International Journal of Scientific & Engineering Research, Volume 3, Issue 3, August-2012
ISSN 2229-5518

Comparative Analysis of k-means and Enhanced K-means clustering algorithm for data mining

Neha Aggarwal,Kirti Aggarwal, Kanika gupta

ABSTRACT-K-Means Clustering is an immensely popular clustering algorithm for data mining which partitions data into different clusters on the basis of similarity between the data points and aims at maximizing the intra-class similarity and minimizing the inter-class similarity. This Algorithm suffers from the limitation of being time consuming and producing different results with different centroids chosen randomly. The first limitation is solved using the Enhanced K-Means algorithm. This paper shows the comparison of Basic K-Means and Enhanced K-Means algorithm which shows that Enhanced K- Means is more efficient than Basic K-Means Algorithm.

Index Terms—basic K-Means Clustering, centroids, Clustering, computational time complexity, enhanced k-means,iterations, local optimum

.

1. INTRODUCTION

cluster analysis is grouping of objects into a number of more or less homogeneous subgroups on the basis of an often subjectively chosen measure of similarity (i.e., chosen subjectively based on its ability to create "interesting" clusters), such that the similarity between objects within a subgroup is larger than the similarity between objects belonging to different subgroups. Cluster analysis divides data into meaningful or useful groups (clusters). If meaningful clusters are the goal, then the resulting clusters should capture the "natural" structure of the data..

K-Means is a clustering algorithm that deals with numerical attribute values (NAs) primarily, although it can also be applied to categorical datasets with binary values, by viewing the binary values as numerical. The k-Means clustering algorithm for numerical datasets requires the user to specify the number of clusters to be produced and the algorithm builds and refines the specified number of clusters. But due to number of iterations in the loop , the basic k-means is computationally more time consuming and also it produces different results with different dataset..

2. BASIC K-MEANS TECHNIQUE

Suppose that a dataset of n data points x1, x2, …, xn such that each data point is in Rd, the problem of finding the minimum variance clustering of the dataset into k clusters is that of finding k points {mj} (j=1, 2, …, k) in Rd such that

$$\sum_{i,j=1}^{n} [\min\ d2(xi, mj)]$$

is minimized, where d(xi, mj) denotes the Euclidean distance between xi and mj. The points {mj} (j=1, 2, …, k) are known as cluster centroids. The problem in above Equation is to find k cluster centroids, such that the average squared Euclidean distance (mean squared error, MSE) between a data point and its nearest cluster centroid is minimized.

The k-means algorithm provides an easy method to implement approximate solution to this Equation. The reasons for the popularity of k-means are ease and simplicity of implementation, scalability, speed of convergence and adaptability to sparse data.

The k-means algorithm can be thought of as a gradient descent procedure, which begins at starting cluster centroids, and iteratively updates these centroids to decrease the objective function in the Equation listed above. The k-means always converge to a local minimum. The particular local minimum found depends on the starting cluster centroids. The problem of finding the global minimum is NP-complete. The k-means algorithm updates cluster centroids till local minimum is found. Algorithm 1 shows the k-means algorithm.

Algorithm 1

K-Means Clustering Algorithm

1. MSE = largenumber;

2. Select initial cluster centroids {mj}jk=1;

3. Do

4. OldMSE = MSE;

5. MSE1 = 0;

6. For j=1 to k

7. mj=0; nj=0;

8. endfor

9. For i = 1 to n

10. For j = 1 to k

11. Compute squared Euclidean distance d2 (xi, mj);

12. endfor

13. Find the closest centroid mj to xi;

14. mj = mj+xi; nj = nj+1;

15. MSE1 = MSE1 + d2 (xi, mj);

16. endfor

17. For j = 1 to k

18. nj = max (nj, 1); mj = mj/nj;

19. endfor

20. MSE = MSE1;

while (MSE < OldMSE)

Before the k-means algorithm converges, distance and centroid calculations are done while loops are executed a number of times, say l, where the positive integer l is known as the number of k-means iterations. The precise value of l varies depending on the initial starting cluster centroids even on the same dataset. So the computational time complexity of the algorithm is O(nkl), where n is the total number of objects in the dataset, k is the required number of clusters we identified and l is the number of iterations, k≤n,l≤n.

2.1 Scenario based on K-means

Following is an example of original k-mean clustering in which the centroids are taken randomly.

Suppose we have several objects (8 Employees of an Organization) and each object has two attributes or features as shown in table below. Our goal is to group these objects into K=3 groups based on the two features (Experience in no. of yrs and Annual Salary).

TABLE1
Employee Data Set

EMPLOYEE	SALARY(ATTR 1)	EXPERIENCE(ATTR2)
Emp1	0.5	2
Emp2	1	3
Emp3	2	3.5
Emp4	3	4
Emp5	3.5	5
Emp6	4	6
Emp7	4.5	7
Emp8	5	7.5

Each employee represents one point with two attributes (X, Y) that we can represent in coordinate in an attribute space as shown in figure 1

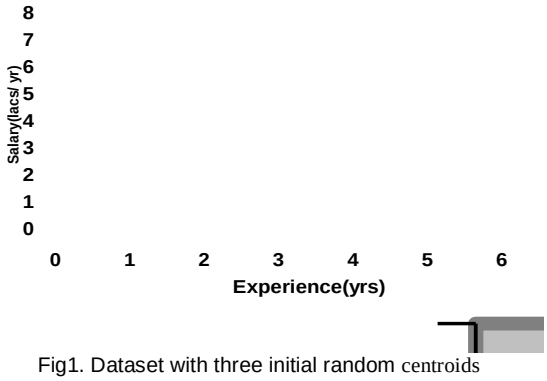

Fig1. Dataset with three initial random centroids

Initial value of centroids : Suppose we use Emp1, Emp2 and Emp3 as the first centroids. Let c1, c2 , c3 denote the coordinate of the centroids, then c1 (0.5, 2) ,c2 (1,3) , c3(2, 3.5) are centroids

1.**Objects-Centroids distance** : we calculate the distance between cluster centroid to each object. Let us use Euclidean distance, then we have distance matrix at iteration 0 is

$$D0=\begin{bmatrix} 0 & 1.118 & 2.12 & 3.2 & 4.24 & 5.315 & 6.4 & 7.1 \\ 1.118 & 0 & 1.118 & 2.23 & 3.2 & 4.24 & 5.315 & 6.02 \\ 2.12 & 1.118 & 0 & 1.118 & 2.12 & 3.2 & 4.35 & 5 \end{bmatrix}$$

2. Each column in the distance matrix symbolizes the object. The first row of the distance matrix corresponds to the distance of each object to the first centroid and the second row is the distance of each object to the second centroid. Similarly for third.

3.**Objects Clustering** : We assign each object based on the minimum distance. Thus, Emp1 is assigned to group 1, Emp2 to group 2, Emp3, Emp4 …..Emp8 to group 3 . The element of Group matrix below is 1 if and only if the object is assigned to that group.

$$G1=\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&1&1&1&1&1\end{bmatrix}$$

4. I**teration-1, determine centroids** : Knowing the members of each group, now we compute the new centroid of each group based on these new memberships. Group 1 only has one member thus the centroid remains in c1 (0.5,2). Group 2 also has one member, thus the centroid remains C2 (1,3). Group 3 has 6 members so the cenroid becomes the average coordinate among the six members:C3= ((2+3+3.5+4+4.5+5)/6 , (3.5+4+5+6+7+7.5)/6)) = (3.66 , 5.5)

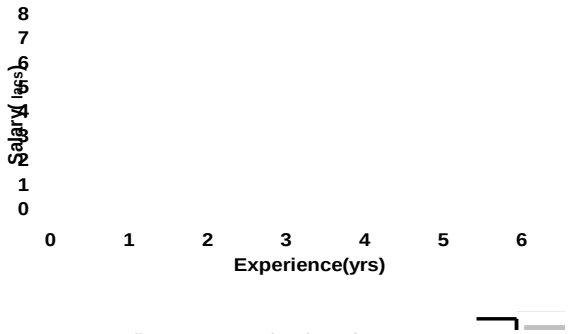

fig2. Dataset after iteration1

5. **Iteration-1, Objects-Centroids distances** : The next step is to compute the distance of all objects to the new centroids. Similar to step 2, we have distance matrix at iteration 1 is

$$D1=\begin{bmatrix}0&1.118&2.12&3.20&4.24&5.315&6.4&7.1\\1.118&0&1.118&2.236&3.2&4.24&5.315&6.02\\4.71&3.65&2.76&1.64&0.525&0.604&1.72&2.41\end{bmatrix}$$

6. **Iteration-1, Objects clustering**: Similar to step 3, we assign each object based on the minimum distance. Based on the new distance matrix, we move Emp3 to Group 2 while all the other objects remain as before. The Group matrix is shown below

$$G1=\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&1&0&0&0&0&0\\0&0&0&1&1&1&1&1\end{bmatrix}$$

7. **Iteration 2, determine centroids**: Now we repeat step 4 to calculate the new centroids coordinate based on the clustering of previous iteration.

C1= (.5 ,2) C2= (1.5 , 3.25) C3= (4 , 5.9)

8. **Iteration-2, Objects-Centroids distances** : Repeat step 2 again, we have new distance matrix at iteration 2 as

$$D2=\begin{bmatrix}0&1.118&2.12&3.2&4.24&5.315&6.4&7.1\\1.6&0.56&0.56&1.67&2.66&3.72&4.8&5.5\\5.24&4.17&3.12&2.15&1.03&0.1&1.21&1.88\end{bmatrix}$$

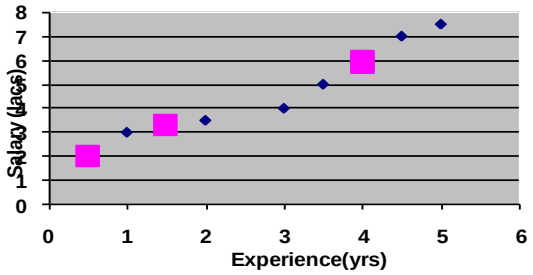

fig3. Dataset after iteration2

9. **Iteration-2, Objects clustering**: Again, we assign each object based on the minimum distance.

$$G2=\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&1&1&0&0&0&0\\0&0&0&0&1&1&1&1\end{bmatrix}$$

10. **Iteration-3, determine centroids**:

C1 = (0.5 ,2) C2 = (2, 3.5) C3 = (4.25 , 6.37)

11**. Iteration-3, Objects-Centroids distances** :

$$D2=\begin{bmatrix}0&1.118&2.12&3.2&4.24&5.315&6.4&7.1\\2.12&1.118&0&1.118&2.12&3.2&4.3&5\\5.76&4.66&3.65&2.68&1.56&0.45&0.67&1.35\end{bmatrix}$$

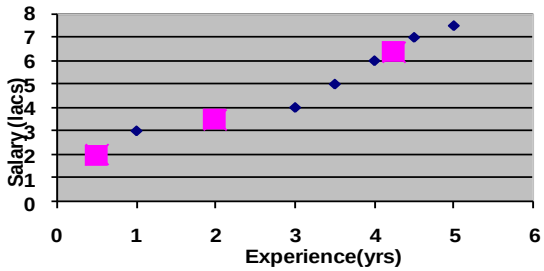

fig3 dataset after iteration3

12. **Iteration-3, Objects clustering**:

$$G3=\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&1&1&0&0&0&0\\0&0&0&0&1&1&1&1\end{bmatrix}$$

We obtain result that G3 = G2. Comparing the grouping of last iteration and this iteration reveals that the objects does

not move group anymore. Thus, the computation of the k-mean clustering has reached its stability and no more iteration is needed. We get the final grouping as the results shown in Table.

TABLE2

CLUSTERING RESULTS

EMPLOYEE	SALARY(ATTRI 1)	EXPERIENCE(ATTRI 2)	GROUP
Emp1	0.5	2	1
Emp2	1	3	2
Emp3	2	3.5	2
Emp4	3	4	2
Emp5	3.5	5	3
Emp6	4	6	3
Emp7	4.5	7	3
Emp8	5	7.5	3

2.2 Limitations of K-Means

- It is computationally very expensive as it involves several distance calculations of each data point from all the centroids in each iteration.
- The final cluster results heavily depends on the selection of initial centroids which causes it to converge at local optimum.

3. AN EFFICIENT ENHANCED K-MEAN CLUSTERING TECHNIQUE

The following algorithm makes k-means more efficient by removing the first limitation i.e. it limits the number of computations to some extent. The idea makes k-means more efficient, especially for dataset containing large number of clusters. Since, in each iteration, the k-means algorithm computes the distances between data point and all centers, this is computationally very expensive especially for huge datasets. Therefore, we do can benefit from previous iteration of k-means algorithm. For each data point, we can keep the distance to the nearest cluster. At the next iteration, we compute the distance to the previous nearest cluster. If the new distance is less than or equal to the previous distance, the point stays in its cluster, and there is no need to compute its distances to the other cluster centers. This saves the time required to compute distances to k−1 cluster centers.

Following fig. explains the idea.

Fig. 4.(a) represents the dataset points and the initial 3 centroids.

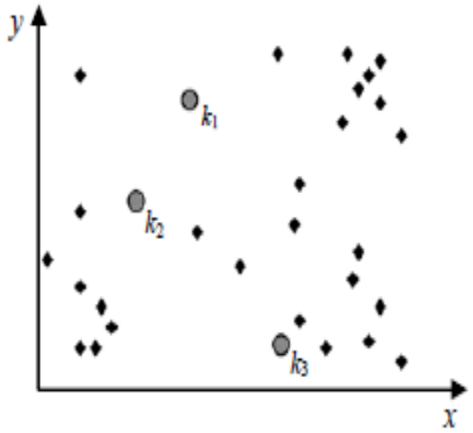

Figure 4(a) : Initial Centroids to a dataset

Fig.4(b) shows points distribution over the initial 3 centroids, and the new centroids for the next iteration

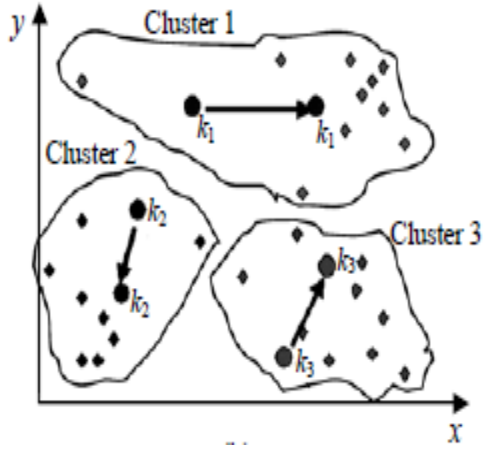

Figure 4 (b) : Recalculating the position of the centroids

Fig. 4(c) shows the final clusters and their centroids.

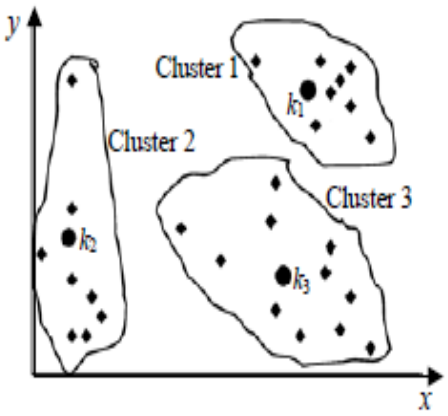

Figure 4 (c) : Final position of the Centroids

When we examine Fig. 4(b), in Clusters 1, 2 we note that, the most points become closer to their new center, only one point in Cluster 1, and 2 points in Cluster 2 will be redistributed (their distances to all centroids must be computed), and the final clusters are presented in Fig. 4(c). Based on this idea, the proposed algorithm saves a lot of time.

In the proposed method, we write two functions. The first function is the basic function of the k-means algorithm, that finds the nearest center for each data point, by computing the distances to the k centers, and for each data point keeps its distance to the nearest center.

The first function is shown in Algorithm 2, which is similar to that in Algorithm 1, with adding a simple data structure to keep the distance between each point and its nearest cluster. This function is called distance().

Algorithm 2:

An Efficient Enhanced k-Mean Clustering Algorithm : First Function

Function distance()

//assign each point to its nearest cluster

1. For i = 1 to n

2. For j = 1 to k

3. Compute squared Euclidean distance d2(xi, mj);

4. endfor

5. Find the closest centroid mj to xi;

6. mj = mj+xi; nj = nj+1;

7. MSE = MSE + d2(xi, mj);

8. Clusterid[i] = number of the closest centroid;

9. Pointdis[i] = Euclidean distance to the closest centroid;

10. endfor

11. For j = 1 to k

12. mj = mj/nj;

13. endfor

In Line 3 the function finds the distance between point number i and all k centroids. Line 5 searches for the closest centroid to point number i, say the closest centroid is number j. Line 6 adds point number i to cluster number j, and increase the count of points in cluster j by one. Lines 8 and 9 are used to enable us to execute the proposed idea; these two lines keep the number of the closest cluster and the distance to the closest cluster. Line 12 does centroids recalculation.

The other function is shown in Algorithm3, which is the same as Algorithm 2 and is called distance_new(). Line 1 finds the distance between the current point i and the new cluster center assigned to it in the previous iteration, if the computed distance is smaller than or equal to the distance to the old center, the point stays in its cluster that was assigned to in previous iteration, and there is no need to compute the distances to the other k−1 centers. Lines 3~5 will be executed if the computed distance is larger than the distance to the old center, this is because the point may change its cluster, so Line 4 computes the distance between the current point and all k centers. Line 6 searches for the

closest center, Line 7 assigns the current point to the closest cluster and increases the count of points in this cluster by one, Line 8 updates mean squared error. Lines 9 and 10 keep the cluster id, for the current point assigned to it, and its distance to it to be used in next call of that function (i.e. next iteration of that function). This information is kept in Line 9 and Line 10 allows this function to reduce the distance calculation required to assign each point to the closest cluster, and this makes the function faster than the function distance in Algorithm 2

Algorithm 3:

An Efficient Enhanced k-Mean Clustering Algorithm : Second Function

```
Function distance_new()
//Assign each point to its nearest cluster
1. For i = 1 to n
        Compute squared Euclidean distance
            d2 (xi, Clusterid[i]);
        If (d2 (xi, Clusterid[i] ) <= Pointdis[i] )
            Point stay in its cluster;
2.      Else
3.          For j = 1 to k
4.              Compute squared Euclidean distance
d2(xi, mj);
5.          endfor
6.      Find the closest centroid mj to xi;
7.      mj = mj+xi; nj = nj+1;
8.      MSE = MSE + d2(xi, mj);
9.      Clustered[i] = number of the closest centroid;
10.     Pointdis[i] = Euclidean distance to the closest
centroid;
11. endfor
12. For j = 1 to k
13.     mj = mj/nj;
14. endfor
```

Now, when the above algorithm for Enhanced K-Means is applied on the employee dataset used in K-Means, then the results produced are shown in the table below

Table 3

EMPLOYEE	**ATTRIBUTE1: X(Experience in yrs)**	**ATTRIBUTE2: Y(SalaryLacs/annual)**	**GROUP**
Emp1	0.5	2	1
Emp2	1	3	1
Emp3	2	3.5	2
Emp4	3	4	2
Emp5	3.5	5	3
Emp6	4	6	3
Emp7	4.5	7	3
Emp8	5	7.5	3

3.1 COMPLEXITY

As discussed before, the k-means algorithm converges to local minimum. Before the k-means converges, the centroids computed number of times, and all points are assigned to their nearest centroids, i.e., complete redistribution of points according to new centroids, this takes O(nkl), where n is the number of points, k is the number of clusters and l is the number of iterations.

In the enhanced k-means algorithm, to obtain initial clusters, this process requires O(nk). Here, some points remain in its cluster, the others move to another cluster. If the point stays in its cluster this require O(1), otherwise require O(k). If we suppose that half points move fr

om their clusters, this requires O(nk/2), since the algorithm converges to local minimum, the number of points moved from their clusters decreases in each iteration. So we expect the total cost is $nk \Sigma_{i=1}^{l} 1/i$.

Even for large number of iteration, $nk \Sigma_{i=1}^{l} 1/nk$ is much less than nkl. So the cost of using enhanced k-means algorithm

Approximately is O(nk), not O(nkl).

This can be shown through a table and a graph

TABLE4

COMPARISON OF TWO ALGORITHMS

Number of Clusters	**k-Mean Time (Sec.)**	**Enhanced k-Mean Time (Sec.)**
2	55	48
3	62	57

4	67	62
5	74	68

Graphically the comparison can be shown as:

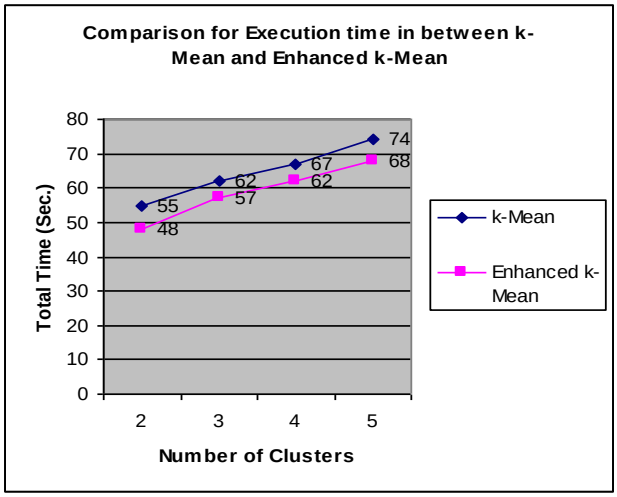

Fig5 comparison of k-means and enhanced k-means

4.CONCLUSION

In this paper, we have discussed the basic k-means clustering technique along with its algorithm and an appropriate example. Then we pointed out the limitation of k-means technique of having more computational complexity, and solution for the same. The Enhanced K-Means Algorithm and Basic K-Means algorithm have been implemented on the same dataset and the Enhanced K-Means is proved to bre efficient than Basic K-Means.

5.REFERENCES

[1]M. R. Anderberg; Cluster Analysis for Applications: Academic Press, New York, 1973.

[2]Anil K. Jain and Richard C. Dubes; Algorithms for Clustering Data: Prentice Hall, Engle wood Cliffs, New Jersey, 1988.

[3]A. El-Hamdouchi and P. Willet; Comparison of Hierarchic Agglomerative Clustering Methods for Document Retrieval: The Computer Journal, Vol. 32, No. 3, 1989.

[4]L. Kaufman, P. J. Rousseeuw; Finding Groups in Data: An Introduction to Cluster Analysis: John Wiley & Sons,1990.Fröhlich, B. and Plate, J. 2000. The cubic mouse: a new device for three-dimensional input. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems

[5]Paul Bradley and Usama Fayyad; Refining Initial Points for K-Means Clustering: Proceedings of the Fifteenth International Conference on Machine Learning ICML98, pp. 91-99, Morgan Kaufmann, San Francisco, 1998.

[6]A. K. Jain, M. N. Murty, and P. J. Flynn; Data clustering: a review: ACM Computing Surveys, pp. 264-323, 1999. Sannella, M. J. 1994 Constraint Satisfaction and Debugging for Interactive User Interfaces. Doctoral Thesis. UMI Order Number: UMI Order No. GAX95-09398., University of Washington.

[7]D. Pelleg and A. Moore; X-means: Extending k-means with efficient estimation of the number of clusters: In Proceedings of the Seventeenth International Conference on Machine Learning, San Francisco, pp. 727-734, 2000.

[8]Maria Halkidi, Yannis Batistakis, Michalis Vazirgiannis; On Clustering Validation Techniques: Journal of Intelligent Information Systems, Vol. 17, pp. 107–145, 2001.

[9]T,Chiu, D.Fang, J.Chen, Y.Wang : A Robust and Scalable Clustering Algorithm for Mixed type attributes in large Database environment : Int.Conf. on Knowledge Discovery and Data Mining, pp. 263-268, 2001

[10]P. Pantel; Clustering by Committee: Ph.d. dissertation, Department of Computing Science, University of Alberta, 2003.

[11]Kirti Aggarwal,Neha Aggarwal,Priyanka Makkar" Analysis of K-Means Clustering Algorithm for Data Mining", national conference on emerging trends in electronics and infotmation technology,2012.

[12]Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and AngelaY. Wu. An efficient k-means clustering algorithm: Analysis and implementation. IEEE Trans. Pattern Anal. Mach. Intell., 24(7):881–892, 2002

[13]Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus F. M. Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg. Top 10 algorithms in data mining. Knowl. Inf. Syst., 14(1):1–37, 2008.

[14]Periklis Andritsos, Data Clustering Techniques, Department of Computer Science, March 11, 2002

[15] Tan P.-N., Steinbach M., Kumar V.: Introduction to Data Mining,490-497, Addison-Wesley, 2006

[16]Kirti Aggarwal,Neha Aggarwal,Sunita Bhardwaj,Nikita Taneja: An Effective Enhanced k-mean Clustering Algorithm for Data Mining, international cofrence on emerging trends in engineering and management,"in press".

1. Neha Aggarwal is currently pursuing masters degree program in computer science engineering in maharishi dayanand university, India, PH-07838035766. E-mail: aggarwal.neha83@gmail.com
2. Kirti Aggarwal is currently working as an assistant professor imn MRCE, Faridabad, India, PH-09899174007. E-mail: kirtiban-sal06@gmail.com.
3. Kanika gupta is currently working as an assistant professor in MRCE, Faridabad, India, PH-09911784844. E mail: kanikagup-ta.1987@gmail.com.