An Efficient Density based Improved K- Medoids Clustering algorithm

Raghuvira Pratap A
M.Tech,CSE Department
P.V.P Siddhartha Institute of Technology
Vijayawada-07, AP

K Suvarna Vani
Assistant Professor,CSE Department
V.R. Siddhartha Engineering College
Vijayawada -07, AP

J Rama Devi
Sr.Assistant Professor,CSE Department
P.V.P Siddhartha Institute of Technology
Vijayawada-07, AP

Dr.K Nageswara Rao
Professor & Head,CSE Department
P.V.P Siddhartha Institute of Technology
Vijayawada-07, AP

***Abstract*— Clustering is the process of classifying objects into different groups by partitioning sets of data into a series of subsets called clusters. Clustering has taken its roots from algorithms like k-medoids and k-medoids. However conventional k-medoids clustering algorithm suffers from many limitations. Firstly, it needs to have prior knowledge about the number of cluster parameter k. Secondly, it also initially needs to make random selection of k representative objects and if these initial k medoids are not selected properly then natural cluster may not be obtained. Thirdly, it is also sensitive to the order of input dataset.**

Mining knowledge from large amounts of spatial data is known as spatial data mining. It becomes a highly demanding field because huge amounts of spatial data have been collected in various applications ranging from geo-spatial data to bio-medical knowledge. The database can be clustered in many ways depending on the clustering algorithm employed, parameter settings used, and other factors. Multiple clustering can be combined so that the final partitioning of data provides better clustering. In this paper, an efficient density based k-medoids clustering algorithm has been proposed to overcome the drawbacks of DBSCAN and kmedoids clustering algorithms. The result will be an improved version of kmedoids clustering algorithm. This algorithm will perform better than DBSCAN while handling clusters of circularly distributed data points and slightly overlapped clusters.

Keywords- Clustering; DBSCAN; Centroid; Medoid; k-medoids.

I. INTRODUCTION

Numerous applications require the management of spatial data, i.e. data related to space. Spatial Database Systems (SDBS) (Gueting 1994) are database systems for the management of spatial data. Increasingly large amounts of data are obtained from satellite images, X-ray crystallography or other automatic equipment. Therefore, automated knowledge discovery becomes more and more important in spatial databases.

Clustering algorithms are attractive for the task of class identification. However, the application to large spatial databases raises the following requirements for clustering algorithms:

(1) Minimal requirements of domain knowledge to determine the input parameters, because appropriate values are often not known in advance when dealing with large databases.

(2) Discovery of clusters with arbitrary shape, because the shape of clusters in spatial databases may be spherical, drawn-out, linear, elongated etc.

(3) Good efficiency on large databases, i.e. on databases of significantly more than just a few thousand objects.

Clustering is considered as one of the important techniques in data mining and is an active research topic for the researchers. The objective of clustering is to partition a set of objects into clusters such that objects within a group are more similar to one another than patterns in different clusters. So far, numerous useful clustering algorithms have been developed for large databases, such as K-MEDOIDS [1], CLARANS [2], BIRCH [3], CURE [4], DBSCAN [5], OPTICS [6], STING [7] and CLIQUE [8]. These algorithms can be divided into several categories. Three prominent categories are partitioning, hierarchical and density-based. All these algorithms try to challenge the clustering problems treating huge amount of data in large databases. However, none of them are the most effective. In density-based clustering algorithms, which are designed to discover clusters of arbitrary shape in databases with noise, a cluster is defined as a high-density region partitioned by low-density regions in data space. DBSCAN (Density Based Spatial Clustering of Applications with Noise) [5] is a typical Density-based clustering algorithm. In this paper, we present a new algorithm which overcomes the drawbacks of DBSCAN and k-medoids clustering algorithms.

II. LITERATURE SURVEY

A. ***DBSCAN:*** *Density Based Spatial Clustering of Applications with Noise.*

In this section, we present the algorithm DBSCAN (Density Based Spatial Clustering of Applications with Noise) which is designed to discover the clusters and the noise in a spatial database. Ideally, we would have to know the appropriate parameters Eps and MinPts of each cluster and at least one point from the respective cluster. Then, we could retrieve all points that are density-reachable from the given point using the correct parameters. But there is no easy way to get this information in advance for all clusters of the database. However, there is a simple and effective heuristic to determine the parameters Eps and MinPts of the "thinnest", i.e. least dense, cluster in the database. Therefore, DBSCAN uses global values for Eps and MinPts, i.e. the same values for all clusters. The density parameters of the “thinnest” cluster are good candidates for these global parameter values specifying the lowest density which is not considered to be noise.

The idea of it was:

1. ε-neighbor: the neighbors in ε semi diameter of an object
2. Kernel object: certain number (*MinP*) of neighbors in ε semi diameter.
3. To a object set *D*, if object *p* is the ε-neighbor of *q*, and *q* is kernel object, then *p* can get “direct density reachable” from *q*.
4. To a ε, *p* can get “direct density reachable” from *q*; *D* contains *Minp* objects; if a series object $p_1, p_2, \ldots.. p_n, p_1 = q.\ p_n = q.$ *then* p_{i+1} can get “direct density reachable” from p_i.,
$p_i \in D, 1 \leq i \leq n.$
5. To ε and *MinP*, if there exist an object $o(o \in D)$ *p* and *q* can get “direct density reachable” from *o*,*p* and *q* are density connected.

Density Reachability and Density Connectivity:

Density reachability is the first building block in dbscan. It defines whether two distance close points belong to the same cluster. Points p1 is density reachable from p2 if two conditions are satisfied: (i) the points are close enough to each other: distance (p1, p2) <e, (ii) there are enough of points in is neighborhood: |{r: distance(r, p2)}|>m, where r is a database point.

Density connectivity is the last building step of dbscan. Points p0 and pn are density connected, if there is a sequence of density reachable points p1,i2,...,i(n-1) from p_0 to p_n such that p(i+1) is density reachable from pi. A dbscan cluster is a set of all density connected points.

Explanation of DBSCAN Steps

- DBScan requires two parameters: epsilon (eps) and minimum points (minPts). It starts with an arbitrary starting point that has not been visited. It then finds all the neighbor points within distance eps of the starting point.
- If the number of neighbors is greater than or equal to minPts, a cluster is formed. The starting point and its neighbors are added to this cluster and the starting point is marked as visited. The algorithm then repeats the evaluation process for all the neighbors’ recursively.
- If the number of neighbors is less than minPts, the point is marked as noise.
- If a cluster is fully expanded (all points within reach are visited) then the algorithm proceeds to iterate through the remaining unvisited points in the dataset.

Advantages of DBSCAN

- DBScan requires two parameters: epsilon (eps) and minimum points (minPts). It starts with an arbitrary starting point that has not been visited. It then finds all the neighbor points within distance eps of the starting point.
- If the number of neighbors is greater than or equal to minPts, a cluster is formed. The starting point and its neighbors are added to this cluster and the starting point is marked as visited. The algorithm then repeats the evaluation process for all the neighbors’ recursively.
- If the number of neighbors is less than minPts, the point is marked as noise.
- If a cluster is fully expanded (all points within reach are visited) then the algorithm proceeds to iterate through the remaining unvisited points in the dataset.

Disadvantages of DBSCAN

- DBScan requires two parameters: epsilon (eps) and minimum points (minPts). It starts with an arbitrary starting point that has not been visited. It then finds all the neighbor points within distance eps of the starting point.
- DBSCAN cannot cluster data sets well with large differences in densities, since the minPts-epsilon combination cannot be chosen appropriately for all clusters then

B. *K-Medoids*

K-medoid is a classical partitioning technique of clustering that clusters the data set of *n* objects into *k* number of clusters [1, 3]. This k: the number of clusters required is to be given by user. This algorithm operates on the principle of minimizing the sum of dissimilarities between each object and its corresponding reference point. The algorithm randomly chooses the k objects in dataset D as initial representative objects called medoids. A medoid can be defined as the object of a cluster, whose average dissimilarity to all the objects in the cluster is minimal i.e. it is a most centrally located point in the given data set. Then for all objects in the dataset, it assigns each object to the nearest cluster depending upon the object’s distance to the cluster medoid. After every assignment of a data object to particular cluster the new medoid is decided.

1) Input

k: the number of clusters.
D: a data set containing n objects.

2) Output

A set of k clusters.

3) Algorithm

1. Randomly choose k objects in D as the initial representative objects;
2. for all objects in the data set D

```
        a. Find the cluster C which is nearest to
           object i by using the dissimilarity measure;
        b. assign object i to cluster C;
        c. set the member object in cluster C having
           minimum intra cluster variance as new
           centroid of C
3. Display statistics of clusters obtained.
```

The problem is K-Medoids does not generate the same result with each run, because the resulting clusters depend on the initial random assignments. It is more robust than kmedoids in the presence of noise and outliers; however it's processing is more costly than the k-medoid method. Lastly, the optimal number of clusters k is hard to be predicted, so it is difficult for a user without prior knowledge to specify the value of k.

Problems with kmedoids clustering algorithm

The algorithm is simple and has nice convergence but there are number of problems with this. Some of the weaknesses of k-mediods are

- When the numbers of data are not so many, initial grouping will determine the cluster significantly.
- The result is circular cluster shape because based on distance.
- The number of cluster, K, must be determined beforehand. Selection of value of K is itself an issue and sometimes it's hard to predict beforehand the number of clusters that would be there in data.
- We never know the real cluster, using the same data, if it is inputted in a different order may produce different cluster if the number of data is few.
- Experiments have shown that outliers can be a problem and can force algorithm to identify false clusters.
- We never know which attribute contributes more to the grouping process since we assume that each attribute has the same weight.

III. Proposed Algorithm

```
Let D is the Dataset with k points
k be the number of clusters to be found
l be the number of clusters initially found by density based clustering algorithm
ε be the Euclidean neighborhood radius
ή Minimum number of neighbors required in ε neighborhood to form a cluster
p can be any point in D
N is a set of points in ε neighborhood of p
c=0
for each unvisited point p in dataset D

{
    N = getNeighbors (p, ε)
    if (sizeof(N) < ή)
        mark p as NOISE
    else
        ++ c
        mark p as visited
        add p to cluster c
        recurse (N)
}
Now will have m clusters
for each detected clusters {
    find the cluster centers Cmby taking the representative object.
    find the total number of points in each cluster
}
If m>k {
    # Join two or more as follows
    select two cluster based on density and number of points satisfying the application criteria and joint them and find the new cluster center and repeat it until achieving k clusters.
    Finally we will have Ck centers
} else {
    l =k-m
    # split one or more as follows
    if ( m >=l ) {
        Select a cluster based on density and number of points satisfying the application criteria and split it using kmedoids clustering algorithm and repeat it until achieving k clusters.

        Finally we will have Ck centers
    }
Apply one iteration of k-medoid clustering with k and new Ck centers as the initial parameters and label all the clusters with k labels.
```

Note: in our simulation of the algorithm, we only assumed overlapped clusters of circular or spheroid in nature. So the criteria for splitting or joining a cluster can be decided based on the number of expected points in a cluster or the expected density of the cluster (derived by using the number of points in a cluster and the area of the cluster)

IV. Evaluation and Results

Metrics Used For Evaluation

In order to measure the performance of a clustering and classification system, a suitable metric will be needed. For evaluating the algorithms under consideration, we used Rand Index and Run Time as two measures

A. Performance in terms of time

We evaluated the three algorithms DBSCAN, k-medoid and DBkmedoids in terms of time required for clustering. The Attributes of Multidimensional Data:

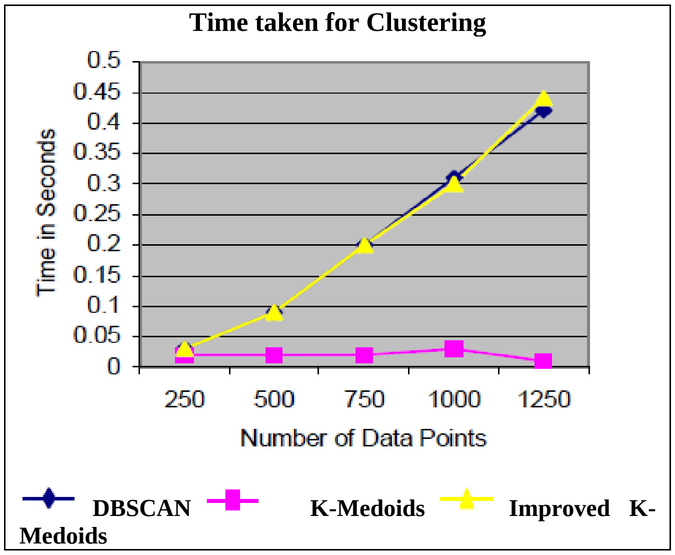
The Number of Classes: 5

The Number of Dimensions: 2

The Number of Points per Class: 50, 100, 150, 200,250

The Standard Deviation: 7.000000e-001

S.No.	Total No. of Records in Proteins Data Base	Time taken for Classification (in Seconds)		
		DBSCAN	K-Medoids	Density based Improved K-Medoids
1	250	0.03	0.02	0.03
2	500	0.09	0.02	0.09
3	750	0.2	0.02	0.2
4	1000	0.31	0.03	0.3
5	1250	0.42	0.01	0.44



B. *Performance in terms of accuracy*

The Rand index or Rand measure is a commonly used technique for measure of such similarity between two data clusters. This measure was found by W. M. Rand and explained in his paper "Objective criteria for the evaluation of clustering methods" in Journal of the American Statistical Association (1971).

Given a set of n objects S = {O1, ..., On} and two data clusters of S which we want to compare: X = {x1, ..., xR} and Y = {y1, ..., yS} where the different partitions of X and Y are disjoint and their union is equal to S; we can compute the following values:

a is the number of elements in S that are in the same partition in X and in the same partition in Y,
b is the number of elements in S that are not in the same partition in X and not in the same partition in Y,
c is the number of elements in S that are in the same partition in X and not in the same partition in Y,
d is the number of elements in S that are not in the same partition in X but are in the same partition in Y.

Intuitively, one can think of a + b as the number of agreements between X and Y and c + d the number of disagreements between X and Y. The Rand index, R, then becomes, The Rand index has a value between 0 and 1 with 0 indicating that the two data clusters do not agree on any pair of points and 1 indicating that the data clusters are exactly the same.

The Attributes of Multidimensional Data:

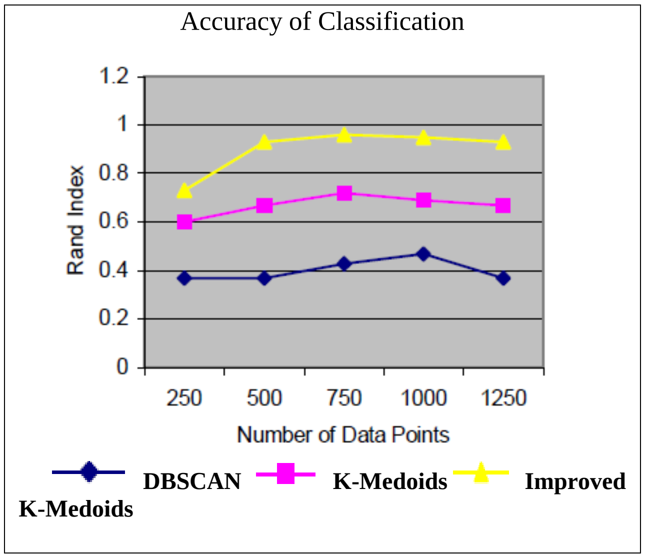
The Number Of Classes: 5

The Number Of Dimensions: 2

The Number Of Points Per Class: 50 , 100, 150, 200,250

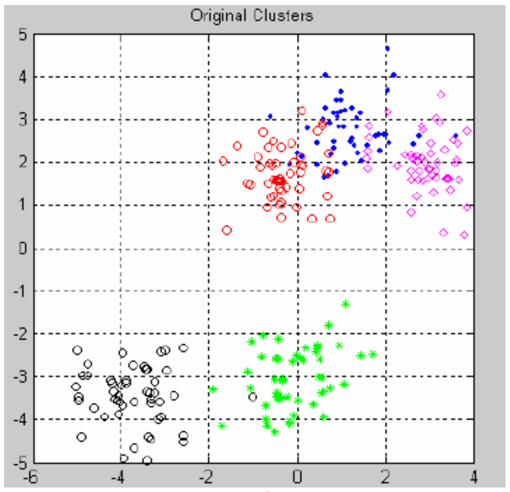
The standard Deviation: 7.000000e-001

S.No.	Total No. of Records in Proteins Data Base	**Classification Accuracy** Measured by Rand Index (Found using Original Class Labels and Calculated Class Labels)		
		DBSCAN	K-Medoids	Density based Improved K-Medoids
1	250	0.03	0.02	0.03
2	500	0.09	0.02	0.09
3	750	0.2	0.02	0.2
4	1000	0.31	0.03	0.3
5	1250	0.42	0.01	0.44

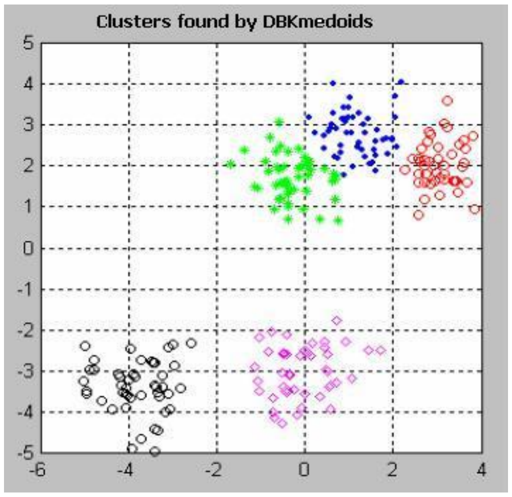


The Clustering and Outlier Detection Results

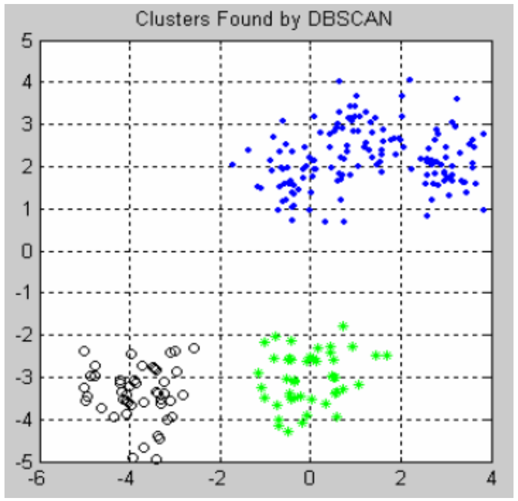
The following show the clusters and outliers marked with DBSCAN and DBkmedoids



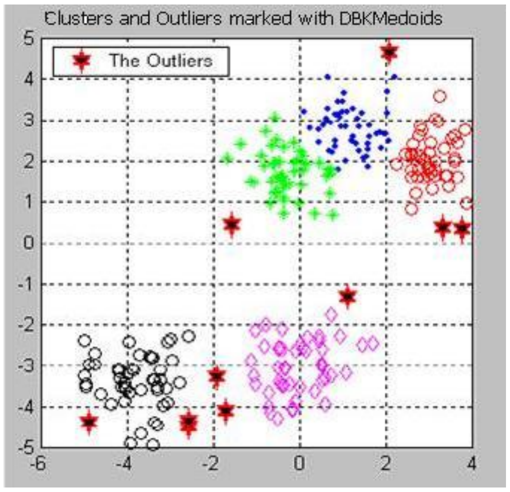
Original Custers



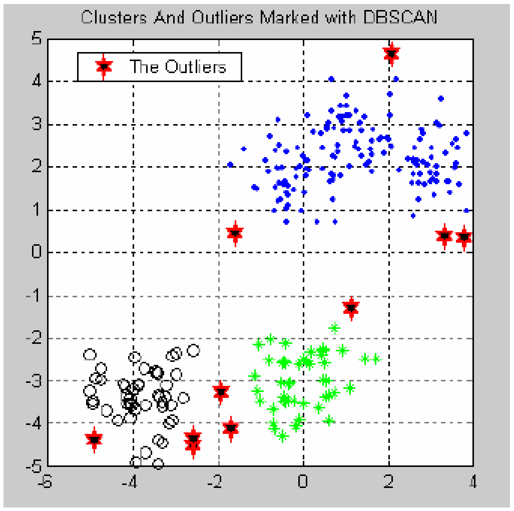
Clusters found by DBKmedoids



Clusters found by DBSCAN



Clusters and outliers marked with DBKMedoids



Clusters and Outliers marked with DBSCAN

From the plotted results, it is noted that DBkmedoids perform better than DBSCAN.

V. Conclusion

This Clustering is an efficient way of reaching information from raw data and Kmeans, Kmedoids are basic methods for it. Although it is easy to implement and understand, Kmeans and Kmedoids have serious drawbacks. The proposed clustering and outlier detection system has been implemented using Weka and tested with the proteins data base created by Gaussian distribution function. The data will form circular or spherical clusters in space. As shown in the tables and graphs, the proposed Density based Kmedoids algorithm performed very well than DBSCAN and k-medoids clustering in term of quality of classification measured by Rand index. One of the major challenges in medical domain is the extraction of comprehensible knowledge from medical diagnosis data.

There is lot of scope for the proposed Density based KMedoidss clustering algorithm in different application areas such as medical image segmentation and medical data mining. Future works may address the issues involved in applying the algorithm in a particular application area.

REFERENCES

[1] Shu-Chuan Chu;Roddick, J.F.;Tsong-Yi Chen;Jeng-Shyang Pan "Efficient search approaches for k-medoids-based algorithms", TENCON '02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering,Feb 2003.

[2] Shu-Chuan Chu,John F.Roddick,J.S. Pan "An Efficient K -Medoids-Based Algorithm Using Previous Medoid Index, Triangular Inequality Elimination Criteria, and Partial Distance Search" 4th International Conference on Data Warehousing and Knowledge Discovery,2002.

[3] Raymond T. Ng and Jiawei Han, CLARANS: A Method for Clustering Objects for Spatial Data Mining, IEEE TRANSACTIONS ON KNOWLEDGE and DATA ENGINEERING, Vol. 14, No. 5, SEPTEMBER 2002.

[4] K. Mumtaz1 and Dr. K. Duraiswamy, A Novel Density based improved k-means Clustering Algorithm – Dbkmeans International Journal on Computer Science and Engineering, Vol. 02, No. 02, 2010, 213-218.

[5] Zhang T, Ramakrishnan R., Livny M., and BIRCH: An efficient data clustering method for very large databases, In: SIGMOD Conference, pp.103~114, 1996.

[6] K. Alsabti, S. Ranka, and V. Singh, ªAn Efficient k-means Clustering Algorithm,º Proc. First Workshop High Performance Data Mining, Mar. 1998.

[7] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases", Proc. 1996 Int. Conf. Knowledge Discovery and Data Mining (KDD'96), Portland, OR, August 1996.

[8] Matheus C.J.; Chan P.K.; and Piatetsky-Shapiro G. 1993. Systems for Knowledge Discovery in Databases, IEEE Transactions on Knowledge and Data Engineering 5(6): 903-913.

[9] Ng R.T. and Han J. 1994. Efficient and Effective Clustering Methods for Spatial Data Mining, Proc. 20th Int. Conf. on Very Large Data Bases, 144-155. Santiago, Chile.

[10] Indukuri, R. K. R. (2011). Dominating Sets and Spanning Tree based Clustering Algorithms for Mobile Ad hoc Networks. IJACSA - International Journal of Advanced Computer Science and Applications, 2(2). Retrieved from http://ijacsa.thesai.org.

[11] Kumar, V. (2011). Knowledge discovery from database Using an integration of clustering and classification. IJACSA - International Journal of Advanced Computer Science and Applications, 2(3), 29-33. Retrieved from http://ijacsa.thesai.org.

[12] Jayech, K. (2011). Clustering and Bayesian network for image of faces classification. IJACSA - International Journal of Advanced Computer Science and Applications, (Special Issue). Retrieved from www.ijacsa.thesai.org.