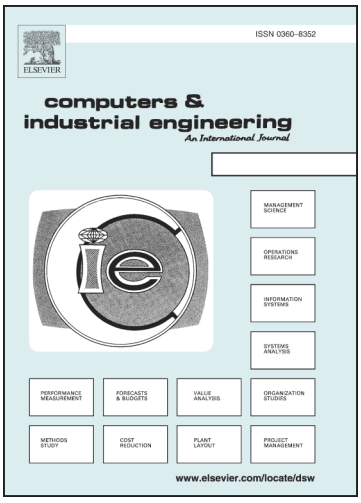


Development of new seed with modified validity measures for *k*- means clustering

S. Manochandar, M. Punniyamoorthy, R.K. Jeyachitra

PII: S0360-8352(20)30024-3
DOI: https://doi.org/10.1016/j.cie.2020.106290
Reference: CAIE 106290

To appear in: *Computers & Industrial Engineering*

Received Date: 22 July 2018
Revised Date: 5 July 2019
Accepted Date: 10 January 2020

Please cite this article as: Manochandar, S., Punniyamoorthy, M., Jeyachitra, R.K., Development of new seed with modified validity measures for *k*- means clustering, *Computers & Industrial Engineering* (2020), doi: https://doi.org/10.1016/j.cie.2020.106290

Title: Development of new seed with modified validity measures for *k*-means clustering

Author 1: S. Manochandar

Ph. D Research Scholar,
Department of Management Studies,
National Institute of Technology,
Tiruchirappalli,
Tamilnadu, INDIA.
Email: mano.chandar90@gmail.com
Phone No: +91-9443694100

Author 2: M. Punniyamoorthy (Corresponding author)

Professor,
Department of Management Studies,
National Institute of Technology,
Tiruchirappalli,
Tamilnadu, INDIA.
Email : punniya@nitt.edu ; mpuniya@yahoo.co.in
Phone No:+91- 9443866660 ; +91-9489066223

Author 3: R. K. Jeyachitra

Associate Professor,
Department of Electronics and Communication Engineering,
National Institute of Technology,
Tiruchirappalli,
Tamilnadu, INDIA.
Email : jeyachitra@nitt.edu
Phone No:+91- 9443145540

Development of new seed with modified validity measures for *k*- means clustering

Abstract

Conventional *k*-means clustering is the widely used partitional method, mainly adapted to machine learning and pattern recognition problems. This algorithm is highly sensitive to initial centroid points, but it cannot guarantee to arrive at a better solution because initial centroids are computed randomly for the given cluster. In this paper, we have developed a new initialization method for *k*-means clustering. We have also made an effort to improve the Dunn Index and introduced a new validity ratio based on the silhouette index. The sum of squared error, Dunn Index, silhouette index, modified Dunn Index, and silhouette validity ratio were used as criteria to evaluate the performance of the initialization algorithm. Various benchmark datasets have been used to assess the effectiveness of the proposed initialization algorithm, and we compared the results with conventional *k*-means and *k*-means++ algorithms. The results have shown that the sum of squared error and number of iterations obtained by our proposed initialization algorithm are minimum. A precision chart is used to test the consistency of the initialization algorithm. The comparative analysis, based on the modified Dunn Index, and silhouette validity ratio have proved that the proposed initialization algorithm has performed better than the other initialization algorithms.

Keywords : Clustering, Dunn Index, *k*-means, *k*-means++, Performance Measure, Silhouette Index.

---

1. Introduction

Clustering is one of the typical problems in machine learning techniques and computational geometry (Arthur, & Vassilvitskii, 2007). It is the process of organizing data objects into a set of separate groups known as clusters (Goyal, & Kumar, 2014). Clustering is an unsupervised classification technique (Jain, Murthy, & Flynn, 1999; Hussain, & Haris, 2019). Cluster analysis is an important tool for various applications, such as data mining, pattern recognition, image processing (Hamad, Thomaseey & Bruniaux, 2017), remote sensing, knowledge discovery, neural networks, artificial intelligence, supply chain application (Yin, Khoo, & Chong, 2013), Natural Language processing (Allen, Sui, & Parker, 2017), market segmentation (Kuo, Ho, & Hu, 2002) and statistics, especially when there is an unsupervised data or unknown class labels (Jain, 2010; Cheung, 2003; Levine & Domany, 2001).

Clustering algorithms are broadly classified into hierarchical and non-hierarchical or partitioning clustering algorithms (Jain, 2010). In the hierarchical algorithm, the given dataset is divided into smaller datasets by using the dendrogram tree (Jothi, Mohanty, & Ojha, 2019). Meanwhile, the non-hierarchical clustering algorithm constructs a single partition of a dataset into *k* number of clusters, in which more similar objects are in the same cluster (Erisoglu, Calis, & Sakallioglu, 2011). The main difference between hierarchical and non-hierarchical clustering is that '*k*' is unknown in the case of hierarchical clustering but is known in the non-hierarchical clustering.

The most popular and fastest clustering algorithm is the *k*-means clustering algorithm (Hartigan, & Wong, 1979, Boobord, Othman, & Bakar, 2015; Jain, Murthy, & Flynn, 1999; Capó, Pérez, & Lozano, 2018; Paea, & Baird, 2018). It requires prior knowledge of the number of clusters (*k*) (Khan, & Ahmad,

2004). Generally, for the given data $X=\{x_1, x_2, \ldots x_n\} \subset R^f$, where '*n'* is the total number of observations or data objects with '*f*' dimensions or variables. The popular *k*-means divides input data *(X)* into '*k*' clusters $C=\{C_1, \ldots C_k\}$, $\bigcup_{p=1}^{k} C_p = X$, $C_p \cap C_q = \Phi$, $1 \leq p \neq q \leq k$. Then it begins to form the clusters of the dataset until a sum of squared error (*SSE*) criterion function is minimized. The squared "error" is generally calculated through the Euclidean distance between cluster data points with the corresponding cluster centroids. It is very sensitive to the specified initial cluster centers (Erisoglu, Calis, & Sakallioglu, 2011). Furthermore, it results in different types of clusters depending on the random choice of initial centroids (Celebi, Kingravi, & Vela, 2013; Sathiya, & Kavitha, 2014). Moreover, the random determination of the initial centroid in the *k*-means clustering algorithm results in a poor cluster structure, and it increases the number of iterations (Capó, Pérez, & Lozano, 2018). The problem of finding better results with the *k*-means clustering algorithm can only be solved by an exhaustive choice of starting points by using several replicates with random starting points (Seber, 1984). This is one of the shortcomings of the conventional *k*-means clustering. To overcome this shortcoming, in this paper we have proposed new initialization processes to the *k*-means clustering algorithm to obtain a minimum *SSE* with lesser number of iterations and thus to get better results. A statistical test (t-test) is applied to validate the significance of the proposed initialization algorithm. The Dunn Index is the ratio between minimum distances between the data objects of different clusters (d_{min}) and the largest within-cluster distance (d_{max}). Here, 'd_{min}' is the single minimum value chosen from the $\frac{k(k-1)}{2}$ combinations of the '*k*' clusters. Similarly, 'd_{max}' is the single maximum value chosen from the '*k*' clusters. The 'd_{min}' and 'd_{max}' representative value is used to validate the performance of the clustering results. Whereas 'd_{min}' and 'd_{max}' values are estimated from the minimum of two representative clusters and the maximum of three representative clusters. So the calculating values are not the representation of all the clusters. Therefore, we have proposed a modified Dunn Index to address this issue. However, while we analyze the silhouette index, we can measure the element-wise validity or determine whether the particular data objects belong to that cluster. In silhouette index, two parameters, i.e., *a(i)*, and *b(i)* are required to determine whether the data object belongs to that particular cluster. Based on this logic, a new validity ratio is developed.

The main contributions of this paper are listed below:

i. The new initialization procedure for computing initial cluster centroids for the *k*-means clustering to avoid randomization and to improve the algorithm's performance.
ii. A modified Dunn Index (*MDI*) is derived by including all the '*k*' representatives at the same time to improve the validation of clusters.
iii. A new validity ratio, termed as Silhouette Validity Ratio (*SVR*), is developed in order to measure the clustering strength.
iv. A precision chart is constructed based on the maximum *SSE* and minimum *SSE* to test the consistency of the initialization algorithm.
v. A t-test is adopted to test the statistical significance of the proposed initialization algorithm with conventional *k*-means and *k*-means++ algorithms.

This paper is organized as follows. Section 2 presents the background of the study, describing the conventional *k*-means and *k*-means++ algorithms. The proposed initialization algorithm is explained in

Section 3. Section 4 describes the proposed performance measure and validity measure. Section 5 discusses the experimental results of the proposed work. Finally, Section 6 presents our conclusions.

2. Related work

Over the years, many works have been carried out to improve the performance of the conventional *k*-means clustering algorithm. Our paper has mainly focused on the initial seed selection for *k*-means clustering and performance or validity measure used to evaluate the clustering results. The literature related to this work was collected and discussed in this section. The literature is grouped on the basis of the direction of the study i.e., new seed selection and comparative study on the different initialization algorithm for *k*-means clustering.

2.1. Literature related to new seed selection

Bradley, and Fayyad (1998) have proposed a set of refined initial starting points for the *k* means clustering algorithm. The dataset was divided into small random subsamples. The minimum error value was computed for these subsamples by executing *k* means clustering, and then it was used as an initial cluster centers. Khan, and Ahmad (2004) have introduced a cluster center initialization algorithm (CCIA) for the computation of initial centroids for the *k*-means clustering. It was based on the similarity of data patterns. Meanwhile, Arthur, and Vassilvitskii (2007) have discussed a randomized seeding technique. A Φ (log *k*) competitive algorithm with which optimal clustering was obtained. The authors have conducted experiments to measure the speed and accuracy of the *k*-means algorithm. Arai, and Barakbah (2007) introduced a method to get initial centroids for *k*-means clustering. This is a hybrid of *k*-means and hierarchical algorithm. Initially, the *k*-means clustering is applied with random initialization after the solution converges the hierarchical clustering is performed to get a better solution. This is mainly used for a larger number of clusters or a higher number of attributes.

The cohesion degree of the neighborhood of a data object and the coupling degree between neighborhoods of data objects were defined based on the rough set model specifically based on the neighborhood by Cao, Liang, and Jiang (2009). They proposed a new initialization method and measured the corresponding time complexity. The authors have studied the influence of the three norms on clustering and compared the *k*-means clustering results obtained with the three different initialization methods. Yedla, Pathakota, and Srinivasa (2010) have suggested the initialization method by normalizing the data objects if the variables have both the positive and negative values. Otherwise, compute the distance by considering the origin as zero; based on the closest distance partition the data objects into '*k*' clusters and then the initial centroids are computed for the *k*-means clustering algorithm. Erisoglu, Calis, and Sakallioglu (2011) have proposed an algorithm to compute initial cluster centers for the *k*-means algorithm. Their algorithm chose two main axes from *'f'* variables, and then the initial centroids were calculated based on these axes. The two axes are chosen based on the less correlation between the variables. At the same time the coefficient of deviation is high.

Li (2011) has defined the nearest neighbor pair and put forward four assumptions about it. These assumptions were based on the center initialization method for the *k*-means algorithm over datasets with

two clusters. Mahmud, Rahman, and Akthar (2012) have proposed a modified *k* means algorithm that has utilized the appropriate centroid. The authors have claimed that this algorithm reduces the number of iterations. Tzortizis, and Likas (2014) have proposed a weighted version of the *k*-means algorithm. The weights were assigned to clusters relative to their variance, and the initialization problem was optimized. Additionally, Goyal, and Kumar (2014) have proposed an algorithm for determining the initial centroids of *k*-means clustering for uniform and non-uniform datasets. Kumar, and Sahoo (2014) have introduced a new initialization process for *k*-means clustering on the basis of the binary search techniques. The minimum and maximum value of each attributes is chosen after that the binary search property is applied for computing the initial cluster centers. A density-based *k* means cluster centroid for the initialization algorithm was proposed and discussed by Dalhatu, and Sim (2016). Yang et al., (2017) proposed an initialization algorithm for *k*-means clustering based on the hybrid distance, which is a combination of Euclidean and density-based distance.

2.2. Literature related to comparison of different initialization algorithm

Pena, Lozano, and Larranage (1999) have empirically compared four initialization methods for the *k*-means algorithm: Random (Lloyd, 1982), Forgy (1965), MacQueen (1967), and Kaufman, and Rousseeuw (1990). The authors have conducted experiments by using each of the four initialization methods to determine the probability distribution of the square error values of the final cluster structure produced by the *k*-means algorithm independently on any of the initial cluster structure. The experimental results have illustrated that both the Random and the Kaufman initialization methods have outperformed the other methods. He et al., (2004) compared three categories of initialization methods for k-means clustering i.e., random sampling methods, distance optimization methods, and density estimation methods. Random Sampling Methods: R-SEL (Forgy,1965) and R-MEAN (He, Tan, & Tan, 2002; Thiesson et al., 1997); Distance Optimization Methods: SCS (Tou, & Gonzalez, 1974) and KKZ (Katsavounidis, Kuo, & Zhang, 1994); Density Estimation Methods: KR (Kaufman, & Rousseeuw,1990). The results show that the distance based methods i.e., SCS and KKZ provides a better solution in terms of cluster separation than the other methods.

Celebi, Kingravi, and Vela (2013) have presented an overview of the algorithm for the placement of initial cluster centers. They compared the eight linear time complexity initialization methods: Forgy (1965), MacQueen (1967), Maximin (Gonzalez, 1985; Katsavounidis, Kuo, & Zhang, 1994), Bradley and Fayyad (1998), *k*-means++ (Arthur, & Vassilvitskii, 2007), greedy k-means++, Var-Part (Su, & Dy, 2007), and Principal Component Analysis-Part (PCA-part) (Su, & Dy, 2007). The experimental analysis was conducted using nonparametric statistical tests. The analysis revealed that popular initialization algorithms such as Forgy(1965), Macqueen (1967) and Maximin (Gonzalez, 1985; Katsavounidis, Kuo, & Zhang, 1994) provides unsatisfactory results, comparatively these methods are better alternatives based on the computational complexity. Similarly, Bradley and Fayyad (1998), greedy *k*-means++, Var-part (Su, & Dy, 2007) and PCA-part (Su, & Dy, 2007) provides good initial centroids for the *k*-means clustering algorithm.

The above literature review on the initialization processes of selecting initial cluster centers indicated to us that the random selection of initial centroids leads to an increase in the number of iterations and does

not guarantee unique clustering. The research aimed at finding an optimum method for obtaining the initial cluster center points is on-going and branching in different directions. Therefore, we were motivated to work on this topic.

Table 1 summarizes the literature review. The main content, initial centroid computation process, direction and performance criteria used in the literatures are listed in the table. The direction of the literature is mainly focused on new seed selection techniques or comparison of different seed selection methods. Based on the direction the literatures are divided into two groups. The initial centroid computation process is classified into two subdivisions: Classification based on considering the sample point coordinates as an initial cluster center or allocating the sample points into the cluster and then computing the initial cluster centroids. The performance criteria used in the literatures are classification accuracy, Precision, Recall, Rand index and Adjusted Rand index are measured through the confusion matrix by considering the class labels, maximum value provides the best clustering results and minimum misclassification error provides the better results, these measures are also termed as external validity measure, and computational complexity is measured through the elapsed time and CPU time, minimum value shows the better performance. Information Gain is used to measure the amount of information gained by the clustering algorithm, this is calculated based on the deviation of total and weighted entropy i.e., with in variance calculated for each cluster, maximum value provides the better results. Initial and Final *SSE* are computed using the initial centroids points and after convergence achieved. Average *SSE* and Number of iterations are calculated by replicating the algorithm for several times. Finally, Dunn's index, Silhouette index, Davies Bouldin index, Calinski–Harabasz index, Krzanowski–Lai index are internal validity measures and these indices are calculated based on the compactness and separateness of the clusters.

Table 1
Summary of the literature

Direction	References	Contents	Initial centroids computing process	Performance Criteria *[Optimal value]*
New Seed	Bradley, and Fayyad (1998)	Initial centroids are computed on the basis of the technique for estimating the modes of distribution.	Computing the initial centroids from the sample points	Information Gain [*Maximum*] Distortion / *SSE* *[Minimum]*
	Khan, and Ahmad (2004)	Individual attributes can provide some information about the initial cluster centroids. Here, *k*-means clustering with random initialization is executed for each variable. Cluster the data objects on the basis of the pattern, these points are then considered as initial cluster centroids.	Computing the initial centroids from the sample points	Misclassification error [*Minimum*] Cluster center proximity index [*Minimum*]
	Arthur, and Vassilvitskii (2007)	The first centroid is chosen randomly; then the remaining centroids are computed based on the probability of the distance from the centroids to the remaining data points.	Considering the sample points coordinates as initial centroids	Average *SSE* [*Minimum*] Minimum *SSE* [*Minimum*] Average Elapsed Time [*Minimum*]
	Arai, and Barakbah (2007)	Hybrid of *k*-means and hierarchical clustering algorithm. Mainly used for higher attributes data or large number of clusters	Computing the initial centroids from the sample points	Misclassification Error [*Minimum*] Computational Time [*Minimum*]
	Cao, Liang, and Jiang	Initial '*k*' centroids are chosen based on the cohesion degree of the	Considering the sample points coordinates as	Accuracy [*Maximum*]

(2009)	neighborhood of an object, and the coupling degree between neighborhoods of objects is defined based on the rough set model.	initial centroids	Precision [*Maximum*] Recall [*Maximum*]
Yedla, Pathakota, and Srinivasa (2010)	The dataset is normalized by subtracting the data points with corresponding minimum values of the variable. Then the initial centroids are computed based on the distance calculated by zero as an origin point	Computing the initial centroids from the sample points	Accuracy [*Maximum*] Elapsed Time [*Minimum*]
Erisogulu, Calis, and Sakallioglu (2011)	Two variables are chosen based on the maximum value of the coefficient of determination, and then the minimum correlation value is chosen.	Considering the sample points coordinates as initial centroids	Misclassification error [*Minimum*] Rand index *[Maximum]* Wilk's lambda *[Minimum]*
Li (2011)	Initial centroids are computed on the basis of the neighbor pairs.	Considering the sample points coordinates as initial centroids	Accuracy *[Maximum]* CPU time [*Minimum*]
Mahmud, Rahman, and Akhtar (2012)	The weighted average score is computed from the dataset. Based on the score, the initial cluster centroids are computed.	Considering the sample points coordinates as initial centroids	Average Time [*Minimum*]
Tzortizis, and Likas (2014)	A weighed version of the *k*-means clustering is introduced based on the cluster variance.	Computing the initial centroids from the sample points	SSE *[Minimum*] Computation Time [*Minimum*]
Goyal, and Kumar (2014)	Initial centroids are computed based on the zero as an origin point. The effectiveness of the algorithm is tested through real life dataset.	Computing the initial centroids from the sample points	Classification Accuracy [*Maximum*]
Dalhatu, and Sim (2014)	Initial cluster centroids are computed based on the maximum density value	Considering the sample points coordinates as initial centroids	Classification Accuracy [*Maximum*]
Kumar, and Sahoo (2014)	Initial cluster centers are computed based on the property of binary search technique	Computing the initial centroids from the sample points	Accuracy [*Maximum*] SSE [*Minimum*]
Yang et al. (2017)	Hybrid distance-based initialization algorithm is introduced.	Computing the initial centroids from the sample points	Accuracy *[Maximum]* Adjusted Rand index *[Maximum]* Rand index *[Maximum]* Dunn's index *[Maximum]* Silhouette index *[Maximum]* Davies Bouldin index [*Minimum*] Calinski–Harabasz index [*Maximum*] Krzanowski–Lai index *[Maximum]*
Pena, Lozano, and Larranaga (1999)	Comparison of four initialization methods for *k*-means clustering: i. Random (Lloyd, 1982)	Considering the sample points coordinates as initial centroids	Average *SSE* [*Minimum*] Average iteration [*Minimum*]
	ii. Forgy (1965)	Considering the sample points coordinates as initial centroids	

		iii.	Macqueen (1967)	Considering the sample points coordinates as initial centroids	
		iv.	Kaufman, and Rousseeuw (1990)	Considering the sample points coordinates as initial centroids	
Comparison of different initialization methods	He et al., (2004)		Comparison of five initialization methods for *k*-means clustering:		Number of Iterations [*Minimum*] Cluster Compactness [*Minimum*] Cluster Separation [*Maximum*]
		i.	R-SEL (Forgy, 1965)	Considering the sample points coordinates as initial centroids	
		ii.	R-MEAN (He, Tan, & Tan, 2002; Thiesson et al., 1997)	Computing the initial centroids from the sample points	
		iii.	SCS (Tou, & Gonzalez, 1974)	Considering the sample points coordinates as initial centroids	
		iv.	KKZ (Katsavounidis, Kuo, & Zhang,1994)	Considering the sample points coordinates as initial centroids	
		v.	KR (Kaufman, & Rousseeuw, 1990)	Considering the sample points coordinates as initial centroids	
	Celebi, Kingravi, and Vela (2013)		Eight initialization algorithms are used in this comparative study:		Initial *SSE* [*Minimum*] Final *SSE* [*Minimum*] Normalized Rand *[Maximum]* Number of iterations [*Minimum*] CPU Time [*Minimum*]
		i.	Forgy (1965)	Considering the sample points coordinates as initial centroids	
		ii.	MacQueen (1967)	Considering the sample points coordinates as initial centroids	
		iii.	Maximin (Gonzalez, 1985; Katsavounidis, Kuo, & Zhang, 1994)	Considering the sample points coordinates as initial centroids	
		iv.	Bradley and Fayyad (1998)	Computing the initial centroids from the sample points	
		v.	Greedy *k*-means	Considering the sample points coordinates as initial centroids	
		vi.	Var-part (Su, & Dy, 2007)	Computing the initial centroids from the sample points	
		vii.	PCA-Part (Su, & Dy, 2007)	Computing the initial centroids from the sample points	

(Note: [ ] indicates the optimal value for performance criteria)

For an easy understanding, denotation and abbreviations adopted in this paper are listed in Table 2.

Table 2
Notations and Abbreviations used in this paper

Notation & abbreviations	Explanation
$X=\{x_1,x_2 \ldots x_n\}$	Data matrix with *'n'* number of observations /data objects
f	Number of variables or features
n	Number of observations
k	Number of clusters
μ_p	Centroid of the p^{th} cluster, $p=\{1,2,\ldots..k\}$
Y	New matrix
λ_{max}	Maximum eigenvalue for 'Y'
V_i^2	Squared value of the eigenvector related to λ_{max}
V_{min}	Minimum of squared value of the eigenvector (V_i^2)
V_{max}	Maximum of squared value of the eigenvector (V_i^2)
$G(i)$	Index used for initial cluster formation for i^{th} data object, $i=\{1,2\ldots.., n\}$
$Max(G(i))$	Maximum value of $G(i)$
$Min(G(i))$	Minimum value of $G(i)$
$SC(i)$	Score value for cluster of each data object (integral_part of $G(i)$)
C_p	p^{th} clusters, $p=\{1,2,\ldots.k\}$
SSE	Sum of squared error for all 'k' clusters
R	Number of replications

UCL	Upper control limit
LCL	Lower control limit
dev	Deviation of mean range values between two different initialization algorithm
IA	Initialization Algorithm
d_{min}	The minimum distance between two objects from different clusters
d_{max}	Maximum distance of two objects from the same cluster
MDI	Modified Dunn Index
D_p	Within Distance of the '*p*' clusters, $p=\{1,2,....k\}$
WD(p,q)	Average with in distance of the *p* and *q* clusters
I_l	Set of independent combinations of clusters
a(i)	Average dissimilarity of i^{th} data point to all other points in the same cluster C_p, $p=\{1,2....k\}$
b(i)	Minimum average dissimilarity of i^{th} data point to all data points in other clusters C_q, $q=\{1,2....k\}, p\neq q$
$MSIL_p$	Cluster mean silhouette C_p
SIL(i)	Silhouette measure for i^{th} data object
GS	Global Silhouette Index
n_p	Number of data objects in the p^{th} cluster
n_q	Number of data objects in the q^{th} cluster
DBCS(i)	Deviation of compactness of i^{th} data object within the cluster (*a(i)*) and Separateness of i^{th} data object between the cluster(*b(i)*)
SVR	Silhouette validity ratio
$N_{positive}$	Frequency of *DBCS(i)* values greater than zero values
$N_{negative}$	Frequency of *DBCS(i)* values lesser or equal to zero values

2.3 Conventional k-means algorithm

In the *k*-means algorithm, homogeneous groups are identified by minimizing the clustering error, which is defined as the sum of the squared Euclidean distances between each data point and corresponding cluster center. The number of clusters is denoted as '*k'* and is a user-defined parameter. Then, '*k*' initial cluster centroids are arbitrarily generated within the given data points, and each point is assigned to the cluster with the closest centroid. The mean value of the data points within each cluster is taken into account for updating the cluster centroids. Due to the formation of new centroids, there is a possibility that the data points move from the existing cluster to the other clusters. The data points are assigned to suitable clusters based on the new centroids. Updating of centroids is repeated until no data points change clusters or the centroids remain as the same. The *k*-means clustering algorithm is an iterative algorithm that renews the results in every iteration. Before starting the algorithm, one has to choose a number of clusters *k* to look for. The conventional *k*-means clustering algorithm consists of the following steps (Han, Pei, & Kamber 2011).

Algorithm 1: *k*-means clustering

Input: *X*: a data set containing '*n*' data objects
k: the number of clusters,
Output: Set of Centroids (μ_p)
1: Arbitrarily choose *k* objects from *X* as the initial cluster centers;
2: repeat
3: (re)assign each data object to the cluster to which the data object is the most similar, based on the mean value of the data objects in the cluster,
4: Update the cluster centers(μ_p),
// where μ_p is the cluster centers, $p=\{1,2,...,k\}$ i.e., calculate the mean value of the data objects for each cluster//
5: Until no change
6: return μ_p

The goal of the *k*-means clustering algorithm is to minimize the *SSE* over all *k* clusters. It is given by (Jain, 2010).

$$SSE = \sum_{p=1}^{k} \sum_{x_i \in C_p} \left\| x_i - \mu_p \right\|^2 , \quad (1)$$

Where '*SSE*' is the sum of squared error value, $X = \{x_i\}$, $i=\{1....n\}$ is the data objects and μ_p is the centroid of the cluster, $p=\{1,2,...k\}$.

2.4 The k-means++ clustering algorithm

Arthur, and Vassilvitskii (2007) have proposed an initialization process for the *k*-means clustering algorithm. It was based on choosing a random starting center with a specific probability, and was called the *k*-means++ algorithm. The initial centroids points are obtained by using a specific probability. This algorithm is widely used in many software, like MATLAB, WEKA, and R. The steps involved in the *k*-means++ algorithm are as follows.

Algorithm 2 : *k*-means++ clustering

Input: *X*: a data set containing *'n'* data objects
k: the number of clusters,
Output: Set of Centroids (μ_p)

1: Choose an initial center μ_1 uniformly at random from *X*
2: Choose the next center μ_q, selecting $\mu_q = x' \in X$ with probability $\frac{Mdist(x')^2}{\sum_{i=1}^{n} Mdist(x_i)^2}$

//where Mdist(x) denotes the minimum distance between data object x to the previously computed centroids; x' is the centroid selected from the given data objects X//

3: Repeat Step (2) until we have chosen a total of '*k*' Centers
4: Proceed as with the conventional *k*-means algorithm
5: return μ_p

The conventional *k*-means clustering algorithm suffers from serious limitations. For example, the solution depends heavily on the initial positions of cluster centers. Even though the specific probability is used in *k*-means++ clustering for computing initial centroids, the first centroid point is chosen randomly. Therefore, it can produce a different cluster structure, *SSE* and iteration for each trail. In this paper, we have developed a new initialization process for the *k*-means clustering algorithm. The performance of the proposed algorithm is compared with the conventional *k*-means and *k*-means++ clustering algorithm.

3. Proposed *k*-Centroid Initialization Algorithm (P*k*CIA)

The random initialization problem is avoided by adequately setting initial centroid points for the conventional *k*-means clustering (Bradley & Fayyad, 1998; Pena, Lozano, & Larranaga, 1999; Khan, & Ahmad, 2004; Tzortzis, & Likas, 2014; Yedla, Pathakota, & Srinivasa, 2010). We have proposed a new initialization algorithm (P*k*CIA) for *k*-means clustering. This P*k*CIA is mainly based on the Eigenvector of the new matrix (*Y*) as an index for computing initial cluster centroids.

The steps involved in the process are as follows. First, let us consider the *X* matrix as the dataset.

$$X = \begin{bmatrix} x_{11} & x_{12} & .. & .. & x_{1f} \\ x_{21} & x_{22} & .. & .. & x_{2f} \\ ... & ... & .. & .. & .. \\ ... & ... & .. & .. & .. \\ x_{n1} & x_{n2} & .. & .. & x_{nf} \end{bmatrix}_{n\ X\ f}, \quad (2)$$

In the above Eq. (2) '*n*' indicates number of observations and '*f*' indicates number of variables. Let *k* be the number of clusters to be formed. Consider a new matrix as *Y*. This new matrix is defined as follows:

$$Y = X * X^T, \quad (3)$$

A symmetric matrix is $Y \in R^{n\ x\ n}$, which is computed as follows:

$$Y = \begin{bmatrix} (x_{11} * x_{11}) + (x_{12} * x_{12}) + ... + (x_{1f} * x_{1f}) & ..... & ..... & ..... & (x_{11} * x_{n1}) + (x_{12} * x_{n2}) + ... + (x_{1f} * x_{nf}) \\ (x_{21} * x_{11}) + (x_{22} * x_{12}) + ... + (x_{2f} * x_{1f}) & ..... & . & . & . \\ . & ..... & . & . & . \\ . & ..... & . & . & . \\ (x_{n1} * x_{11}) + (x_{n2} * x_{12}) + ... + (x_{nf} * x_{1f}) & ..... & ..... & ..... & (x_{n1} * x_{n11}) + (x_{n2} * x_{n2}) + ... + (x_{nf} * x_{nf}) \end{bmatrix}, \quad (4)$$

Where $Y=[y_{ij}]^{n\ x\ n}$, and the matrix consists of a scalar quantity equal to the sum of the pairwise product of the observations. The properties of the 'y_{ij}' are as follows: y_{ij}=0, where the *'i'* or *'j'* observations are zero vectors and $y_{ij}=y_{ji}$, are symmetrical. *Y* should be positive definite. Thus, the Eigenvalues λ_i and corresponding '*i*' Eigenvectors (*V*) for the matrix *Y* have to be determined. Where 'λ' is the diagonal matrix which consists of *n* Eigenvalues *and V* is the *n x n* matrix, which is computed for each eigenvalue. The maximum Eigenvalue λ_{max} is chosen from the Eigenvalues λ_i, and the corresponding vector should be noted. The corresponding eigenvector for the λ_{max} that is the first principal component provides us with information about the patterns in the data. The integral part (i.e., the integer value before the decimal points) of the parameter *G(i)* is used to define the appropriate clusters for the given dataset. Where *G(i)* is calculated as follows

$$G(i) = 1 + \left(\left(\frac{V_i^2 - min(V_i^2)}{max(V_i^2) - min(V_i^2)} \right) * k \right), \quad (5)$$

Where V^2 is the squared Eigenvector related to the $\lambda_{max.}$ and V_i^2 is the squared value of the Eigenvector for the i^{th} data object. The squared Eigenvector value is used in the *G(i)* calculation to avoid the negative value, $min(V_i^2)$ is the minimum value of the squared Eigenvector, $max(V_i^2)$ is the maximum value of the

squared Eigenvector, and $i=\{1,2\ldots\ldots n\}$. $\frac{V_i^2-min(V_i^2)}{max(V_i^2)-min(V_i^2)}$ is the normalization expression, which is used to convert squared Eigenvector value in the range of 0 to 1, '*k*' is the number of clusters to be formed, the values obtained as the result of the product of *'k'* and normalized expression fall in the range of 1 to '*k*' except when $V_i^2 = min(V_i^2)$, the *G(i)* become zero, to avoid this '1' is added with the expression.

$$SC(i) = intergral\_part(G(i)), \tag{6}$$

Where *SC(i)* is the score value of the cluster for the sample point (*i*). In our proposed method, *SC(i)* is used to allocate the data objects into their appropriate clusters. While, we calculate *SC(i)*, if *max(SC(i)) > k*, this leads to the formation of more than '*k*' number of clusters to be formed. If *max(SC(i)) < k*, there will be the possibility of getting a lesser number of clusters. The third possible situation is *max(SC(i)) = k* for computing initial cluster centroids. These three conditions are the expected results from the *SC(i)* calculation. For all three conditions, the initial allocation of sample points into the clusters is carried out as follows:

The *G(i)* values are sorted in ascending order correspondingly order the data objects.

$$RG = Max(G(i)) - Min(G(i)), \tag{7}$$

Where '*RG*' is the range of *G(i)* is computed by using Eq. (7), *Max(G(i))* is the maximum value in the *G(i)* vector, and *Min(G(i))* is the minimum value in the *G(i)* vector. After finding range value, the cutoff per clusters is needed for initially allocating the data objects into their corresponding clusters. This is done by using Eq. (8)

$$CPC = \frac{RG}{k} \tag{8}$$

Where '*CPC*' is the cutoff value generated per cluster based on the range of *G(i)*. The *CPC* expression is used to find the '*k*' partitions. However, the number of data points in each partition may vary. This process provides an initial centroid for the *k*-means clustering algorithm.

The algorithm for a new initialization for *k*-means clustering (P*k*CIA) is shown below:

Algorithm 3: P*k*CIA

Input : *X*: a dataset containing *'n'* data objects
k is the number of clusters
Output: Set of Centroids (μ_p)
1: Compute $\boldsymbol{Y}$ using Eq. (2)
// where Y is the n x n symmetric matrix//
2: Find $\boldsymbol{\lambda}$ and $\boldsymbol{V_i}$ for $\boldsymbol{Y}$
// where 'λ' is the Eigenvalue, 'Vi' is the Eigenvector//
3: Choose $\boldsymbol{\lambda_{max}}$ and corresponding $\boldsymbol{V_i}$
// where 'λ max' is the maximum Eigenvalue, Vi is the corresponding Eigenvector (n x 1)//

4: Compute ***G(i)*** using Eq. (5)
//where G(i) is the (n x 1) vector which is termed as reference index generated based on Vi^2 values //
5: Compute ***RG*** using Eq. (7)
//where RG is the range of the G(i)//
6: Sort the data objects based on ***G(i)*** values in ascending order
7: Based on the ***RG,*** partitions the data objects into '*k*' clusters
8: Compute $\boldsymbol{\mu_p}$ (initial *k* centroids),
// where p={1, 2,…*k*}//
9: Proceed as with the conventional *k*-means algorithm
10: return μ_p

This P*k*CIA provides the initial centroids to avoid the randomization in the *k*-means and *k*-means++ clustering algorithm.

4. Performance measure

We have used the average *SSE* ($\overline{SSE}$) (Arthur & Vassilvitskii, 2007) and the average number of iterations ($\overline{iter}$) (Pena, Lozano, and Larranage, 1999) to validate the results obtained from the proposed initialization algorithm for *k*-means clustering.

The $\overline{SSE}$ is calculated from the set of minimum *SSE* values. The minimum *SSE* values computed for the converged solution for each random initialization. Similarly, the average number of iterations is computed based on the converged solution.

Moreover, the precision chart and statistical t-test are applied to evaluate the performance of the proposed initialization algorithm.

4.1 Precision chart

The precision chart is a control chart, which is used to calculate the precision of the initialization algorithm for the *k*-means clustering. Here, the chart contains a variable upper control limit, mean range value, and a variable lower control limit. The range values are calculated for each replication as

$$Range(r) = Max\ SSE(r) - Min\ SSE(r), \tag{9}$$

Maximum *SSE (Max SSE)* is the sum of squared error values computed after the initialization phase is completed. Minimum *SSE* (*Min SSE*) is the sum of squared error values computed after convergence achieved for every initialization. This is called as one replication. Where *r={1,2,….R}*, *R* is the total number of replications.

The mean range ($\overline{Range}$) is the summation of ranges of *SSE* divided by the total number of replications (*R*)

$$\overline{Range} = \frac{\sum_{r=1}^{R} Range(r)}{R} \tag{10}$$

The upper control limit (*UCL*) is calculated by

$$UCL = D_4 * \overline{Range} \tag{11}$$

The lower control limit (*LCL*) is calculated by

$$LCL = D_3 * \overline{Range} \tag{12}$$

Where 'D_3' and 'D_4' are the lower limit and upper limit table values (Rocke, 1992) for the corresponding sample sizes, respectively. Here, the sample size is the number of iterations for each replication.

4.2 t-test

The performance of the preprocessing of datasets in the data mining process is determined by assessing accuracy and efficiency. The random initialization in *k*-means clustering leads to several replications to find out the best solution. The average *SSE* (Arthur, & Vassilvitiski, 2007; Pena, Lozano, & Larranaga, 1999) is one of the performance criteria used to evaluate the performance of the clustering algorithm. Furthermore, a statistical significance test is needed to evaluate the results obtained from the different initialization algorithms. To test the significance, the hypotheses are framed as follows: Null hypothesis (H_0): There is no significant difference between the mean of range values (H_0: μ_{dev}=0). Research Hypothesis (H_1): There is a significant difference between the mean of range values (H_1: μ_{dev}>0).

$$dev(r) = Range_{IA}(r) - Range_{IA'}(r) \tag{13}$$

Where '*dev*' is the deviation of the range of *SSE* values obtained from the different initialization algorithms for each replication. $Range_{IA}$ are the range values for each replication for the reference initialization algorithm and $Range_{IA'}$ are the range values for each replication for other than the reference initialization algorithm. The '*t*' value is calculated as follows:

$$t = \frac{\overline{dev}}{\sqrt{\frac{\sum_{r=1}^{R}(dev(r))^2 - \left(\frac{\left(\sum_{r=1}^{R} dev(r)\right)^2}{R}\right)}{R(R-1)}}}, \tag{14}$$

In the Eq. (14) $\overline{dev}$ is the average deviation. The degrees of freedom are (*R-1*).

4.3. Validity measure

In general, the goodness of a clustering result is measured by intra-cluster compactness and inter-cluster dispersions (Ansari et al., 2011). The commonly used cluster validity indices are the Dunn Index and silhouette index.

The Dunn Index attempts to identify cluster sets that are compactness within the cluster and well-separated between the cluster for any number of '*k*' (Ansari et al., 2011, Saitta, Raphael, & Smith, 2008). Meanwhile, the silhouette Index is used to validate the effectiveness of the clustering based on the pairwise difference between and within cluster distances of any two data objects. By maximizing the silhouette index value, we are able to determine the optimal number of clusters (Ansari et al., 2011).

4.3.1 Dunn Index (DU)

The Dunn Index (*DU*) defines the ratio between minimum distances between the data objects of different clusters and the largest within cluster distance. (Dunn, 1973)

$$DU = \frac{d_{min}}{d_{max}}, \tag{15}$$

Where 'd_{min}' denotes the minimum distance calculated between two data objects from different clusters, and 'd_{max}' denotes the maximum distance of two data objects from the same cluster. The Dunn Index ranges from 0 to ∞ and the objective is to attain maximum *DU* value. (Ansari et al., 2011).

4.3.2 Modified Dunn Index (MDI)

Dunn index is one of the representative ratios to measure the separateness (d_{min}) and compactness (d_{max}). The representative measure namely 'd_{min}' and 'd_{max}' may be the outcomes of any one of the combinations for the clusters '*p*' and '*q*' as shown in Table.3. Let us consider the set of clusters, $C=\{C_p, C_q, C_m, C_z\}$, where $m, z \neq p, q$.

Table 3:
Combination of outcomes of Dunn index for a pair of clusters C_p and C_q

'd_{min}' (Separateness)	'd_{max}' (Compactness)	Outcome $\left(\frac{\text{Separateness}}{\text{Compactness}}\right)$	Left out cluster from representation
C_p & C_q	C_p or C_q	$\left(\frac{C_p \,\&\, C_q}{C_p \; or \; C_q}\right)$	C_m and C_z Either C_p or C_q
C_m & C_z	C_p or C_q	$\left(\frac{C_m \,\&\, C_z}{C_p \; or \; C_q}\right)$	Either C_p or C_q
C_p & C_q	C_m or C_z	$\left(\frac{C_p \,\&\, C_q}{C_m \; or \; C_z}\right)$	Either C_m or C_z
C_m & C_z	C_m or C_z	$\left(\frac{C_m \,\&\, C_z}{C_m \; or \; C_z}\right)$	C_p and C_q Either C_m or C_z

For example, Dunn index is the representative value derived from any one of the outcomes for the given clusters C_p, C_q, C_m, and C_z. When the Dunn index is applied in practice, one has to deal with a large number of clusters. Then the issue pronounced in the fourth column of Table 3 will be more as more number of clusters may not have representation.

The above mentioned issue is addressed in a modified Dunn index (*MDI*), Which is

$$MDI = \frac{\sum_{p=1}^{k-1}\left(\frac{d_{pq}}{WD(p,q)}\right)}{l}, \quad p=\{1,2,...k-1\}, q=\{p+1, p+2.....,k\}, p<q \tag{16}$$

$$d_{pq} = \min_{\substack{i\in I_p \\ j\in I_q}} \left\| x_i^{\{p\}} - x_j^{\{q\}} \right\|, \quad p<q, \quad \forall i \in I_p, \forall j \in I_q \tag{17}$$

In the Eq. (17) d_{pq} is the minimum distance between 'p' and 'q' pair of clusters; '*i'* represents the data objects that belongs to the p^{th} cluster, whereas 'I_p' is the set of data objects that belongs to the p^{th} cluster, $I_p = \{x_i\}$, $i=\{1, 2, ...n_p\}$, n_p= number of data objects in the p^{th} cluster, Similarly *'j'* represents the data objects that belong to the q^{th} cluster; whereas 'I_q' is the set of data objects that belongs to the q^{th} cluster, $I_q = \{x_j\}$, $j=\{1, 2, ...n_q\}$, n_q= number of data objects in the q^{th} cluster. Where I_l is the representation of the set of independent combinations I_l = $\{(C_1, C_2), (C_1, C_3) ....(C_1, C_q), ...(C_p, C_q)\}$, $p=\{1,2,...k-1\}$, $q=\{p+1,p+2,....k\}, p<q$.

$$WD(p,q) = \frac{D_p + D_q}{2}, \tag{18}$$

In the Eq. (18) '*WD(p,q)'* is the average within distance of combination for p and q clusters.

$$D_p = \max_{\substack{i,j\in I_p \\ i\neq j}} \left\| x_i^{\{p\}} - x_j^{\{p\}} \right\|, \tag{19}$$

In the Eq. (19) 'D_p' is the maximum distance between two data objects from the p^{th} cluster. *MDI* is calculated by considering all the independent combinations of the clusters. The objective is to maximize the *MDI* value which varies between the 0 to ∞. The average values used when computing the modified Dunn Index leads to an increased value of the index i.e., the magnitude of the modified Dunn index is higher than the Dunn index which allows for easy interpretation. The algorithm for *DU* and *MDI* is listed below:

Algorithm 4: Dunn Index (*DU)*

Input: *X*: data objects consist of '*n*' observation
CL: Class_label
//where CL is the Class_label generated for each data object after the convergence achieved for positioning the data objects into the appropriate cluster //

Output: Dunn index value (*DU*)

1: For each combination (*p,q*), Compute the distance between each x_i with all the x_j
//where $x_i \epsilon p$ and $x_j \epsilon q$, p={1,2,..k-1}, q={p+1,p+2,...k}, i={1,2,...n_p}, j={1,2,...,n_q}//
2: Find the minimum distance among the distance estimated in step (2) from each (*p, q*) combination
3: Repeat the step (1) and (2) for all the $\frac{k(k-1)}{2}$ combinations
4: Form the set of minimum distance
// The set consists of $\frac{k(k-1)}{2}$ minimum distance//
5: For each '*k*', Compute distance among the x_i within the cluster
6: Identify the maximum distance among the distance calculated in step (5)
7: Form a *'k'* set of maximum distance
// The set consists of 'k' maximum distance//
8: Select the minimum distance (d_{min}), among the set of minimum distance in step (4)
9: Select the maximum distance (d_{max}), among the set of maximum distance in step (7)
10: Compute $DU = \frac{d_{min}}{d_{max}}$
11: return *DU*

The input requires the data objects (*X*) and '*CL*', where '*CL*' is the class label generated for each data objects after the convergence achieved by the clustering algorithm (i.e., the class label is generated after the *Min SSE* calculated), which helps to place the data objects in to appropriate cluster. In Dunn Index algorithm step (8) the minimum distance is selected among the set of minimum distance generated from $\frac{k(k-1)}{2}$ independent pairwise cluster combinations, which is the single representative value (d_{min}) for separateness. Similarly, step (9) the maximum distance is selected among the set of maximum distance generated from the *'k'* clusters, which is the single representative value (d_{max}) for compactness (data density). These are the remarks identified in the Dunn Index. In *MDI* algorithm the step (1) to (6) is the same steps used for Dunn index is carried out in the *MDI*.

Algorithm 5: Modified Dunn Index (*MDI*)

Input: *X*: data objects consist of '*n*' observation
CL: Class_label
//where CL is the Class_label generated for each data object after the convergence achieved for positioning the data objects into the appropriate cluster //

Output: Modified Dunn index value (*MDI*)

Step (1) to (6) follow the steps as same as Dunn Index

7: Compute Average maximum distance (*WD(p,q)*) using step (6)

8: Form a set of average maximum distance (*WD*) value for $\frac{k(k-1)}{2}$ pair of cluster combinations
9: Compute ratio of *WD* and corresponding minimum distance from step (4)
// The ratio is calculated for each pair of cluster combination, which consists of $\frac{k(k-1)}{2}$ *ratio value //*
10: Compute *MDI* by taking average of ratio generated from step (9)
11: return *MDI*

In Modified Dunn Index *(MDI)*, remarks mentioned in Dunn Index are addressed in steps (7) to (10) the ratio of separateness and compactness is computed for all the $\frac{k(k-1)}{2}$ independent pairwise cluster combinations, finally the average value treated as a *MDI*.

4.3.3 The silhouette index

A cluster silhouette index is a measure of cluster goodness. It was first described by Rousseew (1986). Let us consider that there are '*k*' number of clusters to be formed, the global silhouette index (*GS*) is the mean of the silhouettes of all the clusters, and is given by

$$GS = \frac{1}{k}\sum_{p=1}^{k} MSIL_p \ , \tag{20}$$

The silhouette of the entire dataset is the average of silhouette scores of all individual clusters. This is a measure of how appropriately the data points have been clustered (Ansari et al., 2011).

The mean of the silhouette for a given '*k*' cluster is called the cluster mean silhouette (*MSIL*), and is denoted as $MSIL_p$ where '*i*' represents the data objects which belong to the p^{th} cluster, $i \epsilon I_p$, $p=\{1,2,....k\}$, $i=\{1,2,...., n_p\}$, 'n_p' represents the number of data objects in the p^{th} cluster. It is given by

$$MSIL_p = \frac{1}{n_p}\sum_{i \in I_p} SIL(i) \quad , \tag{21}$$

Where *SIL(i)* is the silhouette measure which is calculated as follows:

$$SIL(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \ , \tag{22}$$

The expression (22) is used for finding the silhouette measure for the i^{th} reference point.

$$a(i)=\frac{1}{n_p-1}\sum_{\substack{i,j\in C_p \\ i\neq j}}\left\|x_i-x_j\right\| \quad , \quad \forall j\in C_p \tag{23}$$

Where *a(i)* is the average dissimilarity of the i^{th} data object to all other data objects in the same cluster C_p , '*j*' represents the data objects other than i^{th} data object. The set of mean distances is computed by finding out the distance between the sample reference point (*i*) and points in each other clusters. The minimum mean distance from the set of mean distances is selected. This minimum mean distance is the representative of *b(i)* for the i^{th} sample point. The *b(i)* calculation is as follows:

$$b(i)=\min_{q\neq p}\{d_2(x_i,C_q)\} \quad , \quad i\in p \tag{24}$$

Let us assume, that the i^{th} sample point belongs to the p^{th} cluster. Then $d_2(x_i, C_q)$ is calculated for each cluster other than the p^{th} cluster, where the reference sample point (*i*) belongs.

$$d_2(x_i,C_q)=\frac{1}{n_q}\sum_{j\in q}d_1(x_i,x_j) \quad , \tag{25}$$

Where 'd_2' is the average values of the calculated dissimilarity values for each cluster, '*i'* is the data object belongs to the p^{th} cluster. *'j'* is the data objects belongs to the q^{th} cluster, 'n_q' is the number of data objects in the q^{th} cluster.

$$d_1(x_i,x_j)=\left\|x_i-x_j\right\| \quad , \quad i\in p, j\in q, i\notin q, j\notin p, i\neq j, p,q=\{1,2....k\}, p\neq q \tag{26}$$

Where 'd_1' is the dissimilarity between i^{th} and j^{th} data object, for a reference point (*i*) in a p^{th} cluster, calculate the dissimilarity value with all points in each cluster other than its own cluster. The algorithm for *b(i)* calculation is as follows

Algorithm 6: *b(i)* calculation

Input *X*: Data matrix with '*n*' data objects
CL: Class_label
//where CL is the Class_label generated for each data object after the convergence achieved for positioning the data objects into the appropriate cluster //
Output: '*b*' values (*n x1* vector values)
1: Compute dissimilarity value (d_1) between the reference point x_i with all x_j using Eq. (26)

//where $x_i \in p$, $x_j \in q, i \notin q, j \notin p$ $i=\{1,2,..n_p\}, j=\{1,2,..n_q\}, p \neq q$ *//*

2: Compute average values (d_2) from the calculated d_1 for each cluster, using Eq. (25)
// for i^{th} *reference point, there are (k-1)* d_2 *values are generated//*
3: Select minimum of all the average values (*b(i)*) of d_2, using Eq. (24)
// where b(i) is the minimum mean distance selected from the set of d_2 *values//*
4: return *b(i)* values

Silhouette ranges between [-1, 1]; a value near to '1' indicates that the data object affected to the correct cluster, whereas a value near '-1' indicates that the data object affected to another cluster. i.e., the data object is wrongly clustered. This is evidenced in the expression (22). In addition, for any data object *(i)*, when *b(i) < a(i)*, then *SIL(i)* assumes negative value and *b(i) = a(i)*, then *SIL(i)* assumes zero value. This situation leads to imply that the i^{th} data object is not placed in the correct cluster. Similarly, when *b(i) > a(i)*, then *SIL(i)* assumes positive value, then one can infer that the i^{th} data object is placed in the correct cluster. These two situations lead to the development of a new validity ratio which is termed as Silhouette Validity Ratio(*SVR*).

4.3.4 Silhouette validity ratio (SVR)

In the silhouette index computation, the within cluster mean distance of the i^{th} data object is measured through *a(i)*, which is the measure for the error term and *b(i)* is the minimum mean distance, chosen from the set of mean distances. In *SIL(i)* expression the numerator becomes negative, when *a(i) > b(i)*, this shows that i^{th} data object should be affected to other cluster as the error value *a(i)* is more than *b(i)*. Similarly, the numerator becomes positive, when *a(i) < b(i)*, this denotes that i^{th} data object is affected to the correct cluster. Based on this logic the silhouette validity ratio (*SVR*) is proposed. This ratio helps in identifying the better method, which is used in the classification. This measure ascertains the belongingness of data objects with respect to each cluster for the given dataset. Specifically, this index enables in assessing the effectiveness of the clustering methods in terms of bringing out negative values for data object which mostly in the boundary region of the cluster. The *SVR* expression is as follows:

$$SVR = \frac{N_{negative}}{(N_{negative} + N_{positive})}, \tag{27}$$

$$DBCS(i) = b(i) - a(i), \tag{28}$$

Where '*DBCS(i)*' is the deviation between within mean distance of i^{th} data object and minimum mean distance between the i^{th} data object to each data objects in other clusters. Where '$N_{negative}$' is the number of elements which possess *DBCS(i)* values as negative and zero and '$N_{positive}$' denotes the number of elements which possess *DBCS(i)* as positive. The objective is to minimize the *SVR*, which ranges from 0 to 1. The effectiveness of the clustering methods is assessed by applying this ratio to find the number of data objects with negative values. Then the best method is chosen which brings less number of negative value, which is an indirect measure of accuracy. The data objects which possess negative values mostly lie in the

boundary region of the cluster. Thus this ratio also helps in assessing the ability of its own cluster by keeping the data objects intact.

The main advantages in *SVR* are as follows: The negative *DBCS* values denote that the data objects lie on the boundary region of the cluster. Thus it acts as an indicator to detect the fuzzy elements. For the same dataset, various clustering algorithms result in different cluster structure. Due to this variation, the number of data objects lie in the boundary region also gets varied. The number of negative elements decides the effectiveness of the clustering method.

5. Experimental results

The practical applicability of our proposed initialization algorithm is tested experimentally and evaluated by using different benchmark datasets. The datasets are collected from the UCI machine learning repository (http://archive.ics.uci.edu/ml/), R datasets (https://vincentarelbundock.github.io/Rdatas-ets/datasets.html), KEEL datasets (https://sci2s.ugr.es/keel/category.php?cat=clas), and Clustering basic benchmark datasets (http://cs.joensuu.fi/sipu/datasets/).

5.1 Summary of datasets

Twenty-eight datasets are used for this study. These datasets consisted of 47–5069 cases (*n*), 2–90 features (*f*), and 2–31 clusters (*k*). The class label is either known or unknown. The proposed algorithms are tested and analyzed using MATLAB and R software. Our computing system is configured with an Intel™ Core ™, 8.00 GB RAM, and 64-bit operating system. The details of these various datasets are listed in Table 4.

Table 4
Characteristics of datasets

S. No	Datasets	Number of Observations *(n)*	Number of Variables *(f)*	Number of Clusters *(k)*
1	Abalone	4177	8	10
2	Atom	800	3	2
3	Bank Notes	200	6	2
4	Chainlink	1000	3	2
5	Compound	399	2	6
6	Cygobi	47	2	2
7	Data_Akbil	536	9	4
8	UM	258	5	4
9	Hepta	212	3	7
10	Iris	150	4	3
11	Sammon	75	4	5
12	Sensor Discriminator	2212	12	3
13	UID	5069	4	2
14	Xclara	3000	2	3
15	Thyroid	215	5	3
16	A1	3000	2	20
17	Jain	373	2	2
18	R15	600	2	15
19	Glass	214	9	7
20	Flame	240	2	2
21	Ecoli	336	7	8
22	Sonar	208	60	2
23	Pathbased	300	2	3
24	Aggeration	788	2	7
25	Hayes-roth	160	4	3
26	D31	3100	2	31
27	Movement Libra	360	90	15
28	Ionosphere	351	33	2

The random initialization for *k*-means clustering leads to several replications. In this paper, 30 replications are used for the validation process.

5.2 Comparison of the conventional and proposed method through performance measure

The performances of the proposed algorithm are studied in terms of the number of iterations, *SSE*, and consistency of error. Similarly, the performance of the conventional *k*-means clustering algorithm and *k*-means++ algorithm was analyzed, and then the experimental results of the proposed algorithm (P*k*CIA) with conventional *k*-means and *k*-means++ algorithm is compared. We have considered the performance criteria such as the average sum of squared error and the number of iterations in Table 5.

Table 5
Comparison of average *SSE* ($\overline{SSE}$) and number of iterations ($\overline{iter}$) for initialization algorithms

Dataset	*k*-means		*k*-means++		P*k*CIA	
	$\overline{SSE}$	$\overline{iter}$	$\overline{SSE}$	$\overline{iter}$	$\overline{SSE}$	$\overline{iter}$
Abalone	138.25	41.00	136.00	39.40	**125.14**	**32.00**
Atom	299.05	10.36	298.93	9.92	**298.52**	**7.00**
Bank Notes	102.41	5.00	102.41	4.57	**102.40**	**2.00**
Chainlink	515.05	18.20	513.46	17.40	**513.43**	**3.00**
Compound	4647.29	15.70	4642.50	17.20	**3865.94**	**5.00**
Cygobi	8.58	4.80	8.44	4.72	**8.15**	**3.00**
Data_Akbil	0.496	22.12	0.496	20.60	**0.495**	**12.00**
UM	41.33	16.10	41.39	14.70	**41.31**	**6.00**
Hepta	22.81	5.90	11.91	3.52	**7.12**	**2.00**
Iris	89.26	6.20	78.94	6.60	**78.85**	**2.00**
Sammon	328.63	6.60	328.63	6.24	**326.98**	**3.00**
Sensor Discriminator	26752.65	13.60	27269.51	10.92	**20938.60**	**8.00**
UID	2603590.00	11.00	2603590.00	10.20	**2603590.00**	**2.00**
Xclara	611606.00	5.40	611606.00	5.00	**611606.00**	**3.00**
Thyroid	5.82E+14	3.63	5.44E+14	4.38	**4.60E+14**	**3.00**
A1	1.81E+10	4.78	1.78E+10	4.44	**1.21E+10**	**3.00**
Jain	22214.42	8.10	22216.3	7.50	**22208.80**	**5.00**
R15	190.41	5.31	178.32	6.07	**108.62**	**4.00**
Glass	327.32	9.94	323.25	6.81	**292.35**	**5.00**
Flame	3151.52	7.50	3150.48	6.87	**3147.54**	**5.00**
Ecoli	15.22	13.67	15.11	14.44	**14.33**	**7.00**
Sonar	280.71	13.75	280.68	12.62	**280.57**	**12.00**
Pathbased	8973.61	12.08	8971.68	11.08	**8957.91**	**8.00**
Aggeration	12338.4	14.40	11650.1	13.60	**10996.80**	**12.00**
Hayes-roth	359.29	6.82	356.29	6.36	**344.94**	**6.00**
D31	4039.63	16.58	3927.07	16.25	**3773.55**	**15.00**
Movement Libra	327.48	10.07	323.31	9.86	**314.32**	**9.00**
Ionosphere	2387.35	5.83	2387.32	5.66	**2387.29**	**3.00**

The best results obtained from the proposed method are depicted in boldfaced

The values listed in Table 5 clearly shows that the proposed method (P*k*CIA) provides minimum *SSE* values with less number of iterations than the conventional *k*-means and *k*-means++ algorithms. Furthermore, the proposed initialization algorithm for *k*-means clustering could produce a better clustering result for all the datasets. Comparison of precision charts for the initialization algorithm are represented in Fig. 1 and 2.

[Please Insert Figure 1: Comparison of precision charts for the initialization algorithms (Aggregation Dataset)]

[Please Insert Figure 2: Comparison of precision charts for the initialization algorithms (D31 Dataset)]

In Fig. 1 and 2 mean(Range)) is the representation of $\overline{Range}$. The *LCL* and *UCL* are the multiplication of table values for the corresponding number of iterations with the mean of range of *SSE* values. For the random initialization algorithm, the range for each replication is inconsistent and varied between the *LCL* and *UCL*. Conversely, for the P*k*CIA, the range values are consistent for all replications. For illustration purposes, the Aggregation and D31 datasets are listed in Fig. 1 and 2. The chart clearly shows that some of the range values are above the *UCL* limit for conventional *k*-means and *k*-means++. Similar graphs are obtained for all datasets used in this paper. Results showed that the P*k*CIA provided a consistent *SSE* value for each replication; meanwhile, the range values for each replication are lies between the lower and upper control limits and comparatively the range values are minimum than the conventional methods (*k*-means and *k*-means++). Table 6 is the comparison of the mean of range of *SSE* values for the conventional and proposed method.

Table 6
Comparison of mean of range of *SSE* for the conventional and proposed method

Datasets	*k*-means	*k*-means^{++}	P*k*CIA
Abalone	56.74	53.45	**23.76**
Atom	36.58	28.24	**17.60**
Bank Notes	6.72	5.64	**3.12**
Chainlink	91.43	78.26	**55.13**
Compound	202.52	198.76	**141.30**
Cygobi	5.12	4.32	**3.21**
Data_Akbil	1.23	0.94	**0.45**
UM	36.74	32.12	**22.62**
Hepta	33.65	30.12	**19.78**
Iris	5.22	3.88	**2.42**
Sammon	102.32	103.32	**75.65**
Sensor Discriminator	2704.51	2503.12	**1672.34**
UID	25601.02	21601.33	**14601.68**
Xclara	20601.54	19874.17	**11346.53**
Thyroid	2.70E+14	2.57E+14	**1.59E+14**
A1	4.65E+09	3.51E+09	**2.62E+09**
Jain	2681.82	1948.10	**1250.50**
R15	39.85	34.31	**27.89**
Glass	53.06	41.64	**17.60**
Flame	1287.23	1056.3	**874.56**
Ecoli	3.25	3.13	**1.56**
Sonar	92.31	79.94	**63.12**
Pathbased	201.51	198.26	**121.38**
Aggeration	2956.98	2742.23	**2107.00**
Hayes-roth	170.32	125.60	**98.56**
D31	2328.39	2160.23	**1261.17**
Movement Libra	73.96	73.19	**52.56**
Ionosphere	513.48	428.22	**128.31**

The best results obtained from the proposed method are depicted in bold

The value listed in Table 6 clearly show that the mean of range of *SSE* for the proposed initialization algorithm is less than that of the conventional algorithms (*k*-means and *k*-means++) for all the datasets. Then paired t-test was conducted based on the mean of range of *SSE* for 30 replications. The results are listed in Table 7.

Table 7
Paired t-test results for the conventional with proposed method

Datasets	*k*-means & *k*-means++		*k*-means & P*k*CIA		*k*-means++ & P*k*CIA	
	t-test	p-value	t-test	p-value	t-test	p-value
Abalone	0.10	0.921032	**4.58**	0.000081**	**4.41**	0.00013**
Atom	2.10	0.044542*	**6.23**	0.000001**	**4.13**	0.00028**
Bank Notes	1.05	0.302390	**7.14**	0.000000**	**6.43**	0.00000**
Chainlink	0.03	0.976273	**4.88**	0.000035**	**3.41**	0.00193**
Compound	1.09	0.284688	**6.3**	0.000001**	**4.1**	0.00030**

Cygobi	0.45	0.656056	**7.21**	0.000000**	**6.25**	0.00000**
Data_Akbil	0.03	0.976273	**4.3**	0.000176**	**4.42**	0.00013**
UM	2.40	0.023039*	**6.3**	0.000001**	**4.1**	0.00030**
Hepta	1.54	0.134402	**4.13**	0.000281**	**5.32**	0.00001**
Iris	2.23	0.033651*	**4.53**	0.000093**	**4.42**	0.00013**
Sammon	1.11	0.276119	**4.42**	0.000127**	**5.32**	0.00001**
Sensor Discriminator	1.34	0.190651	**7.32**	0.000000**	**6.69**	0.00000**
UID	0.78	0.441711	**6.5**	0.000000**	**4.42**	0.00013**
Xclara	1.89	0.068791	**4.3**	0.000176**	**4.1**	0.00030**
Thyroid	1.21	0.236050	**4.1**	0.000305**	**3.9**	0.00052**
A1	0.62	0.540099	**4.08**	0.000322**	**3.29**	0.00263**
Jain	1.11	0.276119	**4.01**	0.000389**	**3.63**	0.00108**
R15	1.87	0.071610	**4.82**	0.000042**	**3.49**	0.00157**
Glass	2.04	0.050551	**5.01**	0.000025**	**4.09**	0.00031**
Flame	1.99	0.056093	**4.78**	0.000047**	**3.92**	0.00050**
Ecoli	0.55	0.586531	**3.69**	0.000922**	**3.33**	0.00238**
Sonar	1.17	0.251131	**4.77**	0.000048**	**4.66**	0.00007**
Pathbased	1.39	0.175107	**5.11**	0.000019**	**6.03**	0.00000**
Aggeration	0.75	0.459297	**3.69**	0.000922**	**3.57**	0.00123**
Hayes-roth	2.09	0.045497*	**3.98**	0.000422**	**4.04**	0.00036**
D31	1.37	0.181201	**8.81**	0.000000**	**7.54**	0.00000**
Movement Libra	1.34	0.190651	**3.99**	0.000411**	**3.09**	0.00439**
Ionosphere	1.08	0.289043	**5.45**	0.000007**	**4.32**	0.00017**

*The significant results are depicted in bold; ** significance level at 0.01; * significance level at 0.05*

This table clearly shows that deviation of the mean of range of *SSE* values between the proposed initialization algorithm (P*k*CIA) and the conventional initialization algorithm (*k*-means and *k*-means++) are significant. Moreover, for all datasets the deviation of mean of range of *SSE* values for the conventional initialization algorithm is higher than that of the proposed initialization algorithm. Therefore, H_1 is accepted at the significance level of 0.05 and 0.01 for degrees of freedom 29. This shows that the proposed algorithm (P*k*CIA) has provided better initial and final *SSE* values than the conventional methods (*k*-mean and *k*-means++). The results obtained from the performance measure for the P*k*CIA show the superiority over the conventional *k*-means and *k*-means++ algorithm. The results obtained from the validity measure are discussed in the sub-section.

5.3 Comparison of the conventional and proposed method through validity measure

The *DU* and *MDI* are calculated on the basis of the cluster structure formed for the minimum *SSE* values. The minimum *SSE* values for all initialization algorithms are chosen from the 30 replications. Table 8 is the comparison of *DU* and *MDI* of all initialization algorithms.

Table 8
Comparison of *DU and MDI* values of all initialization algorithms

Datasets	*k* means		*k* means++		P*k*CIA	
	DU	*MDI*	*DU*	*MDI*	*DU*	*MDI*
Abalone	0.013	0.651	0.013	0.651	**0.018**	**0.679**
Atom	0.038	0.329	0.038	0.329	**0.041**	**0.393**
Banknotes	0.100	0.439	0.100	0.439	**0.230**	**0.439**
Chainlink	0.018	0.018	0.018	0.018	**0.018**	**0.018**
Compound	0.025	0.353	0.025	0.353	**0.035**	**0.459**
Cygobi	0.040	0.081	0.040	0.081	**0.070**	**0.084**
Data Akbil	0.060	0.190	0.060	0.190	**0.060**	**0.190**
UM	0.068	0.228	0.068	0.228	**0.102**	**0.236**
Hepta	0.600	1.319	0.600	1.319	**1.080**	**1.684**
Iris	0.100	0.740	0.100	0.740	**0.100**	**0.740**
Sammon	0.130	0.288	0.130	0.288	**0.160**	**0.298**
Sensor Discriminator	0.019	0.153	0.019	0.153	**0.024**	**0.820**
UID	0.008	0.890	0.008	0.890	**0.008**	**0.890**
Xclara	0.050	1.060	0.050	1.060	**0.050**	**1.060**
Thyroid	0.031	0.031	0.031	0.031	**0.091**	**0.091**
A1	0.019	0.213	0.019	0.219	**0.023**	**0.241**

Jain	0.012	0.012	0.012	0.012	**0.019**	**0.019**
R15	0.129	0.221	0.129	0.221	**0.194**	**0.356**
Glass	0.052	0.113	0.052	0.113	**0.060**	**0.138**
Flame	0.038	0.039	0.038	0.038	**0.038**	**0.038**
Ecoli	0.049	0.082	0.049	0.082	**0.051**	**0.130**
Sonar	0.131	0.131	0.131	0.131	**0.131**	**0.131**
Pathbased	0.032	0.434	0.032	0.434	**0.035**	**0.510**
Aggeration	0.029	0.586	0.029	0.585	**0.033**	**0.674**
Hayes-roth	0.209	0.391	0.208	0.391	**0.213**	**0.395**
D31	0.016	0.047	0.016	0.047	**0.021**	**0.056**
Movement Libra	0.079	0.226	0.079	0.226	**0.115**	**0.384**
Ionosphere	0.071	0.072	0.071	0.072	**0.071**	**0.072**

The best results obtained from the proposed method are depicted in bold

The higher value of *DU* and *MDI* shows better results. The proposed method provides a higher *DU* value than conventional methods. Comparatively *MDI* values for the proposed method higher for most of the datasets. The best method is chosen on the basis of the maximum *DU* and *MDI* values. Here, *MDI* is introduced as an alternative for *DU* by including all the representative values. To validate the modified Dunn index rank comparison and t-test is adopted. The rank comparison of the *DU* and the *MDI* for different initialization algorithms are listed in Table 9. The rank arrives for the data given in Table 8.

Table 9
Rank comparison for the *DU* and the *MDI*

Datasets	*DU*			*MDI*		
	k-means	*k*-means++	P*k*CIA	*k*-means	*k*-means++	P*k*CIA
Abalone	2	2	1	2	2	1
Atom	2	2	1	2	2	1
Banknotes	2	2	1	1	1	1
Chainlink	2	2	1	2	2	1
Compound	2	2	1	2	2	1
Cygobi	2	2	1	2	2	1
Data Akbil	2	2	1	2	2	1
UM	2	2	1	2	2	1
Hepta	2	2	1	2	2	1
Iris	1	1	1	1	1	1
Sammon	2	2	1	2	2	1
Sensor Discriminator	2	2	1	2	2	1
UID	1	1	1	1	1	1
Xclara	1	1	1	1	1	1
Thyroid	2	2	1	2	2	1
A1	3	2	1	3	2	1
Jain	2	2	1	2	2	1
R15	2	2	1	2	2	1
Glass	2	2	1	2	2	1
Flame	1	1	1	1	1	1
Ecoli	2	2	1	2	2	1
Sonar	1	1	1	1	1	1
Pathbased	2	2	1	2	2	1
Aggeration	2	2	1	2	2	1
Hayes-roth	2	2	1	2	2	1
D31	2	2	1	2	2	1
Movement Libra	2	2	1	2	2	1
Ionosphere	1	1	1	1	1	1

The rank for the different initialization algorithms are equal for the *DU* and *MDI*. The magnitude of the *MDI* is higher than the Dunn index but the rank possesses as same as Dunn index. The significance of the difference for the *DU* and *MDI* are tested through t-test. The difference between the *DU* and *MDI* is listed in Table 10.

Table 10

Comparison of difference of *DU* and *MDI* values for different initialization algorithm

Dataset	$MDI_{k\text{-means}} - DU_{k\text{-means}}$	$MDI_{k\text{-means++}} - DU_{k\text{-means++}}$	$MDI_{PkCIA} - DU_{PkCIA}$

Abalone	0.6380	0.6380	0.6610
Atom	0.2910	0.2910	0.3520
Banknotes	0.3390	0.3390	0.2090
Chainlink	0	0	0
Compound	0.3280	0.3282	0.4236
Cygobi	0.0410	0.0410	0.0140
Data Akbil	0.1290	0.1290	0.1540
UM	0.1600	0.1600	0.1340
Hepta	0.7190	0.7190	0.6040
Iris	0.6400	0.6400	0.6400
Sammon	0.1580	0.1580	0.1380
Sensor Discriminator	0.1340	0.1340	0.7960
UID	0.8820	0.8820	0.8820
Xclara	1.0100	1.0100	1.0100
Thyroid	0.0005	0	0
A1	0.1944	0.1995	0.2178
Jain	0	0	0
R15	0.0920	0.0920	0.1619
Glass	0.0614	0.0614	0.0780
Flame	0.0002	0	0
Ecoli	0.0327	0.0327	0.0790
Sonar	0	0	0
Pathbased	0.4024	0.4024	0.4749
Aggeration	0.5570	0.5568	0.6415
Hayes-roth	0.1825	0.1823	0.1814
D31	0.0306	0.0306	0.0350
Movement Libra	0.1462	0.1458	0.2695
Lymphography	0	0	0
Ionosphere	0.6380	0.6380	0.6610

A statistical t-test is conducted to prove that there is a significant difference existing between the mean values of *DU* and *MDI*. The results obtained from the statistical t-test are listed in Table 11.

Table 11:
Comparison of t-test results for *DU* and *MDI*

Methods		Mean	t-calculated	t-table (df=27,Sig=0.05)	p-value	Remarks
k-means	*DU*	0.077	**4.662**	2.052	0.000076	**H_1 Accepted**
	MDI	**0.333**				
k-means++	*DU*	0.077	**4.665**	2.052	0.000076	**H_1 Accepted**
	MDI	**0.333**				
P*k*CIA	*DU*	0.109	**5.061**	2.052	0.000026	**H_1 Accepted**
	MDI	**0.401**				

The t-test showed that there is a significant difference existing between the Dunn index and modified Dunn index. The mean *DU* for *k*-means, *k*-means++ and P*k*CIA is 0.077,0.077, and 0.109, mean *MDI* is 0.333,0.333 and 0.401, this significance of the results shows that the magnitude of the *MDI* is higher than the Dunn Index. Since, the proposed method having higher Dunn and Modified Dunn index value further this shows that the proposed initialization algorithm provides a better solution than the conventional method. The *DU* and *MDI* results show that the P*k*CIA provides better compactness value within the clusters and separateness value between the clusters than the conventional method (*k*-means and *k*-means++).

The comparison of *GS* and *SVR* for the conventional and proposed method is discussed here. For illustration purpose, the negative values ($N_{negative}$) are presented for two dimensional datasets such as D31 and Aggregation. The pictorial representation of $N_{negative}$ values is marked in red colour.

[Please Insert Figure 3: Representation of $N_{negative}$ values for D31 dataset]

[Please Insert Figure 4: Representation of $N_{negative}$ values for Aggregation dataset]

Fig 3 (a) and 4 (a) is the representation of negative values ($N_{negative}$) obtained from the conventional methods (*k*-means and *k*-means++) for D31 and Aggregation dataset. Fig 3(b) and 4(b) is the representation of negative values ($N_{negative}$) obtained from the proposed method (P*k*CIA) for D31 and Aggregation Dataset. The P*k*CIA establish superiority over conventional methods. This is evident through the '$N_{negative}$' values. The P*k*CIA assumes '$N_{negative}$' values of 2 and 6 for D31 and Aggregation dataset, but the conventional method is 14 for both the dataset. The negative *DBCS(i)* shows that the particular data objects (*i*) lie in the boundary region of the cluster. Comparatively, proposed method (P*k*CIA) having a higher number of positive values ($N_{positive}$) than the conventional methods, this shows that most of the data objects are positioned in the appropriate cluster. This is the critical information that we have deduced from the *SVR*. Similarly, for all the dataset the comparison of negative values ($N_{negative}$) for the *k*-means, *k*-means++ and P*k*CIA is shown in Table 12.

Table 12
Comparison of '$N_{negative}$' values for conventional methods and P*k*CIA

Datasets	*k*-means	*k*-means^{++}	P*k*CIA
Abalone	61	61	**0**
Atom	14	14	**6**
Bank Notes	20	20	**0**
Chainlink	8	8	**6**
Compound	22	22	**19**
Cygobi	0	0	**0**
Data_Akbil	29	29	**23**
UM	5	5	**0**
Hepta	1	1	**0**
Iris	1	1	**0**
Sammon	0	0	**0**
Sensor Discriminator	100	100	**100**
UID	230	230	**199**
Xclara	0	0	**0**
Thyroid	3	3	**2**
A1	14	14	**0**
Jain	2	2	**1**
R15	3	3	**0**
Glass	6	6	**4**
Flame	2	2	**2**
Ecoli	20	20	**14**
Sonar	5	5	**5**
Pathbased	1	1	**0**
Aggeration	14	14	**6**
Hayes-roth	3	3	**2**
D31	14	14	**2**
Movement Libra	12	12	**9**
Ionosphere	44	44	**43**

The best results obtained from the proposed method are boldfaced

This shows that the negative data objects lie in the boundary region of the clusters in P*k*CIA is lesser than the conventional methods (*k*-means and *k*-means++). For most of the datasets such as Abalone, Pathbased, R15, A1, UM, Hepta, iris, sammon, XClara, and Banknotes the P*k*CIA algorithm provides the '$N_{negative}$' value zero. This shows that most of data objects are affected to the correct clusters. Similarly, for conventional methods (*k*-means and *k*-means++), provides zero values for Cygobi, Sammon, and Xclara. Comparatively, the proposed method shows better performance in terms of providing a lesser number of negative values. The comparison of global silhouette value (*GS*) and silhouette validity ratio *(SVR)* for the conventional and proposed method is shown in Table 13.

Table 13
Comparison of *GS* and *SVR* for conventional and proposed algorithm

Datasets	*k* means		*k* means++		P*k*CIA	
	GS	*SVR*	*GS*	*SVR*	*GS*	*SVR*
Abalone	0.362	0.015	0.362	0.015	**0.377**	**0**
Atom	0.350	0.017	0.350	0.017	**0.360**	**0.007**
Banknotes	0.439	0.100	0.439	0.100	0.439	**0**
Chainlink	0.280	0.008	0.280	0.008	**0.310**	**0.006**
Compound	0.353	0.055	0.353	0.055	**0.459**	**0.048**
Cygobi	0.190	0	0.190	0	**0.530**	**0**
Data Akbil	0.189	0.054	0.189	0.054	**0.194**	**0.043**
UM	0.162	0.019	0.162	0.019	**0.196**	**0**
Hepta	0.580	0.005	0.580	0.005	**0.700**	**0**
Iris	0.390	0.007	0.390	0.007	**0.560**	**0**
Sammon	0.200	0	0.200	0	**0.200**	**0**
Sensor Discriminator	0.380	0.045	0.380	0.045	**0.380**	**0**
UID	0.600	0.045	0.600	0.045	**0.600**	**0.039**
Xclara	0.700	0	0.700	0	**0.700**	**0**
Thyroid	0.558	0.014	0.558	0.014	**0.577**	**0.009**
A1	0.560	0.005	0.570	0.005	**0.617**	**0**
Jain	0.475	0.005	0.475	0.005	**0.484**	**0.003**
R15	0.466	0.005	0.466	0.005	**0.725**	**0**
Glass	0.424	0.028	0.424	0.028	**0.448**	**0.019**
Flame	0.360	0.008	0.360	0.008	**0.360**	**0**
Ecoli	0.214	0.059	0.214	0.059	**0.248**	**0.042**
Sonar	0.177	0.024	0.177	0.024	**0.177**	**0**
Pathbased	0.502	0.003	0.502	0.003	**0.512**	**0**
Aggeration	0.478	0.018	0.478	0.018	**0.550**	**0.008**
Hayes-roth	0.220	0.019	0.220	0.019	**0.234**	**0.012**
D31	0.523	0.004	0.523	0.004	**0.540**	**0.0007**
Movement Libra	0.232	0.033	0.232	0.033	**0.251**	**0.025**
Ionosphere	0.289	0.125	0.289	0.125	**0.296**	**0.123**

The best result obtained from the proposed initialization are depicted in boldfaced

The P*k*CIA achieved better indices values for all the datasets than the conventional *k*-means and *k*-means++ algorithm. While we comparing *SVR* values in Table 13 most of the values are zero for the proposed initialization algorithm. This shows that most of the *DBCS* values are positive for the proposed method. This is a sign for most of the data objects are correctly classified by the proposed method.

In addition, two graphical representations are presented based on the ratio values calculated through '$N_{positive}$' and '$N_{negative}$' values. The first graph is constructed based on the number of *'$N_{positive}$'* elements per data object. For better understanding purpose, the ratio is calculated through a number of positive *DBSC(i)* data object divided by the total number of data objects $\left(\frac{N_{positive}}{n}\right)$, where '$N_{postive}$' is the elements which possess positive *DBSC(i)* values and '*n*' is the total number of data objects. In order to represent graphically, the *x*-axis is the representation of the total number of data objects, and the *y*-axis is the ratio $\left(\frac{N_{positive}}{n}\right)$ values. The graphical representation is shown below:

[Please Insert Figure 5: Comparison of $N_{positive}$ per data objects for different initialization algorithm]

Fig. 5 clearly shows that the proposed method (P*k*CIA) provides a better result than the conventional method.

The second graph is generated based on the number of '$N_{negative}$' data objects per cluster. The graphical representation (Fig. 6) is also done by sorting the dataset based on the number of clusters in *x*-axis and ratio value in *y*-axis which is calculated by number of incorrectly classified data objects divided

by number of cluster $\left(\frac{N_{negative}}{k}\right)$, where '$N_{negative}$' is the number of data objects incorrectly classified and 'k' is the number of clusters. The proposed method provides better results than conventional methods (*k*-means and *k*-means++). In addition, the peak values show that the datasets belong to the fuzzy clustering. Thus, this graph act as a tool to detect the fuzziness in the data.

[Please Insert Figure 6: Comparison of $N_{negative}$ per cluster for different initialization algorithm]

A limitation of our proposed initialization algorithm is their high computation time. This is because the computational complexity is high for the calculation of eigenvalues for large datasets. However, this computational complexity provided good initial centroids for the *k*-means clustering algorithm. The proposed initialization algorithm exhibited fixed centroid points. The proposed method provided a consistent and minimum error value with a lower number of iterations; In addition, it has provided better validity indices than the conventional methods.

6. Conclusion

The performance of *k*-means clustering highly depends on the initial centroids. A fundamental issue related to the *k*-means clustering is its consistency in obtaining good clustering because of choosing random initial points as centroids. In this paper, we proposed a new initialization procedure (P*k*CIA) for the *k*-means clustering algorithm to address this issue. The average sum of squared error, the average number of iterations, Dunn index, and silhouette index were used as performance measures to validate the clustering results. The Dunn Index is calculated by two distance which is 'd_{min}' and 'd_{max}'. A shortcoming of the present Dunn Index is that there are 'k' number of clusters available but only two representative values are considered for calculating the Dunn Index. However, we addressed this issue by deriving a modified Dunn Index by incorporating all the representative values from the $\frac{k(k-1)}{2}$ combinations. The silhouette index measures the element-wise validity; the value ranged between -1 and 1. A negative value shows that the points do not belong to that cluster; similarly, a positive value indicates that the point belongs to that particular cluster. Next, the silhouette validity ratio is derived by analyzing the positive and negative values of elements belonging and not belonging to a specific cluster. The proposed initialization algorithm is applied to various benchmark datasets, and the results were compared to those of the conventional *k*-means and *k*-means++ algorithms. The proposed initialization algorithm (P*k*CIA) has performed better than the other algorithms, by obtaining the minimum *SSE* with a less number of iterations. Comparison of the average sum of squared error and number of iterations clearly showed that the proposed method produced better results than the conventional *k*-means and *k*-means++ algorithms. A statistical test is conducted on the mean of the range values, revealing a significant difference between the conventional and proposed initialization algorithm. A precision chart is constructed to test the consistency by calculating the *SSE* values obtained from the different initialization algorithms for each replication. The precision charts clearly show that the proposed method has provided minimum and consistent range values for all the datasets. The overall validity of the clusters is measured through the Dunn Index and

modified Dunn Index, while the element-wise validity is measured through the silhouette index and silhouette validity ratio. The results have shown that the proposed initialization algorithm offers better solutions than conventional *k*-means and *k*-means++ algorithms.

References

Allen, T. T., Sui, Z., & Parker, N. L. (2017). Timely decision analysis enabled by efficient social media modeling. *Decision Anal*, *14*(4), 250-260.

Ansari, Z., Azeem, M. F., Ahmed, W., & Babu, A. V. (2015). Quantitative evaluation of performance and validity indices for clustering the web navigational sessions. arXiv preprint arXiv:1507.03340.

Arai, K., & Barakbah, A. R. (2007). Hierarchical K-means: an algorithm for centroids initialization for K-means. *Reports of the Faculty of Science and Engineering*, *36*(1), 25-31.

Arthur, D., & Vassilvitskii, S. (2007, January). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* (pp. 1027-1035). Society for Industrial and Applied Mathematics.

Boobord, F., Othman, Z., & Bakar, A. A. (2015). A WK-means Approach for Clustering. *International Arab Journal of Information Technology*, *12*(5), 489-493.

Bradley, P. S., & Fayyad, U. M. (1998, July). Refining Initial Points for K-Means Clustering. In *ICML* (Vol. 98, pp. 91-99).

Cao, F., Liang, J., & Jiang, G. (2009). An initialization method for the K-Means algorithm using neighborhood model. *Computers & Mathematics with Applications*, *58*(3), 474-483.

Capó, M., Pérez, A., & Lozano, J. A. (2018). An efficient K-means clustering algorithm for massive data. *Journal of LATEX Class Files*, arXiv preprint arXiv:1801.02949.

Celebi, M. E., Kingravi, H. A., & Vela, P. A. (2013). A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert systems with applications*, *40*(1), 200-210.

Cheung, Y. M. (2003). k*-Means: A new generalized k-means clustering algorithm. *Pattern Recognition Letters*, *24*(15), 2883-2893.

Dalhatu, K., & Sim, A. T. H. (2016). Density base k-mean's cluster centroid initialization algorithm. *International Journal of Computer Application*, *137*(11), 0975-8887.

Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters.

Erisoglu, M., Calis, N., & Sakallioglu, S. (2011). A new algorithm for initial cluster centers in k-means algorithm. *Pattern Recognition Letters*, *32*(14), 1701-1705.

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *biometrics*, *21*, 768-769.

Gonzalez, T. F. (1985). Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, *38*, 293-306.

Goyal, M., & Kumar, S. (2014). Improving the initial centroids of K-means clustering algorithm to generalize its applicability. *Journal of The Institution of Engineers (India): Series B*, 95(4), 345-350.

Hamad, M., Thomassey, S., & Bruniaux, P. (2017). A new sizing system based on 3D shape descriptor for

morphology clustering. *Computers & Industrial Engineering*, *113*, 683-692.

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Applied statistics*, *28*(1), 100–108.

He, J., Lan, M., Tan, C. L., Sung, S. Y., & Low, H. B. (2004, July). Initialization of cluster refinement algorithms: A review and comparative study. In 2004 *IEEE International Joint Conference on Neural Networks* (IEEE Cat. No. 04CH37541) (Vol. 1, pp. 297-302).

He, J. Tan, A.H, and Tan, C.L. (2002). ART-C: A neural architecture for selforganization under constraints. In Proceedings of International Joint Conference on Neural Networks (IJCNN), pp 2550–2555.

Hussain, S. F., & Haris, M. (2019). A k-means based co-clustering (kCC) algorithm for sparse, high dimensional data. *Expert Systems with Applications*, *118*, 20-34. https://doi.org/10.1016/j.eswa.2018.09.006.

Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern recognition letters*, *31*(8), 651-666.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, *31*(3), 264-323.

Jothi, R., Mohanty, S. K., & Ojha, A. (2019). DK-means: a deterministic k-means clustering algorithm for gene expression analysis. *Pattern Analysis and Applications*, *22*(2), 649-667.

Katsavounidis, I., Kuo, C. C. J., & Zhang, Z. (1994). A new initialization technique for generalized Lloyd iteration. *IEEE Signal processing letters*, *1*(10), 144-146.

Kaufman, L. and Rousseeuw, P.J. (1990). Finding Groups in Data. An Introduction to Cluster Analysis. John Wiley & Sons, Inc., Canada

Khan, S. S., & Ahmad, A. (2004). Cluster center initialization algorithm for K-means clustering. *Pattern recognition letters*, *25*(11), 1293-1302.

Kumar, Y., & Sahoo, G. (2014). A new initialization method to originate initial cluster centers for K-Means algorithm. *International Journal of Advanced Science and Technology*, *62*, 43-54.

Kuo, R. J., Ho, L. M., & Hu, C. M. (2002). Cluster analysis in industrial market segmentation through artificial neural network. *Computers & Industrial Engineering*, *42*(2-4), 391-399.

Levine, E., & Domany, E. (2001). Resampling method for unsupervised estimation of cluster validity. *Neural computation*, *13*(11), 2573-2593.

Li, C. S. (2011). Cluster center initialization method for k-means algorithm over data sets with two clusters. *Procedia Engineering*, *24*, 324-328.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE transactions on information theory*, *28*(2), 129-137.

MacQueen, J.B. (1967). Some Methods for Classification and Analysis of Multivariate Observations. Proc. Symp. Math. and Probability, 5th., Berkeley, 1, 281-297, AD 669871. University of California Press, Berkeley, CA https://projecteuclid.org/euclid.bsmsp/1200512992

Mahmud, M. S., Rahman, M. M., & Akhtar, M. N. (2012, December). Improvement of K-means clustering algorithm with better initial centroids based on weighted average. In Electrical & Computer

Engineering (ICECE), 2012 7th International Conference on (pp. 647-650). IEEE.

Paea, S., & Baird, R. (2018). Information architecture (IA): using multidimensional scaling (MDS) and K-means clustering algorithm for analysis of card sorting data. *Journal of Usability Studies*, *13*(3), 138-157.

Pena, J. M., Lozano, J. A., & Larranaga, P. (1999). An empirical comparison of four initialization methods for the k-means algorithm. *Pattern recognition letters*, *20*(10), 1027-1040.

Rocke, D. M. (1992). and RQ charts: robust control charts. *Journal of the Royal Statistical Society: Series D (The Statistician)*, *41*(1), 97-104.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, *20*, 53-65.

Saitta, S., Raphael, B., & Smith, I. F. (2008). A comprehensive validity index for clustering. *Intelligent Data Analysis*, *12*(6), 529-548.

Sathiya, G., & Kavitha, P. (2014). An efficient enhanced K-means approach with improved initial cluster centers. *Middle-East Journal of Scientific Research*, *20*(1), 100-107.

Seber, G. A. F., "Multivariate Observations". New York, Wiley- Interscience Paperback Series. 1984.

Su, T., & Dy, J. G. (2007). In search of deterministic methods for initializing K-means and Gaussian mixture clustering. *Intelligent Data Analysis*, *11*(4), 319-338.

Thiesson, B. Meek, C. Chickering, D. and Heckerman, D. (1997). Learning mixtures of bayesian networks. Technical Report MSR-TR-97-30, Microsoft Research.

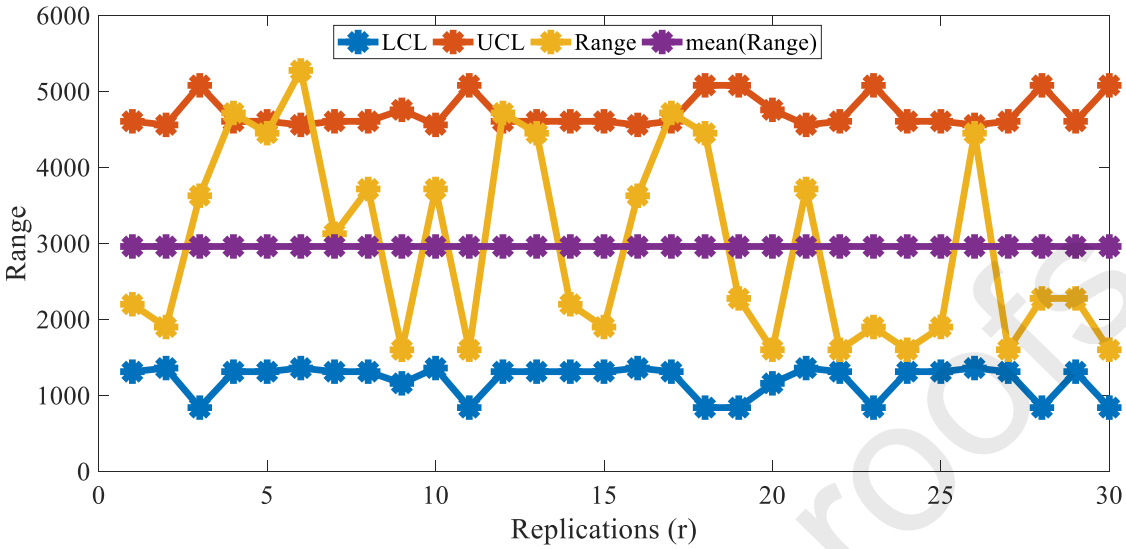
Tou, J.T. and Gonzalez, R.C. (1974). Pattern Recognition Principles. Addison Wesley, Massachusetts, 1974

Tzortzis, G., & Likas, A. (2014). The MinMax k-Means clustering algorithm. *Pattern Recognition*, *47*(7), 2505-2516.

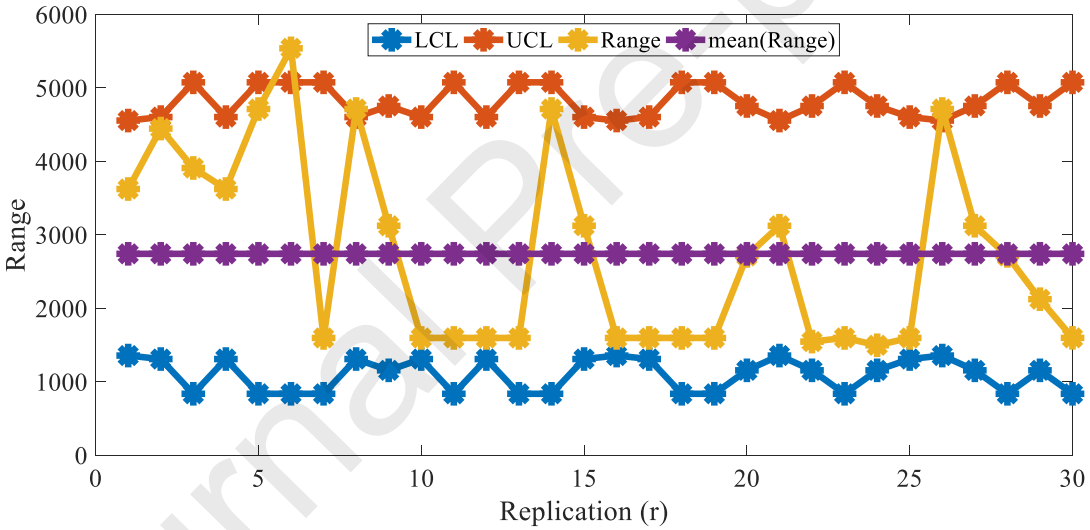
Yang, J., Ma, Y., Zhang, X., Li, S., & Zhang, Y. (2017). An initialization method based on hybrid distance for k-means algorithm. *Neural computation*, *29*(11), 3094-3117.

Yedla, M., Pathakota, S. R., & Srinivasa, T. M. (2010). Enhancing K-means clustering algorithm with improved initial center. *International Journal of computer science and information technologies*, *1*(2), 121-125.

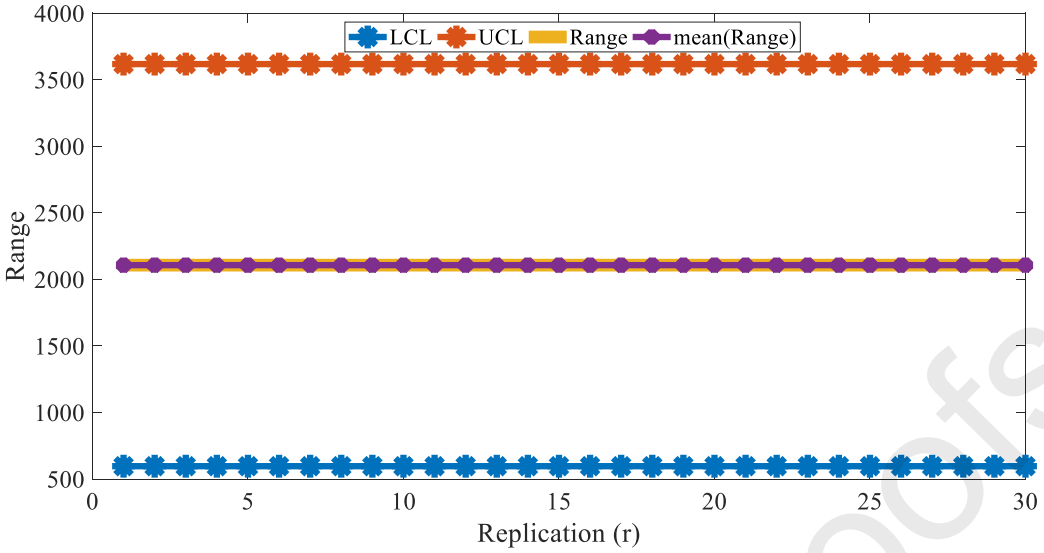
Yin, X. F., Khoo, L. P., & Chong, Y. T. (2013). A fuzzy c-means based hybrid evolutionary approach to the clustering of supply chain. *Computers & Industrial Engineering*, *66*(4), 768-780.



(a) *k*-means algorithm

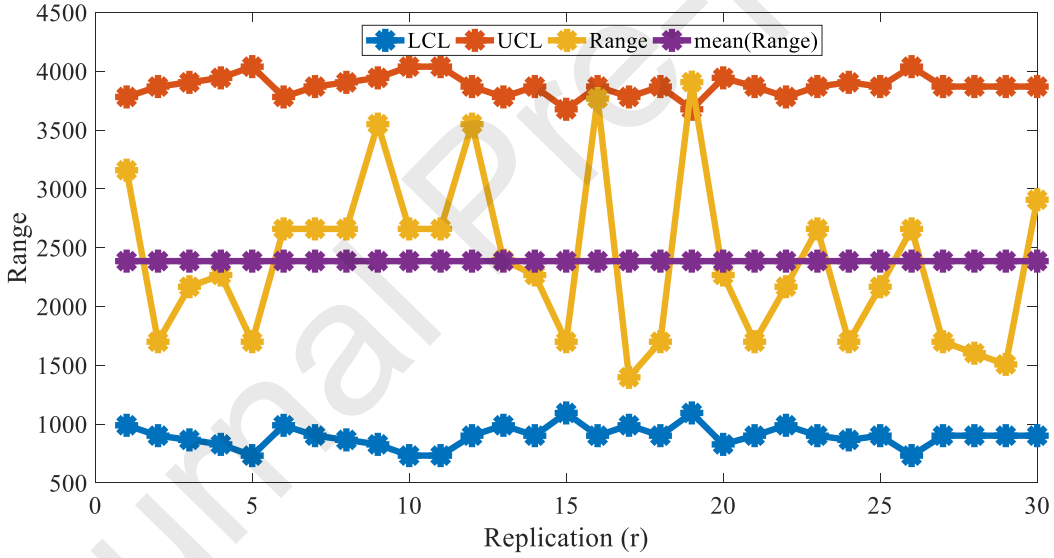


(b) *k*-means++ algorithm

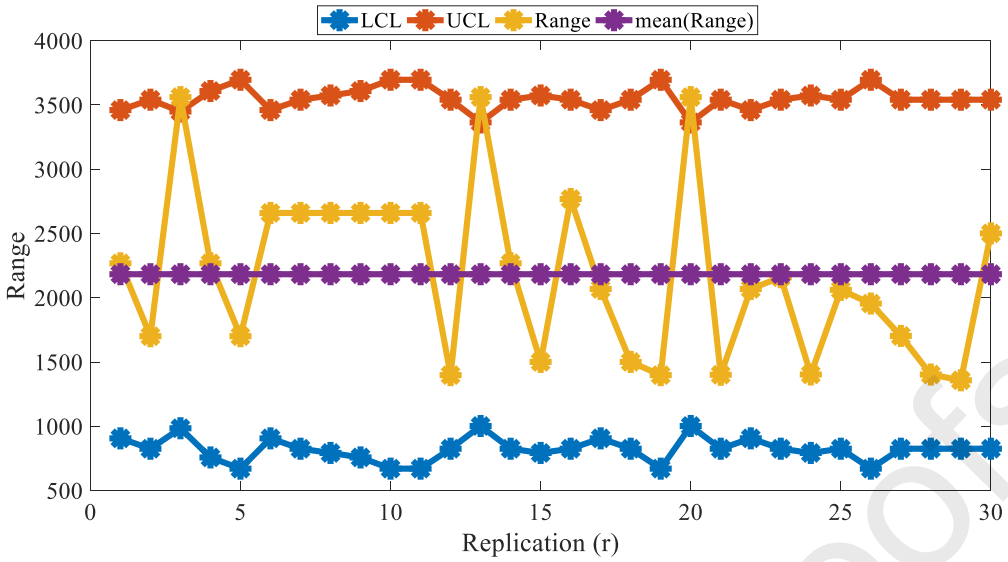


(c)P*k*CIA

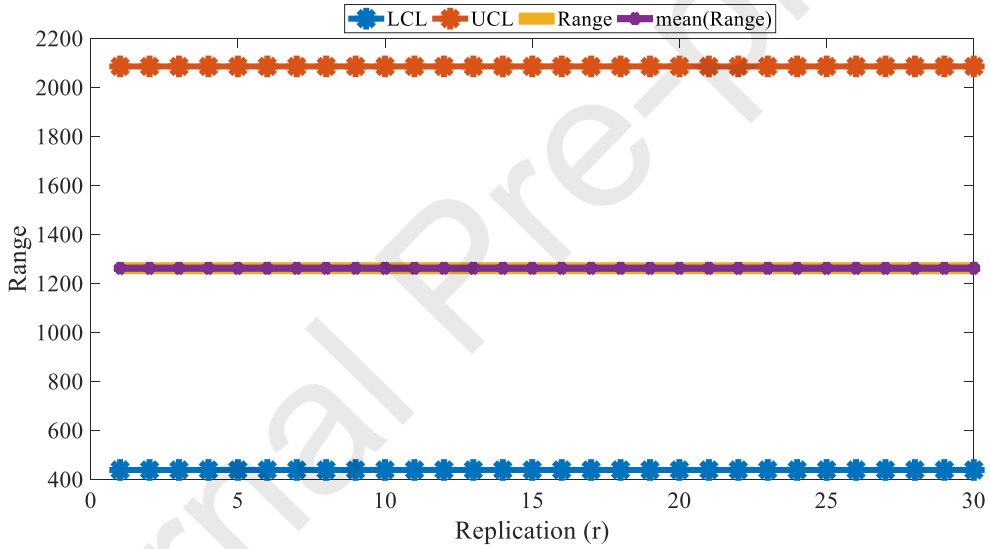
Figure 1: Comparison of precision charts for the initialization algorithms (Aggregation Dataset)



(a) *k*-means algorithm

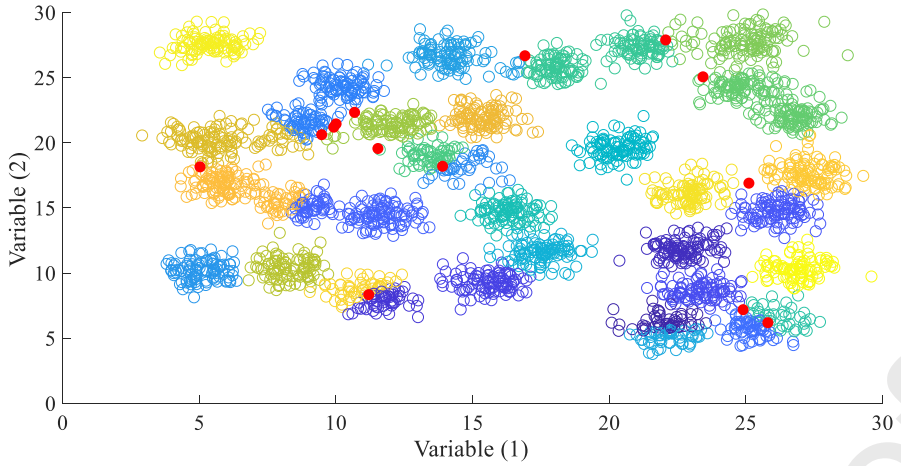


(b) *k*-means++ algorithm

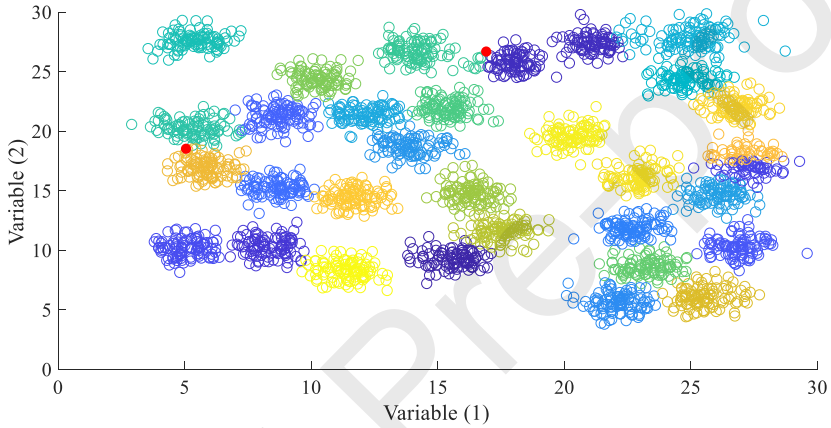


(c)P*k*CIA

Figure 2: Comparison of precision charts for the initialization algorithms (D31 Dataset)

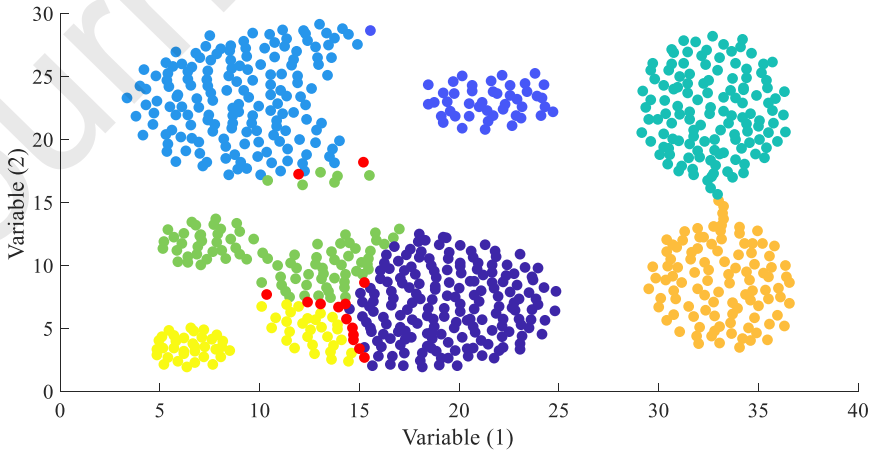


(a) *k*-means and *k*-means ++ Algorithm

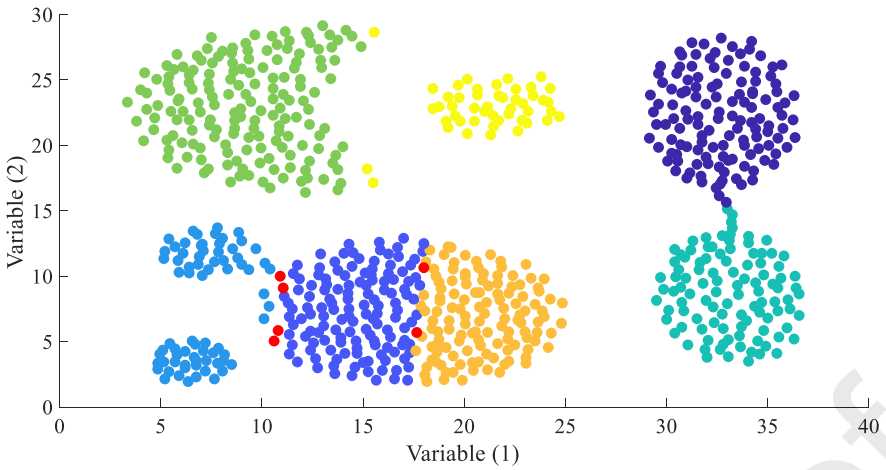


(b) P*k*CIA Algorithm

Figure 3: Representation of $N_{negative}$ values for D31 dataset



(a) *k*-means and *k*-means++ algorithm



(b) P*k*CIA

Figure 4: Representation of $N_{negative}$ values for Aggregation dataset

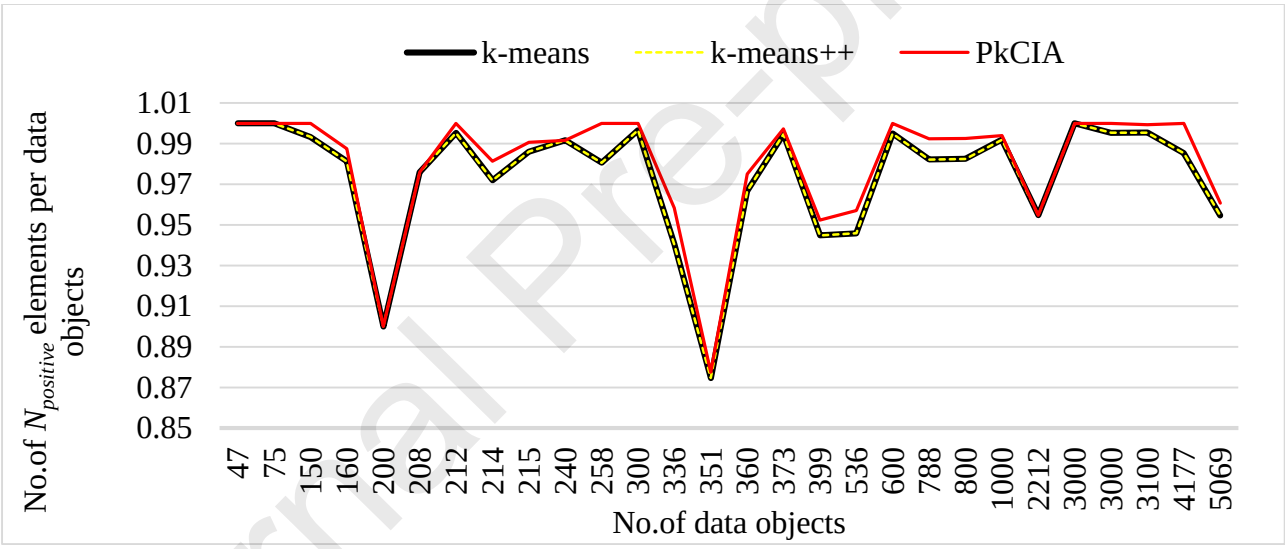


Figure 5: Comparison of $N_{positive}$ per data objects for different initialization algorithm

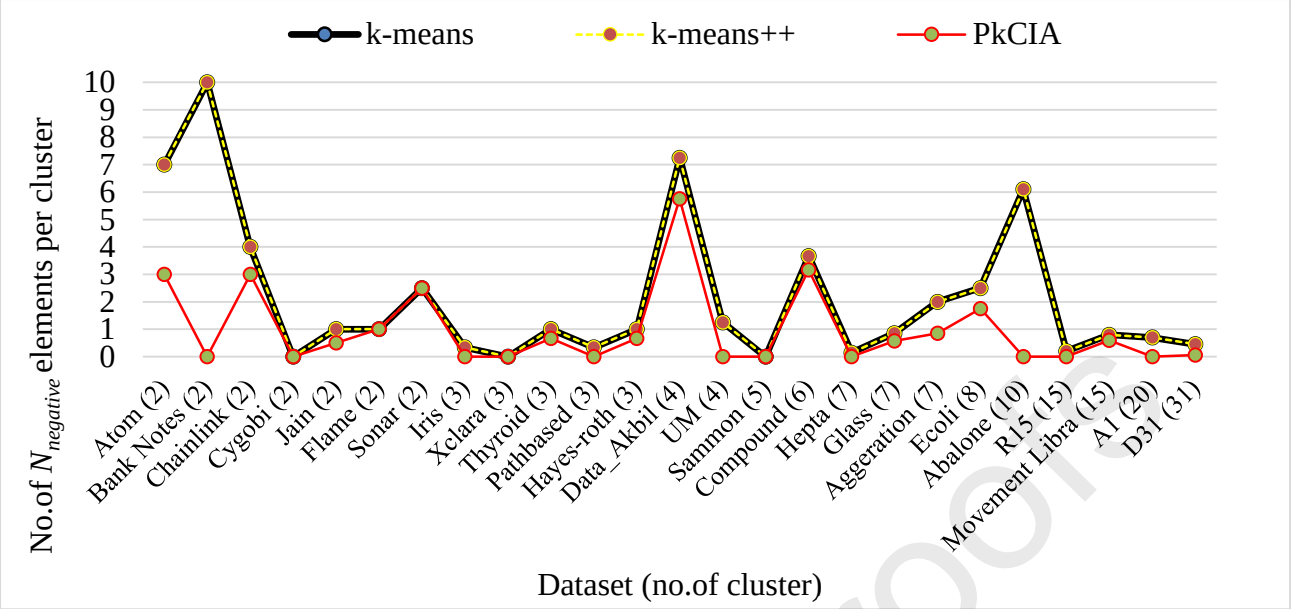


Figure 6: Comparison of $N_{negative}$ per cluster for different initialization algorithm

Highlights

- A performance enhanced initialization for the *k*-means clustering
- A modified Dunn Index as a representative of all the clusters
- Silhouette Validity Ratio to assess the clustering algorithms
- A precision chart to measure the consistency of the clustering algorithm