

Sensitivity to initial seeds

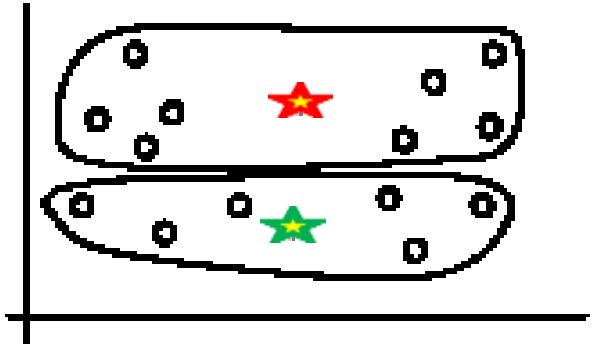




Random selection of seeds (centroids)



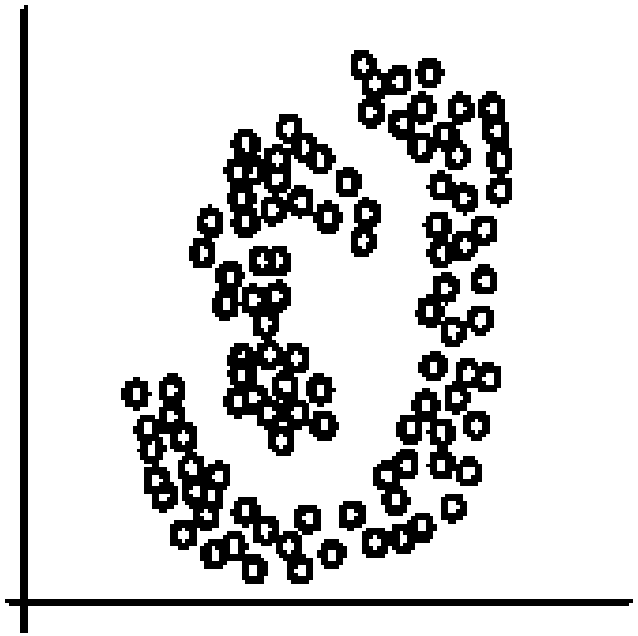
Iteration 1



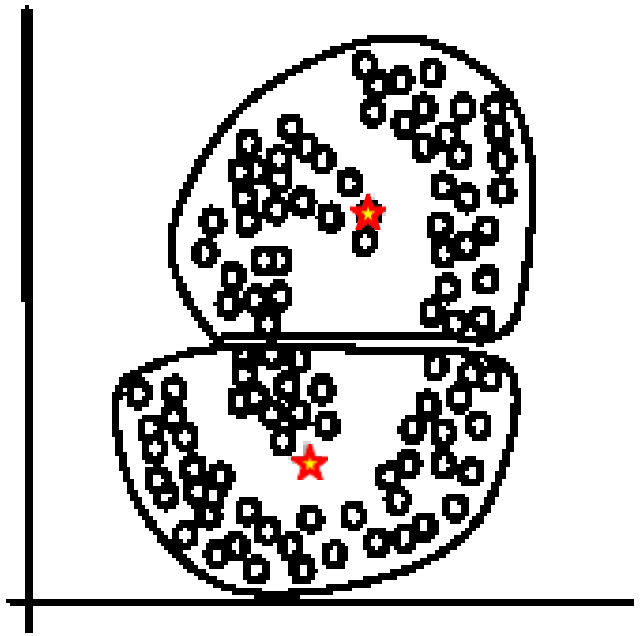
Iteration 2

Special data structures

The *k*-means algorithm is not suitable for discovering clusters that are not hyper-ellipsoids (or hyper-spheres).



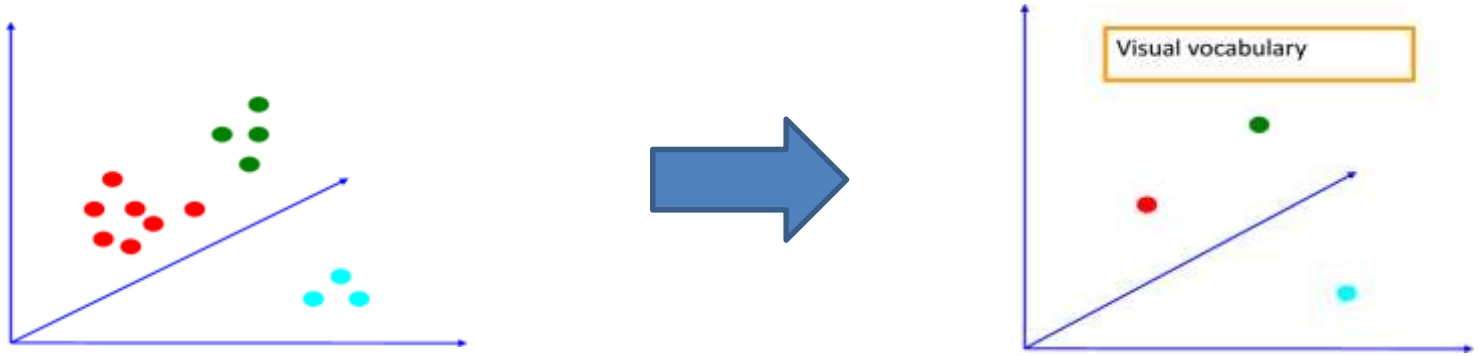
(A): Two natural clusters



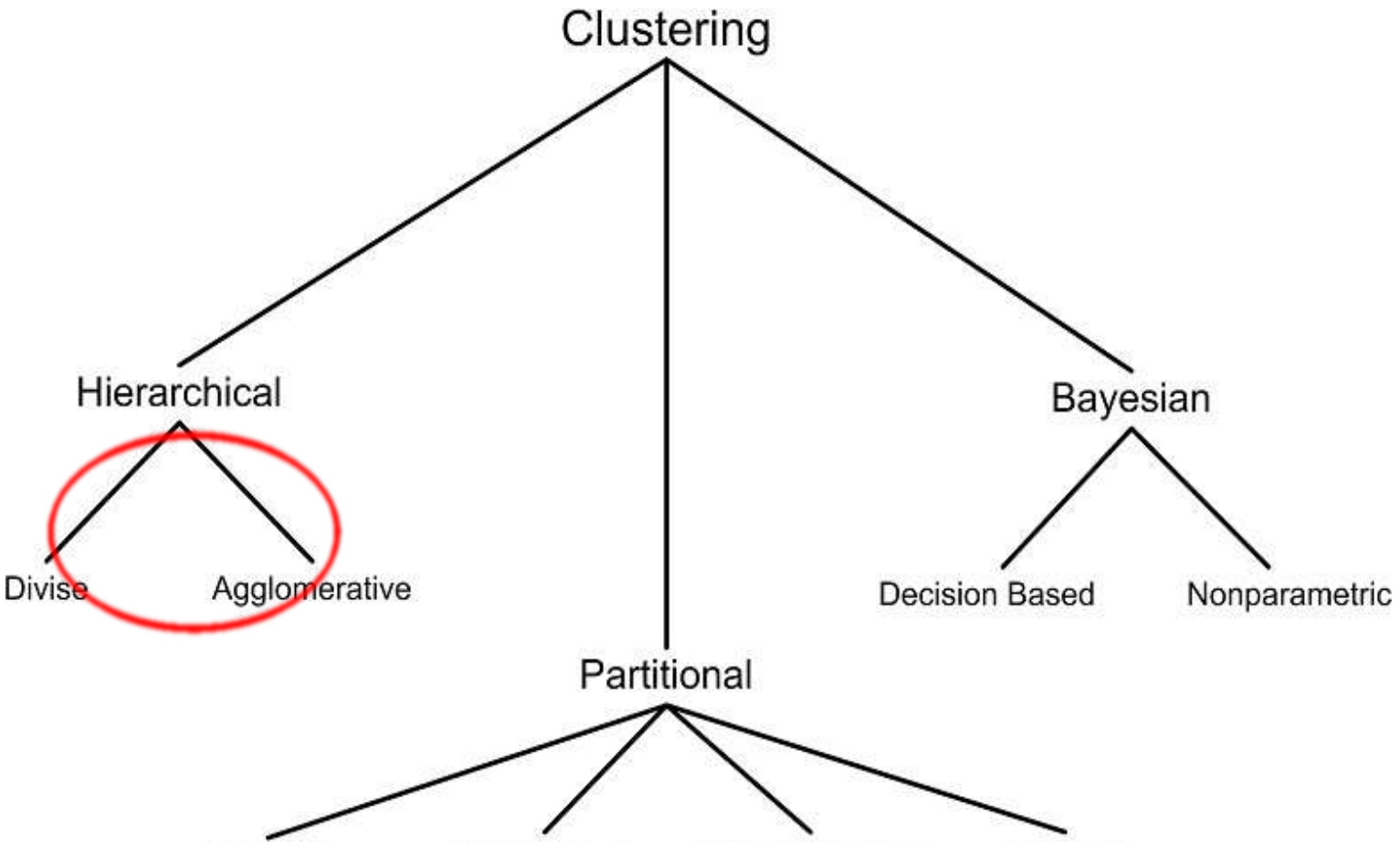
(B): *k*-means clusters

K-means summary

Despite weaknesses, *k*-means is still the most popular algorithm due to its simplicity and efficiency
No clear evidence that any other clustering algorithm performs better in general
Comparing different clustering algorithms is a difficult task. No one knows the correct clusters!

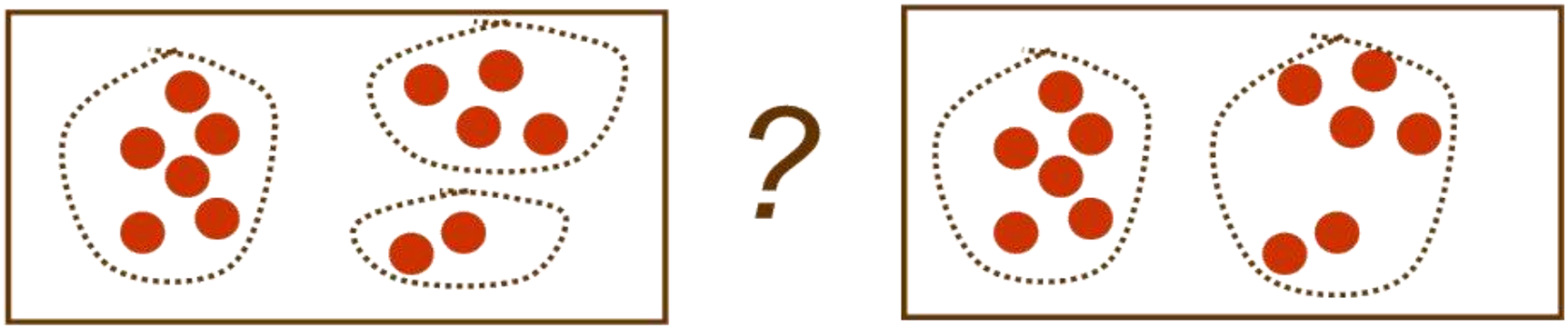


Clustering techniques



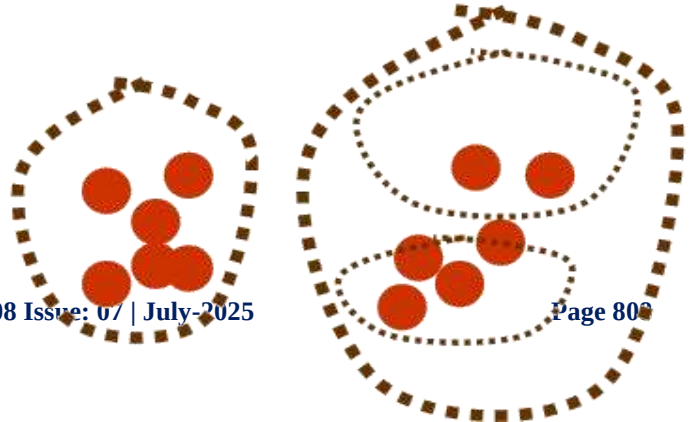
Hierarchical clustering

- Up to now, considered “flat” clustering

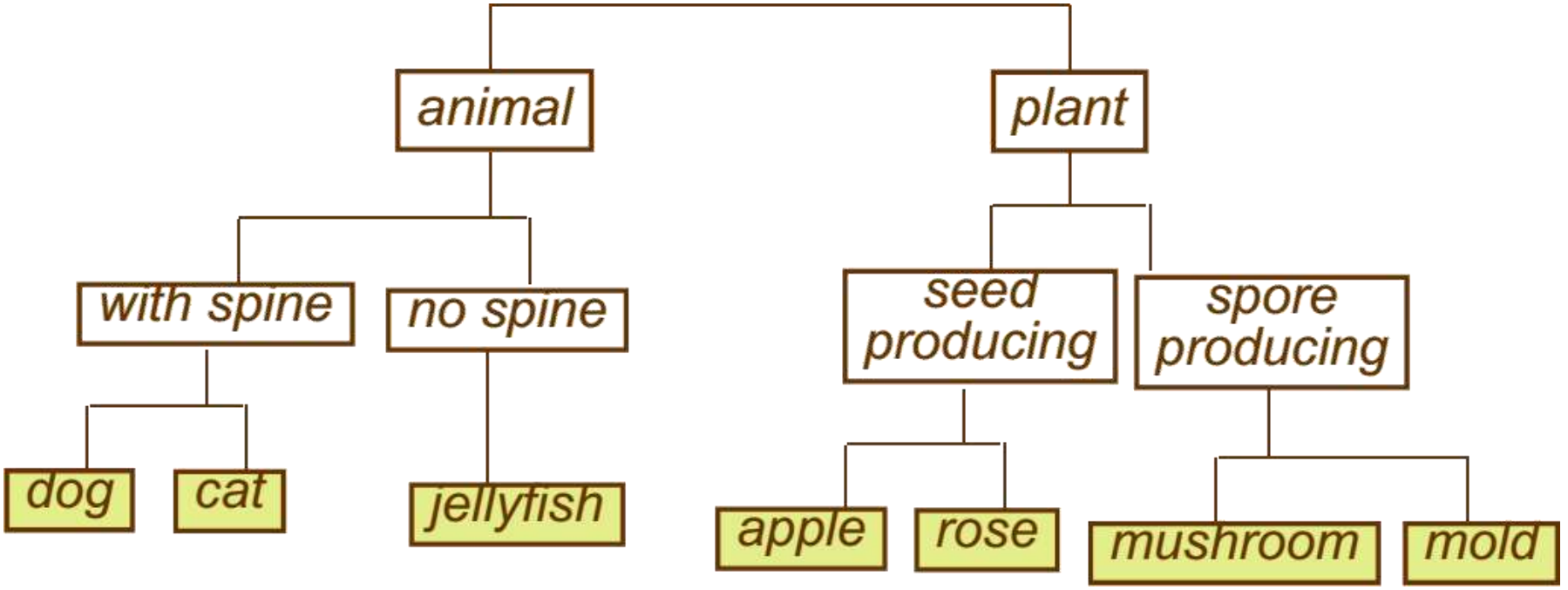


- For some data, hierarchical clustering is more appropriate than “flat” clustering

- Hierarchical clustering



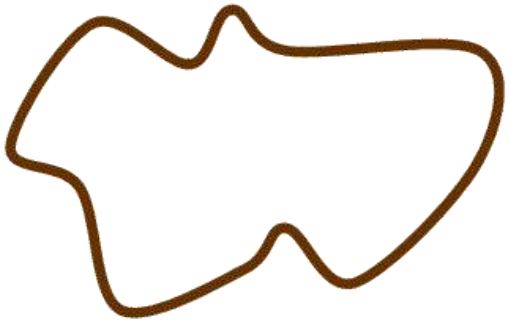
Example: biological taxonomy

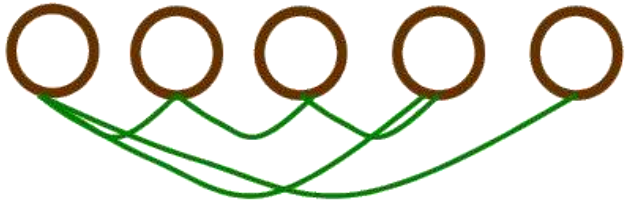


Divisive vs. Agglomerative

- Agglomerative is faster to compute, in general
- Divisive may be less “blind” to the global structure of the data

Divisive	*Agglomerative*
when taking the first step (split), have access to all the data; can find the best possible split in 2 parts	when taking the first step merging, do not consider the global structure of the data, only look at pairwise structure





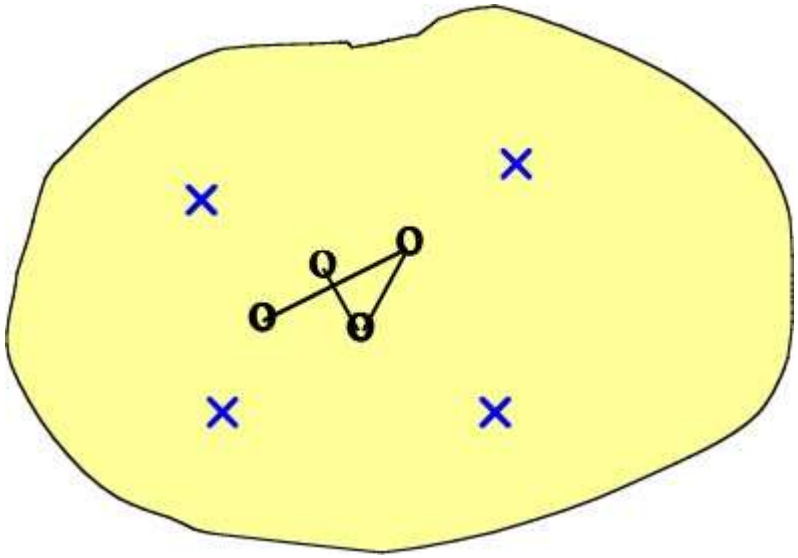
Competitive learning algorithm:

Kohonen Self Organization Maps (K-SOM)

- Initialize the units to have random weights
- Repeat
  - Find the weight vector which is closest to the presented input vector. Call this the winner or the winning vector.
  - Modify the winner so as to move closer to the input vector
    - modifying weights so as to make them more similar to the values in the input vector.

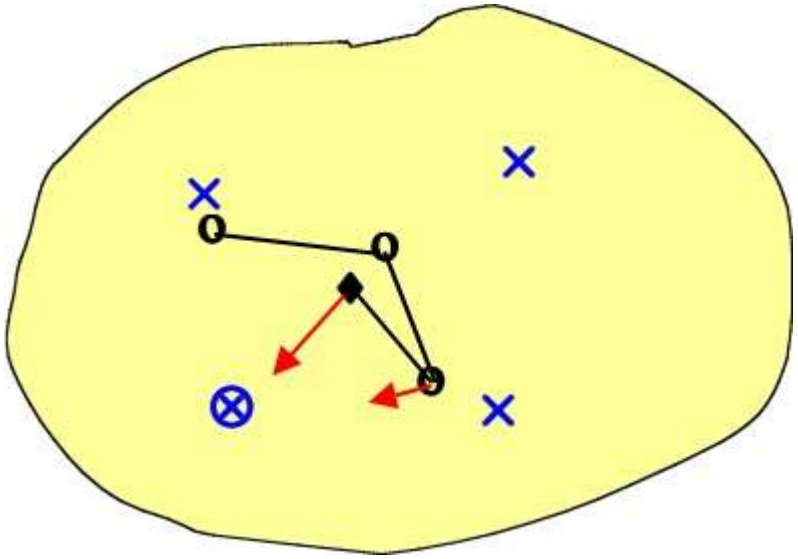
K-SOM example

- Four input data points (crosses) in 2D space.
- Four output nodes in a discrete 1D output space (mapped to 2D as circles).
- Random initial weights start the output nodes at random positions.



K-SOM example

- Continue to randomly pick data points for training, and move the winning neuron and its neighbors (by a smaller increment) towards the training data points.
- Eventually, the whole output grid unravels itself to represent the input space.



Summary

Clustering has a long history and still is in active research

- There are a huge number of clustering algorithms, among them: Density based algorithm, Sub-space clustering, Scale-up methods,
  Neural networks based methods, Fuzzy clustering, Co-clustering …
- More are still coming every year

- Clustering is hard to evaluate, but very useful in practice
- Clustering is highly application dependent (and to some extent subjective)

- Competitive learning in neuronal networks performs clustering analysis of the input data

Conclusion

Clustering, as a key technique in unsupervised learning, offers powerful tools for discovering patterns and structures in unlabeled data. By grouping similar data points based on inherent characteristics, clustering enables insights that are often difficult to achieve through traditional analysis. This paper has reviewed several major clustering algorithms—including K-Means, Hierarchical Clustering, DBSCAN, and Gaussian Mixture Models—highlighting their methodologies, strengths, and limitations. The choice of algorithm depends largely on the nature of the dataset and the desired outcome. As data continues to grow in complexity and volume, effective clustering will remain essential in domains such as data mining, marketing, biology, and artificial intelligence.

Continued research and innovation in this area will further enhance our ability to analyze and interpret complex datasets without the need for labeled inputs.