An Improvement on Initial Centre Points of K-Means Clustering Algorithm

Akinwale Adio, Alabi Orobosade and Onafowora Temitope

Department of Computer Science, Federal University of Agriculture, Abeokuta, Nigeria

DOI: https://doi.org/10.51244/IJRSI.2025.12040155

Received: 21 April 2025; Accepted: 25 April 2025; Published: 24 May 2025

ABSTRACT

A large volume of new data is generated every moment of the day by various domains of social media, bank transactions, and online websites. One of the vital means in dealing with this data is to classify it into a set of clusters. At present, many clustering algorithms have been proposed and improved due to the face of large-scale and high-dimensional data, however, they have not achieved better effects and do not generalize well. This paper improves the k-means algorithm, where its results are sensitive to the initial clustering centre, and the clustering results of different similarity measurement methods would be different. The incorporation of in-built polygon sizes with their respective fuzzy linguistic hedge satisfactions for the classification of data points made clusters distinct from each other. To verify the effectiveness of the proposed methods, two sets of data were used as experimental data sets, which produced a good clustering solution with low processing times.

Keywords: k-means, Inertia method, inter-cluster, fuzzy linguistic, polygon

INTRODUCTION

Clustering is a type of unsupervised learning where data points are grouped into different sets based on their degree of similarity, such as Euclidean based distance, correlation-based distance, Pearson correlation distance, Spearman correlation distance, etc. There are various types of clustering such as hierarchical clustering (agglomerative clustering with bottom-up approach, divisive clustering with top down approach for examples, Clustering using representative, Balanced Iterative Reducing Clustering), partitioning clustering (k-means clustering for example Clustering Large Application based upon Randomized Search, fuzzy c-means clustering, mean shift clustering). Other types are density-based such as Density Spatial Clustering of Application with Noise, Ordering Points to Identify Clustering Structure, grid clustering such as Statistical Information Grid, Clustering in Quest, etc.

Clustering has been applied in various sectors such as market segmentation, where customers are found in terms of behaviour or attributes or similar products, image segmentation or compression in terms of similar regions or colour separation, and document clustering based on topics or contents. Other areas include data summarization, such as image processing, vector quantization, similar trends detection in dynamic data, social network analysis of generating sequences in images, videos, or audio, and biological data analysis. Clustering could be used for anomaly detection for fraud or intrusion, a recommender system for products or services based on past purchases or performances. One of the properties of cluster states is that the points within a cluster should be similar to each other. This paper aims to minimize the distance between the points within a cluster, focusing on the k-means clustering method.

K-means is very fast, robust, and easier to understand. It works by iteratively assigning data points to the nearest cluster centroid and updating the centroid until they stabilize. It is easy to implement, can work with a large number of variables, and can quickly change clusters when the centroids are recomputed. The

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

algorithm requires a prior specification of the number of cluster centres. The random initialization step causes the k-means algorithm to be non-deterministic, meaning that cluster assignment will vary if you run the same algorithm twice on the same dataset. The algorithm might create clusters that do not quite match the actual data distribution, leading to some clusters being too small or too large compared to others. This might lead to clusters that do not accurately reflect the true structure of the data. The weakness of the k-means clustering is that we do not know how many clusters we need by just running the model. We need to test ranges of values to decide on the best value of k. The k value in the k-means algorithm is a crucial parameter that determines the number of clusters to be formed in the dataset. It is important to choose the right value of k, as a small value can result in under-clustered data, and a large value can cause over-clustering. In the k-means algorithm, the processing mode of abnormal data and the similarity calculation method will affect the clustering division.

K-means++ algorithm, an improved k-means, ensures a smarter initialization of the centroids and improves the quality of the clustering. The improved algorithm minimizes the chance of picking centres that are close to one another while avoiding the systematic pitfall of the farthest-first traversal method. Inertia method makes sure that the first property of all the data points in a cluster should be similar to each other is satisfied. Calculation of intra-cluster distance and inter-cluster distance makes sure that the second property of the data points from different clusters should be as different as possible to have more meaningful clusters. The time complexity of the k-means algorithm is O(NTK), where N is the total number of data sets, K is the total number of partitions, and T is the number of iterations in the relatively high clustering process. This paper improves the initial centre points of the k-means clustering algorithm so that the clustering result has a low fluctuation.

LITERATURE REVIEW

Many clustering techniques have been developed in different application fields, but they have not achieved a better effect due to at times datasets with different data types. Although researchers have proposed variants of the k-means algorithm to overcome these impediments of assignment of centroids and number of clusters and ability to handle various data types, they are, however domain domain-specific and do not generalize well.

[1] defined the concepts of neighbourhood coupling and separation of objects by using the upper and lower approximations of object neighbourhood and neighbourhood model. They proposed an initial clustering centre selection algorithm, which showed significant improvement. [2] proposed an improvement of the k-means++ algorithm to improve the way of the initial centroids were selected. The first centroid was selected in a random number and all the remaining centroids were picked by proportional probability to the shortest distance from all the existing centroids. [3] proposed a clustering algorithm based on maximum and minimum distance and weighted similarity calculation (MW-k-means). In the MW-k-means, the initial cluster centre is selected according to the idea of the maximum and minimum distance, and then the weighted similarity is used to divide each data point into corresponding clusters. This method avoided the clustering seeds from being close to each other in the initial value of the selection of the k-means algorithm. The MW-k-means integrates the advantages of weighted k-means (w-k-means) and maximum k-means (m-k-means) with faster convergence speed, more accurate clustering results, and higher stability, which can be used for practical clustering.

[4] described the k-means algorithm of separating data and objects into different clusters. They noted that objects with similar properties are put in one cluster and objects with different properties are put in a separate cluster. This permitted large datasets to be easily scaled with k-means clustering system learning techniques, which work in an unsupervised manner. This gave a strong guarantee of the convergence and generalized the clusters for the various shapes and structures. [5] compared many k-means algorithms with a series of experiments. Experimental results showed that the method was reasonable for the selection of the center point, and the calculation of similarity produced a better clustering effect. [6] proposed initial cluster based

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

on the relative distance of each data point from the initial centroids. These clusters were subsequently fine-tuned by using a heuristic approach.

[7] worked on the variants for solving the problem of initialization of the k-means algorithm. [8] proposed a simple and efficient clustering algorithm to solve the problem of the cluster centre point not being well-determined. The modified algorithm was built on a kd-tree data structure for the data points. The modified algorithm could avoid entering the local optimal solution. [9] presented a modified k-means algorithm based on self-paced learning theory. This method of self-paced learning theory was used to select a competing training subset that helps the k-means algorithm to build an initial cluster model. The generalization ability of the algorithm was then improved by subsequently adding training subsets found by self-paced learning until the model reached an optimal performance. The authors proposed this algorithm and demonstrated its performance on several real datasets.

[10] proposed a co-clustering algorithm for dealing with sparse and high-dimensional data that outperformed the basic k-means. [11] incorporated cuckoo search along with the k-means algorithm. The cuckoo search algorithm was modified by the authors, which helped to reach a better solution since the search step size factor was changed. The modified cuckoo search algorithm was robust enough to reach a near-optimal solution. [12] proposed an initial estimation method for computing the covariance matrix for the k-means algorithm using Mahalanobis distance. It involved finding a group of points comprised of neighbours with high density that represent the centroids of the selected clusters. These provided an approximate estimate of the covariance matrix, which was updated successively using the proposed method. [13] analysed the stability of the k-means clustering algorithm in a more practical scenario, the parameter of cluster number was chosen by stability-based model selection.

[14] reported a k-means algorithm that generated the initial centroids using pair-wise computed distances, which put all the closer points based on the threshold value for the minimum number of clusters required to build a cluster. [15] worked on k-means algorithm that is characterized by sorting all the dataset records using a table in which the sorted table was divided into multiple sub-tables that equalled the number of the desired initial centroids. [16] presented the k-means algorithm that monitored the centroids, which would not change anymore while the algorithm was still running. [17] proposed a novel k-means clustering algorithm, which reformulated the classical k-means objective function as a trace maximization problem and then replaced it with a new formulation. The proposed algorithm does not need to calculate the cluster centers in each iteration and requires fewer additional intermediate variables during the optimization process. In addition, the authors proposed an efficient iterative re-weighted algorithm to solve the involved optimization problem and provided the corresponding convergence analysis. The proposed algorithm keeps a consistent computational complexity as Lloyd's algorithm, $O(ndk)$, but shows a faster convergence rate in experiments. Extensive experimental results on real-world benchmark datasets showed the effectiveness and efficiency of the proposed algorithm.

[18] presented a process that utilizes the farthest first and canopy algorithms to reduce the number of iterations in the k-means clustering. This improvement focused on enhancing the centroid initialization method for each cluster. The research study utilized ten datasets from the UCI machine learning repository to test the effectiveness of improving the k-means clustering algorithm. It compared the number of iterations and the sum of squares error as performance metrics. The results showed that the farthest first and canopy algorithms significantly reduced the number of iterations in the k-means clustering when compared to random centroid initialization. These research findings confirmed the hypothesis that the method of centroid initialization for each cluster impacted the iteration process of k-means clustering.

Due to their popularity and ease of use, k-means clustering methods were used in conjunction with deep learning for tasks as image segmentation and handwriting recognition. A more recent work used a fully connected deep convolutional neural network along with k-means, and was used to perform pixel matching between a segmented image and a convoluted image. [19] proposed the k-means clustering algorithm based on the optimization of artificial fish swarm, which can obtain the global optimal division. This paper aims to

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

present an improvement of the initial value of the k-means algorithm for good quality of clusters' results with low processing time, where the optimal picking value of k from k-means is subjective for interpretation with the underlying structure of the dataset.

TYPE OF DISTANCE MEASURE

Several studies have suggested a wide range of distance metrics for data clustering. This section briefly describes seven widely used distance measures.

Euclidean Distance Measure

The Euclidean distance between the points **x** and **y** in two-dimensional plane is given by:

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

In case there is an extension to n dimensions and the points **x** and **y** are of the form

$$x = (x_1, x_2, x_3 + \ldots + x_n) \qquad y = (y_1, y_2, y_3 + \ldots + y_n)$$

then the Euclidean distance is given as follows:

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

Manhattan distance

The Manhattan distance, also called **taxicab** distance or **city block** distance, is another popular distance metric. The Manhattan distance between the points **x** and **y** is given by:

$$d(x, y) = |x_1 - y_1| + |x_2 - y_2|$$

In n-dimensional space, where each point has n coordinates, the Manhattan distance is given by:

$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

Though the Manhattan distance does not give the shortest distance between any two given points, it is often preferred in applications where the feature points are located in a high-dimensional space.

Minkowski distance

Minkowski distance is named after a German mathematician, Henmann Minkowski and is in a normed vector space which is given by:

$$d(x, y) = \left(\sum_{i=i}^{n} |x_i - y_i|^p\right)^{1/p} \qquad for\ p \geq 1$$

It is to note that for **p = 1**, the Minkowski distance equation takes the same form as that of Manhattan distance and when p = 2, it is equivalent to the Euclidean distance:

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

Squared Euclidean distance

The squared Euclidean distance between the point's **p** and **q** in two-dimensional plane is given by:

$$d^2(p, q) = (p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2 + ... + (p - q_i)^2 + ... + (p_n - q_n)^2$$

In case there is an extension to n dimensions at the point's **p** and **q** then the squared Euclidean distance is given as follows:

$$d^2(p, q) = \sum_{i=1}^{n} (p_i - q_i)^2$$

Eisen cosine correlation distance

Eisen cosine correlation distance is given as:

$$d_{eisen}(x, y) = 1 - \frac{\sum_{i=1}^{n} x_i \, y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2}}$$

Mahalanobis distance

Mahalanobis distance is the distance between two points in multivariate space. For uncorrelated variables, the Euclidean distance equals the Mahalanobis distance. The Mahalanobis distance quantities relative distance from the centroid, in which the centroid is the point in multivariate space where all means from all variables intersect. The Mahalanobis distance between two objects is formally defined as:

$$d_{mahalanobis} = \left[(x_B - x_A)^T * c^{-1} * (x_B - x_A) \right]^{0.5}$$

where x_B and x_A are a pair of objects and c is the sample covariance matrix. T is the transpose operation, which flips a matrix over its diagonal. A different version of the formula uses distances from each observation to the central mean, and the equation is given as follows:

$$d_i = \left[(x_i - \bar{x})^T * c^{-1} * ((x_i - \bar{x})) \right]^{0.5}$$

where x_i is an object vector and $\bar{x}$ is the arithmetic mean vector

Chebyshev distance

Chebyshev distance, also known as maximum value distance, is used to examine the absolute magnitude of the differences between the coordinates of a pair of objects and the equation is given as follows:

$$d_{Chebyshev}(p, q) = \max_i \left(\left| p_i - q_i \right| \right)$$

Properties of distance measure

i. No negative distance

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

Distances should always be non-negative. This means, it should be greater than or equal to zero.

$$d(x_i, x_j) \geq 0 \; ; \forall x_i, x_j, \in S$$

ii. Distances are positive

$$d(x_i, x_j) = 0; \quad x_i = x_j \quad \forall x_i, x_j, \in S$$

Distances are positive, except for the distance from a point to itself

iii. Distance is symmetric

If **x** and **y** are two points in a metric space, then the distance between **x** and **y** should be equal to the distance between **y** and **x**

$$d(x_i, x_j) = d(x_j, x_i) \; \forall x_i, x_j, \in S$$

iv. Triangle inequality

Given three points **x**, **y** and **z,** the distance metric should satisfy the triangle inequality

$$d(x_i, x_k) \leq d(x_i, x_j) + d(x_j, x_k) \qquad \forall x_i, x_j, x_k \in S$$

3.2 k-means and k-means++ algorithms

Let $X = [x_1, x_2, x_3, ..., x_n]$ be a set of data points and $V = [v_1, v_2, v_3, ..., v_n]$ be the set of centres.

i. randomly select 'c' cluster centre
ii. calculate the distance between each data point and the cluster centre
iii. assign the data point to the cluster centre whose distance from the cluster centre is the minimum of all the cluster centres
iv. recalculate the new cluster centre using $v_i = (\frac{1}{c_i})\sum_{j=1}^{c_i} x_i$ where c_i represent the number of data points in i^{th} cluster
v. recalculate the distance between each data point and the newly obtained cluster centres
vi. if no data point was reassigned, then stop; otherwise, repeat from step c

k-means algorithm aims at minimizing an objective function known as the squared error function, given by

$$J(v) = \sum_{i=1}^{M}\sum_{k=1}^{K} w_{ik} \left\| x_i - \mu_k \right\|^2$$

where $\| x_i - \mu_k \|$ Is the Euclidean distance between x_i and μ_k

w_{ik} = 1 for the data point x_i if it belongs to cluster k, otherwise w_{ik} = 0. It is noted that μ_k is the centroid of x_i's cluster.

This is a minimization problem of two parts.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

part(1): There is a need to minimize J(v) with respect to w_{ik} and treat μ_k fixed.

part(2): There is a need to minimize J(v) with respect to μ_k and treat w_{ik} fixed.

Technically, differentiation J(v) concerning w_{ik} first (part(1)) and update cluster assignment and differentiation of J(v) with respect to μ_k (part(2)) and recomputed the centroid after the cluster assignment was needed.

Therefore, part(1), which would assign the data point x_i to the closest cluster, judged by its sum of squared distances from the cluster's centroid, is $\frac{\partial J}{\partial w_{ik}} = \sum_{i=1}^{M}\sum_{k=1}^{K}[x_i - \mu_k\|^2$

$$w_{ik} = \begin{cases} 1 \ \ if \ \ k = \arg\min_j \left\|x_i - \mu_k\right\|^2 \\ 0 \qquad\qquad\qquad otherwise \end{cases}$$

part(2), which recomputed the centroid of each cluster to reflect the new assignment, is given as $\frac{\partial J}{\partial w_{ik}} = 2\sum_{i=1}^{M} w_{ik}(x_i - \mu_k) = 0$

where $\mu_k = \frac{\sum_{i=1}^{M} w_{ik} x^i}{\sum_{i=1}^{M} w_{ik}}$

METHODOLOGY

Selection of k in k-means or k-means++

Steps 1 and 2 of k-means or k-means++ are about choosing the number of clusters (k). These two steps are modified using the sizes of polygons as centroids and fuzzy linguistic hedges for clustering satisfactions. Table 1 illustrates the procedure for the sizes of the polygon and fuzzy linguistic hedges.

Table 1: Sizes of the polygon and fuzzy linguistic hedges

Polygon size	Vertices number	Centroid/cluster number	Fuzzy linguistic hedges
Straight line =2	2	2	(high, low)
Triangle = 3	3	3	(high, average, low)
Quadrilateral = 4	4	4	(excellence, good, fair, poor)
Pentagon = 5	5	5	(very high, high, average, low, very low)
----	---	---	---
Decagon = 10	10	10	(extremely good, very very good, very good, good, more or less good, average, more or less low, low, very low, extremely low)

i. Read the variables and visualize the points in x and y axis.

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

ii. The first cluster is chosen uniformly at the first data points of x and y axis as 'a'
iii. Using the first point, intra-cluster is computed for grouping all points together as well as monitoring the distance d(x) of each data point that has been chosen uniformly.
iv. The second cluster is chosen by computing squared distance $(d(x))^2$ as 'b', which is the farthest from the first cluster a. The first and second would form a straight line with two vertices or two centroids. Inter-cluster is computed using the first and second clusters by applying fuzzy linguistic hedges.
v. The third cluster is chosen by computing squared distance $(d(x))^2$ as 'c', which is farthest from the second cluster 'b'. The first, second, and third clusters would form a triangle with three vertices or three centroids. Inter-cluster is computed using the first, second, and third clusters by applying fuzzy linguistic hedges.
vi. The fourth cluster is chosen by computing squared distance $(d(x))^2$ as 'd' which is farthest from the third cluster 'c'. The first, second, third, and fourth clusters would form a quadrilateral with four vertices or four centroids. Inter-cluster is computed using the first, second, third, and fourth clusters by applying fuzzy linguistic hedges.
vii. The fifth cluster is chosen by computing squared distance $(d(x))^2$ as 'e', which is farthest from the fourth cluster 'd'. The first, second, third, fourth, and fifth clusters would form a pentagon with five vertices or five centroids. Inter-cluster is computed using the first, second, third, fourth, and fifth clusters by applying fuzzy linguistic hedges. This process would continue for other polygon sizes of hexagon, heptagon, octagon, nonagon, decagon, hendecagon, etc, with their respective fuzzy linguistic hedges.

Fuzzy linguistic hedges

Let the centroid be the universe of discourse, C = [0, 10]. Then, the fuzzy sets 'high' and 'low' may subjectively be defined as follows:

$$f_{low}(C) = \begin{cases} 1 & 0 \prec c \le 5, \\ (1 + ((c-5)/1.5)^2)^{-1}, & 5 \prec c \le 10, \end{cases}$$

$$f_{high}(C) = \begin{cases} 0 & 0 \prec c \le 5, \\ (1 + ((c-5)/1)^{-2})^{-1}, & 5 \prec c \le 10, \end{cases}$$

where f_{low} and f_{high} are the membership functions of the fuzzy sets "low" and "high" respectively. This translates into high and low centroids of certain data points.

These two fuzzy linguistic hedges of high and low could be extended into three as low, average, and high centroids of other certain data points. The same is also true for four fuzzy linguistic hedges of poor, fair, good, and excellent, centroids of data points after visualization. For instance, fuzzy linguistic hedges of five centroids could be constructed as matching techniques. Let F_i and F_j be two different fuzzy points represented by fuzzy sets of the universe of discourse C, respectively, and let VH (very high), H (high), A (average), L (low), and VL (very low) be standard fuzzy sets of the universe of discourse of C, where

$C = \{0\%, 2\%, 4\%, 6\%, 8\%, 10\%\}$ and the corresponding awarded degrees of the standard fuzzy sets are as follows:

$very\,high\ (VH) = \{(0\%, 0.0), (2\%, 0.0), (4\%, 0.8), (6\%, 0.9), (8\%, 0.9), (10\%, 1.0)\}$

$high\ (H) = \{(0\%, 0.0), (2\%, 0.0), (4\%, 0.8), (6\%, 0.9), (8\%, 0.9), (10\%, 0.8)\}$

$average\ (A) = \{(0\%, 0.0), (2\%, 0.1), (4\%, 0.8), (6\%, 0.9), (8\%, 0.4), (10\%, 0.2)\}$

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

$low(L) = \{(0\%, 0.4), (2\%, 0.4), (4\%, 0.6), (6\%, 0.6), (8\%, 0.2), (10\%, 0.0)\}$

$very\,low(VL) = \{(0\%, 1.0), (2\%, 1.0), (4\%, 0.4), (6\%, 0.2), (8\%, 0.0), (10\%, 0.0)\}$

Based on the vector representative method, the fuzzy sets VH, H, A, L and VL could be represented as follows: $\bar{VH}$, $\bar{H}$, $\bar{A}$, $\bar{L}$, and $\bar{VL}$

where

$\bar{VH} = \langle 0.0, 0.0, 0.8, 0.9, 0.9, 1.0 \rangle$

$\bar{H} = \langle 0.0, 0.0, 0.8, 0.9, 0.9, 0.8 \rangle$

$\bar{A} = \langle 0.0, 0.1, 0.8, 0.9, 0.4, 0.2 \rangle$

$\bar{L} = \langle 0.4, 0.4, 0.6, 0.6, 0.2, 0.0 \rangle$

$\bar{VL} = \langle 1.0, 1.0, 0.4, 0.2, 0.0, 0.0 \rangle$

For example, a company with variable income data points could be grouped into five as very high income, high income, average income, low income, and very low income. This is applied to polygon size six (hexagon), size seven (heptagon), size eight (octagon), size nine (nonagon) and etc,. The satisfaction level of fuzzy linguistic hedges for a polygon of size ten (decagon) is illustrated in Table 2.

Table 2: Satisfaction level of fuzzy linguistic hedges for decagon

Fuzzy linguistic hedges	Degrees of clustering
extremely good income	90%-100% (0.90 – 1.00)
very good income	80% - 89% (0.80 – 0.89)
good income	70% - 79% (0.70 – 0.79)
more or less good income	60% - 69% (0.60 – 0.69)
average income	50% - 59% (0.50 – 0.59)
more or less Low income	40% - 49% (0.40 – 0.49)
low income	30% - 39% (0.30 – 0.39)
very low income	20% - 29% (0.20 – 0.29)
very very low income	10% - 19% (0.10 – 0.19)
extremely low income	0% - 9% (0.0 – 0.9)

This would be translated into ten groups of certain data points of a variable in visualization mode and could be applied to other polygon sizes, eleven (hendecagon), and so on.

After all data points have been assigned to clusters based on the fuzzy linguistic hedges using degrees of satisfaction, recalculate the centroids of the clusters by taking weighted means, variation, co-variation, and standard deviation of all data points assigned to each cluster.

Repeat steps e and f of the above k-means or k-means++ algorithm in section 4 until convergence occurs when the centroids are no longer changed significantly. Once the convergence is reached, the algorithm produces the final cluster centroids. These sizes of polygons with their fuzzy linguistic hedges could be

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

computed as built-in functions and called upon during the applications of the k-means clustering algorithm. The time complexity of the k-means algorithm, O(NTK), where N is the total number of data sets, K is the total number of partitions, and T is the number of iterations in the clustering process, would be drastically reduced as k is now O(1). In this case, k would now represent the calling in-built functions that would reduce the time complexity greatly.

COLLECTION OF DATASETS

The datasets used for this research work are health analytics dataset that was downloaded from https://www.kaggle.com/jinalgada/starter-health-analytics-b8146c6b-2 and rain in Australia dataset that fetched from https://www.kaggle.com/jsphyg/weather-dataset-rattle-package. The health analytics dataset has 284 rows and 644 columns of data, and the rain in Australia dataset also has 145461 rows and 23 columns of data

Centering

Centering as a data preprocessing method involves moving the coordinate system within the dataset to a new reference point, which is usually the origin of the coordinate system in a k-dimensional space. Proper centering removes the postulated offsets without changing the structural model of the data. When it is performed, interval-scale data behave as though they are ratio-scale data. The weighted median centre which allows to specify a weight field as the number of trips associated with each feature is employed. This permits to locate Euclidean distance to all weighted features in the dataset. The median centre is calculated as $d_i^t = \sqrt{(X_i - X^t)^2 + (Y_i - Y^t)^2 + (Z_i - Z^t)^2}$. The median center only returns a single point, although there may be more than one location (solution) that would minimize the distance to all features.

Principal Component Analysis

Principal Component Analysis (PCA) employed comprises of five stages as follows:

i. Subtract the mean of each variable.

Before the PCA, the standardization was performed which transformed all the variables to the same scale. This is done through $z = \frac{value - mean}{s\tan dard\ deviaton}$

ii. Calculate the Covariance Matrix

The covariance matrix was computed to remove redundant information and make sure that all the possible pairs of variables are correlated. For example, a 4-dimenssional data set of 4 variables x, y, z, w of 4 X 4 data matrix is as follows: $\begin{bmatrix} \mathrm{cov}(x,x), \mathrm{cov}(x,y), & \mathrm{cov}(x,z), \mathrm{cov}(x,w) \\ \mathrm{cov}(y,x), \mathrm{cov}(y,y), & \mathrm{cov}(y,z), \mathrm{cov}(y,w) \\ \mathrm{cov}(z,x), & \mathrm{cov}(z,y), & \mathrm{cov}(z,z), \mathrm{cov}(z,w) \\ \mathrm{cov}(w,x), \mathrm{cov}(w,y), & \mathrm{cov}(w,z), \mathrm{cov}(w,w) \end{bmatrix}$

iii. Compute the Eigenvalues and Eigenvectors

The eigenvector and eigenvalue are the linear algebra concepts to determine the principal components of the data. Every eigenvector has an eigenvalue which means they are pairs. The eigenvalues are simply the coefficients attached to eigenvectors, which give the *amount of variance carried in each principal component.*

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

iv. Sort Eigenvectors by corresponding Eigenvalues in descending order and select a subset from the rearranged Eigenvalue matrix

v. Recast data along the principal components

The final data set was done by multiplying the transpose of the original data set by the transpose of the feature vector as follows: $final\_data\_set = features\_vector^T * standardized\_original\_dataset^T$

After computing the eigenvectors and ordering them by their eigenvalues in descending order, allow us to find the principal components in order of significance

Outlier removal

Interquartile Range (IQR), a statistical dispersion representing the middle 50% of a dataset is used. It calculates by finding the difference between the 75th percentile (Q3) and the 25th percentile (Q1). The IQR method identifies outliers by setting boundaries based on Q1 and Q3: Lower Bound = Q1 - (1.5 * IQR) and Upper Bound = Q3 + (1.5 * IQR). Any data point outside these bounds is considered an outlier.

Applications of weighted Euclidean distances

Standardized Euclidean distance

The standardized Euclidean distance between two J-dimensional vectors can be written as

$$d(x, y) = \sqrt{\sum_{j=1}^{J} (\frac{x_j}{s_j} - \frac{y_j}{s_j})^2}$$

Where s_j is the sample of the standard deviation of the j-th variable

This equation could also be expressed as follows: $d(x, y) = \sqrt{\sum_{j=1}^{J} \frac{1}{s_j} (x_j - y_j)^2}$

$$= \sqrt{\sum_{j=1}^{J} w_j (x_j - y_j)^2}$$

Where $w_j = \frac{1}{s_j}$ is the inverse of the j-th variance

This w_j is used as a weight attached to the j-th variable.

There is compensatory effect produced by standardization of the weight of the variable. This is done by attaching weight of means, variation and co-variance.

Experiment and results

To verify the effectiveness of the sizes of polygons with their corresponding fuzzy linguistic hedge satisfaction, two sets of data, as mentioned in section 5, were used as experimental data sets alongside the following methods:

i. Euclidean distance with regular selection of the value of k
ii. Euclidean distance with random selection of the value of k

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

iii. Minkowski distance measure
iv. Weighted Euclidean distance using standard deviation
v. Weighted Euclidean distance using means
vi. Weighted Euclidean distance using variance
vii. Weighted Euclidean distance using co-variance

Euclidean distance will work fine as long as the dimensions are equally weighted and are independent of each other. This can technically be overcome by scaling the variables of computing the standard deviation, covariance, which can make it vary within a particular range between 0 and 1.

The experiments were performed many times following the modification of the initialization of the k value in the k-means using the sizes of polygons with their fuzzy linguistic hedges. The functions of the sizes of polygons with their corresponding fuzzy linguistic hedges have been computed and invoked to visualize the fitness of the data points. The experimental results for each run, one, two, and three, for the health analytics dataset with their respective methods are shown in Table 3. The average clustering processing time for each method was calculated as indicated in Table 3. The same process was also done with the Australia rain dataset, which is illustrated in Table 4. The fastest clustering time for each method was illustrated in Table 5 using the health analytics dataset. Information retrieval times using both health analytics and Australia rain datasets with various search terms were done. The results of information retrieval times for the two datasets and their methods were illustrated in Tables 6 and 7. Table 8 illustrates the ranking of the methods based on the retrieval time of the search item using the Australia rain dataset. As depicted in tables 5 and 8, weighted Euclidean distance using standard deviation determined the fastest clustering time, following by weighted Euclidean distance using co-variance, weighted Euclidean distance using variance, Minkowski distance measure, weighted Euclidean distance using means, Euclidean distance with random selection of the value of k and Euclidean distance with regular selection of the value of k while Minkowski distance measure determined the fastest retrieval time followed by Euclidean distance with regular selection of the value of k, weighted Euclidean distance using co-variance, Euclidean distance with random selection of the value of k, weighted Euclidean distance using variance, weighted Euclidean distance using standard deviation and weighted Euclidean distance using means.

The results revealed that the data clustered with the Minkowski method in place of Euclidean distance, returned the searched item in the least time.

The results in each of the scenarios reveal that clustering performed with weighted Euclidean distance using standard deviation consistently took the least time, making it the fastest method of getting experimental data sets clustered with the k-means clustering algorithm.

Table 3: Clustering time with the “Health Analytics” Dataset

1^{ST} RUN							
	Euclidean Distance		Minkowski	Weighted Euclidean Distance			
	Regular selection	Random Selection	Minkowski	Standard Deviation	Means	Variance	Co-variance
Time taken (ms)	676	513	509	503	508	509	504
	503	505	503	503	503	503	504
	502	501	502	502	503	501	502
	504	501	502	502	502	502	501
AVERAGE	546.25	505	504	502.5	504	503.75	502.75
2^{ND} RUN							
Time taken (ms)	553	504	504	502	507	504	502
	501	502	502	501	501	501	505

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

	501	502	503	502	501	502	502
	501	501	502	501	502	502	502
AVERAGE	514	502.25	502.75	501.5	502.75	502.25	502.75
3RD RUN							
Time taken (ms)	516	505	504	501	506	505	501
	504	502	501	502	502	502	502
	502	502	502	502	501	501	502
	503	501	501	501	501	501	501
AVERAGE	506.25	502.5	502	501.5	502.5	502.25	501.25
Total Average (ms)	544.58	506	507.6	503.4	506	505.9	505.5

Table 4: Clustering time with the "Rain in Australia" Dataset

IST RUN							
	Euclidean Distance		Minkowski	Weighted Euclidean Distance			
	Regular selection	Random Selection	Minkowski	Standard Deviation	Means	Variance	Co-variance
Time taken (ms)	900	538	510	503	509	510	521
	503	503	507	503	506	503	505
	503	503	508	503	503	503	503
	511	504	505	503	503	503	503
AVERAGE	604.25	512	507.5	503	505.25	504.75	508.75
2ND RUN							
Time taken (ms)	566	509	510	505	514	513	504
	505	506	503	503	505	503	504
	503	503	505	505	503	505	503
	504	505	503	503	504	505	504
AVERAGE	519.5	505.75	505.25	504	506.5	506.5	503.75
3RD RUN							
Time taken (ms)	531	509	508	504	514	515	503
	503	506	503	503	503	504	505
	503	503	506	503	503	504	504
	503	503	504	503	506	503	504
AVERAGE	510	505.25	505.25	503.25	506.5	506.5	504
Total Average	522.17	502.9	503.2	501.8	503.1	502.75	502.3

Table 5: Ranking of the methods to determine clustering time in health analytics dataset

	Euclidean Distance	Minkowski	Random Selection	Standard Deviation	Means	Variance	Co-variance
GRAND Average (ms)	553.4	504.5	505.4	502.6	504.6	504.3	503.9
Rank	7th	4th	6th	1st	5th	3rd	2nd

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

Table 6 : Information Retrieval time with the "Health Analytics" Dataset using different search terms

	Euclidean Distance(ms)		Minkowski (ms)		Weighted Euclidean Distance (ms)		
Search term	Regular selection	Random Selection	Minkowski	Standard Deviation	Means	Variance	Co-variance
18501	500	502	500	501	502	502	502
	500	504	500	501	501	501	503
	500	502	501	502	502	502	502
AVERAGE	500.75	502.25	501	501.75	501.75	501.75	502.25

	Euclidean Distance(ms)		Minkowski (ms)	Weighted Euclidean Distance (ms)				
Search term	Euclidean Distance	Random Selection	Minkowski	Standard Deviation		Means	Variance	Co-variance
	500	502	501	502		501	502	501
	501	501	500	501		501	504	502
	501	502	500	501		502	503	502
5769	501	502	500	502		502	503	502
AVERAGE	500.75	501.75	500.25	501.5		501.5	503	501.75
Total Average	500.75	502	500.625	501.625	501.625		502.375	502

Table 7: Information Retrieval time with the "Rain in Australia" Dataset using different search terms

	Euclidean Distance(ms)		Minkowski (ms)	Weighted Euclidean Distance (ms)			
Search term	Euclidean Distance	Random Selection	Minkowski	Standard Deviation	Means	Variance	Co-variance
	504	502	503	502	502	502	
	500	502	500	502	502	502	
	501	502	501	533	533	502	
19.8	500	502	500	501	501	502	
AVERAGE	501.5	502	501	509.5	509.5	502	

	Euclidean Distance(ms)		Minkowski (ms)	Weighted Euclidean Distance (ms)			
Search term	Euclidean Distance	Random Selection	Minkowski	Standard Deviation	Means	Variance	Co-variance
	500	502	501	501	502	502	502
	501	502	500	502	502	501	503
	501	504	500	501	502	502	502
21.5	500	502	501	502	502	502	502
AVERAGE	500.5	502.5	500.5	501.5	502	501.75	502.25
Total Average	501	502.25	500.75	505.5	505.75	501.875	501.125

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

Table 8: Ranking of the methods to determine retrieval time in Australia rain datasets

	Euclidean Distance	Random selection	Minkowski	Standard Deviation	Means	Variance	Co-variance
GRAND Average (ms)	500.875	502.125	500.688	503.563	503.688	505.125	502.063
Rank	2nd	4th	1st	6th	7th	5th	3rd

The Stand-Alone Classic k-means algorithm (SaCKmeans) used in the work of Lu, W. (2020) was selected for the benchmark with the weighted Euclidean distance using standard deviation (SDKmeans) since it produced the best clustering result among all the others. The SDKmeans took a shorter time to cluster the dataset in each of the 3 runs as indicated in table 9. Hence, it could be conclusively verified that the SDKmeans outperformed the SaCKmeans in terms of the time taken to cluster the two experimental data sets

Table 9 : Comparison of SDKmeans with SaCKmeans

Runs	SDKMeans	SaCKmeans
	Time taken (ms)	
	529.2	524.9
	504.1	522.1
1	504.4	521
	504.5	530
AVERAGE	510.6	524.5
	504.1	522.1
	503	521.6
2	503.3	522.2
	502.3	521.6
AVERAGE	503.2	521.9
	503.9	521.8
	503.6	522
3	504	521.9
	502.8	522.4
AVERAGE	503.6	522
TOTAL AVERAGE	505.8	522.8

CONCLUSION

The choice of the first point along the first x and y axis does not permit the value of the first centroid to vary in each run time of the experiments. The values of k in the k-means or k-means++ were strictly adjusted to the sizes of the polygon with their corresponding fuzzy linguistic hedges during the running times of the experiments. Every time the experiment was conducted, the number of clusters generated was equally the same with the sizes of the polygon alongside the satisfaction degrees of fuzzy linguistic hedges. The combination of sizes polygon, and fuzzy linguistic hedge handled millions of data points in a matter of seconds and allowed the k-means algorithm to converge rapidly.

REFERENCES

1. Johannes Schneider and Michail Vlachos, Fast parameter less density-based clustering via random projections , Association for computing machinery, New York, ACM press, pp 861-866, 2013

ISSN No. 2321-2705 | DOI: 10.51244/IJRSI |Volume XII Issue IV April 2025

2. Ioan-Daniel Borka, Radu-Emil Precup and Alexandra-Bianca Borlea, Improvement of k-means cluster quality by post processing resulted clusters, Precedia. Computer science , 199, pp 63 -70, 2022
3. YanPing Zhao and XiaoLai Zhou, K-means clustering algorithm and its improvement research, Journal of Physics: Conference series, vol. 1873, 2021 2nd International workshop on electronic communication and artificial intelligence (IWECAI 2021), 12-14 March 2021, Nanjing, China
4. Manish Suyal and Sanjay Sharma, A review of analysis of k-means clustering machine learning algorithm based on unsupervised learning, Journal of artificial intelligence and system, 6 85 – 95, 2024
5. Mohiuddin Ahmed, Raihan Seraj and Syed Mohanned Shamsul Islam, The k-means algorithm: A comprehensive survey and performance evaluation, Journal of Electronics 9(1295) pp 1-12, 2020, doi:10.3390/electronics9081295
6. Fahim A. M., Salem A. M., Turkey F. A., and Ramadam M. A., An efficient enhanced k-means clustering algorithm, Journal of Zhejiang University, China, 10(7), 1626 – 1633, 2006
7. Rai P., and Singh S., A survey of clustering techniques, Int. computing appl, 7, 1-5, 2010
8. Kanungo T., Mount D. M., Netenyahu N. S., Piatko C. D., and Silverman R., An efficient k-means clustering algorithm: analysis and implementation, IEEE trans. Pattern and marching intell. 24, 870 – 892, 2002
9. Yu H., Wen G., Gan J., Zheng W., and Lei C., Self-paced learning for k-means clustering algorithm, Journal of pattern recognition, 2018
10. Hussain S. F. and Harris M. A., K-means based co-clustering algorithm for sparse, high dimensional data, Expert system, 118, 20-34, 2019
11. Ye S., Huang X., Teng Y., and Li Y., k-means clustering algorithm based on improved cuckoo search algorithm and its application, In proceedings of the 2018 IEEE 3rd international conference on big data analysis, Shanghai, China, 9 – 12, March 2018, pg 422 – 426
12. Melnykov I., and Melnykov V., on k-means algorithm with the use of Mahalanobis distance, Stat. probab letter, 84, pp 88 – 95, 2014
13. Bubeck S., Meila M., and Von Luxburg U., How the initialization affects the stability of the k-means algorithm, ESAIM probab. Stat 16, pp 436 – 452, 2012
14. Nasser S, Alkhaldi R, Vert G. A modified fuzzy K-means clustering using expectation maximization. In Proc 2006 IEEE Int Conf Fuzzy Syst, Vancouver, BC, Canada, p. 231–235, 2006
15. Nazeer KAA, Kumar SDM, Sebastian MP. Enhancing the K-means clustering algorithm by using a O(n logn) heuristic method for finding better initial centroids. In Proc 2011 2nd Int Conf Emerg Appl Inf Technol, Washington, DC, USA; 261–264, 2011.
16. Kaukoranta T, Franti P, Nevalainen O. A fast exact GLA based on code vector activity detection. IEEE Trans Image Process
17. Ziheng Li, Rong Wang and Xuelong Li, An effective and efficient algorithm for k'means clustering with new formulation, IEEE transaction on knowledge and data engineering vol 75 isuue 4, 2023
18. Narongsak Chayangkoon Chatchai Kasemtaweechok, Improving the performance of k-mean clustering to reduce work by addressing the gap in finding, International computer Science and Engineering, 2023 27th (ICEC)
19. Das S., Abraham A., and Koner A., Automatic clustering using an improved differential evaluation algorithm, IEEE transaction on system, Man cybernetics-part system humans, 38(1), pp 218-237, 2008
20. Lu, W. Improved k-means clustering algorithm for big data mining under hadoop parallel framework, Journal of grid computing 18, pp 239 – 250, 2020