

Data Analytics

Course Taught at IIFT

Day 17: Unsupervised Learning - Clustering Techniques

Dr. Tanujit Chakraborty

www.ctanujit.org

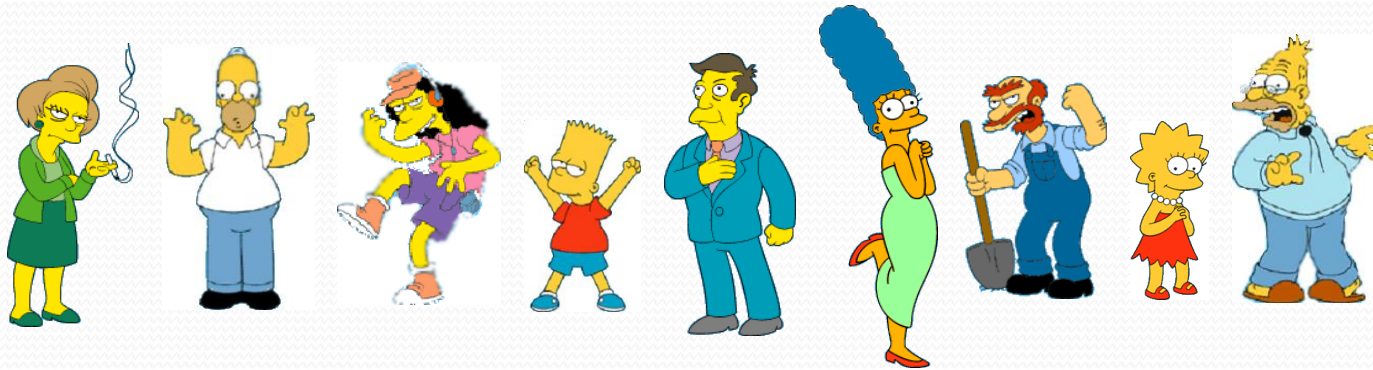
Topics to be covered...

- Introduction to clustering
- Clustering techniques
- Partitioning algorithms
- Hierarchical algorithms
- Density-based algorithm

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.
 - high intra-class similarity
 - low inter-class similarity
- More informally, finding natural groupings among objects.

What is a natural grouping among these objects?



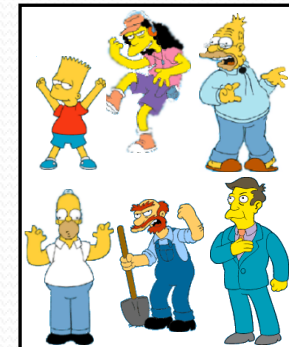
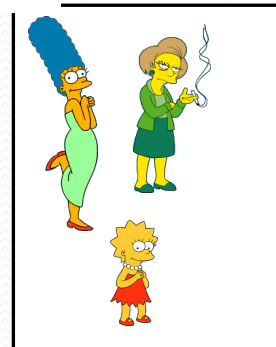
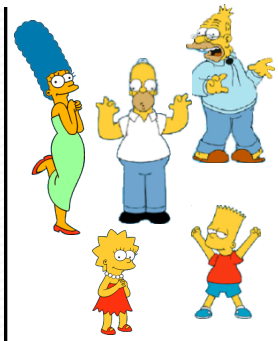
Clustering is subjective

Simpson's Family

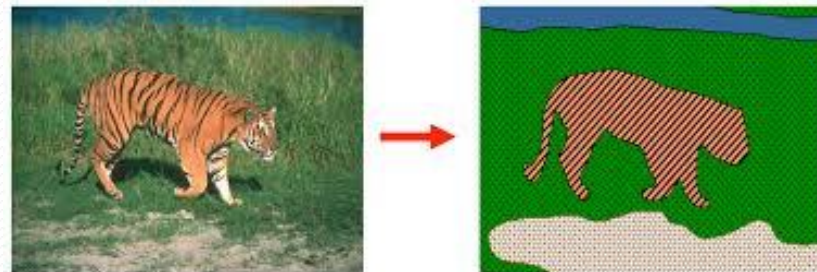
School Employees

Females

Males



Computer vision application : Image segmentation

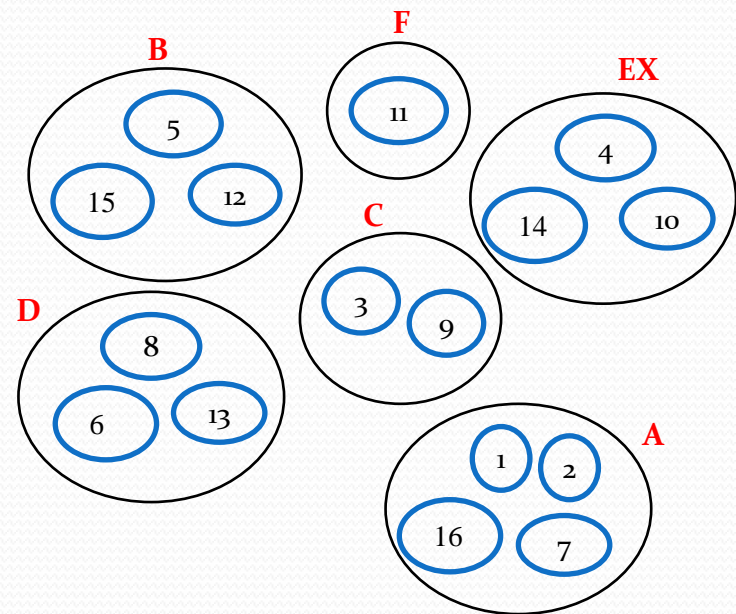


Supervised Vs. Unsupervised Learning

Table 1 : Tabulation of Marks

Roll No	Mark	Grade
1	80	A
2	70	A
3	55	C
4	91	EX
5	65	B
6	35	D
7	76	A
8	40	D
9	50	C
10	85	EX
11	25	F
12	60	B
13	45	D
14	95	EX
15	63	B
16	88	A

Figure 1: Group representation of dataset in Table 1



Introduction to Clustering

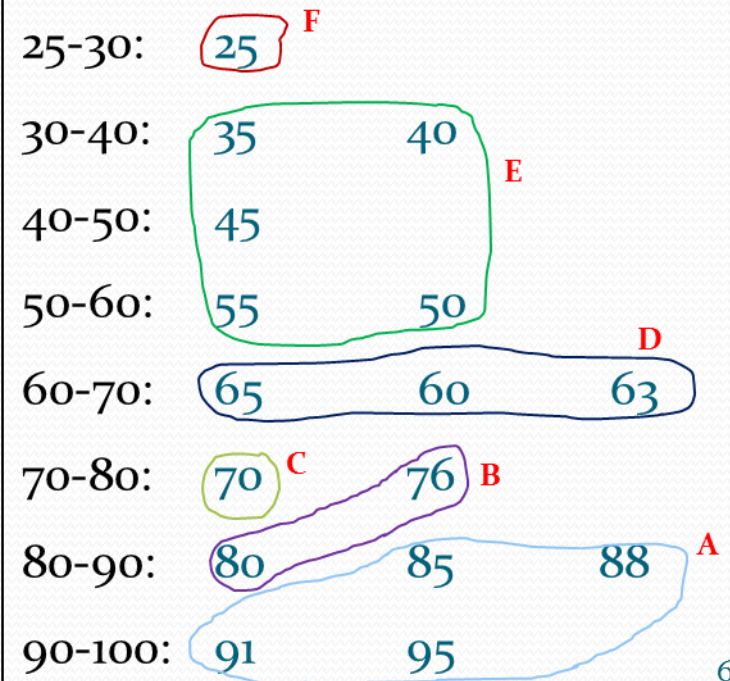
- It is evident that there is a simple mapping between Table 1 and Fig 1.
- The fact is that groups in Fig 1 are already predefined in Table 1. This is similar to classification, where we have given a dataset where **groups of data are predefined**.
- Consider another situation, where 'Grade' is not known, but we have to make a grouping.
- Put all the marks into a group if any other mark in that group does not exceed by 5 or more.
- This is similar to “**Relative grading**” concept and grade may range from A to Z.

Introduction to Clustering

- Figure 2 shows another grouping by means of another simple mapping, but the difference is **this mapping does not based on predefined classes**.
 - In other words, this grouping is accomplished by finding **similarities between data according to characteristics** found in the actual data.
 - Such a group making is called **clustering**.
- Figure 2 : Alternative grouping without predefined classes**

25-30:	25	F	
30-40:	35	40	E
40-50:	45		
50-60:	55	50	D

Figure 2 : Alternative grouping without predefined classes



Introduction to Clustering

Example 1 : The task of clustering

In order to elaborate the clustering task, consider the following dataset.

Table 2: Life Insurance database

Martial Status	Age	Income	Education	Number of children
Single	35	25000	Under Graduate	3
Married	25	15000	Graduate	1
Single	40	20000	Under Graduate	0
Divorced	20	30000	Post-Graduate	0
Divorced	25	20000	Under Graduate	3
Married	60	70000	Graduate	0
Married	30	90000	Post-Graduate	0
Married	45	60000	Graduate	5
Divorced	50	80000	Under Graduate	2

With certain similarity or likeliness defined, we can classify the records to one or group of more attributes (and thus mapping being non-trivial).

Introduction to Clustering

- Clustering has been used in many application domains:
 - Image analysis
 - Document retrieval
 - Machine learning, etc.
- When clustering is applied to real-world database, many problems may arise.
 1. The (best) number of cluster is not known.
 - There is not correct answer to a clustering problem.
 - In fact, many answers may be found.
 - The exact number of cluster required is not easy to determine.

Introduction to Clustering

2. There may not be any a priori knowledge concerning the clusters.

- This is an issue that what data should be used for clustering.
- Unlike classification, in clustering, we have not supervisory learning to aid the process.
- Clustering can be viewed as similar to **unsupervised learning**.

3. Interpreting the semantic meaning of each cluster may be difficult.

- With classification, the labeling of classes is known ahead of time. In contrast, with clustering, this may not be the case.
- Thus, when the clustering process is finished yielding a set of clusters, the exact meaning of each cluster may not be obvious.

Definition of Clustering Problem

Definition 1: **Clustering**

Given a database $D = \{t_1, t_2, \dots, t_n\}$ of n tuples, the clustering problem is to define a mapping $f : D \rightarrow C$, where each $t_i \in D$ is assigned to one cluster $c_i \in C$. Here, $C = \{c_1, c_2, \dots, c_k\}$ denotes a set of clusters.

- Solution to a clustering problem is devising a mapping formulation.
- The formulation behind such a mapping is to establish that a tuple within one cluster is **more like** tuples within that cluster and not similar to tuples outside it.

Data Representations for Clustering

- Input data to algorithm is usually a vector (also called a “tuple” or “record”)
- Types of data
 - Numerical
 - Categorical
 - Boolean
- Example: Clinical Sample Data
 - Age (numerical)
 - Weight (numerical)
 - Gender (categorical)
 - Diseased (Boolean)
- Must also include a method for computing similarity of or distance between vectors

Definition of Clustering Problem

- Hence, mapping function f in Definition 1 may be explicitly stated as

$$f : D \rightarrow \{c_1, c_2, \dots, c_k\}$$

where i) each $t_i \in D$ is assigned to one cluster $c_i \in C$.

ii) for each cluster $c_i \in C$, and for all $t_{ip}, t_{iq} \in c_i$ and there exist $t_j \notin c_i$ such that

$$\text{similarity}(t_{ip}, t_{iq}) > \text{similarity}(t_{ip}, t_j) \text{ AND } \text{similarity}(t_{iq}, t_j)$$

- In the field of cluster analysis, this **similarity** plays an important part.
- Now, we shall learn how similarity (this is also alternatively judged as “dissimilarity”) between any two data can be measured.

What do we need for clustering?

- 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



Large d , small s



Large s , small d

- The diagram shows two clusters of blue dots. Each cluster is enclosed by a red circle. Both red circles are enclosed by a larger green rectangle. This illustrates a criterion function for evaluating clustering.

Criterion function to evaluate clustering

- Algorithm to evaluate clustering
 - For example by optimizing criterion function

Distance (dissimilarity) measures

- Euclidean distance
 - Translation invariant

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^d (x_i^k - x_j^k)^2}$$

- Manhattan (city block) distance
 - Approximation to Euclidean distance
 - Cheaper to compute

$$d(x_i, x_j) = \sum_{k=1}^d |x_i^k - x_j^k|$$

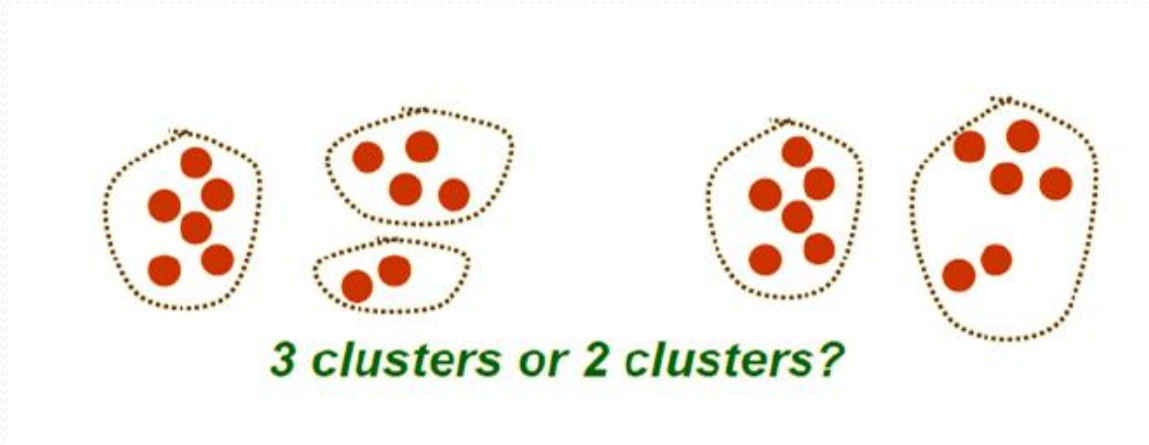
- They are special cases of Minkowski distance:
 - p is a positive integer

$$d_p(x_i, x_j) = \left(\sum_{k=1}^d |x_i^k - x_j^k|^p \right)^{\frac{1}{p}}$$

Cluster evaluation (a hard problem)

- **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?



- Possible approaches
 - Fix the number of cluster to k
 - Find the best clustering according to the criterion function

Clustering techniques

- Clustering has been studied extensively for more than 40 years and across many disciplines due to its broad applications.
- As a result, many clustering techniques have been reported in the literature.
- Let us categorize the clustering methods. In fact, it is difficult to provide a crisp categorization because many techniques overlap to each other in terms of clustering paradigms or features.
- A broad taxonomy of existing clustering methods is shown in the next slide.
- It is not possible to cover all the techniques in this lecture series. We emphasize on major techniques belong to partitioning and hierarchical algorithms.

Clustering Techniques

Partitioning methods

- k-Means algorithm [1957, 1967]
- k-Medoids algorithm
- k-Modes [1998]
- Fuzzy c-means algorithm [1999]

- PAM [1990]
- CLARA [1990]
- CLARANS [1994]

Hierarchical methods

Divisive

- DIANA [1990]

Agglomerative methods

- AGNES [1990]
- BIRCH [1996]
- CURE [1998]
- ROCK [1999]
- Chamelon [1999]

Density-based methods

- STING [1997]
- DBSCAN [1996]
- CLIQUE [1998]
- DENCLUE [1998]
- OPTICS [1999]
- Wave Cluster [1998]

Graph based methods

- MST Clustering [1999]
- OPOSSUM [2000]
- SNN Similarity Clustering [2001, 2003]

Model based clustering

- EM Algorithm [1977]
- Auto class [1996]
- COBWEB [1987]
- ANN Clustering [1982, 1989]

Clustering techniques

- Various Clustering Methods:
 - Partitioning
 - k-Means algorithm
 - PAM (k-Medoids algorithm)
 - Hierarchical
 - DIANA (divisive algorithm)
 - AGNES
 - ROCK
 - } (Agglomerative algorithm)
 - Density – Based
 - DBSCAN

k-Means Algorithm

- k-Means clustering algorithm proposed by J. Hartigan and M. A. Wong [1979].
- Given a set of n distinct objects, the k-Means clustering algorithm partitions the objects into k number of clusters such that intracluster similarity is high but the intercluster similarity is low.
- In this algorithm, user has to specify k , the number of clusters and consider the objects are defined with numeric attributes and thus using any one of the distance metric to demarcate the clusters.

k-Means Algorithm

The algorithm can be stated as follows.

- First it selects k number of objects at random from the set of n objects. These k objects are treated as the **centroids or center of gravities** of k clusters.
- For each of the **remaining objects**, it is assigned to one of the **closest centroid**. Thus, it forms a **collection of objects assigned to each centroid** and is called a **cluster**.
- Next, the centroid of each cluster is then updated (by calculating the mean values of attributes of each object).
- The assignment and update procedure is continued until it reaches some stopping criteria (such as, number of iteration, centroids remain unchanged or no assignment, etc.)

k-Means Algorithm

Algorithm 1: k-Means clustering

Input: D is a dataset containing n objects, k is the number of cluster

Output: A set of k clusters

Steps:

1. Randomly choose k objects from D as the initial cluster centroids.
2. **For** each of the objects in D **do**
 - Compute distance between the current objects and k cluster centroids
 - Assign the current object to that cluster to which it is closest.
3. Compute the “cluster centers” of each cluster. These become the new cluster centroids.
4. Repeat step 2-3 until the convergence criterion is satisfied
5. Stop

k-Means Algorithm

Note:

- 1) Objects are defined in terms of set of attributes. $A = \{A_1, A_2, \dots, A_m\}$ where each A_i is continuous data type.
- 2) **Distance computation**: Any distance such as L_1, L_2, L_3 or cosine similarity.
- 3) **Minimum distance** is the measure of closeness between an object and centroid.
- 4) **Mean Calculation**: It is the mean value of each attribute values of all objects.
- 5) **Convergence criteria**: Any one of the following are termination condition of the algorithm.
 - Number of maximum iteration permissible.
 - No change of centroid values in any cluster.
 - Zero (or no significant) movement(s) of object from one cluster to another.
 - Cluster quality reaches to a certain level of acceptance.

Illustration of k-Means clustering algorithms

Table 3: 16 objects with two attributes A_1 and A_2 .

A_1	A_2
6.8	12.6
0.8	9.8
1.2	11.6
2.8	9.6
3.8	9.9
4.4	6.5
4.8	1.1
6.0	19.9
6.2	18.5
7.6	17.4
7.8	12.2
6.6	7.7
8.2	4.5
8.4	6.9
9.0	3.4
9.6	11.1

Fig 3: Plotting data of Table 3

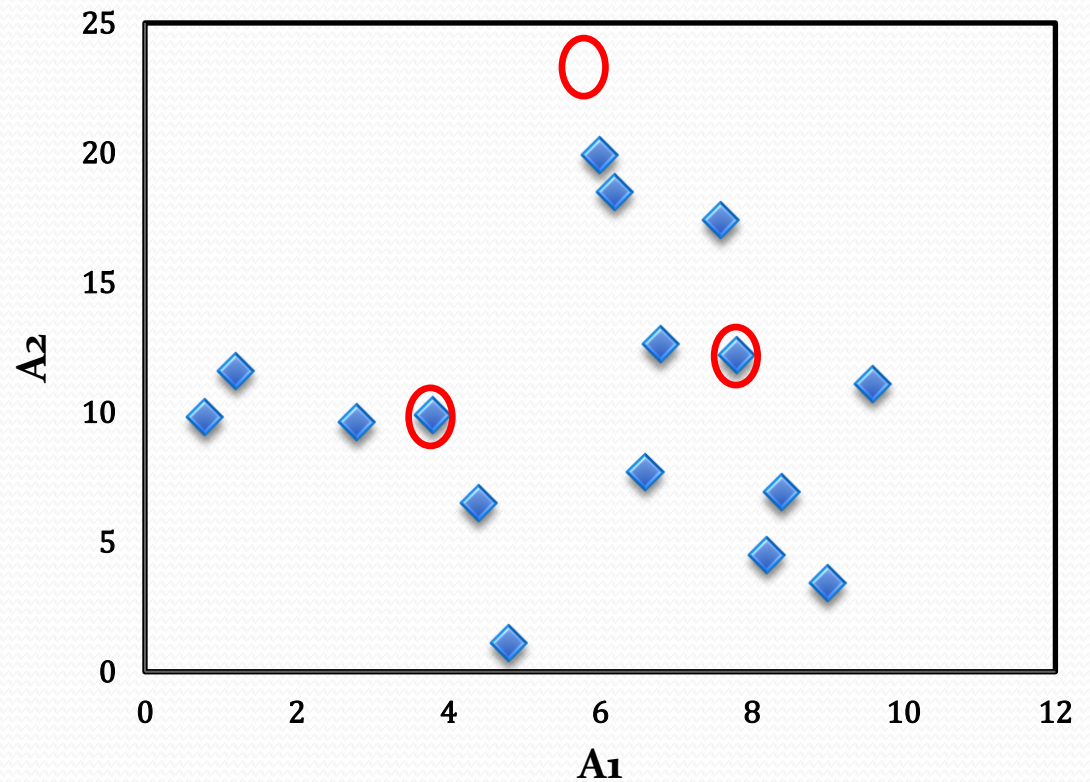


Illustration of k-Means clustering algorithms

- Suppose, $k=3$. Three objects are chosen at random shown as circled (see Fig 3). These three centroids are shown below.

Initial Centroids chosen randomly

Centroid	Objects	
	A1	A2
c_1	3.8	9.9
c_2	7.8	12.2
c_3	6.2	18.5

- Let us consider the Euclidean distance measure (L_2 Norm) as the distance measurement in our illustration.
- Let d_1 , d_2 and d_3 denote the distance from an object to c_1 , c_2 and c_3 respectively. The distance calculations are shown in Table 4.
- Assignment of each object to the respective centroid is shown in the right-most column and the clustering so obtained is shown in Fig 4.

Illustration of k-Means clustering algorithms

Table 4: Distance calculation

A_1	A_2	d_1	d_2	d_3	cluster
6.8	12.6	4.0	1.1	5.9	2
0.8	9.8	3.0	7.4	10.2	1
1.2	11.6	3.1	6.6	8.5	1
2.8	9.6	1.0	5.6	9.5	1
3.8	9.9	0.0	4.6	8.9	1
4.4	6.5	3.5	6.6	12.1	1
4.8	1.1	8.9	11.5	17.5	1
6.0	19.9	10.2	7.9	1.4	3
6.2	18.5	8.9	6.5	0.0	3
7.6	17.4	8.4	5.2	1.8	3
7.8	12.2	4.6	0.0	6.5	2
6.6	7.7	3.6	4.7	10.8	1
8.2	4.5	7.0	7.7	14.1	1
8.4	6.9	5.5	5.3	11.8	2
9.0	3.4	8.3	8.9	15.4	1
9.6	11.1	5.9	2.1	8.1	2

Fig 4: Initial cluster with respect to Table 4

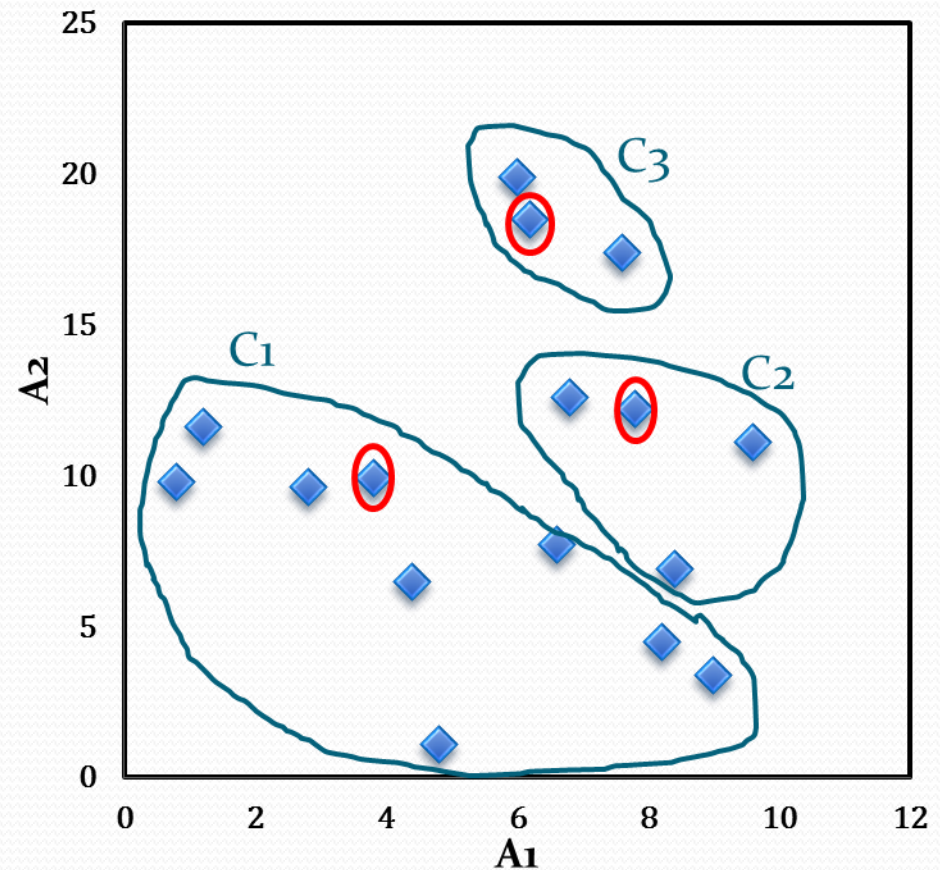


Illustration of k-Means clustering algorithms

The calculation new centroids of the three cluster using the mean of attribute values of A_1 and A_2 is shown in the Table below. The cluster with new centroids are shown in Fig 5.

Calculation of new centroids

New Centroid	Objects	
	A_1	A_2
c_1	4.6	7.1
c_2	8.2	10.7
c_3	6.6	18.6

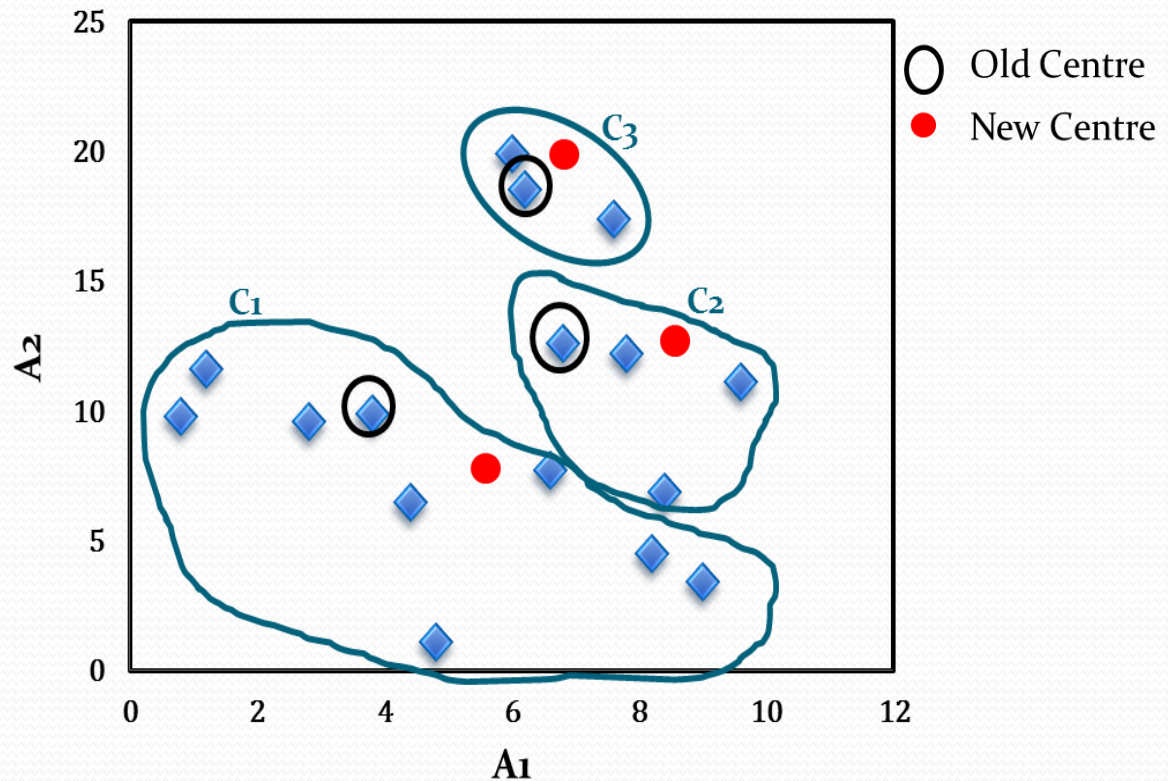


Fig 5: Initial cluster with new centroids

Illustration of k-Means clustering algorithms

We next reassign the 16 objects to three clusters by determining which centroid is closest to each one. This gives the revised set of clusters shown in Fig 6.

Note that point p moves from cluster C_2 to cluster C_1 .

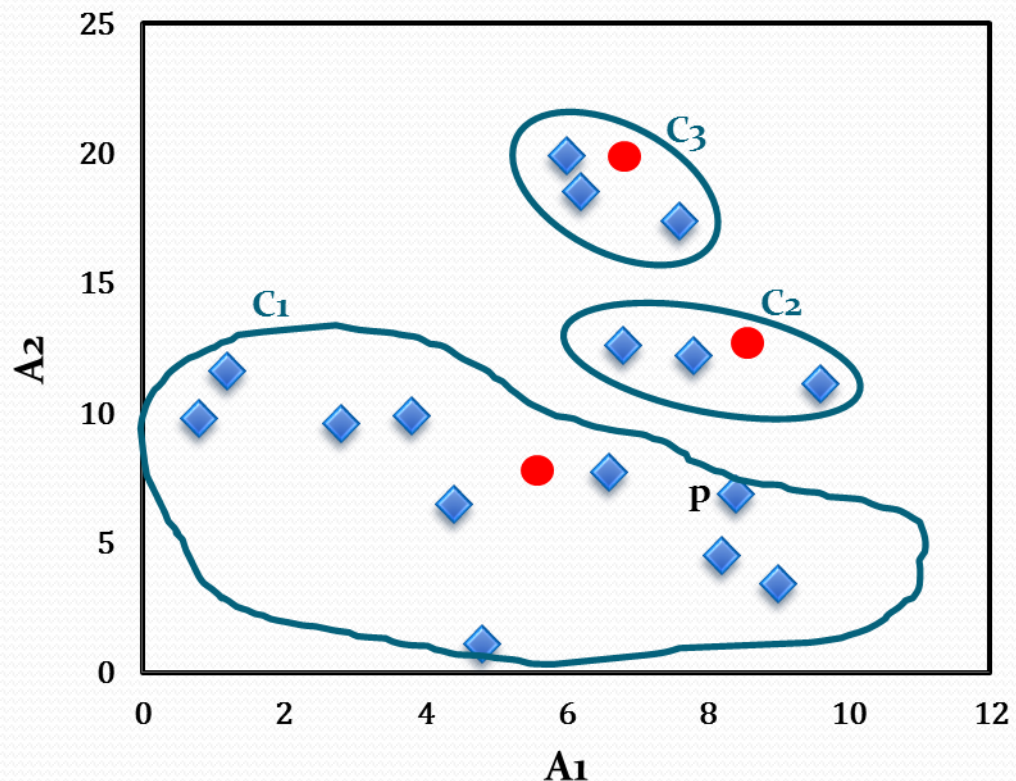


Fig 6: Cluster after first iteration

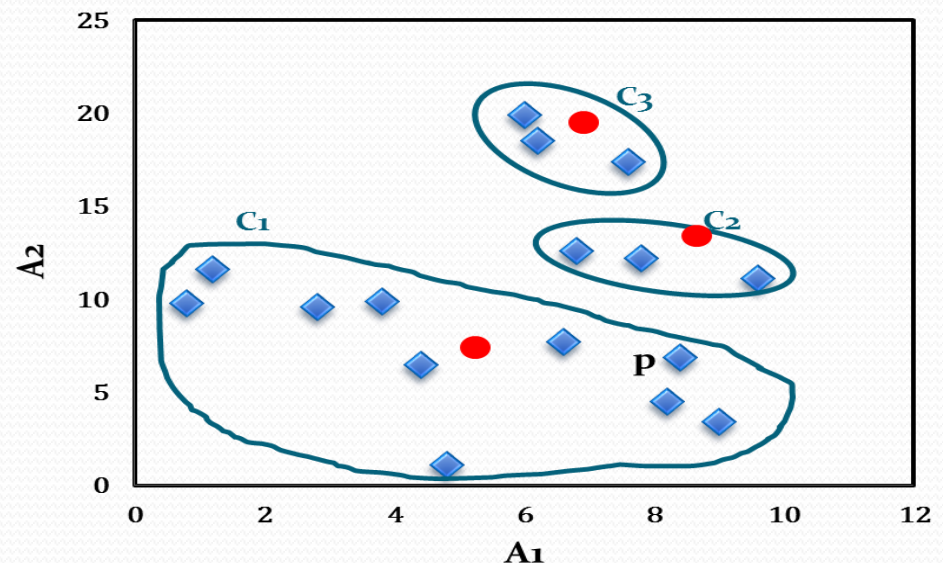
Illustration of k-Means clustering algorithms

- The newly obtained centroids after second iteration are given in the table below. Note that the centroid c_3 remains unchanged, where c_2 and c_1 changed a little.
- With respect to newly obtained cluster centres, 16 points are reassigned again. These are the same clusters as before. Hence, their centroids also remain unchanged.
- Considering this as the termination criteria, the k-means algorithm stops here. Hence, the final cluster in Fig 7 is same as Fig 6.

Cluster centres after second iteration

Centroid	Revised Centroids	
	A1	A2
c_1	5.0	7.1
c_2	8.1	12.0
c_3	6.6	18.6

Fig 7: Cluster after Second iteration



Comments on k-Means algorithm

Let us analyse the k-Means algorithm and discuss the pros and cons of the algorithm. We shall refer to the following notations in our discussion.

- **Notations:**
 - x : an object under clustering
 - n : number of objects under clustering
 - C_i : the i -th cluster
 - c_i : the centroid of cluster C_i
 - n_i : number of objects in the cluster C_i
 - c : denotes the centroid of all objects
 - k : number of clusters

Comments on k-Means algorithm

1. Value of k :

- The k-means algorithm produces only one set of clusters, for which, user must specify the desired number, k of clusters.
- In fact, k should be the **best guess** on the number of clusters present in the given data. Choosing the best value of k for a given dataset is, therefore, an issue.
- We may not have an idea about the possible number of clusters for high dimensional data, and for data that are not scatter-plotted.
- Further, possible number of clusters is hidden or ambiguous in image, audio, video and multimedia clustering applications etc.
- There is no principled way to know what the value of k ought to be. We may try with successive value of k starting with 2.
- The process is stopped when two consecutive k values produce more-or-less identical results (with respect to some cluster quality estimation).
- Normally $k \ll n$ and there is heuristic to follow $k \approx \sqrt{n}$.

Comments on k-Means algorithm

Example 2: k versus cluster quality

- Usually, there is some objective function to be met as a goal of clustering. One such objective function is **sum-square-error** denoted by **SSE** and defined as

$$SSE = \sum_{i=1}^k \sum_{x \in C_i} (x - c_i)^2$$

- Here, $x - c_i$ denotes the error, if x is in cluster C_i with cluster centroid c_i .
- Usually, this error is measured as distance norms like L_1 , L_2 , L_3 or Cosine similarity, etc.

Comments on k-Means algorithm

Example 3: k versus cluster quality

- With reference to an arbitrary experiment, suppose the following results are obtained.

k	SSE
1	62.8
2	12.3
3	9.4
4	9.3
5	9.2
6	9.1
7	9.05
8	9.0

- With respect to this observation, we can choose the value of $k \approx 3$, as with this smallest value of k it gives reasonably good result.
- Note: If $k = n$, then $SSE=0$; However, the cluster is useless! This is another example of overfitting.

Comments on k-Means algorithm

2. Choosing initial centroids:

- Another requirement in the k-Means algorithm to choose initial cluster centroid for each k would be clusters.
- It is observed that the k-Means algorithm terminate whatever be the initial choice of the cluster centroids.
- It is also observed that initial choice influences the ultimate cluster quality. In other words, the result may be trapped into local optima, if initial centroids are NOT chosen properly.
- One technique that is usually followed to avoid the above problem is to choose initial centroids in multiple runs, each with a different set of randomly chosen initial centroids, and then select the best cluster (with respect to some quality measurement criterion, e.g. SSE).
- However, this strategy suffers from the combinational explosion problem due to the number of all possible solutions.

Comments on k-Means algorithm

2. Choosing initial centroids:

- A detail calculation reveals that there are $c(n, k)$ possible combinations to examine the search of global optima.

$$c(n, k) = \frac{1}{k!} \sum_{i=1}^k (-1)^{k-i} \binom{k}{i} (i)^n$$

- For example, there are $o(10^{10})$ different ways to cluster 20 items into 4 clusters!
- Thus, the strategy having its own limitation is practical only if
 - 1) The sample is relatively small (~ 100 - 1000), and
 - 2) k is relatively small compared to n (i.e.. $k \ll n$).

Comments on k-Means algorithm

3. Distance Measurement:

- To assign a point to the closest centroid, we need a proximity measure that should quantify the notion of “closest” for the objects under clustering.
- Usually Euclidean distance (L_2 norm) is the best measure when object points are defined in n-dimensional Euclidean space.
- Other measure namely cosine similarity is more appropriate when objects are of document type.
- Further, there may be other type of proximity measures that appropriate in the context of applications.
- For example, Manhattan distance (L_1 norm), Jaccard measure, etc.

Comments on k-Means algorithm

3. Distance Measurement:

Thus, in the context of different measures, the **sum-of-squared error** (i.e., objective function/convergence criteria) of a clustering can be stated as under.

Data in Euclidean space (L_2 norm):

$$SSE = \sum_{i=1}^k \sum_{x \in C_i} (c_i - x)^2$$

Data in Euclidean space (L_1 norm):

The Manhattan distance (L_1 norm) is used as a proximity measure, where the objective is to minimize the **sum-of-absolute error** denoted as **SAE** and defined as

$$SAE = \sum_{i=1}^k \sum_{x \in C_i} |c_i - x|$$

Comments on k-Means algorithm

Distance with document objects

Suppose a set of n document objects is defined as d document term matrix (DTM) (a typical look is shown in the below form).

Document	Term			
	t_1	t_2	t_n
D_1	f_{11}	f_{12}		f_{1n}
D_2	f_{21}	f_{22}		f_{2n}
\vdots				
D_n	f_{n1}	f_{n2}		f_{nn}

Here, the objective function, which is called Total cohesion denoted as TC and defined as

$$TC = \sum_{i=1}^k \sum_{x \in C_i} \cos(x, c_i)$$

where $\cos(x, c_i) = \frac{x \cdot c_i}{\|x\| \|c_i\|}$

Where $x \cdot c_i = \sum_j x_j c_{ij}$ and $\|x\| = \sqrt{\sum_j^p x_j^2}$

$\hat{x} = \sum_{j=1}^p \hat{x}_j$ $\hat{c}_i = \sum_{j=1}^p \hat{c}_{ij}$ $\|c_{ij}\| = \sqrt{\sum_j^p c_{ij}^2}$

Comments on k-Means algorithm

Note: The criteria of objective function with different proximity measures

1. SSE (using L_2 norm) : To **minimize** the SSE.
2. SAE (using L_1 norm) : To **minimize** the SAE.
3. TC(using cosine similarity) : To **maximize** the TC.

Comments on k-Means algorithm

Note:

- 1) When SSE (L_2 norm) is used as objective function and the objective is to minimize, then the cluster centroid (i.e. mean) is the mean value of the objects in the cluster.
- 2) When the objective function is defined as SAE (L_1 norm), minimizing the objective function implies the cluster centroid as the median of the cluster.

Comments on k-Means algorithm

4. Type of objects under clustering:

- The k-Means algorithm can be applied only when the mean of the cluster is defined (hence it named k-Means). The cluster mean (also called centroid) of a cluster C_i is defined as

$$c_i = \frac{1}{n_i} \sum_{x \in C_i} x$$

- In other words, the mean calculation assumed that each object is defined with numerical attribute(s). Thus, we cannot apply the k-Means to objects which are defined with categorical attributes.
- More precisely, the k-means algorithm require some definition of cluster mean exists, but not necessarily it does have as defined in the above equation.
- In fact, the k-Means is a very general clustering algorithm and can be used with a wide variety of data types, such as documents, time series, etc.



How to find the mean of objects with composite attributes?

Comments on k-Means algorithm

Case 1: SSE

We know,

$$SSE = \sum_{i=1}^k \sum_{x \in C_i} (c_i - x)^2$$

To minimize SSE means, $\frac{\partial(SSE)}{\partial c_i} = 0$

Thus,

$$\frac{\partial}{\partial c_i} \left(\sum_{i=1}^k \sum_{x \in C_i} (c_i - x)^2 \right) = 0$$

Or,

$$\sum_{i=1}^k \sum_{x \in C_i} \frac{\partial}{\partial c_i} (c_i - x)^2 = 0$$

Comments on k-Means algorithm

Or,

$$\sum_{x \in C_i} 2(c_i - x) = 0$$

Or,

$$n_i \cdot c_i = \sum_{x \in C_i} x$$

Or,

$$c_i = \frac{1}{n_i} \sum_{x \in C_i} x$$

- Thus, **the best centroid for minimizing SSE of a cluster is the mean of the objects in the cluster.**

Comments on k-Means algorithm

Case 2: SAE

We know,

$$SAE = \sum_{i=1}^k \sum_{x \in C_i} |c_i - x|$$

To minimize SAE means, $\frac{\partial(SAE)}{\partial c_i} = 0$

Thus,

$$\frac{\partial}{\partial c_i} \left(\sum_{i=1}^k \sum_{x \in C_i} |c_i - x| \right) = 0$$

Or,

$$\sum_{i=1}^k \sum_{x \in C_i} \frac{\partial}{\partial c_i} |c_i - x| = 0$$

Comments on k-Means algorithm

Or,

$$\sum_{x \in C_i} \left\{ (x - c_i) \Big|_{\text{if } x > c_i} + (c_i - x) \Big|_{\text{if } c_i > x} \right\} = 0$$

Solving the above equation, we get

$$c_i = \text{median} \{x | x \in C_i\}$$

- Thus, the best centroid for minimizing SAE of a cluster is the median of the objects in the cluster.



Interpret the best centroid for maximizing TC (with Cosine similarity measure) of a cluster.

The above mentioned discussion is quite sufficient for the validation of k-Means algorithm.

Comments on k-Means algorithm

5. Complexity analysis of k-Means algorithm

Let us analyse the time and space complexities of k-Means algorithm.

Time complexity:

The time complexity of the k-Means algorithm can be expressed as

$$T(n) = O(n \times m \times k \times l)$$

where n = number of objects

m = number of attributes in the object definition

k = number of clusters

l = number of iterations.

Thus, time requirement is a linear order of number of objects and the algorithm runs in a modest time if $k \ll n$ and $l \ll n$ (the iteration can be moderately controlled to check the value of l).

Comments on k-Means algorithm

5. Complexity analysis of k-Means algorithm

Space complexity: The storage complexity can be expressed as follows.

It requires $n \times m$ space to store the objects and $n \times k$ space to store the proximity measure from n objects to the centroids of k clusters.

Thus the total storage complexity is

$$S(n) = O(n \times (m + k))$$

That is, space requirement is in the linear order of n if $k \ll n$.

Comments on k-Means algorithm

Advantages:

- k-Means is simple and can be used for a wide variety of object types.
- It is also efficient both from storage requirement and execution time point of views. By saving distance information from one iteration to the next, the actual number of distance calculations, that must be made can be reduced (specially, as it reaches towards the termination).



How similarity metric can be utilized to run k-Means faster? What is the updation in each iteration?

Limitations:

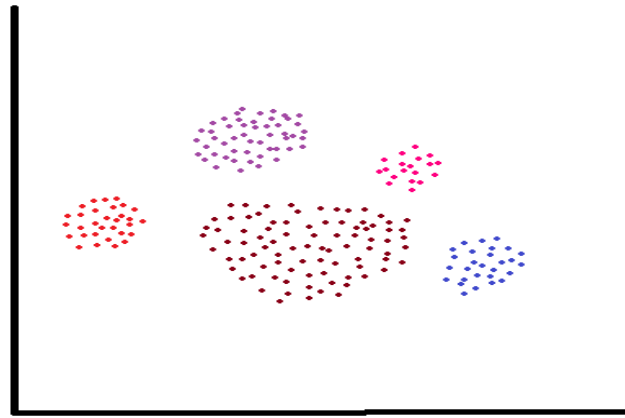
- The k-Means is not suitable for all types of data. For example, k-Means does not work on categorical data because mean cannot be defined.
- k-means finds a local optima and may actually minimize the global optimum.

Comments on k-Means algorithm

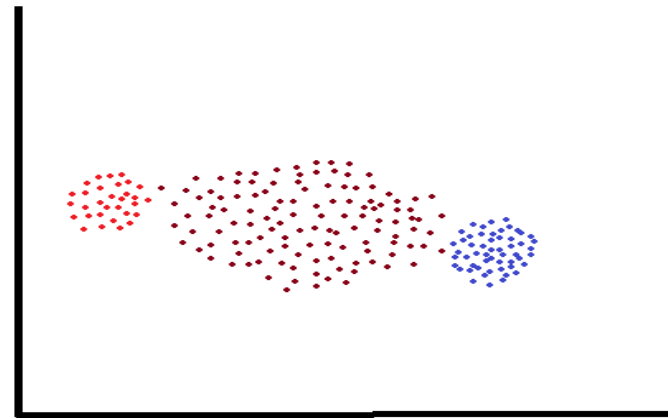
Limitations :

- k-means has **trouble clustering data that contains outliers**. When the SSE is used as objective function, outliers can unduly influence the cluster that are produced. More precisely, in the presence of outliers, the cluster centroids, in fact, not truly as representative as they would be otherwise. It also influence the SSE measure as well.
- k-Means algorithm **cannot handle non-globular clusters**, clusters of different sizes and densities (see Fig in the next slide).
- k-Means algorithm not really beyond the scalability issue (and not so practical for large databases).
- More to Explore: Weighted k-Means, MW k-Means, Spectral Clustering.

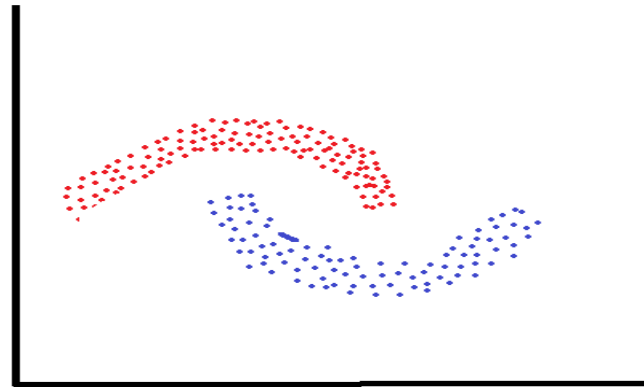
Comments on k-Means algorithm



Cluster with different sizes



Cluster with different densities



Non-convex shaped clusters

Fig : Some failure instance of k-Means algorithm

The k-Medoids algorithm

Now, we shall study a variant of partitioning algorithm called k-Medoids algorithm.

Motivation: We have learnt that the k-Means algorithm is sensitive to outliers because an object with an “extremely large value” may substantially distort the distribution. The effect is particularly exacerbated due to the use of the SSE (sum-of-squared error) objective function. The k-Medoids algorithm aims to diminish the effect of outliers.

Basic concepts:

- The basic concepts of this algorithm is to **select an object as a cluster center** (one representative object per cluster) instead of taking the mean value of the objects in a cluster (as in k-Means algorithm).
 - We call this cluster representative as a **cluster medoid** or simply **medoid**.
1. Initially, it selects a random set of k objects as the set of medoids.
 2. Then at each step, all objects from the set of objects, which are not currently medoids are examined one by one to see if they should be medoids.

The k-Medoids algorithm

- That is, the k-Medoids algorithm **determines** whether there is an object that should replace one of the current medoids.
- This is accomplished by looking all pair of medoid, non-medoid objects, and then choosing a pair that improves the objective function of clustering the best and exchange them.
- The sum-of-absolute error (SAE) function is used as the objective function.

$$SAE = \sum_{i=1}^k \sum_{x \in C_i, x \notin M \text{ and } c_m \in M} |x - c_m|$$

Where c_m denotes a medoid

M is the set of all medoids at any instant

x is an object belongs to set of non-medoid object, that is, x belongs to some cluster and is not a medoid. i.e. $x \in C_i, x \notin M$

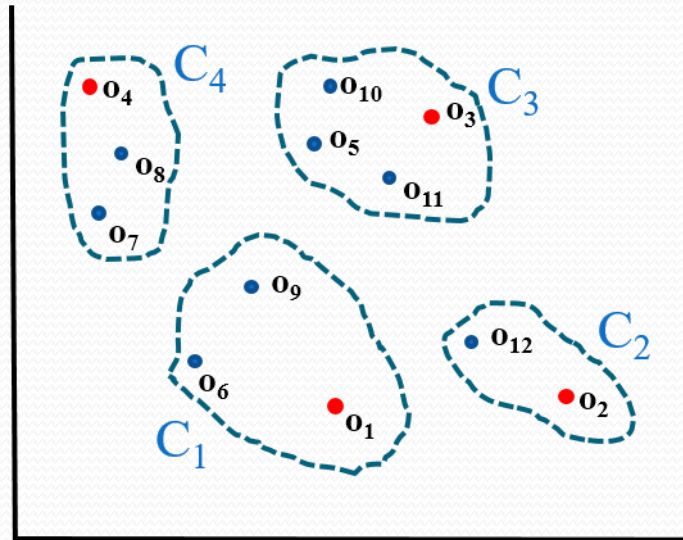
PAM (Partitioning around Medoids)

- For a given set of medoids, at any iteration, it select that exchange which has minimum SAE.
- The procedure terminates, if there is no any change in SAE in syuccessive iteration (i.e. there is no change in medoid).
- This k-Medoids algorithm is also known as PAM (Partitioning around Medoids).

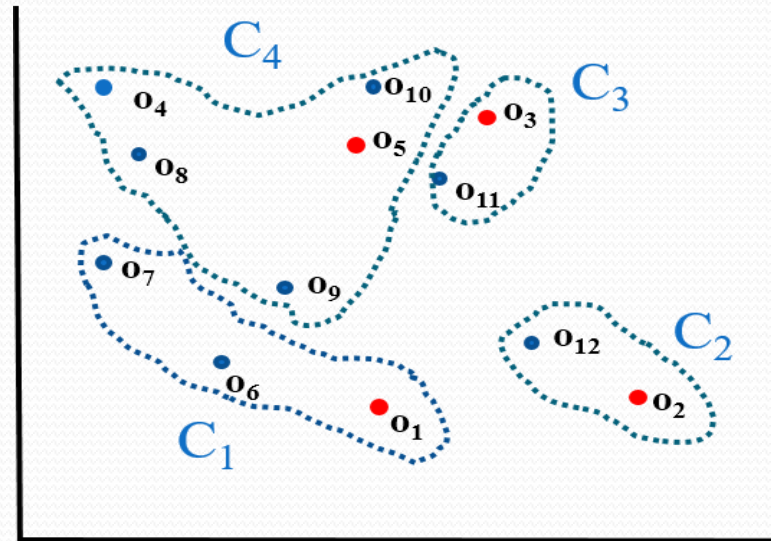
Illustration of PAM

- Suppose, there are set of 12 objects $O(o_1, o_2, \dots, o_{12})$ and we are to cluster them into four clusters. At any instant, the four cluster C_1, C_2, C_3 and C_4 are shown in Fig. (a). Also assume that o_1, o_2, o_3 , and o_4 are the medoids in the clusters C_1, C_2, C_3 and C_4 , respectively. For this clustering we can calculate SAE.
- There are many ways to choose a non-medoid object to be replaced any one medoid object. Out of these, suppose, if o_5 is considered as candidate medoid instead of o_4 , then it gives the lowest SAE. Thus, the new set of medoids would be o_1, o_2, o_3 , and o_5 . The new cluster is shown in Fig (b).

PAM (Partitioning around Medoids)



(a) Cluster with o_1, o_2, o_3 , and o_4 as medoids



(b) Cluster after swapping o_4 and o_5 (o_5 becomes the new medoid).

Fig : Illustration of PAM

PAM (Partitioning around Medoids)

PAM algorithm is thus a procedure of iterative selection of medoids and it is precisely stated in Algorithm 2.

Algorithm 2: PAM

Input: Database of objects D .

k , the number of desired clusters.

Output: Set of k clusters

Steps:

1. Arbitrarily select k medoids from D .
2. **For** each object o_i not a medoid **do**
3. **For** each medoid o_j **do**
4. Let $M = \{o_1, o_2, \dots, o_{i-1}, o_i, o_{i+1}, o_k\}$ //Set of current medoids
 $M' = \{o_1, o_2, \dots, o_{j-1}, o_j, o_{j+1}, o_k\}$ //set of medoids but swap with non-medoids o_j
5. Calculate $cost(o_i, o_j) = SAE|_M - SAE_{M'}$,
6. **End** of 2 for loop

PAM (Partitioning around Medoids)

Algorithm 2: PAM

7. Find o_i, o_j for which the $\text{cost}(o_i, o_j)$ is the smallest.
8. Replace o_i with o_j and accordingly update the set M .
9. Repeat step 2 - step 8 until $\text{cost}(o_i, o_i) \leq 0$.
10. Return the cluster with M as the set of cluster centers.
11. Stop

Comments on PAM

1. Comparing k-Means with k-Medoids:

- Both algorithms need to fix k , the number of clusters prior to the algorithms. Also, both algorithms arbitrarily choose the initial cluster centroids.
- The k-Medoid method is more robust than k-Means in the presence of outliers, because a medoid is less influenced by outliers than a mean.

2. Time complexity of PAM:

- For each iteration, PAM considers $k(n - k)$ pairs of objects o_i, o_j for which a cost $cost(o_i, o_j)$ is determined. Calculating the cost during each iteration requires that the cost be calculated for all other non-medoids o_j . There are $n - k$ of these. Thus, the total time complexity per iteration is $n(n - k)^2$. The total number of iterations may be quite large.

3. Applicability of PAM:

- PAM does not scale well to large databases because of its computational complexity.

Other variants of k-Medoids algorithms

- There are some variants of PAM that are targeted mainly large datasets are CLARA (Clustering LARge Applications) and CLARANS (Clustering Large Applications based upon RANdomized Search), it is an improvement of CLARA.

References:

For PAM and CLARA:

- L. kaufman and P. J. Rousseew, “Finding Groups in Data: An introduction to cluster analysis”, John and Wiley, 1990.

For CLARANS:

- R. Ng and J. Han, “Efficient and effective clustering method for spatial Data mining”, Proceeding very large databases [VLDB-94], 1994.

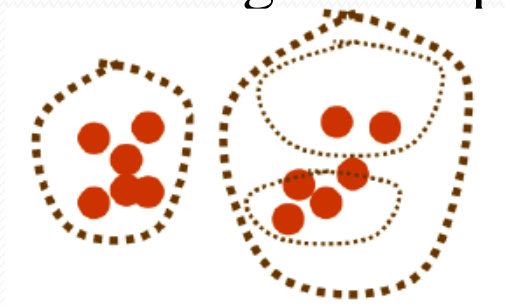
Hierarchical clustering

- Consider flat clustering



- For some data hierarchical clustering more appropriate than 'flat' clustering

- Hierarchical clustering



Types of Hierarchical clustering

Divisive (top down) clustering

- Starts with all data points in one cluster, the root, then
 - 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, i.e., each cluster with only a single point

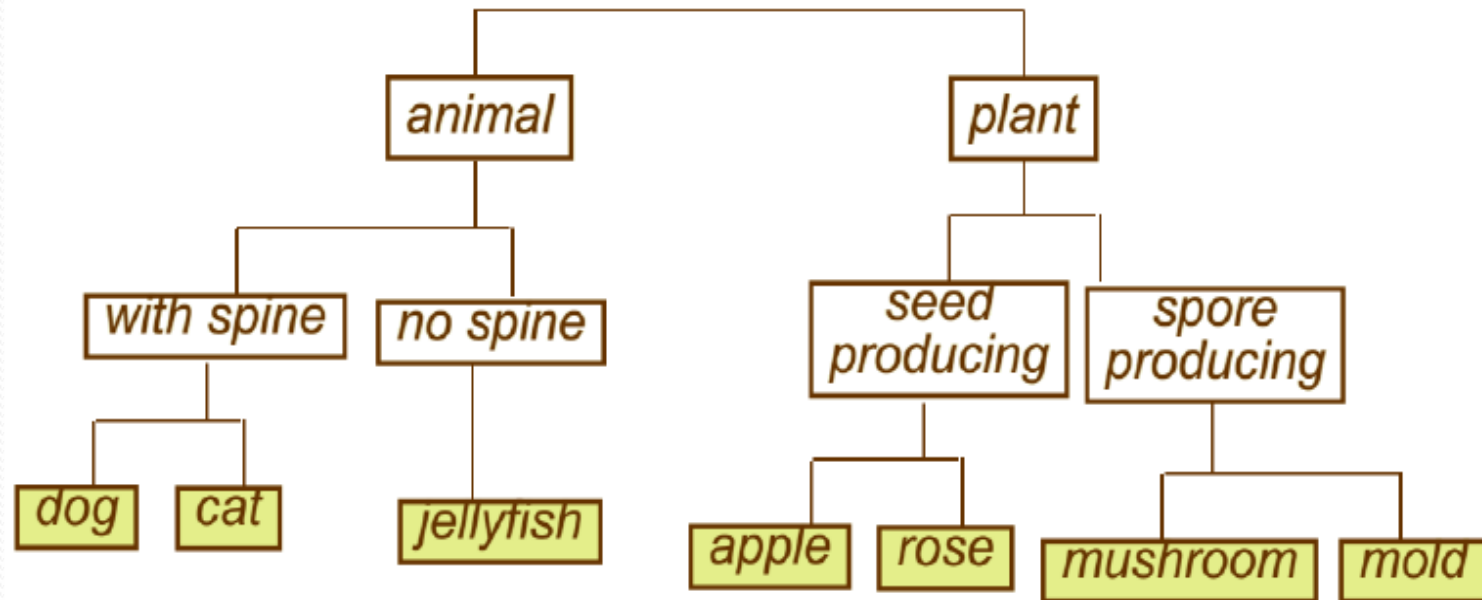
Agglomerative (bottom up) clustering

- The dendrogram is built from the bottom level by
 - merging the most similar (or nearest) pair of clusters
 - stopping when all the data points are merged into a single cluster (i.e., the root cluster).
- The number of dendrograms with n leafs = $(2n - 3)! / [(2^{n-2}) (n - 2)!]$

Summary of Hierarchical Clustering Methods

- No need to specify the number of clusters in advance.
- Hierarchical nature maps nicely onto human intuition for some domains
- They do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects.
- Like any heuristic search algorithms, local optima are a problem.
- Interpretation of results is (very) subjective.

Example: biological taxonomy





Any question?

You may send your question(s) at ctanujit@gmail.com