e-PGPathshala
Subject : Computer Science
Paper: Data Analytics
Module No 28: CS/DA/28 -Clustering - 3
Quadrant 1 – e-text

1.1 Learning Objectives

- Know about clustering algorithms for arbitrary shapes clusters
- Understand the steps in CURE, CLIQUE and ProClus algorithms

1.2 Introduction to Extension of *k*-means to clusters of arbitrary shapes -The CURE Algorithm

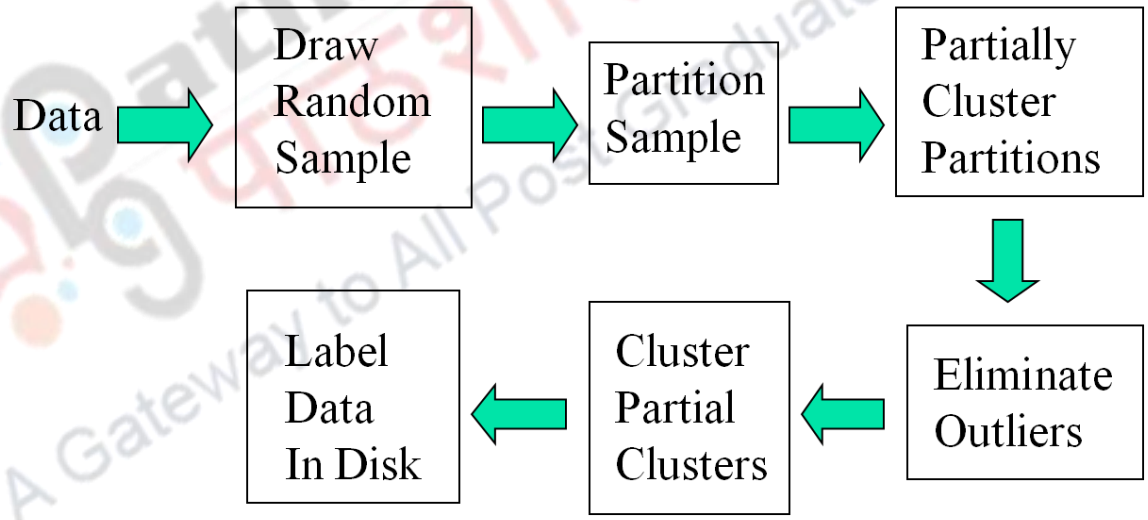
A new scalable algorithm called CURE is introduced, which uses random sampling and partitioning to reliably find clusters of arbitrary shape and size. CURE clusters a random sample of the database in an agglomerative fashion, dynamically updating a constant number c of well-scattered points R1, . . . , Rc per cluster to represent each cluster's shape. To assign the remaining, unsampled points to a cluster, these points Ri are used in a similar manner to centroids in the k-means algorithm – each data point that was not in the sample is assigned to the cluster which contains the point Ri closest to the data point. To handle large sample sizes, CURE divides the random sample into partitions which are pre-clustered independently, then the partially-clustered sample is clustered further by the agglomerative algorithm. A heap is used to keep track of which clusters to merge at each iteration, and a kd-tree is used to find the closest representative point to a given point x. The time complexity of CURE is O(n 2 log n) in general, and in the case when the data points lie in a two-dimensional space, the time complexity can be shown to reduce to O(n 2).

A new algorithm for detecting arbitrarily-shaped clusters at large-scale is presented and named CURE, for "Clustering Using Representatives". The algorithm works by pre-clustering a sample of the entire dataset, then using representative points within the sample to assign the remainder of the dataset. The sample is clustered using an agglomerative clustering algorithm which keeps track

of the representative points in each cluster, as well as the nearest neighbor of each cluster at each step. The goal of CURE is to detect clusters of arbitrary shapes and sizes.

1.3 Contribution of CURE, ideas:

- CURE employs anew hierarchical algorithm that adopts a middle ground between centroid- based and all-points approach.
- A constant number of well scattered points in a cluster are chosen as representative. This point catch all the possible form that could have the cluster.
- The cluster with the closest pair of representative points are the cluster that are merged at each step of CURE.
- Random Sampling and partitioning are used for reducing the data set of input.



Six Steps in CURE Algorithm

CURE's Advantages

More accurate:

Adjusts well to geometry of non-spherical shapes.

Scales to large datasets

Less sensitive to outliers

More efficient:

Space complexity: O(n)

Time complexity: $O(n^2 \log n)$ ($O(n^2)$ if dimensionality of data points is small)

Features:

Random Sampling

- Key idea: apply CURE to a random sample drawn from the data set rather than the entire data set.
- Advantages:Smaller size, Filtering outliers.
- Concerns: may miss out or incorrectly identify certain clusters!
- Experimental results show that, with moderate sized random samples, we were able to obtain very good clusters

Partitioning for Speedup

- Partition the sample space into p partitions, each of size n/p.
- Partially cluster each partition until the final number of clusters in each partition reduces to n/(pq). (q > 1)
- Collect all partitions and run a second clustering pass on the n/p partial clusters
- Tradeoff: sample size vs. accuracy

Labeling Data on Disk

- Input is a randomly selected sample.
- Have to assign the appropriate cluster labels to the remaining data points
- Each data point is assigned to the cluster containing the representative point closest to it

- Advantage: using multiple points enables CURE to correctly to distribute the data points when clusters are non-spherical or non-union

Outliers Handling

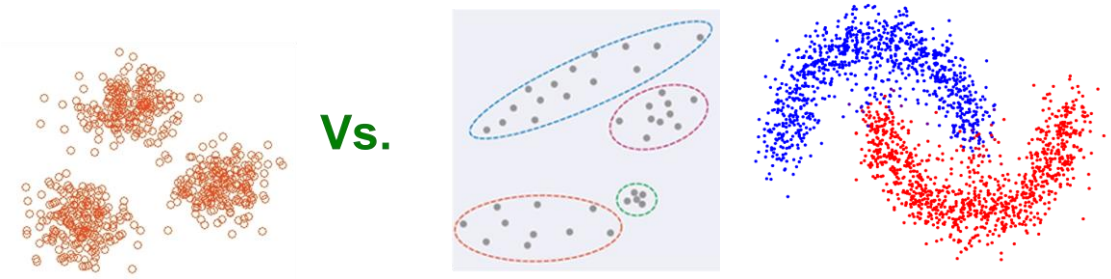
- Random sampling filters out a majority of the outliers.
- The remaining few outliers in the random sample are distributed all over the sample space and gets further isolated.
- The clusters which are growing very slowly are identified and eliminated as outliers.
- Use a second level pruning to eliminate merging-together outliers: outliers form very small clusters.

Problem with BFR/*k*-means:

- Assumes clusters are normally distributed in each dimension
- And axes are fixed – ellipses at an angle are not OK

CURE (Clustering Using REpresentatives):

- Assumes a Euclidean distance
- Allows clusters to assume any shape
- Uses a collection of representative points to represent clusters



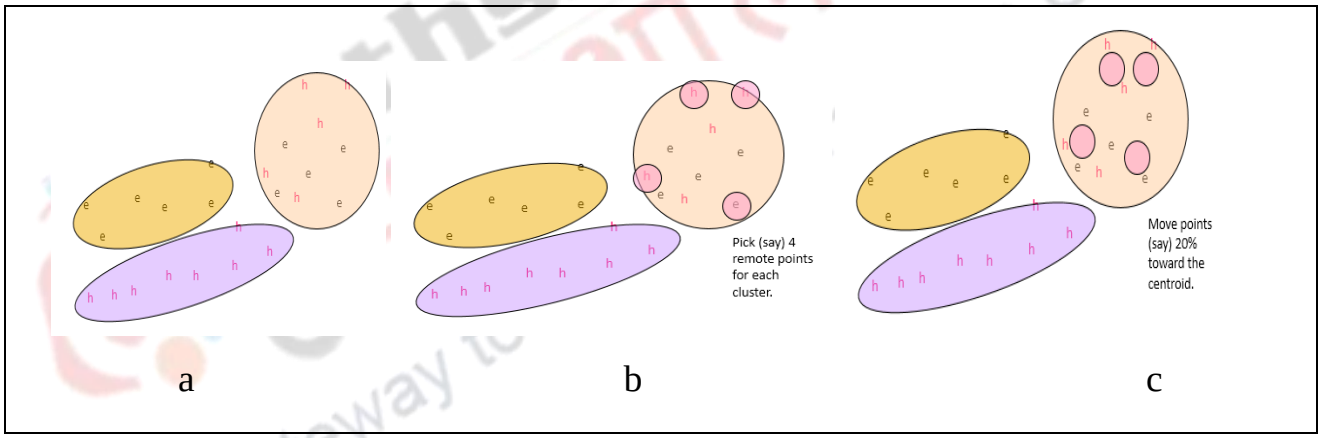
1.4 2 Phase algorithm

Pass 1:

- Pick a random sample of points that fit in main memory

Initial clusters:

- Cluster these points hierarchically – group nearest points/clusters
- Pick representative points:
- For each cluster, pick a sample of points, as dispersed as possible
- From the sample, pick representatives by moving them (say) 20% toward the centroid of the cluster



a :Initial Clusters

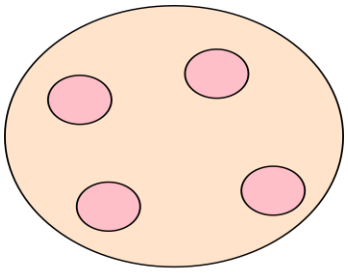
b : Pick Dispersed Points(Pick(say) 4 remote points for each cluster)

c :Pick Dispersed Points(Move points (say) 20%toward the centroid)

Pass 2:

- Now, rescan the whole dataset and visit each point ***p*** in the data set
- Place it in the "closest cluster"
  - Normal definition of "closest":

- Find the closest representative to ***p*** and assign it to representative's cluster



p

1.5 CLIQUE

A clique is defined as a maximal complete sub-graph of a given graph—i.e., a group of people where everybody is connected directly to everyone else. The word "maximal" means that no other nodes can be added to the clique without making it less connected. Essentially, a clique consists of several overlapping closed triads, and inherits many of the culture-generating, and amplification properties of closed triads.

A clique must generate consensus or fall apart—which is why in the vernacular, cliques are often viewed in conflict with other cliques. It is, in fact, very easy to agree about conflict, and having a common enemy (or a group of common enemies) helps cliques unite

- Clustering In QUEst
- Agrawal, Gehrke, Gunopulos, Raghavan (SIGMOD'98).
- Automatically identifying subspaces of a high dimensional data space that allow better clustering than original space
- CLIQUE can be considered as both density-based and grid-based

- It partitions each dimension into the same number of equal length interval
- It partitions an m-dimensional data space into non-overlapping rectangular units
- A unit is dense if the fraction of total data points contained in the unit exceeds the input model parameter
- A cluster is a maximal set of connected dense units within a subspace

The Major Steps

- Partition the data space and find the number of points that lie inside each cell of the partition.
- Identify the subspaces that contain clusters using the Apriori principle
- Identify clusters:
  - Determine dense units in all subspaces of interests
  - Determine connected dense units in all subspaces of interests.
- Generate minimal description for the clusters
  - Determine maximal regions that cover a cluster of connected dense units for each cluster
  - Determination of minimal cover for each cluster

1.6 Flow Chart of CLIQUE:

- Bottom-up to find dense units
- Further Prune subspaces using g MDL principle
- Generating g Minimal number of Regions, each region cover one cluster
- Firstly y, greedily y find a number of maximal rectangles

- Generate a minimal cover

Basic Idea of CLIQUE:

- Monotonicity:
    - If a collection of points S is a cluster in a Kdimensional space, then S is also part of a cluster in any (k-1) dimensional projections of this space.

Strength and Weakness of CLIQUE

- Strength :
    - It automatically finds subspaces of the highest dimensionality such that high density clusters exist in those subspaces
    - It is insensitive to the order of records in input and does not presume some canonical data distribution
    - It scales linearly with the size of input and has good scalability as the number of dimensions in the data increases
- Weakness :
    - The accuracy of the clustering result may be degraded at the expense of simplicity of the method

1.7 PROCLUS

Projected cluster: subset of data points, together with a subset of dimensions, such that the points are closely clustered in the corresponding subspace.

PROCLUS (Projected Clustering) is a projected clustering algorithm which is partitioned in nature. In a high dimensional data, all dimensions are not relevant to each cluster. (Aggarwal et al 1999) referred that the problem of handling very high dimensional data can be solved by picking the projected dimensions and discovering clusters in those projected sub spaces. The main goal of their algorithm is to pick up the relevant dimensions on which data points are mostly correlated. Pruning the irrelevant dimensions from the data set greatly reduces the noise. Alternatively, feature selection methods can also be used for selecting relevant

dimensions, but, sometimes it will lead to loss of information. The concept of discovering 29 clusters in a sub set of dimensions has been observed for the first time by Aggarwal et al (1998). The output of the PROCLUS algorithm will be k+1 partition of the data such as C1, C2 .. Ck and O. The partitions are called projected clusters and O is called as outlier

Objective of PROCLUS

- PROCLUS (**Pro**jected **Clus**tering)
  - Algorithm to find out the clusters and the dimensions for the corresponding clusters
  - Splits out those Outliers (points that do not cluster well) from the clusters

Input and Output for PROCLUS

Input:

- The set of data points
- Number of clusters, denoted by k
- Average number of dimensions for each clusters, denoted by L

Output:

- The clusters found, and the respective dimensions to such clusters

Three Phase for PROCLUS:

- Initialization Phase
- Iterative Phase
- Refinement Phase

- The best set of mediods is found by a hill climbing process. Hill climbing simply means process of improving the set of mediods.

1.7.1 Initialization Phase

The purpose of initialization phase is to select representative points from each cluster. In full dimensional space, piercing set of mediods is identified with the help of greedy method which is proposed in (Gonsales 1985). In PROCLUS, greedy technique is used to find the good superset of the piercing set of mediods. Initialization phase also reduces the size of the original data set.

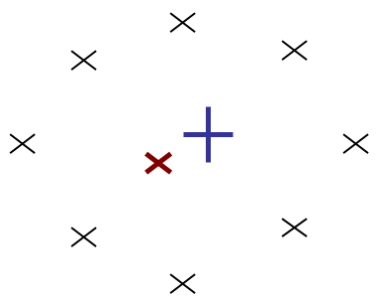
- Choose a sample set of data point randomly.
- Choose a set of data point which is probably the mediods of the clusters

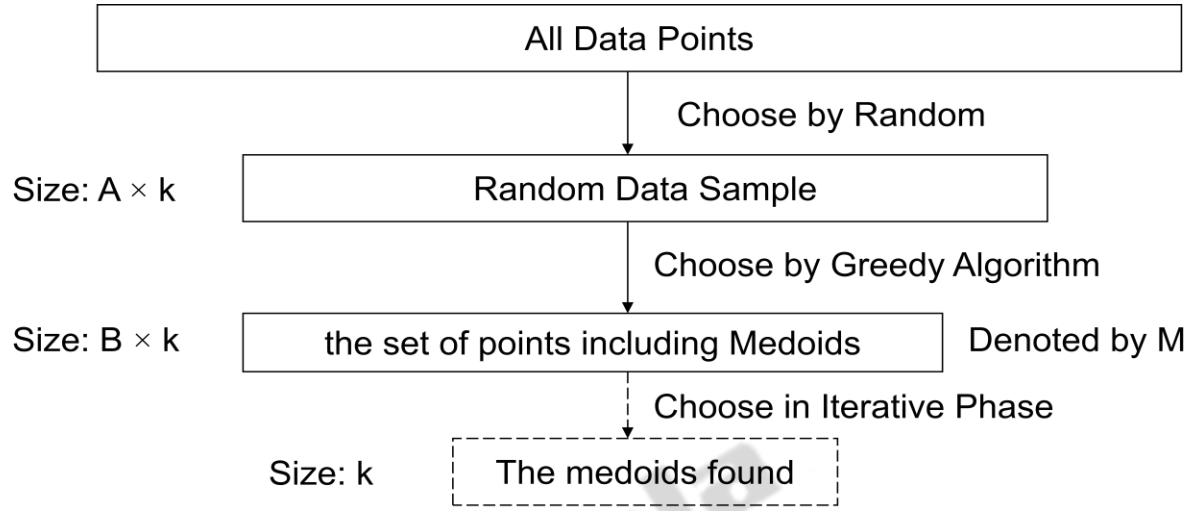
Choosing Mediods of clusters

Medoids are representative objects of a data set or a cluster with a data set whose average dissimilarity to all the objects in the cluster is minimal. Medoids are similar in concept to means or centroids, but medoids are always members of the data set. Medoids are most commonly used on data when a mean or centroid cannot be defined such as 3-D trajectories or in the gene expression context. The term is used in computer science in data clustering algorithms.

For some data sets there may be more than one medoid, as with medians. A common application of the medoid is the k-medoids clustering algorithm, which is similar to the k-means algorithm but works when a mean or centroid is not definable. This algorithm basically works as follows. First, a set of medoids is chosen at random. Second, the distances to the other points are computed. Third, data are clustered according to the medoid they are most similar to. Fourth, the medoid set is optimized via an iterative process.

- Medoid for a cluster is the **data point** which is nearest to the center of the cluster





Greedy Algorithm

A greedy algorithm is an algorithmic paradigm that follows the problem solving heuristic of making the locally optimal choice at each stage with the hope of finding a global optimum.

Greedy algorithms work by recursively constructing a set of objects from the smallest possible constituent parts. Recursion is an approach to problem solving in which the solution to a particular problem depends on solutions to smaller instances of the same problem. The advantage to using a greedy algorithm is that solutions to smaller instances of the problem can be straightforward and easy to understand. The disadvantage is that it is entirely possible that the most optimal short-term solutions may lead to the worst possible long-term outcome.

Greedy algorithms have some advantages and disadvantages:

- It is quite easy to come up with a greedy algorithm (or even multiple greedy algorithms) for a problem.
- Analyzing the run time for greedy algorithms will generally be much easier than for other techniques (like Divide and conquer). For the Divide and conquer technique, it is not clear whether the technique is fast or slow. This is

because at each level of recursion the size of gets smaller and the number of sub-problems increases.

- The difficult part is that for greedy algorithms you have to work much harder to understand correctness issues. Even with the correct algorithm, it is hard to prove why it is correct. Proving that a greedy algorithm is correct is more of an art than a science. It involves a lot of creativity.

  - Avoid to choose the mediods from the same clusters.
  - Therefore the way is to choose the set of points which are most far apart.
  - Start on a random point

	A	B	C	D	E
A	0	1	3	6	7
B	1	0	2	4	5
C	3	2	0	5	2
D	6	4	5	0	1
E	7	5	2	1	0

A Randomly Choosed first
Set {A}

Choose E
Set {A, E}

Minimum Distance to the points in the Set

A	B	C	D	E
-	1	3	6	7

A	B	C	D	E
-	1	2	1	-

1.7.2 Iterative Phase

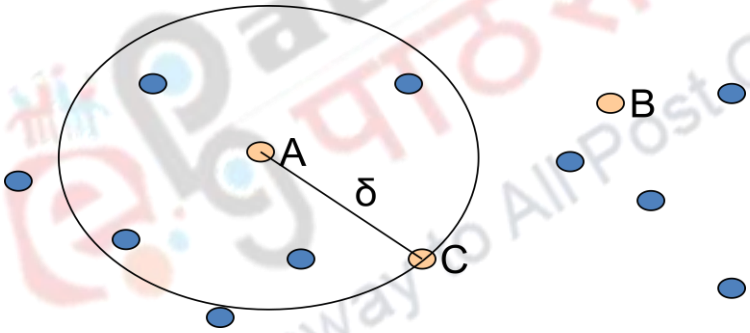
In this phase, bad mediods are iteratively replaced with good mediods in order to improve the quality. Further, an appropriate set of dimensions for each mediod is also identified. Given the mediods and their associated dimensions, all points are assigned to the nearest mediods. Quality of a set of mediods is evaluated as the average Manhattan segmental distance from the points to the centroids of the clusters to which they belong.

- From the Initialization Phase, we got a set of data points which should contains the mediods. (Denoted by M)
- This phase, we will find the best mediods from M.

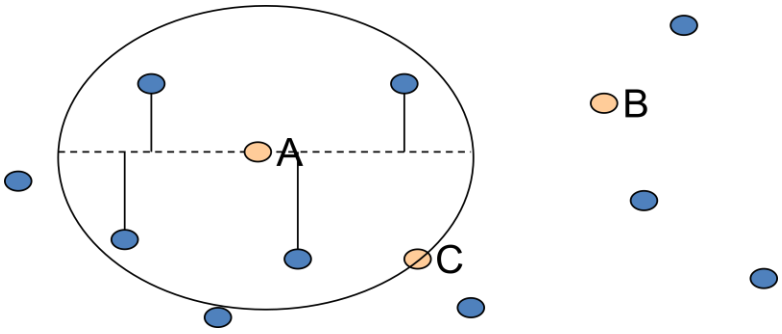
- Randomly find the set of points $M_{current}$, and replace the “bad” mediods from other point in M if necessary.
- For the mediods, following will be done:
    - Find Dimensions related to the mediods
    - Assign Data Points to the mediods
    - Evaluate the Clusters formed
    - Find the bad medoid, and try the result of replacing bad medoid
- The above procedure is repeated until we got a satisfied result

Find Dimensions

- For each medoid m_i, let D be the nearest distance to the other medoid
- All the data points within the distance will be assigned to the medoid m_i



- For the points assigned to medoid m_i, calculate the average distance $X_{i,j}$ to the medoid in each dimension j



- Calculate the mean Y_i and standard deviation σ_i of $X_{i,j}$ along j
- Calculate $Z_{i,j} = (X_{i,j} - Y_i) / \sigma_i$
- Choose $k \times L$ most negative of $Z_{i,j}$ with at least 2 chosen for each mediods

Assign Points

- For each data point, assign it to the medoid m_i if its Manhattan Segmental Distance for Dimension D_i is minimum, the point will be assigned to m_i

Manhattan Segmental Distance

The Manhattan distance function computes the distance that would be traveled to get from one data point to the other if a grid-like path is followed. The Manhattan distance between two items is the sum of the differences of their corresponding components. It is defined as the distance between two points measured along axes at right angles. In a plane with p_1 at (x_1, y_1) and p_2 at (x_2, y_2), it is $|x_1 - x_2| + |y_1 - y_2|$.

- Manhattan Segmental Distance is defined relative to a dimension.
- The Manhattan Segmental Distance between the point x_1 and x_2 for the dimension D is defined as:

$$d_D(x_1, x_2) = \frac{\Sigma_{i \in D} |x_{1,i} - x_{2,i}|}{|D|}$$

Evaluate Clusters

- For each data points in the cluster i, find the average distance $Y_{i,j}$ to the centroid along the dimension j, where j is one of the dimension for the cluster.
- Calculate the follows:

$$w_i = \frac{\sum_j Y_{i,j}}{|D_i|} \qquad E = \frac{\sum_{i=k}^{k} |C_i| \cdot w_i}{N}$$

- The value will be used to evaluate the clusters. The lesser the value, the better the clusters.
- Try to compare the case when a bad medoid is replaced, and replace the result if the value calculated above is better
- The bad medoid is the medoid with least number of points.

1.7.3 Refinement Phase

Once the best set of mediods is found, one more pass over the data is performed to improve the quality of the clustering. For each cluster C1…Ck formed during iterative phase, the dimensions associated with each mediod are discarded and new dimensions are identified. Once the new set of dimensions have been computed, all data points are reassigned to 30 the mediods relative to these new set of dimensions until there is no further change in dimensions and mediods.

- Redo the process in Iterative Phase once by using the data points distributed by the result cluster, but not the distance from mediods
- Improve the quality of the result
- In iterative phase, we don't handle the outliers, and now we will handle it.

Handle Outliers

- For each medoid m_i with the dimension D_i, find the smallest Manhattan segmental distance Δ_i to any of the other mediods with respect to the set of dimensions D_i

$$\Delta_i = \min_{j \neq i} d_{D_i}(m_i, m_j)$$

- Δ_i is the sphere of influence of the medoid m_i
- A data point is an outlier if it is not under any spheres of influence.

Summary of PROCLUS

Input:	Database containing n objects and k (Number of clusters) and l (number of dimensions)

Output:	Formation of k clusters
Phase 1:	Initialization Phase. The best set of mediods is found by hill climbing process. The size of the original data set is alsoreduced.
Phase 2:	Iterative Phase: Bad mediods are replaced with good mediods. Appropriate set of dimensions are also identified.
Phase 3:	Refinement Phase: Mediods and dimensions are refined.

Summary

- Algorithms for deriving clusters of arbitrary shapes require different distance metrics
- Finding subspaces and relevant dimensions are necessary for good clustering.
- Detailed discussion about CURE, CLIQUE and PROCLUS algorithms

Pathshala A Gateway to All Post Graduate Courses