IOP Conf. Series: Journal of Physics: Conf. Series **1489** (2020) 012029 doi:10.1088/1742-6596/1489/1/012029

On Some Data Pre-processing Techniques For K-Means Clustering Algorithm

Dauda Usman[1], Fatima Sani Stores[2]

[1]Department of Mathematics and Computer Science,
Faculty of Natural and Applied Sciences,
Umaru Musa Yar'adua University, Katsina-Nigeria.

[2]Department of Business Administration,
Faculty of Social and Management Sciences,
Al-Qalam University, Katsina-Nigeria.

E-mail: dauusman@gmail.com

Abstract. This paper analyzed the performance of the basic K-Means clustering algorithm with two major data pre-processing techniques and superlative similarity measure with automatic initialization of seed values on the dataset. Further experiment was conducted with simulated data sets to prove the accuracy of the new method. The new method presented in this paper gave a good and promising performance for the different types of data sets. The sum of the squares clustering errors reduced significantly for the new method as compared with basic K-Means method whereas inter-distances between clusters are preserved to be as large as possible for better clusters identification.

Keywords: data pre-processing, data standardization, K-Means clustering, singular value decomposition, similarity measures

1. Introduction

Cluster analysis is an important exploratory tool widely used in many applications. It consist of a process in discovering groups of objects such that the objects in a group will be similar (or related) to one another and dissimilar with (or unrelated) to the objects in other groups. Cluster analysis is a technique for creating groups of objects such that each cluster contains points that are similar and unique [1]. The goal is to assign objects in a data set into meaningful classes such that objects in the same class are more similar to each other than to those in other classes.

One of the most popular clustering methods is the K-Means clustering algorithm [2]. It is a system ordinarily used to directly segment sets of data into *k* groups. K-Means algorithm generates a fast and efficient solution. The basic K-Means algorithm works with the objective to minimize the mean square distance from each data point to its nearest center.

A good clustering technique will generate a high quality clusters and high intra-cluster similarity with low inter-cluster similarity. The quality of clustering results relies heavily on the distance measure used by the technique and its implementation and also by its ability to discover some or all of the hidden patterns [3].

A distance measuring function is used to measure the similarity among objects, in such a way that more similar objects have lower dissimilarity value. Several distance measures can be employed for clustering tasks. Each measure has its own advantage and disadvantage. The selection of different measures is a problem dependent. Hence, choosing an appropriate distance measure for K-Mean clustering algorithm can greatly reduce the burden of the algorithm.

The weakness of the basic K-Means clustering method is that, it is sensitive to the selection of the initial partition and may convergence to a local minimum of the criterion function value. A local minimum is the least value that is located within a set of points which may or may not be a global minimum and it is not the lowest value in the entire set. Its computational complexity is also very high, especially for large dimension dataset. An ad-hoc solution to these problems is by choosing a set of different initial partition and the initial partition that gives the smallest sum of squares error is taken as the solution.

Many attempts were made by different researchers to improve the performance of the basic K-Means algorithm. [4] motivated the use of a deterministic divisive hierarchical method, which is referred to as PCA-Part (principal components analysis partitioning) for initialization and the method brings about faster convergence of the K-Means clustering with fewer number of iterations compared to basic K-Means method. Their experimental results improved the clustering quality as compared to some other clustering methods. But, there is no mention of the dimensionality of the data and the computation time needed in sorting the centers is quite high in this paper.

{5} presented a new strategy to accelerate the K-Means clustering algorithm through the Partial Distance (PD) logic. The proposed strategy avoids many unnecessary distance calculations by applying efficient PD approach. The experimental results reveal the efficiency on the proposed strategy when applied to different data sets. However, the new presented strategy gains more computation time saving than the basic PD technique. This is because the initial distance d is greater than the minimum distance ($d > d_{\min}$) produced from the new technique is very small. This method assumes a random selection of the initial centroid which is what is obtainable with the basic K Means method.

[6] designed an experiment on different types of normalization each of which was tested against the ID3 methodology using the HSV dataset, taken three factors into account number of leaf nodes, accuracy and tree growing time and accomplished comparisons between different learning methods as they were applied to each normalization procedure. They also designed a new matrix to test for the best normalization method based on the factors and their priorities and stated that accurate result depends on the type of analysis and the kind of data to be normalized.

[7] Extended the K-Means clustering algorithm by applying global normalization before performing the clustering on distributed datasets, without necessarily putting the whole data into one location. The effectiveness of the proposed normalization based distributed K-Means clustering algorithm was compared against distributed K-Means clustering algorithm and normalization based centralized *K*-Means clustering algorithm. The quality of clustering was also compared by three normalization procedures for the proposed distributed clustering algorithm. The comparison study implies that the distributed clustering results depend on the type of normalization procedure.

For the two methods presented above, it is observed that there is no suggestion as to the techniques of dimension reduction which is a very serious disadvantage of the basic K-Means algorithm. Thus, to obtain an optimum solution for K-Means clustering algorithm, the data need to be pre-processed before the K-Means cluster analysis [3]. This pre-processing process consists of data standardization to scale the dataset to fall within a specified range of values so that any attribute with larger value will not dominate the attribute with a smaller value. Moreover, for a very high dimensional dataset, a singular value decomposition (SVD) can be used to reduce the dimension [8].

The singular value decomposition is a powerful tool employed for the solution of systems of linear equations and the approximation of a matrix in the field of matrix computation and analysis. It is also a very popular technique used for linear transformation which is employed in the compression of data and its visualization [9]. The main aim of conducting SVD is the reduction of the dimension of high dimensional dataset that consist of very large number of variables that are interrelated which retain to a large degree the variation present in the original dataset. The principal components (PCs) are new uncorrelated ordered variables whose initial variables are synonymous with the original variables [10], [11].

As K-Means is highly dependent on its initial center position [8], an alternative way of center initialization method is also required to make the algorithm more effective and efficient. To overcome the above weaknesses, this paper focused on developing a good clustering technique by two data preprocessing techniques, selection an appropriate distance measure and the initialization of the center points for K-Means clustering algorithm.

2. Materials and Methods

The method is a combination of two data pre-processing techniques and selection of better similarity measure for K-means analysis. A method of z-score is used to scale the data to fall within a specified range of values, so that any variable with larger value will not dominate the variable with smaller value. Furthermore, the singular value decomposition to obtain a reduced data containing possibly uncorrelated variables. Third, the resulting reduced data will be applied to the K-Means clustering algorithm using Euclidean distance in obtaining the distances between the centers and their points attached to each center. All the analysis is carried out using MATLAB M-File 7.6 (R2008a) package. The steps for the technique are as follows:

2.1. Step 1

Consider the z-score method of data standardization. For convenience, let $\boldsymbol{X} = (\boldsymbol{X}_1', \boldsymbol{X}_2', \dots, \boldsymbol{X}_n')$ is the d-dimensional raw data set. This results in an $n \times d$ data matrix given by:

$$\boldsymbol{X} = (\boldsymbol{X}_1', \boldsymbol{X}_2', \dots, \boldsymbol{X}_n') = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1d} \\ x_{21} & x_{22} & \dots & x_{2d} \\ \vdots & \vdots & \dots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nd} \end{bmatrix}$$

The z-score of $\boldsymbol{X}_i$ is given as: $Z(\boldsymbol{X}_i) = \frac{x_i - \bar{x}_i}{S_{x_i}}$ (1)

where

x_i, is the raw scores to be standardized. $\bar{x}_i$ and S_{x_i} are the sample mean and sample standard deviation respectively.

2.2. Step 2

In step 2, we proceed to find the singular value decomposition of the standardized dataset in Equation 1. After that a reduced projected data will be obtained. Consider the standardized data matrix $\boldsymbol{X}$, with zero mean and variance 1. Mathematically, the transformation is defined by a set of p-dimensional

vectors of loadings (loadings here means the weight by which each standardized original variable should be multiplied to get the component score).

Let $\boldsymbol{X}_{n\times d} = \boldsymbol{U}_{n\times d}\boldsymbol{D}_{d\times d}\boldsymbol{V}'_{q\times d}$ (2)

where
$\boldsymbol{U}'\boldsymbol{U} = \boldsymbol{I}, \boldsymbol{V}'\boldsymbol{V} = \boldsymbol{I}$; the column of $\boldsymbol{U}$ are orthogonal eigenvectors of $\boldsymbol{X}\boldsymbol{X}'$, the column of $\boldsymbol{V}$ are orthogonal eigenvectors of $\boldsymbol{X}'\boldsymbol{X}$ and $\boldsymbol{D}$ is the diagonal matrix containing the square roots of eigenvalues from $\boldsymbol{U}$ or $\boldsymbol{V}$ in descending order. And to find $\boldsymbol{U}$ we have to start with $\boldsymbol{X}'\boldsymbol{X}$.

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \text{ and } \quad \boldsymbol{X}' = \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{n1} \\ x_{12} & x_{22} & \cdots & x_{n2} \\ \vdots & \vdots & \cdots & \vdots \\ x_{1d} & x_{2d} & \cdots & x_{nd} \end{bmatrix}$$

$$\boldsymbol{X}'\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{n1} \\ x_{12} & x_{22} & \cdots & x_{n2} \\ \vdots & \vdots & \cdots & \vdots \\ x_{1d} & x_{2d} & \cdots & x_{nd} \end{bmatrix} = \begin{bmatrix} x_{11*} & x_{12*} & \cdots & x_{1n*} \\ x_{21*} & x_{22*} & \cdots & x_{2n*} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n1*} & x_{n2*} & \cdots & x_{nn*} \end{bmatrix}$$

To get the eigenvalues that corresponds to the eigenvectors of $\boldsymbol{X}'\boldsymbol{X}$. The equation: $X\hat{\imath} = \lambda\hat{\imath}$ defines the eigenvector and by applying this to $\boldsymbol{X}'\boldsymbol{X}$ gives:

$$\begin{bmatrix} x_{11*} & x_{12*} & \cdots & x_{1n*} \\ x_{21*} & x_{22*} & \cdots & x_{2n*} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n1*} & x_{n2*} & \cdots & x_{nn*} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \lambda \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

This can be written as a set of equations

$$x_{11*}y_1 + x_{12*}y_2 + \cdots + x_{1n*}y_1 = \lambda y_1 \quad (3)$$

$$x_{21*}y_2 + x_{22*}y_2 + \cdots + x_{2n*}y_2 = \lambda y_2 \quad (4)$$

$$x_{n1*}y_2 + x_{n2*}y_2 + \cdots + x_{nn*}y_n = \lambda y_n \quad (5)$$

where $\lambda_1 > \lambda_2 > \cdots > \lambda_n$

Rearranging the above Equations (3, 4 & 5) and solving for λ by putting the determinant of the coefficient matrix as zero. This will result in our eigenvalues λ also plugging λ back into the original equation will get us our eigenvectors. These eigenvectors are column vectors in a matrix which is ordered by the corresponding eigenvalues size. This implies that, the eigenvector of the largest eigenvalues is found in column one, the next eigenvector in size of the eigenvalues is found in column two and so on until we arrive at the eigenvectors of the smallest eigenvalues which is placed in the last column of our matrix. Then, we convert the matrix into an orthogonal matrix which we do by applying the Gram-Schmidt orthogonalization process to the column vectors to obtain $\boldsymbol{U}$.

Thus $\boldsymbol{U} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix}$

The calculation of $\boldsymbol{V}$ is similar and same procedure with $\boldsymbol{U}$, but $\boldsymbol{V}$ is based on $\boldsymbol{X}\boldsymbol{X}'$.

Thus $\boldsymbol{V}$ is $= \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \cdots & \vdots \\ x_{q1} & x_{q2} & \cdots & x_{qd} \end{bmatrix}$ and $\boldsymbol{V'} = \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{q1} \\ x_{12} & x_{22} & \cdots & x_{q2} \\ \vdots & \vdots & \cdots & \vdots \\ x_{1n} & x_{2n} & \cdots & x_{qd} \end{bmatrix}$

For $\boldsymbol{D}$ taking the square roots of the eigenvalues that are non-zero and using them as the elements in the diagonal by putting the largest in d_{11}, the next largest in d_{22} and so on until the smallest value ends up in d_{dd}, these values indicate the variance of the linearly independent components along each dimension. The non-zero eigenvalues of $\boldsymbol{U}$ and $\boldsymbol{V}$ are always the same, that is why it does not matter which one we take them from. The diagonal entries in $\boldsymbol{D}$ are the singular values of $\boldsymbol{X}$, the columns in $\boldsymbol{U}$ are the left singular vectors and the columns in $\boldsymbol{V}$ are the right singular vectors.

$$\boldsymbol{D} = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1d} \\ d_{21} & d_{22} & \cdots & d_{2d} \\ \vdots & \vdots & \cdots & \vdots \\ d_{d1} & d_{d2} & \cdots & d_{dd} \end{bmatrix} = \begin{bmatrix} \sigma_{\bar{1}} & 0 & \cdots & 0 \\ 0 & \sigma_{\bar{2}} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \sigma_{\bar{d}} \end{bmatrix}$$

where

$\sigma_{\bar{1}} \geq \sigma_{\bar{2}} \geq \cdots \geq \sigma_{\bar{d}}$ are the rank ordered set of the singular values. Now we have all the pieces of the puzzle, then full principal components decomposition of $\boldsymbol{X}$ can be given as:

$$= \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & \vdots & \cdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} \sigma_{\bar{1}} & 0 & \cdots & 0 \\ 0 & \sigma_{\bar{2}} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \sigma_{\bar{d}} \end{bmatrix} \begin{bmatrix} x_{11.} & x_{21.} & \cdots & x_{q1.} \\ x_{12.} & x_{22.} & \cdots & x_{q2.} \\ \vdots & \vdots & \cdots & \vdots \\ x_{1n.} & x_{2n.} & \cdots & x_{qd.} \end{bmatrix}$$

$$\widehat{\boldsymbol{X}} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1q} \\ x_{21} & x_{22} & \cdots & x_{2q} \\ \vdots & \vdots & \cdots & \vdots \\ x_{d1} & x_{d2} & \cdots & x_{dq} \end{bmatrix}$$

2.3. Step 3

Given a set of observations, $\widehat{\boldsymbol{X}} = (\widehat{\boldsymbol{X}}_{\boldsymbol{1}}', \widehat{\boldsymbol{X}}_{\boldsymbol{2}}', \ldots, \widehat{\boldsymbol{X}}_{\boldsymbol{n}}')$ where each observation is a p-dimensional real vector, to partition the n observations into k sets $(k \leq n)$, $G = (g_1, g_2, \ldots, g_k)$ compute

$$d_{euclidean}(\boldsymbol{XY}) = \sqrt{(x_i - y_i)^2} = [(x-y)(x-y)']^{\frac{1}{2}} \quad (6)$$

The algorithm proceeds by alternating between two steps

$$G_i = \{x_p : ||x_p - \mu_i||^2 \leq ||x_p - \mu_j||^2 \; \forall j, 1 \leq j \leq k\} \quad (7)$$

where

each x_p is assign to exactly one G. Then update the process by calculating the new centers in the new clusters. The algorithm converges when this assignment no longer changes. Then calculate the total sum of squares error (SSE) that is:

$$SSE = arg_g min \sum_{i=1}^{k} \sum_{x_j \epsilon C_j} ||x_j - \mu_j||^2 \quad (8)$$

where $\mu_i = \frac{1}{n} \sum_{X_i \epsilon C_j} \boldsymbol{X}_i$ denotes the centroid of a cluster c_j and n denotes the number of instances in c_j.

3. Results and Discussions

In this section, a simulation experiment is conducted to compare the basic and new K-Means methods. In order to show the advantage of the new K-Means method, we use small and large p in all our

simulation experiment. That is $(p, n) = (15, 20), (100, 500), (200, 500), (500, 1500)$ and $(1000, 1500)$, where p refers to the number of variables and n is the sample size. The data was generated from multivariate normal distribution $N_p(0, I_p)$ with covariance matrix $\boldsymbol{\Sigma} = b\Gamma$, $b > 0$, and Γ is a symmetric matrix of size $(p \times p)$ with all diagonal elements equal 1 and all off diagonal elements equal ρ where $\rho = 0$, and $b = 1.2$ as in [12]. The $\rho = 0$ values is a representative of no correlation values. For $b = 1.2$ the covariance matrix for the $\rho = 0$ value is:

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1.2 & 0 & \cdots & 0 \\ 0 & 1.2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots & 1.2 \end{bmatrix}; \ \rho = 0$$

The CPU running time required by each experiment and their error sum of squares for the two methods are presented in Table 1. Here a sample of four cluster formations are shown in Figure 1 to Figure 4, for the pairs $(p, n) = (15, 20)$ and $(100, 500)$ owing to convenience and space considerations.

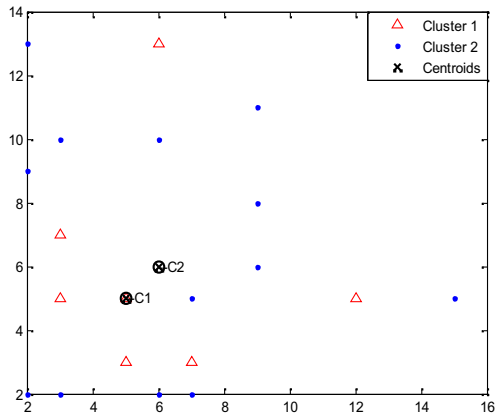

Figure 1 Basic K-Means method with (15, 20)

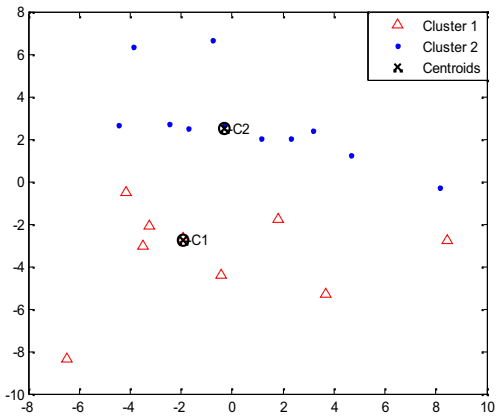

Figure 2 New K-Means method with (15, 20)

Figure 1 shows the cluster formation obtained using the basic K-Means method with simulated data for the pairs (p, n) (15, 20). It can be seen that six points are out of the cluster formation. These six points are on the borders which are marked by the coordinates (2, 2), (0, 9), (0, 13), (3, 0), (6, 0) and (7, 0). They are all found in cluster 2. This is one of the Conventional K-Means drawbacks, which is that the method does not capture all the points within the cluster formation.

Figure 2 shows the new K-Means method. It can be observed that there is no points that are (farthest to the cluster center) outside the cluster formation line. This implies that this method is good for clustering the points. Furthermore, the inter distances between clusters are preserved to be as large as possible.

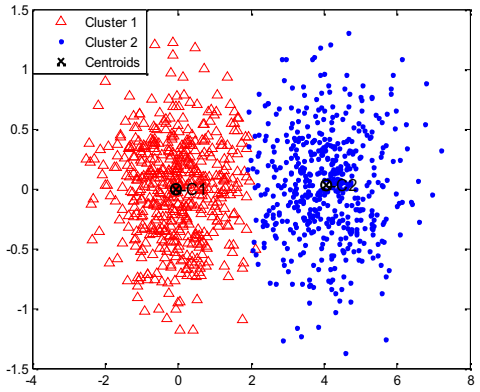

Figure 3 Basic K-Means method with (100, 500)

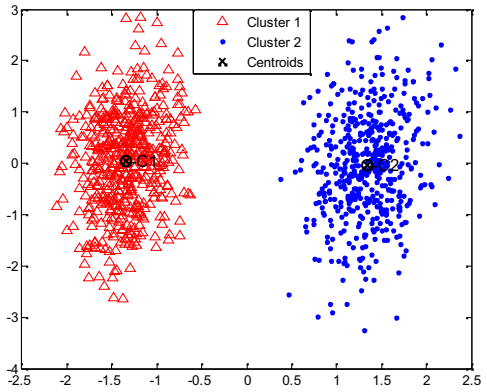

Figure 4 New K-Means method with (100, 500)

Figure 3 shows the cluster formation obtained using the basic K-Means method with simulated data for the pairs (p, n) (15, 20) and (100, 500). The total sum of squares error for the clustering is very high while inter distances between clusters is very small and this makes it difficult to differentiate the center that belongs to either cluster 1 or cluster 2 in the cluster formation.

Figure 4 show the cluster formation obtained using the new K-Means method with simulated data (15, 20) and (100, 500). Here, the total sum of squares error and the running time are much reduced while the inter distances between clusters are preserved to be as large as possible for better identification.

Table 1: The total sum of squares error and CPU time taken

		Basic K-Means method		New K-Means method	
n	p	SSE	CPU time (Sec)	SSE	CPU time (Sec)
20	15	946.24	37	428.05	11
500	100	8731.06	104	5152.31	32
500	200	10386.17	318	59519.53	58
1500	500	14638.08	567	8729.36	174
1500	1000	152137.41	814	9186.25	272

Table 1 shows the total sum of squares error and their respective CPU time for the basic K-Means clustering method and the new K-Means method. This was achieved by the runs of the simulated data sets. The result reveals that the new method performs better in terms of their sum of squares error and the CPU time taken for the execution.

3.1. Discussion

The experimental results obtained with the basic K-Means method yield the expected results bearing in mind that the method is known to be inefficient as can be seen in Figure 1 and 3. The experimental results also implies that the new K-Means method performs very well, providing better total sum of squares error and reduced CPU time taken for the execution as shown in Table 1. However, it is observed that when p is getting larger the CPU running time is also considerably increased. This was observed to be true for both basic and new methods. Furthermore, it was also observed that the clusters are well separated as revealed in Figure 2 and 4. This agrees with the findings of [13], that says compactness and separation are used to measure the significance of clustering results.

4. Conclusion

K-Means is one of the most popular clustering methods. It has become an important multivariate analysis tool in many exploratory studies. However, there exist abundant researches on K-Means in the literature, although none of them is suitable, considering the high complexity of the basic method, especially for high dimensional dataset. Hence, this research centered on two steps: data pre-processing techniques and selection of good similarity measure, we scaled the data to fall within a specified range of values and singular value decomposition to reduce the dimension to a lower level.

The results and findings were validated with simulation experiments. From this experiments it was observed that, the sum of the total clustering errors was reduced as much as possible whereas inter distances between clusters are preserved to be as large as possible for better identification of clusters as in figure 4.

References

[1] Guojun, G., Chaoqun, M. and Jianhong, W. (2007). *Data Clustering Theory, Algorithms and Applications*. American Statistical Association and The Society for Industrial and Applied Mathematics

[2] Tsai, C. Y., and Chiu, C. C. (2008). Developing a Feature Weight Self-Adjustment Mechanism for a *K*-Means Clustering Algorithm. *Computational Statistics and Data Analysis*, 52:4658-4672

[3] Chandrasekhar, T., Thangavel, K. and Elayaraja, E. (2011). Effective Clustering Algorithms for Gene Expression Data. *International Journal of Computer Applications*, 32(4): 25-29

[4] Su, T., and Dy, J. (2004). A Deterministic Method for Initializing *K*-Means Clustering. In *Tools with Artificial Intelligence, 2004. ICTAI 2004. 16th IEEE International Conference*. pp. 784-786

[5] Belal, A. M., Hudaib, M., Huneiti., A. and Hammo, B. (2008). New Efficient Strategy to Accelerate *K*-Means Clustering Algorithm. *American Journal of Applied Sciences*, 5(9): 1247-1250

[6] Alshalabi, L., Shaaban, Z. and Kasasbeh, B. (2006). Data Mining: A Preprocessing Engine. *Journal of Computer Science*. 2(9):735-739

[7] Karthikeyani, V. N. and Thangavel, K. (2009). Impact of Normalization in Distributed K-Means Clustering. *International Journal of Soft Computing* 4(4):168-172

[8] Chris, D. and Xiaofeng, H. (2006). *K*-Means Clustering Via Principal Component Analysis. *Proc. of the 21st International Conference on Machine Learning.* Banff, Canada.

[9] Andrews, H. and Patterson, C. (1976). Singular value decompositions and digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 24:26–53

[10] Ding, C. and He, X. (2004). *K*-Means clustering via Principal Component Analysis. In *Twenty-first international conference on machine learning*. New York: ACM Press.

[11] Jolliffe, I. (2002). *Principal Component Analysis*, 2nd edition. Springer Series in Statistics. New York: Springer-Verlag.

[12] Mason, R. L., Chaou, Y. M. and Young, J. C. (2009). Monitoring Variation in a Multivariate Process when the Dimension is Large Relative to the Sample Size, *Communication in Statistics. Theory and Methods*, 36:(6), 939-951

[13] Berry, M. J. A. and Linoff, G. S. (1997). *Data Mining Techniques for Marketing, Sales and Customer Support*. John Wiley & Sons, Inc., New York,