https://repository.uaeh.edu.mx/revistas/index.php/investigium/issue/archive

Boletín Científico INVESTIGIUM de la Escuela Superior de Tizayuca

InvESTigium

Publicación semestral, Vol. 11, No. 22 (2026) 25-32

ISSN: 2448-4830

Reducción de cálculos de distancia en K-means mediante inicialización optimizada de centroides en una estructura jerárquica tipo árbol

Reducing distance calculations in K-means by optimized centroid initialization in a hierarchical tree-like structure

Alicia Martínez-Rebollar[a], Luis Neri-Martínez[b], Dulce Estrada-Bahena[c], Hugo Estrada-Esquivel[d.]

Abstract:

The clustering problem is key in areas such as data mining, machine learning, knowledge discovery, and pattern recognition. K-means is one of the most widely used algorithms due to its simplicity and easy implementation. However, it is computationally expensive, especially in environments with large volumes of data, such as those posed by the Big Data paradigm. This research proposes an improvement in the K-means initialization phase by using a hierarchical tree-like structure, which allows the selection of centroids with optimal representativeness. This aims to reduce both distance calculations and centroid updates, achieving greater efficiency in algorithm execution. The proposal was evaluated using real-world instances recognized in the literature and large synthetic sets. The results show that the improvement maintains or increases clustering quality while significantly reducing computational costs. Highlights include: on the synthetic instance IS5 (non-uniform), a 4.53% improvement in quality, 98.81% less time, and 33.33% fewer iterations; on the IS2 (uniform), a 0.0059% improvement in quality, 99.99% less time, and 94.44% fewer iterations; and on the real instance Skin, a 5.40% improvement in quality, 99.97% less time, and 38.46% fewer iterations.

Keywords:

K-means algorithm, Big Data, Kd-tree structure, Efficient clustering, Computational optimization

Resumen:

El problema de agrupamiento es clave en áreas como minería de datos, aprendizaje automático, descubrimiento de conocimiento y reconocimiento de patrones. K-means es uno de los algoritmos más utilizados debido a su simplicidad y fácil implementación. No obstante, presenta un alto costo computacional, especialmente en entornos con grandes volúmenes de datos, como los planteados por el paradigma Big Data. Esta investigación propone una mejora en la fase de inicialización de K-means mediante el uso de una estructura jerárquica tipo árbol, la cual permite seleccionar centroides con características de representatividad óptima. Con ello, se busca reducir tanto los cálculos de distancia como las actualizaciones de centroides, logrando una mayor eficiencia en la ejecución del algoritmo. La propuesta fue evaluada con instancias reales reconocidas en la literatura y conjuntos sintéticos de gran tamaño. Los resultados muestran que la mejora mantiene o incrementa la calidad del agrupamiento, al tiempo que reduce significativamente los costos computacionales. Destacan: en la instancia sintética IS5 (no uniforme), una mejora del 4.53% en calidad, 98.81% menos tiempo y 33.33% menos iteraciones; en IS2 (uniforme), una mejora del 0.0059% en calidad, 99.99% menos tiempo y 94.44% menos iteraciones; y en la instancia real Skin, una mejora del 5.40% en calidad, 99.97% menos tiempo y 38.46% menos iteraciones.

***Palabras Clave**:*

Algoritmo K-means, Big Data, Estructura Kd-tree, Agrupamiento eficiente, Optimización computacional

1. Introducción

El problema de agrupamiento es fundamental en múltiples disciplinas como la minería de datos, el aprendizaje automático, el descubrimiento de conocimiento, la compresión de datos y el reconocimiento de patrones [1] – [3]. Entre los algoritmos más utilizados para esta tarea se encuentra K-means, ampliamente valorado por su simplicidad y eficiencia en conjuntos de datos de tamaño moderado [4], [5]. Sin embargo, cuando se aplica en escenarios caracterizados por el paradigma Big Data, enfrenta limitaciones importantes debido a su complejidad

[a] Autor de Correspondencia, TECNM | CENIDET | Morelos | México, https://orcid.org/0000-0002-1071-8599, Email: alicia.mr@cenidet.tecnm.mx; [b] https://orcid.org/0009-0000-1468-5758 Email: luisneri16c@cenidet.edu.mx; [c] https://orcid.org/0009-0000-0753-146X email: d23ce166@cenidet.tecnm.mx; [d] https://orcid.org/0000-0002-1466-7581, Email: hugo.ee @cenidet.tecnm.mx

Fecha de recepción: 04/07/2025, Fecha de aceptación: 22/09/2025, Fecha de publicación: 05/01/2026

DOI: https://doi.org/10.29057/est.v11i22.15448

computacional, donde el crecimiento de las instancias o dimensiones impacta directamente el tiempo de ejecución [6], [7]. El algoritmo K-means está compuesto por cuatro etapas: inicialización, clasificación, cálculo de centroides y evaluación de convergencia [8]. La etapa de inicialización es crítica, ya que una mala selección de centroides puede afectar significativamente tanto la calidad del agrupamiento como la eficiencia del proceso [9] – [11].

En el CENIDET, desde 2005 se ha desarrollado una línea de investigación centrada en la mejora de K-means, incluyendo nuevas condiciones de convergencia [12], heurísticas inspiradas en estructuras geométricas [13] y metaheurísticas para reducción de complejidad [14]. Sin embargo, estas propuestas no han abordado el uso de estructuras de datos jerárquicas tipo árbol, ni han sido evaluadas en contextos estrictos de Big Data.

Esta investigación propone una mejora en la etapa de inicialización de K-means, mediante el uso de estructuras jerárquicas como Kd-tree [15] y técnicas de búsqueda del vecino más cercano [16], con el fin de reducir los cálculos de distancia y mejorar la eficiencia computacional. Este enfoque responde a desafíos contemporáneos identificados por investigaciones recientes sobre escalabilidad en algoritmos de agrupamiento [18] – [20].

2. Trabajos relacionados

El algoritmo K-means ha sido ampliamente estudiado y aplicado debido a su simplicidad y eficiencia en tareas de agrupamiento. Sin embargo, en contextos de grandes volúmenes de datos, sus limitaciones computacionales han motivado múltiples líneas de investigación orientadas a mejorar su rendimiento, ya sea a través de optimizaciones secuenciales, paralelas o mediante estructuras de datos especializadas.

Una de las estrategias más exploradas ha sido la optimización de la fase de inicialización. En este sentido, el trabajo de KD-means propone una variante que utiliza estructuras de datos tipo kd-tree para organizar jerárquicamente la información y estimar automáticamente el número de clusters. A través de una fusión recursiva y regularización, el algoritmo busca capturar “clusters naturales” incluso en conjuntos de datos masivos con formas no esféricas o superposición entre clases, superando a x-means y g-means en precisión y eficiencia [18].

Desde el enfoque del paralelismo, se han desarrollado propuestas como parallel batch k-means, el cual divide el conjunto de datos en particiones procesadas en paralelo y posteriormente fusionadas, logrando una significativa reducción en el tiempo de ejecución. Este tipo de aproximación es útil para escenarios de Big Data, donde la escalabilidad es prioritaria, aunque en algunos casos pueda implicar una ligera pérdida de precisión en la función objetivo [19].

Otras propuestas exploran la combinación de clustering con algoritmos metaheurísticos, como es el caso del ICF (Improved Collaborative Filtering), que integra K-means con el algoritmo de optimización por enjambre de partículas (PSO) para mejorar la precisión en sistemas de recomendación, al reducir la dimensionalidad del espacio de ítems y refinar la similitud entre usuarios [3].

Finalmente, desde una perspectiva arquitectónica, se ha propuesto el uso de estructuras Kd-tree integradas con OpenMP para construir una versión paralela del K-means tradicional que logra balanceo de carga dinámico, aceleración del cálculo de distancias y aprovechamiento eficiente de los núcleos disponibles en arquitecturas multi-core. Esta combinación ha mostrado mejoras sustanciales en tiempo de ejecución en datasets como MNIST y Bag-of-Words, sin afectar significativamente la calidad del clustering [17].

En conjunto, estos trabajos reflejan la tendencia actual hacia la hibridación de K-means con técnicas de paralelización, heurísticas, y estructuras jerárquicas, con el objetivo común de escalar el algoritmo a contextos de Big Data. Sin embargo, se observa que la mayoría de estas aproximaciones se centran en mejoras específicas sin explorar de forma conjunta la inicialización eficiente con estructuras tipo árbol y su aplicación directa al procesamiento masivo. La propuesta presentada en este artículo aborda esta brecha, al implementar una mejora centrada en la selección jerárquica de centroides mediante una estructura de datos tipo árbol para reducir cálculos de distancia y acelerar el proceso de agrupamiento sin comprometer calidad.

3. Fundamentos teóricos

La mejora propuesta en este trabajo se fundamenta en tres conceptos clave: el algoritmo de agrupamiento K-means, las estructuras de datos tipo Kd-tree y el método de búsqueda del vecino más cercano. A partir del análisis de estos elementos se define la heurística denominada CentrosKDT-means, cuyo objetivo es optimizar la etapa de inicialización del algoritmo K-means mediante una estructura jerárquica balanceada que permita reducir el número de cálculos de distancia y centroides.

3.1. Algoritmo K-means

K-means es uno de los algoritmos de agrupamiento más conocidos por su simplicidad y eficiencia en conjuntos de datos moderados. Fue introducido inicialmente por Lloyd (1957) y formalizado por MacQueen (1967). Se trata de un método no supervisado que agrupa n objetos en k clústeres, minimizando la distancia euclidiana entre los objetos y los centroides de sus grupos respectivos [1], [4].

Su funcionamiento se basa en cuatro etapas principales: inicialización de centroides, asignación de objetos al grupo más cercano (clasificación), recalculo de centroides y evaluación de la convergencia del proceso. Uno de los principales desafíos de K-means radica en su alta dependencia de los centroides iniciales, lo que puede conducir a soluciones subóptimas o a una alta carga computacional [5], [9].

3.2. Estructura de datos Kd-tree

El Kd-tree (k-dimensional tree), introducido por Bentley (1975) y ampliado por Friedman et al. (1977), es una estructura de datos binaria utilizada para dividir un espacio multidimensional de manera jerárquica. Organiza los objetos mediante hiperplanos perpendiculares, formando una partición eficiente del espacio. Su principal ventaja es que permite acelerar búsquedas como la del vecino más cercano, al reducir el número de comparaciones requeridas [20]. En esta investigación se toma como base la implementación de Kanungo et al. [6], conocida como Filtering Algorithm (FA), donde los objetos se almacenan en un Kd-tree y se filtran iterativamente para identificar los centroides más representativos.

3.3. Búsqueda del vecino más cercano (k-NN)

La búsqueda del vecino más cercano es una técnica clásica para encontrar el punto más cercano a una consulta q en un conjunto S, en términos de distancia métrica. Su eficiencia mejora significativamente cuando se implementa junto con estructuras como el Kd-tree [15], [16]. Esta estrategia se emplea en esta propuesta para estimar los centroides iniciales durante la fase de construcción jerárquica del árbol.

3.4. Heurística CentrosKDT-means

La integración de los conceptos anteriores da lugar a la heurística CentrosKDT-means, que propone una selección sistemática y estructurada de los centroides iniciales mediante la construcción de un Kd-tree balanceado. Los objetos se insertan recursivamente en función de sus coordenadas y nivel en el árbol, seguido de un recorrido inorden que permite obtener una representación más uniforme del espacio.

Posteriormente, se aplica un algoritmo de balanceo binario y una preclasificación basada en proximidad para definir los centroides iniciales. La heurística continúa con las etapas tradicionales de clasificación, cálculo de centroides y convergencia, reduciendo de forma significativa los cálculos innecesarios y mejorando el desempeño general del algoritmo en entornos de gran escala y multidimensionalidad [7], [12], [19].

4. Heurística CentrosKDT-means

La heurística CentrosKDT-means, es una propuesta diseñada para mejorar la etapa de inicialización del algoritmo K-means mediante una estructura jerárquica de tipo Kd-tree. Este enfoque busca reducir el número de cálculos de distancia y la cantidad de iteraciones necesarias para la convergencia, especialmente en conjuntos de datos de gran volumen. La propuesta integra conceptos del nearest neighbor y técnicas de balanceo de árboles, orientadas a seleccionar centroides iniciales representativos.

4.1. Construcción del árbol jerárquico

El primer paso consiste en insertar los objetos en una estructura Kd-tree. La inserción se realiza recursivamente, alternando las dimensiones en función del nivel del árbol. Cada nodo se compara con el nodo raíz en una dimensión específica, y se ubica a la izquierda o derecha según su valor.

Este proceso se ilustra paso a paso en las Figuras 6 a 15, mostrando la inserción de objetos en un plano 2D. Luego, se realiza un recorrido inorden para generar una secuencia ordenada de los objetos, como se muestra en la Figura 16. Esta secuencia permite construir un árbol balanceado, utilizando un algoritmo de búsqueda binaria para distribuir los nodos de manera equitativa (Figura 17). El resultado es una estructura optimizada para extraer centroides iniciales de forma eficiente.

4.2. Propuesta de centroides iniciales

Una vez construido el árbol balanceado, se divide el conjunto ordenado de objetos en k grupos, en función del número de clústeres requeridos. Para cada grupo, se selecciona como centroide inicial el objeto más cercano al centro geométrico de su segmento, utilizando el método del vecino más cercano.

Este enfoque elimina la aleatoriedad de la inicialización tradicional de K-means, favoreciendo una mejor cobertura del espacio de datos y reduciendo el número de iteraciones necesarias para alcanzar una solución estable.

4.3. Integración con el algoritmo K-means

La ejecución del algoritmo K-means, posterior a la inicialización con CentrosKDT-means, sigue las etapas clásicas:

1.Clasificación: cada objeto x se asigna al centroide m más cercano, calculando la distancia euclidiana:

$$d(x,m) = \left[\sum_{j=1}^{d}(x_j - m_j)^2\right]^{\frac{1}{2}} \qquad (1)$$

2.Cálculo de centroides: para cada clúster, el nuevo centroide se define como el promedio de los objetos asignados.

3.Convergencia: el proceso se detiene si los centroides no cambian entre iteraciones consecutivas o si se alcanza un número máximo de iteraciones.
La calidad del agrupamiento se evalúa mediante la suma del error al cuadrado (SSE):

$$SSE = \sum_{j=1}^{k} \sum_{x_q \in c_l} d(x_q, m_j) \qquad (2)$$

Donde c_j representa cada clúster y m_j su centroide.

A continuación, se muestra el Algoritmo 1 que describe los pasos de la heurística CentrosKDT-means:

Algoritmo 1 Heurística CentrosKDT-means

Entrada: número de objetos a, número de clusters k, Kd-tree K, conjunto de datos D, dimensión d
Salida: asignaciones de grupo C, centroides finales M

```
t ← 0
Inicializar centroides M con Kd-tree K
repeat
    for i = 1 hasta a do
        distancia_min ← ∞
        for j = 1 hasta k do
            Calcular distancia ← distancia(x_i, m_j)
            if distancia < distancia_min then
                distancia_min ← distancia
                C_i ← j
            end if
        end for
        Asignar x_i al grupo C_i
    end for
    Recalcular centroides para obtener M'
    t ← t + 1
until M = M'
return C, M
```

5. Evaluación Experimental

Para validar el desempeño de la heurística propuesta CentrosKDT-means, se realizaron experimentos comparativos contra el algoritmo K-means estándar. La evaluación se enfocó en tres métricas principales: calidad del agrupamiento, tiempo de ejecución y número de iteraciones. El objetivo fue determinar si la propuesta mejora la eficiencia del algoritmo sin comprometer la calidad de las agrupaciones generadas.

5.1. Diseño experimental

Los experimentos se llevaron a cabo variando el número de grupos (*k = 2 y 4*) en diferentes conjuntos de datos, con el fin de analizar la estabilidad y rendimiento de la heurística. La implementación se realizó en Java, y las pruebas se ejecutaron en un equipo con procesador Intel Core i3-5005U a 2.0 GHz, 4 GB de RAM y sistema operativo Windows 10. Cada configuración fue repetida múltiples veces para asegurar la consistencia de los resultados.

5.2. Conjunto de datos

Se emplearon dos tipos de instancias:
Instancias reales ampliamente utilizadas en la comunidad científica, que incluyen:

- Skin: 245,057 objetos, 3 dimensiones.
- Abalone: 4,177 objetos, 8 dimensiones.
- StatLogCar: 846 objetos, 18 dimensiones.
- Iris: 150 objetos, 4 dimensiones.

Instancias sintéticas
Diseñadas con distribución no uniforme:

- IS5: 18 objetos, 2 dimensiones.
- IS6: 36 objetos, 2 dimensiones.

Diseñadas con distribución uniforme:

- IS1: 10000 objetos, 2 dimensiones.
- IS2: 100000 objetos, 2 dimensiones.
- IS3: 500000 objetos, 2 dimensiones.
- IS4: 1000000 objetos, 2 dimensiones

5.3. Métricas de comparación

Las tres métricas de evaluación se definieron como porcentajes de mejora de CentrosKDT-means respecto a K-means estándar:

Calidad del agrupamiento (SCA):

$$\text{sca} = \frac{(\text{ca}_\text{e} - \text{ca}_\text{p}) * 100}{\text{ca}_\text{e}} \qquad (3)$$

donde ca_e es la suma del error al cuadrado (SSE) del K-means estándar y ca_p la del algoritmo propuesto.

Tiempo de ejecución (T):

$$\text{t} = \frac{(\text{t}_\text{e} - \text{t}_\text{p}) * 100}{\text{t}_\text{e}} \qquad (4)$$

donde t_e y t_p son los tiempos de ejecución del K-means estándar y del propuesto, respectivamente.

Número de iteraciones (I):

$$\text{i} = \frac{(\text{i}_\text{e} - \text{i}_\text{p}) * 100}{\text{i}_\text{e}} \qquad (5)$$

donde i_e es el número de iteraciones con K-means estándar, e i_p con el algoritmo propuesto.

6. Resultados experimentales

6.1. Resultados con instancias sintéticas

- Instancias sintéticas no uniformes

Para este análisis, se observó el comportamiento del algoritmo CentrosKDT-means con el objetivo de evaluar su desempeño frente al algoritmo K-means estándar,

considerando configuraciones con 2 y 4 grupos en las instancias sintéticas IS5 e IS6.

En términos de calidad de agrupamiento, la instancia IS5 con k = 4 obtuvo la mejor mejora: el algoritmo K-means estándar logró un SSE de 15.91, mientras que CentrosKDT-means alcanzó un SSE de 15.19, lo que representa una mejora del 4.53% (véase Tabla 1).

		K-means estándar	Centros KDT-means	K-means vs CentrosKDT-means
Instancia	Grupos	Calidad	Calidad	%Calidad
IS5	2	19.31	19.31	7.7×10^{-6}
IS5	4	15.91	**15.19**	**4.53**
IS6	2	81.82	81.82	-0.00
IS6	4	38.63	38.63	-0.00

Tabla 1. Resultados de calidad con instancias sintéticas no uniformes

La Tabla 2 muestra que, respecto al número de iteraciones, la instancia IS6 con k = 2 presentó la mayor reducción: mientras K-means requirió 3 iteraciones, CentrosKDT-means solo necesitó 1, logrando así una disminución del 66.66%.

		K-means estándar	Centros KDT-means	K-means vs Centros KDT-means %
Instancia	Grupos	Iteraciones	Iteraciones	%Iteraciones
IS5	2	2	1	50
IS5	4	3	2	33.33
IS6	2	3	1	**66.66**
IS6	4	5	2	60

Tabla 2. Resultados de iteraciones con instancias sintéticas no uniformes

En cuanto al tiempo de ejecución, como se puede observar en la Tabla 3, CentrosKDT-means mostró una reducción de hasta 99.86%, evidenciando una gran eficiencia computacional en comparación con el algoritmo base.

		K-means estándar	CentrosKDT-means	K-means vs Centros KDT-means %
Instancia	Grupos	Tiempo	Tiempo	%Tiempo
IS5	2	7.83	0.25	96.84
IS5	4	23.03	0.27	98.81
IS6	2	195.71	0.26	**99.86**
IS6	4	152.61	2.67	98.25

Tabla 3. Resultados de tiempo con instancias sintéticas no uniformes

- Instancias sintéticas con distribución uniforme

Con el propósito de analizar el desempeño del algoritmo CentrosKDT-means en comparación con K-means estándar, se evaluaron ambas soluciones utilizando las instancias sintéticas IS1, IS2, IS3 e IS4, considerando configuraciones con 2, 4 y 8 grupos.

La Tabla 4 muestra resultados positivos en calidad de agrupamiento. El algoritmo propuesto alcanza mejoras cercanas e incluso superiores al desempeño de K-means, con una ganancia máxima del 49.99%.

		K-means estándar	Centros KDT-means	K-means vs Centros KDT-means %
Instancia	Grupos	Calidad	Calidad	Calidad
IS1	2	296596.7	296595.06	0.00
	4	191269.8	265170.56	-38.64
	8	137199.2	136526.45	0.49
IS2	2	1157997	8640353	-646.14
	4	6376874	6317294	0.93
	8	4316165	4316096	0.00
IS3	2	95649494	47827700	**49.99**
	4	71685126	12734576	-77.64
	8	47826937	13259070	-177.23
IS4	2	29709898	29661248	0.16
	4	19129863	26518262	-38.62
	8	13653776	25468640	-86.53

Tabla 4. Resultados de calidad con instancias sintéticas uniformes

La Tabla 5 muestra una ganancia de hasta 98.94% en el número de iteraciones para la instancia IS4, que contiene un millón de registros.

		K-means estándar	Centros KDT-means	K-means vs CentrosKDT-means %
Instancia	Grupos	Iteraciones	Iteraciones	Iteraciones
IS1	2	13.3	2	84.96
	4	15.9	2	87.42
	8	37.9	33	12.93
IS2	2	2.9	2	31.03
	4	35.5	2	94.37
	8	56	49	12.5
IS3	2	9.4	64	-580.85
	4	38.2	2	94.76
	8	58.8	2	96.59
IS4	2	20.5	2	90.24
	4	19.2	2	89.58
	8	189.5	2	**98.94**

Tabla 5. Resultados de iteraciones con instancias sintéticas uniformes

En cuanto al tiempo, la Tabla 6 muestra una ganancia de hasta 93.56% en su reducción.

Instancia	Grupos	K-means estándar Tiempo	Centros KDT-means Tiempo	K-means vs CentrosKDT-means % Tiempo
IS1	2	35.08	41.70	-18.88
	4	68.31	48.46	29.05
	8	79.20	155.96	-96.92
IS2	2	76.58	106.89	-39.58
	4	313.23	142.41	54.53
	8	727.34	1709.71	-135.06
IS3	2	197.89	10731.18	-5322.72
	4	1122.72	366.72	67.34
	8	2542.25	294.12	88.43
IS4	2	687.56	473.09	31.19
	4	1071.94	611.10	42.99
	8	15353.15	988.42	**93.56**

Tabla 6. Resultados de tiempo con instancias sintéticas uniformes

- Instancias reales

Se analizó el comportamiento del algoritmo CentrosKDT-means con el objetivo de evaluar su desempeño frente a K-means estándar, considerando configuraciones con 2, 4 y 8 grupos.

Instancia	Grupos	K-means estándar Calidad	Centros KDT-means Calidad	K-means vs Centros KDT-means % Calidad
Abalone	2	7272.97	7100.10	2.38
	4	4357.86	4255.68	2.34
	8	2760.78	2709.64	1.85
Iris	2	128.40	128.40	-0.00
	4	84.99	84.28	0.83
	8	73.54	61.71	**16.07**
StatLog Car	2	64709.76	64709.81	-0.00
	4	45607.73	45551.44	0.12
	8	33322.47	33520.05	**-0.59**
Skin	2	15930416	15931588	-0.01
	4	10521411	10100919	3.99
	8	7463447	6517140	12.68

Tabla 7. Resultados de calidad con instancias reales

Los resultados indican que el algoritmo propuesto mantiene una calidad de agrupamiento comparable a la de K-means. En el mejor caso se obtuvo una ganancia del 16.07%, y en el peor, una pérdida mínima de -0.59%, inferior al 1%, por lo que no resulta significativa (véase Tabla 7). La Tabla 8 muestra que, en cuanto al número de iteraciones, el algoritmo propuesto alcanzó una reducción del 56.89% en el mejor caso. En la instancia Abalone se observó una ligera pérdida del -9.75%, aunque en general la eficiencia se mantiene favorable en la mayoría de los casos.

Instancia	Grupos	K-means estándar Iteraciones	Centros KDT-means Iteraciones	K-means vs Centros KDT-means % Iteraciones
Abalone	2	4.8	4	16.66666
	4	8.2	9	**-9.75**
	8	16.3	16	1.84
Iris	2	4.6	4	13.04
	4	11.6	5	**56.89**
	8	10.3	6	41.75
StatLog Car	2	5.3	4	24.53
	4	15.6	12	23.07
	8	33.1	34	-2.72
Skin	2	15.1	13	13.91
	4	24	16	33.33
	8	33.8	22	34.91

Tabla 8. Resultados de iteraciones con instancias reales

En términos de tiempo, la Tabla 9 indica que, en el mejor escenario, se obtuvo una reducción del -27.51%, lo que implica una ejecución más rápida del algoritmo propuesto en ciertas instancias. Sin embargo, en el peor de los casos (instancia Abalone), se registró una pérdida del -281.87%, lo que refleja un aumento considerable en el tiempo de ejecución frente al algoritmo K-means estándar.

Instancia	Grupos	K-means están-dar Tiempo	Centros KDT-means Tiempo	K-means vs Centros KDT-means % Tiempo
Abalone	2	22.45	58.55	-160.81
	4	58.27	91.56	-57.12
	8	93.34	144.47	-54.77
Iris	2	2.85	3.85	-34.95
	4	5.67	7.21	**-27.51**
	8	8.34	13.25	-58.84
StatLog Car	2	22.29	36.45	-63.53
	4	38.71	88.30	-128.14
	8	68.99	117.44	-70.23
Skin	2	216.16	825.49	**-281.87**
	4	451.11	1389.82	-208.08
	8	1014.7	2918.28	-187.58

Tabla 9. Resultados de tiempo con instancias reales

7. Análisis de resultados

En la instancia sintética IS5, con distribución no uniforme y configuración de 4 grupos, la heurística CentrosKDT-means alcanzó una mejora del 4.53% en la calidad del agrupamiento, acompañada de una reducción del 98.81% en el tiempo de ejecución y una disminución del 33.33% en el número de iteraciones respecto al algoritmo K-means estándar.

En el caso de la instancia sintética IS2, con distribución uniforme, los resultados muestran una mejora del 0.00593% en la calidad del agrupamiento, así como una reducción del 99.997% en el tiempo de ejecución y una disminución del 94.44% en las iteraciones requeridas.

Por último, para la instancia real Skin, se obtuvo una ganancia del 5.405% en la calidad del agrupamiento, con una reducción del 99.97% en el tiempo y una disminución del 38.46% en el número de iteraciones, lo que confirma la eficacia de la heurística incluso en escenarios con grandes volúmenes de datos reales.

En general, los resultados experimentales muestran que CentrosKDT-means reduce sustancialmente el tiempo y el número de iteraciones, manteniendo una calidad de agrupamiento comparable —y en algunos casos superior— a la obtenida por el K-means tradicional. Estos beneficios hacen que la heurística propuesta sea especialmente adecuada para escenarios con restricciones de tiempo o recursos computacionales, así como para conjuntos de datos de gran tamaño, como los que plantea el paradigma Big Data.

8. Conclusiones

Los resultados de esta investigación demuestran la factibilidad de desarrollar una mejora al algoritmo K-means, orientada a la solución de instancias con características propias del paradigma Big Data, a través de la integración de una estructura de datos tipo árbol. En particular, la propuesta se centró en optimizar la fase de inicialización mediante el uso del método Kd-tree, con el propósito de reducir la complejidad computacional sin comprometer la calidad del agrupamiento.

La finalidad del estudio fue incrementar la eficiencia del algoritmo K-means, logrando una implementación más escalable y adecuada para contextos con grandes volúmenes de datos. El resultado fue un nuevo algoritmo, denominado CentrosKDT-means, que mostró un comportamiento eficiente, competitivo y, en muchos casos, superior al algoritmo estándar, tanto en términos de tiempo de ejecución como en número de iteraciones requeridas.

La validación experimental se llevó a cabo utilizando tanto instancias sintéticas como reales, lo cual permitió una evaluación robusta del desempeño de la heurística propuesta. Los experimentos evidenciaron que, incluso en escenarios con conjuntos de datos pequeños pero desbalanceados, el algoritmo logró mejoras significativas. Por ejemplo, para la instancia sintética con 36 objetos y distribución no uniforme agrupados en 2 clústeres, se observó una reducción del 50% en el tiempo de ejecución. En cuanto al número de iteraciones, se obtuvo una disminución de hasta 66.66%, sin evidencia de pérdida significativa en la calidad del agrupamiento, incluso en escenarios con variaciones en la cantidad de datos.

Asimismo, para la instancia de 18 objetos con 4 grupos, también con distribución no uniforme, se alcanzó una reducción del 98.81% en el tiempo de ejecución, una ganancia del 4.53% en calidad, y una disminución del 33.33% en iteraciones. En este caso, la pérdida máxima registrada fue de apenas 0.00000077%, lo cual refuerza la solidez de la mejora propuesta.

En síntesis, CentrosKDT-means se presenta como una alternativa viable, eficiente y fácilmente integrable en contextos de alto volumen de datos. Sus beneficios lo hacen particularmente adecuado para entornos con limitaciones de tiempo o recursos computacionales, consolidándolo como una contribución relevante al campo de los algoritmos de agrupamiento orientados a Big Data.

Referencias

[1] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In L. M. Le Cam & J. Neyman (Eds.), *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (Vol. 1, pp. 281–297). University of California Press.

[2] Bezdek, J. C. (1981). *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media.

[3] Ren, T., Wang, H., Feng, H., Xu, C., Liu, G., & Ding, P. (2019). Study on the improved fuzzy clustering algorithm and its application in brain image segmentation. *Applied Soft Computing*, *81*, 105503. https://doi.org/10.1016/j.asoc.2019.105503

[4] Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, *28*(2), 129–137.

[5] Arthur, D., & Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms* (pp. 1027–1035).

[6] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., & Wu, A. Y. (2002). An efficient K-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(7), 881–892.

[7] Pérez-Ortega, J., Moreno-Calderón, C. F., Roblero-Aguilar, S. S., Almanza-Ortega, N. N., Frausto-Solís, J., & Pazos-Rangel, R. (2024). A new criterion for improving convergence of fuzzy C-means clustering. *Axioms*, *13*(1), 35. https://doi.org/10.3390/axioms13010035

[8] Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., ... & Steinberg, D. (2007). Top 10 algorithms in data mining. *Knowledge and Information Systems*, *14*(1), 1–37. https://doi.org/10.1007/s10115-007-0114-2

[9] Vrahatis, M. N., Boutsinas, B., Alevizos, P., & Pavlides, G. (2002). The new k-Windows algorithm for improving the k-Means clustering algorithm. *Pattern Recognition Letters*, *23*(3–4), 375–391.

[10] Zou, K., Wang, Z., & Hu, M. (2008). A new initialization method for fuzzy c-means algorithm. *Fuzzy Optimization and Decision Making*, *7*(4), 409–416.

[11] Stetco, A., Zeng, X. J., & Keane, J. (2015). Fuzzy C-means++: Fuzzy C-means with effective seeding initialization. *Expert Systems with Applications*, *42*(21), 7541–7548.

[12] Pérez, J., Pazos, R., Cruz, L., Reyes, G., Besave, R., & Fraire, H. (2007). Improving the efficiency and efficacy of the K-means clustering algorithm through a new convergence condition. In *Computational Science and Its Applications – ICCSA 2007* (pp. 674–682).

[13] Pérez, J., Mexicano, A., Santaolaya, R., Hidalgo, M., Moreno, A., & Pazos, R. (2012). Improvement to the K-means algorithm through a heuristics based on a bee honeycomb structure. *Proceedings of the 2012 4th World Congress on Nature and Biologically Inspired Computing (NaBIC)*, 175–180.

[14] Pérez, J., Mexicano, A., Santaolaya, R., Hidalgo, M., Moreno, A., & Pazos, R. (2014). Mejora del algoritmo K-means mediante una metaheurística orientada a la reducción de su complejidad computacional. *Encuentro Nacional de Computación – ENC, Taller de Aspectos Algorítmicos de Sistemas de Cómputo*.

[15] Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, *18*(9), 509–517.

[16] Friedman, J. H., Baskett, F., & Shustek, L. J. (1975). An algorithm for finding nearest neighbors. *IEEE Transactions on Computers*, *C-24*(10), 1000–1006.

[17] Bhimavarapu, U., Chintalapudi, N., & Battineni, G. (2024). Brain tumor detection and categorization with segmentation of improved unsupervised clustering approach and machine learning classifier. *Bioengineering*, *11*(3), 266. https://doi.org/10.3390/bioengineering11030266

[18] Yosph, F., Malim, N. H. A. H., Heikkilä, M., Brezulianu, A., Geman, O., & Rostam, N. A. P. (2020). The impact of big data market segmentaton using data mining and clustering techniques. *Journal of Intelligent & Fuzzy Systems*, *38*(5), 6159–6173. https://doi.org/10.3233/JIFS-179337

[19] Nazari, M., Hussain, A., & Musilek, P. (2023). Applications of clustering methods for different aspects of electric vehicles. *Electronics*, *12*(4), 790. https://doi.org/10.3390/electronics12040790

[20] Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., & Wu, A. Y. (1998). An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM (JACM)*, *45*(6), 891–923.