International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 6, June - 2014

Comparative Study of Clustering Algorithms on Diabetes Data

S. Anuradha, P. Jyothirmai, Y. Tirumala, S. Goutham, V. HariPrasad
Department of CSE
VITS College of Engineering
Visakhapatnam, India

***Abstract*- Diabetes is a common disease that causes to all ages of people which needs to be prevented at early stage so that severe problems can be eliminated in future. In the context of medical sciences, it is required to handle diabetes which is treated as a silent disease which may adverse effects to other parts of the body needs to be treated at its early stage, for which we are proposing a clustering approach on patients data which is aimed at finding out the characteristics that determine the presence of diabetes and to also helps us to track the maximum no. of patients suffering from diabetes.**

Keywords- quality of clusters; k-means; k-medoids; k-NN; spanning tree

I. INTRODUCTION

Clustering is a discovery process of data mining that groups a data set such that the intra-cluster similarity is maximized and the inter-cluster similarity is minimized, which signifies that, a cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to objects in other clusters. Especially by clustering, one can identify dense and sparse regions and, therefore, discover overall distribution patterns and interesting correlations among data attributes [1]. The overall performance of clustering depends on identifying the quality clusters. Thus most of the clustering algorithms are concerned with efficiently determining the set of clusters in the given dataset. Here, different algorithms were used which are targeted to generate quality clusters [2].

In this paper we are applying clustering techniques on the given dataset which contains labeled features which are targeted to a class. Once clustering applied, we are able to determine the quality of clusters which gives us efficient results regarding which cluster has got more number of patients who have been suffering from diabetes. Since the proposed system is a comparative approach we are testing the same dataset with four different clustering techniques namely k-Means, Partitioning Around Medoids (PAM), Minimum Spanning Tree (MST) and k-Nearest Neighbor (k-NN). Among all these, the best algorithm that generates quality clusters is determined for further processing. In this study the clustering technique with highest area calculated is considered for measuring the quality of the cluster [3]. This work is done in four modules as follows.

II. MODULES

Loading Dataset, Normalize the Dataset, Applying Algorithms on the normalized file, and Estimation of Quality of Cluster.

Loading Dataset

In this module we load an external high dimensional dataset, which can be derived from experimental results which is related to diabetes dataset [4], which is the input for our project to perform other operations before we get the final results. The Dataset should contain both metadata and also physical data representing some patient's data.

Normalizing the Dataset

In this module, we take raw input from raw dataset, construct the data in terms of classes and labels, where class represent the feature and label represents the attribute, based on which clustering approach should be performed to find out the quality of clusters which differs from one clustering algorithm to another algorithm. Once the normalized file is generated it can be processed in Clustering Algorithm where different approaches are applied as per the algorithm concepts.

Applying of Algorithms on the Normalized File

In this module, the normalized file will be applied to our four clustering techniques one by one and for each approach of clustering algorithm ,we are going to get a set of clusters based on which we are calculating area of the clusters to find to best algorithm.

Graph Representation

This is the final module, where after performing clustering with all four algorithms we are going to represent a graphical analysis where it will show the no of good quality of clusters formed for the best algorithm generated so that we can know the percentage of patient's records to be preferred and the percentage of records not to prefer.

III. ALGORITHM DEVELOPMENT

k-Means

K-means clustering is a method of grouping items into K groups [5] (where K is the number of pre-chosen groups). The grouping is done by minimizing the sum of squared distances (Euclidean distance [6]) between data items and the corresponding centroid. A centroid is the center of mass of a geometric object of uniform density, though here, we will consider mean vector as centroid. The initial partitioning can be done in a variety of ways. One is dynamically chosen, this method is good enough when the amount of data is expected to grow. The initial cluster means can simply be the first few items of data from the set. For instance, if the data is grouped into 3 clusters, then the initial cluster means will be the first 3 items of data. Another is randomly chosen, this method is almost self-explanatory, where the initial cluster means are randomly chosen values within the same range as the highest and lowest of the data values.

Algorithm: k-Means

Let X = {x_1, x_2, x_3,……., x_n} be the set of data points and V = {v_1, v_2,……., v_c} be the set of centers.
Step 1: Randomly select *'c'* cluster centers.

Step 2: Calculate the distance between each data point and cluster centers.

Step 3: Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.

Step 4: Recalculate the new cluster center using equation, as in (1)

$$v_i = (1/c_i) \sum_{j=1}^{c_i} x_i \qquad (1)$$

Where, c_i is the number of data points in i^{th} cluster.

Step 5: Recalculate the distance between each data point and newly obtained cluster centers.
Step 6: If no data point was reassigned then stop, otherwise repeat from step 3

Illustration of K-Means

Input: data (Table 1), No. of clusters is 3
Step 1: Take empty clusters (C1, C2, C3) Take any arbitrary 3 (no. of clusters) records say R1, R3 and R6 as initial means (M1=R1, M2=R3 and M3=R3) of clusters. C1 mean: R1,C2 mean: R3, C3 mean: R6

Step 2: Calculate the distances between each record with all the means (M1, M2 and M3). And put the record in the cluster Ci which is having the minimum distances with its mean Ri. Here, using the formula given in (2) the distance between a point ***X*** (***X1, X2,*** etc.) and a point ***Y*** (***Y1, Y2,*** etc.) is calculated using the following equation.

$$d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \qquad (2)$$

For example, distances between R1 and M1 is calculated as 0 (here second R1 is mean of C1). Distance between R1and M2 is 57.07 and distance between R1 and M3 is 141.552.

(0-0)*(0-0)+(70-75)*(70-75)+(58-110)*(58-110)+(0-23)*(0-23)

0+25+2704+529

Sqrt (0+25+2704+529)=57.07

Then we will get the cluster as

C1 :{ R1, R4, R5, R7}

Mean M1: {0, 70, 58, 0}

C2: {R3} Mean M2: {0, 75, 110, 23}

C3: {R6, R2} Mean M3: {1, 200, 58, 56}

TABLE I. SAMPLE DATA SET

Step 3: Calculate the mean of the C1-m1 :{ 0.5, 75.5, 59.25, 6} Mean of the C2-m2: {0, 75, 110, 23} and. Mean of the C3-m3: {0.5, 200, 84.5, 56}

Record. No (for identify the record only)	**Pregnancy (1=Yes, 0=No)**	**Plasma glucose concentration (mg/dl)**	**Diastolic blood pressure (mm/hg)**	**Triceps skin fold (mm/cm)**
R1	0	70	58	0
R2	0	200	111	56
R3	0	75	110	23
R4	1	70	59	0
R5	1	70	60	1
R6	1	200	58	56
R7	0	92	60	23

If m1 and M1, m2 and M2, m3 and M3 are equal then the clusters are formed… (C1, C2, C3) End… Otherwise assign the m1 as M1 to C1 and m2 as M2 to C2. Repeat the step 2 and 3.

Finally by repeating the above steps we will get the clusters as

C1: {R1, R4, R5}, C2: {R3, R7} and C3: {R6, R2}

k-Medoids

Compared to the well known k-means algorithm, PAM has more number of features [7]. This algorithm would also accept a dissimilarity matrix. It is more robust because it minimizes a sum of dissimilarities instead of a sum of squared Euclidean distances and allows selecting the number of clusters using mean.

The PAM-algorithm is based on the search for k representative objects or medoids among the observations of the dataset. These observations should represent the structure of the data. With finding a set of k medoids, k number of clusters is constructed by assigning each observation to the nearest medoid. The main aim is to find k-representative objects which minimize the sum of the dissimilarities of the observations to their closest representative object. When medoids are not specified by default, then the algorithm first looks for a good initial set of medoids.

Algorithm: k-Medoids

Step 1: Initialize: randomly select *k* of the *n* data points as the medoids

Step 2: Associate each data point to the closest medoid. (closest, here is defined using valid distance metric most commonly Euclidean distance)

Step 3: For each medoid *m*,

Step 3.1: For each non-medoid data point *o*,

Step 3.2: Swap *m* and *o* and compute the total cost of the configuration

Step 4: Select the configuration with the lowest cost.

Step 5: Repeat steps 2 to 4 until there is no change in the medoid.

Illustration of k-Medoids

Input: data (Table 1), No. of clusters is 3

Step 1: Take empty clusters (C1, C2, C3). Take any arbitrary 3 (no. of clusters) records say R1, R3 and R6 as initial means (M1=R1, M2=R3, M3=R3) of clusters. C1 mean: R1; C2 mean: R3; C3 mean: R6

Step 2: Calculate the distances between each record with all the means (M1, M2, M3) and put the record in the cluster Ci which is having the minimum distances with its mean Ri. Calculate the distance between a point ***X*** (***X1, X2,*** etc.) and a point ***Y*** (***Y1, Y2,*** etc.) using the eq. as in (2). For example, distances between R1 and M1 (here second R1 is mean of C1) is calculated as 0. Distance between R1, M2: 57.07 and the distance between R1, M3: 141.552

(0-0)*(0-0)+(70-75)*(70-75)+(58-110)*(58-110)+(0-23)*(0-23)

0+25+2704+529

sqrt(0+25+2704+529)=57.07

Then we will get the cluster as

C1 :{ R1, R4, R5, R7}; Mean M1: {0, 70, 58, 0} C2: {R3} ; Mean M2: {0, 75, 110, 23} and C3: {R6, R2}Mean M3: {1, 200, 58, 56}

Step 3: Now take any other three records R4, R3 and R2 as medoids m1, m2, m3. And create another clusters c1, c2, c3. Now calculate the cost of each cluster C1, C2, C3, c1, c2, c3. Cost will be the sum of the distance between the each record in cluster and its medoid.

C1: 35.753 and c1: 35.753; C2: 0 and c2: 0; C3: 31.890 and c3: 31.890

Step 4: If C1 and c1, C2 and c2, C3 and c3 are equal then clusters are formed... take C1, C2, C3 and stop the algorithm. Otherwise, repeat the steps 3 and 4. Finding the best records (in step 3) which are having low cost:

- Find the centroid
- Take the nearest records (which are having the minimum distance) from centriod as we need.

Minimum Spanning Tree

A minimum spanning tree in an undirected connected weighted graph is a spanning tree of minimum weight. A spanning tree of a weighted graph is a connected sub graph of G such that, T contains every vertex of G and T does not contain any cycle. MST is a spanning tree with minimum total distance. Through this MST representation, we can convert a multi-dimensional clustering problem to a tree partitioning problem. An MST can be constructed using Prim's algorithm in which vertex is added one at a time. It starts with one vertex of a graph as tree and adds smallest edge that grows the tree by one more vertex. Prim's algorithm is very similar to Kruskal's: whereas Kruskal's "grows" a forest of trees, Prim's algorithm grows a single tree until it becomes the minimum spanning tree. Both algorithms use the greedy approach - they add the cheapest edge that will not cause a cycle. But rather than choosing the cheapest edge that will connect any pair of trees together, Prim's algorithm only adds edges that join nodes to the existing tree.

Algorithm: MST

Step 1: Given N data points, construct the MST using Prim's algorithm.

Step 2: Consider total no. of clusters

Step 3: Take any random index based record Ri from the dataset D which is considered as a start node and remove that record from the D.

Step 4: Add the Random record to the nodelist.

Step 5: Calculate distance of the remaining records Ri with the random record and the record Ri which is getting least distance value, will be added to the nodelist.

Step 6: Process the remaining records until the whole dataset get exhausted.

Step 7: Now the all nodes will be formed in a order based on distances.

Step 8: Take the largest distance from the nodes formed by the total no. of clusters-1

Step 9: Then the clusters are formed with the largest distance node.

Step 10: Repeat the process recursively until the whole nodelist gets empty.

Illustration of MST

Input: Data (Table 1), No. of clusters n=3. Process: let D be: R1, R2, R3, R4, R5, R6, R7. Take any random element R7. Now form the tree (as linear tree) as below:

R7 -> minimum distance from R7 with remaining elements {R1, R2, R3 R4, R5, R6} R7->R1=31.890

R7->R2=123.911 R7->R3=52.810

R7->R4=31.859 R7->R5=31.128

R7->R6=112.951. The minimum distance acquired is R7->R5.

R7 -> R5 -> minimum distance from R5 with remaining elements {R1, R2, R3, R4, R6}. The minimum distance acquired is R5->R4

R7 -> R5 -> R4 -> minimum distance from R4 with remaining elements {R1, R2, R3, R6}. The minimum distance acquired is R4->R1

R7 -> R5 -> R4 ->R1-> minimum distance from R1 with remaining elements {R2, R3, R6}

The minimum distance acquired is R1->R3

R7 -> R5 -> R4 ->R1->R3 minimum distance from R3 with remaining elements {R2, R6}. The minimum distance acquired is R3->R2

R7 -> R5 -> R4 ->R1->R3->R2 minimum distance from R3 with remaining elements {R6}

Then let the resultant tree as R7-> R5 -> R4 -> R1 -> R3 -> R2->R6

And distances: R7, R5: 31.128; R5, R4: 1.414; R4, R1: 1.414; R1, R3: 57.078;

R3, R2: 129.286; R2, R6: 53.009

R7---31.128---R5---1.414---R4---1.414---R1---57.078---R3---129.282---R2---53.009---R6

Now form the clusters as follows:

Take (n-1) splits and get those maximum distances (57.078, 129.286) and break those links.

R7---R5---R4---R1 R3 R2-------R6

So clusters as:

C1: {R7, R5, R4, R1}

C2: {R3} and C3: {R2, R6}

k-Nearest Neighbor

k-Nearest Neighbor (k-NN) is a classification technique that uses a version of the same method. It decides in which class to place a new case by examining some number- the 'k' in k-nearest neighbor-of the most similar cases or neighbors. It counts the number of cases for each class, and assigns the new case to the same class to which most of its neighbors belong.

First, apply k-NN algorithm to find a measure of the distance between attributes in the data and then calculate it. While this is easy for numeric data, categorical variables need special handling. For example, what is the distance between blue and green? You must then have a way of summing the distance measures for the attributes.

First, distances between the cases are calculated and then the set of already classified case is selected to use as the basis for classifying new cases. Finally, the neighborhood in which the comparison is made is decided.

k-NN puts a large computational load on the computer because the calculation time increases as the factorial of the total number of points. While it is a rapid process to apply a decision tree or neural net to a new case, k-NN requires that a new calculation be made for each new case. To speed up k-NN, frequently all the data is kept in memory. Memory-based reasoning usually refers to a k-NN classifier kept in memory.

Algorithm: k-NN

Step 1: $K_1 = \{t_1\}$; add K_1 to K; // t1 initialized the first cluster

Step 2: k=1

Step 3: for i=2 to n do //for t_2 to t_n add to existing cluster or place in new one

Find the t_m in some cluster K_m in K such that $d(t_m,t_i)$ is the smallest;

if $d(t_m,t_i) <= t$ then $K_m = K_m \cup \{t_i\}$
//existing cluster; t is no.of initialized clusters

else

k=k+1; $K_k = \{t_i\}$; add K_k to K //new cluster

Illustration of k-NN Algorithm

Input: Data (Table 1), No. of clusters k=2.

Process: Take an arbitrary threshold value t1= 75. Record R1 is placed in a cluster by itself so we have K1={R1}. We then look at R2 if it should be added to K1 or to be placed in a new cluster. Calculating distance between R1, R2 =151.145 As the distance obtained is greater than threshold value, the record R2 is placed in the other cluster K2. Calculate the distance of record R3 with {R1, R2}

R3->R2=57.07

R3->R1= 129.286

Then the record R3 is compared with the threshold values t1 and t2. The minimum distance is considered R3->R2=57.07 and the record R3 is placed in cluster K1. The process continues until the all the record get processed. Finally, the clusters obtained are

K1= {R1, R3, R4, R5, R7}

K2= {R2, R6}

IV. RESULTS AND DISCUSSIONS

The clusters formed in k-Means algorithm are : C1= {R1, R4, R5}, C2= {R3, R7}, C3= {R6, R2}

The clusters formed with k-medoids algorithm are: C1 {R1, R4, R5, R7}, C2 {R3}, C3 {R2, R6}

The clusters obtained with MST algorithm are C1: {R7, R5, R4, R1} C2: {R3}, C3: {R2, R6}.

The clusters generated with k-NN algorithm are K1= {R1, R3, R4, R5, R7} K2= {R2, R6}.

Comparative Analysis

In this comparative analysis we have first calculated the average distance of each cluster by taking each record in the cluster with the centroid and then after finding all the average

distances of each cluster in each algorithm, we are going to determine the best clustering algorithm based on area of each cluster by calculating the area function of non-polygon area based on the no of 'x' and 'y' points which gives individual areas. Based on the highest area we are determining the best clustering algorithm based on dataset and based on no of clusters given as input by the user. Now among all these, the algorithm with highest area calculated is k-medoids, which is chosen as the best algorithm for generating clusters.

V. CONCLUSION

In this paper, comparison of 4 clustering algorithms is done based on a given dataset that contains information about women diabetics, which will be processed internally based on the clusters formed by the input value of clusters. First we load the dataset in the memory and then perform the clustering algorithms through which we are going to derive the best clustering technique based on the average distance of the cluster and then finding an area of a cluster based on non-polygon area which will give the us the areas of all the clusters, where we are going to take the highest area based cluster as the best algorithm and then based on the general preferences of the symptoms we are going to generate a graph that shows the preferred and non-preferred patients in the best cluster and overall clusters, as in fig. 1, from the given dataset targeting to the best clustering algorithm we achieved.

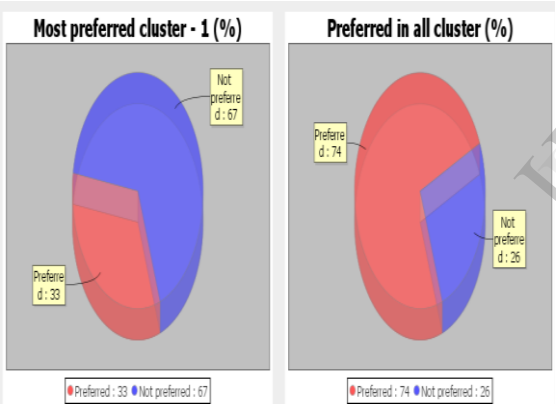

Fig. 1. (a) preferred patients for cluster-1
(b) preferred patients for all clusters

REFERENCES

[1] A. K. Jain, M. N. Murty, P. J. Flynn, 'Data clustering: a review,' ACM Computing Surveys, 31, 1999.

[2] P. Padmaja, V. Srikanth, N. Siddiqui, D. Praveen, B. Ambica, V. B. V. E. Venkata Rao, and V.J.P. Raju Rudraraju, Characteristic evaluation of diabetes data using clustering techniques, journal No.11, November 2008.

[3] Mark S. Aldenderfer, and Roger K. Blashfield, Cluster Analysis, Sage publications, 1984.

[4] www.diabetes.niddk.in.gov website of National institute of Diabetes, Digestive and Kidney Diseases.

[5] Kanungo, T., Mount, D., Piatko, C., Silverman, R., Wu, A.," An efficient k-means clustering algorithm: analysis and implementation," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.24, No.7, pp. 887-892, (2002).

[6] www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Euclidean_and_Euclidean_Squared_Distance_Metrics.html

[7] en.wikipedia.org/wiki/k-medoid