A Novel Density based improved k-means Clustering Algorithm – Dbkmeans

K. Mumtaz[1] and Dr. K. Duraiswamy [2],
[1] Vivekanandha Institute of Information and Management Studies, Tiruchengode, India
[2] KS Rangasamy College of Technology, Tiruchengode, India

Abstract: 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 amount of spatial data being collected is increasing exponentially. So, it far exceeded human's ability to analyze. Recently, clustering has been recognized as a primary data mining method for knowledge discovery in spatial database. 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, a novel density based k-means clustering algorithm has been proposed to overcome the drawbacks of DBSCAN and kmeans clustering algorithms. The result will be an improved version of k-means clustering algorithm. This algorithm will perform better than DBSCAN while handling clusters of circularly distributed data points and slightly overlapped clusters.

Keywords : Clustering, DBSCAN, k-means, DBkmeans.

I. INTRODUCTION

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-MEANS [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-means clustering algorithms.

II. DBSCAN ALGORITHM

Density-Based Spatial Clustering and Application with Noise (DBSCAN) was a clustering algorithm based on density. It did clustering through growing high density area, and it can find any shape of clustering (Rong *et al*., 2004). 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 $p_1, p_2, ..., p_n, p_1 = q$, $p_n = p$, then p_{i+1} can get "direct density reachable" from p_i, $p_i \in D, 1 \le i \le n$
5. To ε and *MinP*, if there exist a 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 p0 to pn 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

1. DBScan does not require you to know the number of clusters in the data a priori, as opposed to k-means.
2. DBScan can find arbitrarily shaped clusters. It can even find clusters completely surrounded by (but not connected to) a different cluster. Due to the MinPts parameter, the so-called single-link effect (different clusters being connected by a thin line of points) is reduced.
3. DBScan has a notion of noise.
4. DBScan requires just two parameters and is mostly insensitive to the ordering of the points in the database.

Disadvantages

1. DBSCAN can only result in a good clustering as good as its distance measure is in the function getNeighbors(P,epsilon). The most common distance metric used is the euclidean distance measure. Especially for high-dimensional data, this distance metric can be rendered almost useless.
2. DBScan does not respond well to data sets with varying densities (called hierarchical data sets).

III. K MEANS ALGORITHM

The naive k-means algorithm partitions the dataset into ‘k’ subsets such that all records, from now on referred to as points, in a given subset "belong" to the same center. Also the points in a given subset are closer to that center than to any other center.

The algorithm keeps track of the centroids of the subsets, and proceeds in simple iterations. The initial partitioning is randomly generated, that is, we randomly initialize the centroids to some points in the region of the space. In each iteration step, a new set of centroids is generated using the existing set of centroids following two very simple steps. Let us denote the set of centroids after the ith iteration by $C^{(i)}$. The following operations are performed in the steps:

(i) Partition the points based on the centroids C(i), that is, find the centroids to which each of the points in the dataset belongs. The points are partitioned based on the Euclidean distance from the centroids.

(ii) Set a new centroid c(i+1) $\in$ C (i+1) to be the mean of all the points that are closest to c(i) $\in$ C (i) The new location of the centroid in a particular partition is referred to as the new location of the old centroid.

The algorithm is said to have converged when recomputing the partitions does not result in a change in the partitioning. In the terminology that we are using, the algorithm has converged completely when $C^{(i)}$ and $C^{(i-1)}$ are identical. For configurations where no point is equidistant to more than one center, the above convergence condition can always be reached. This convergence property along with its simplicity adds to the attractiveness of the k-means algorithm.

The k-means needs to perform a large number of "nearest-neighbour" queries for the points in the dataset. If the data is‘d’ dimensional and there are ‘N’ points in the dataset, the cost of a single iteration is O(kdN). As one would have to run several iterations, it is generally not feasible to run the naïve k-means algorithm for large number of points.

Sometimes the convergence of the centroids (i.e. $C^{(i)}$ and $C^{(i+1)}$ being identical) takes several iterations. Also in the last several iterations, the centroids move very little. As running the expensive iterations so many more times might not be efficient, we need a measure of convergence of the centroids so that we stop the iterations when the convergence criteria is met. Distortion is the most widely accepted measure.

Clustering error measures the same criterion and is sometimes used instead of distortion. In fact k-means algorithm is designed to optimize distortion. Placing the cluster center at the mean of all the points minimizes the distortion for the points in the cluster. Also when another cluster center is closer to a point than its current cluster center, moving the cluster from its current cluster to the other can reduce the distortion further. The above two steps are precisely the steps done by the k-means cluster. Thus k-means reduces distortion in every step locally. The k-Means algorithm terminates at a solution that is locally optimal for the distortion function. Hence, a natural choice as a convergence criterion is distortion. Among other measures of convergence used by other researchers, we can measure the sum of Euclidean distance of the new centroids from the old centroids. In this thesis we always use clustering error/distortion as the convergence criterion for all variants of k-means algorithm.

Definition 1: *Clustering error* is the sum of the squared Euclidean distances from points to the centers of the partitions to which they belong.

Mathematically, given a clustering ϕ, we denote by $\phi(x)$ the centroid this clustering associates with an arbitrary point *x* (so for k-means, $\phi(x)$ is simply the center closest to *x*). We then define a measure of quality for ϕ:

$$distortion_{\phi} = \frac{1}{N}\sum_{x}\left|x-\phi(x)\right|^2$$

Where |a| is used to denote the norm of a vector ‘a’. The lesser the difference in distortion over successive iterations, the more the centroids have converged. Distortion is therefore used as a measure of goodness of the partitioning.

In spite of its simplicity, k-means often converges to local optima. The quality of the solution obtained depends heavily on the initial set of centroids, which is the only non-deterministic step in the algorithm. Note that although the starting centers can be selected arbitrarily, k-means is fully deterministic, given the starting centers. A bad choice of initial centers can have a great impact on both performance and distortion. Also a good choice of initial centroids would reduce the number of iterations that are required for the solution to converge. Many algorithms have tried to improve the quality of the k-means solution by suggesting different ways of ways of sampling the initial centers, but none has been able to avoid the problem of the solution converging to a local optimum.

Problems with kmeans clustering algorithm

The algorithm is simple and has nice convergence but there are number of problems with this. Some of the weaknesses of k-means 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 before hand. Selection of value of K is itself an issue and sometimes its hard to predict before hand 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.
- Sensitive to initial condition. Different initial condition may produce different result of cluster. The algorithm may be trapped in the local optimum.
- We never know which attribute contributes more to the grouping process since we assume that each attribute has the same weight.
- Weakness of arithmetic mean is not robust to outliers. Very far data from the centroid may pull the centroid away from the real one.
- Experiments have shown that outliers can be a problem and can force algorithm to identify false clusters.
- Experiments have shown that performance of algorithms degrade in higher dimensions and can be off by factor of 5 from optimum [10][11].

IV. PROPOSED ALGORITHM

```
Let D is the Dataset with n 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
    η 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) < η)
      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 $C_m$by taking the mean
      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 $C_k$ 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 kmeans clustering algorithm and repeat it
          until achieving k clusters.
          Finally we will have $C_k$ centers
      }
```

Apply one iteration of k-mean clustering with k and new C_k 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)

V. 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, kmeans and DBkmeans in terms of time required for clustering.

The Attributes of Multidimensional Data :

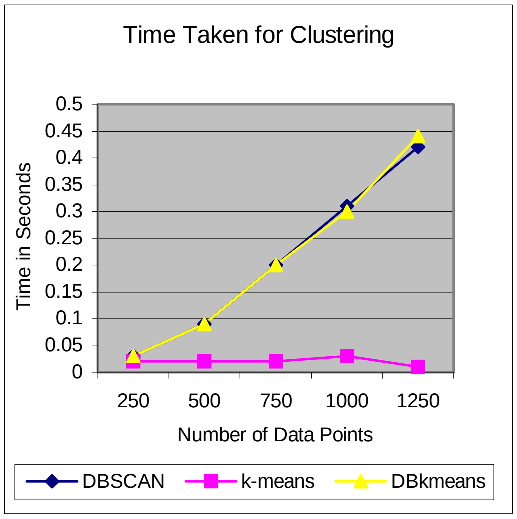
The Number Of Classes: 5

The Number Of Dimensions: 2

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

The standard Deviation: 7.000000e-001

Sl No	Total Number of Records in Synthetic Data	Time Taken for Classification (in seconds)		
		DBSCAN	k-means	DBkmeans
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 :

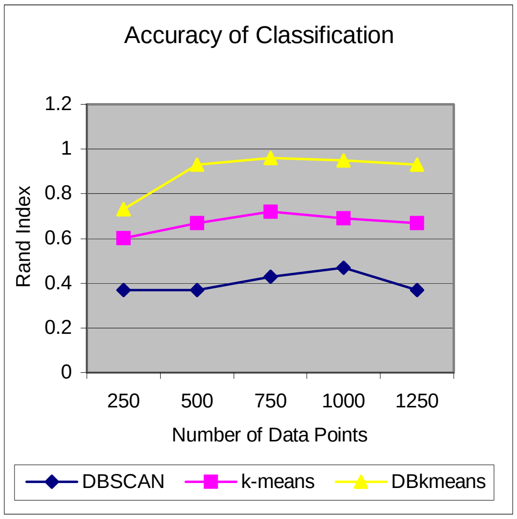
The Number Of Classes: 5

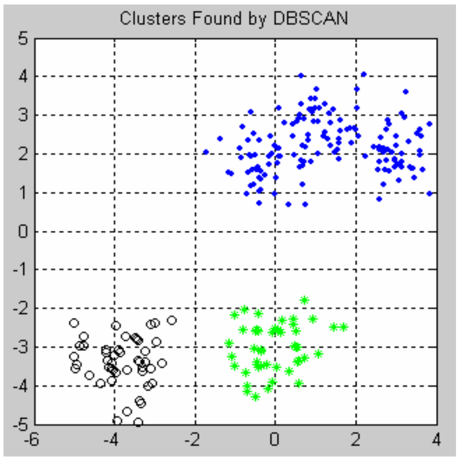
The Number Of Dimensions: 2

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

The standard Deviation: 7.000000e-001

Sl No	Total Number of Records in Synthetic Data	Classification Accuracy Measured by Rand Index (Found using Original Class Labels and Calculated Class Labels)		
		DBSCAN	k-means	DBkmeans
1	250	0.37	0.6	0.73
2	500	0.37	0.67	0.93
3	750	0.43	0.72	0.96
4	1000	0.47	0.69	0.95
5	1250	0.37	0.67	0.93

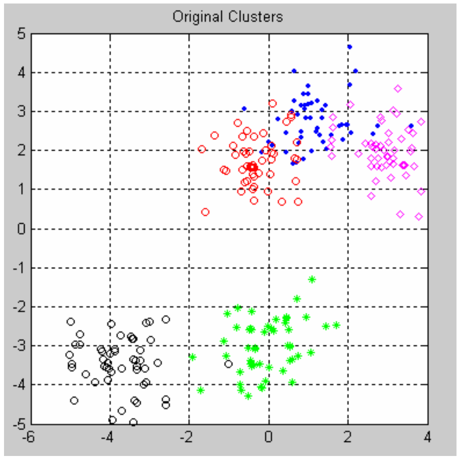




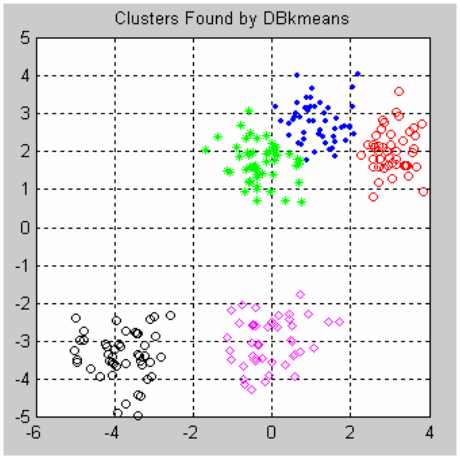
Clusters Marked without outliers

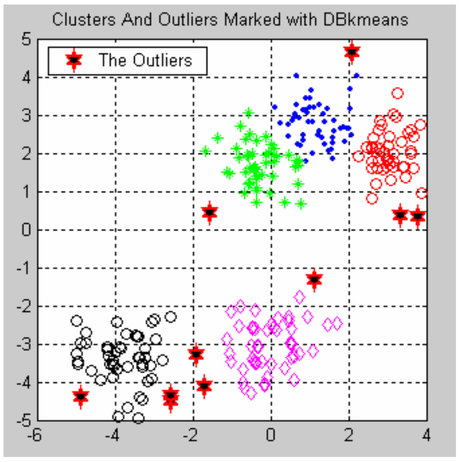
The Clustering and Outlier Detection Results

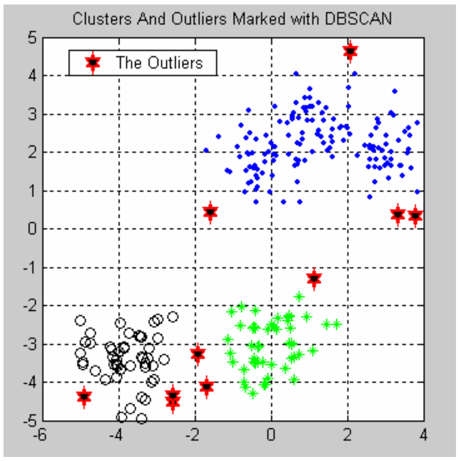
The following show the clusters and outliers marked with DBSCAN and DBkmeans.



The original clusters







Clusters Marked along with outliers

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

VI. CONCLUSION

The proposed clustering and outlier detection system has been implemented using Matlab and tested with the data synthetically created by gaussian distribution function. The data will form circular or spherical clusters in space. As shown in the tables and graphs, the proposed Dbkmeans algorithm performed very well than DBSCAN and k-means 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 Dbkmeans 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.

VII. REFERENCES

[1] Kaufman L. and Rousseeuw P. J., Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley & Sons, 1990.

[2] 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.

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

[4] Guha S, Rastogi R, Shim K., CURE: An efficient clustering algorithm for large databases, In: SIGMOD Conference, pp.73~84, 1998.

[5] Ester M., Kriegel H., Sander J., Xiaowei Xu, A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, KDD'96, Portland, OR, pp.226-231, 1996.

[6] Ankerst M., Markus M. B., Kriegel H., Sander J., OPTICS: Ordering Points To Identify the Clustering Structure, Proc.ACM SIGMOD'99 Int. Conf. On Management of Data, Philadelphia, PA, pp.49-60, 1999.

[7] Wang W., Yang J., Muntz R., STING: A statistical information grid approach to spatial data mining, In: Proc. of the 23rd VLDB Conf. Athens, pp.186~195, 1997.

[8] Rakesh A., Johanners G., Dimitrios G., Prabhakar R., Automatic subspace clustering of high dimensional data for data mining applications, In: Proc. of the ACM SIGMOD, pp.94~105, 1999.

[9] http://en.wikipedia.org/wiki/DBSCAN

[10] http://en.wikipedia.org/wiki/K-means_clustering

[11] http://users.csc.calpoly.edu/~dekhtyar/560-Fall2009/lectures/lec08.466.pdf

[12] http://www.mathworks.com/access/helpdesk/help/