

Análisis de datos abiertos MEData basados en algoritmos de clustering

Daniel Santa Rendón, Daniel Torres González, Héctor Mauricio Guerra Londoño

Proyecto integrador I, código 2508103, grupo 04

Ingeniería de Sistemas, Universidad de Antioquia
Medellín, Colombia

hector.guerra@udea.edu.co

daniel.santar@udea.edu.co

daniel.torresg@udea.edu.co

Resumen—MEData es una propuesta de la alcaldía de Medellín que consta de 237 conjuntos de datos sobre salud, población, seguridad, etc. Con el fin de estar un paso más cerca de convertirse en Smart City; pero alguna información allí almacenada no está en condiciones óptimas (datos con ruido, datos faltantes, información no relevante) para que los expertos en bases de datos las analicen y tomen las decisiones más convenientes. Se tomaron diez bases de datos a las cuales se les realizó un proceso de aprendizaje no supervisado, es decir, transformación de variables categóricas por medio de label-encoder, llenado de valores faltantes usando la estrategia de simple imputer, manejo de outliers o datos atípicos, escalamiento de la base de datos por medio del método MinMax entre 0 y 1, y por último, en la etapa de ingeniería de características, se hizo un análisis de componentes principales (PCA) escogiendo las variables que explican por lo menos el 95 % de la varianza de los datos para cada base de datos; después, se ejecutaron los algoritmos de clustering: K-Means, DBSCAN y GK-Means, y en algunas situaciones otros como Spectral Clustering y minibatch K-Means. Para cada algoritmo se emplearon índices de validación para la selección del número de clusters óptimo.

Palabras Claves: Clustering, Bases de datos, Ingeniería de características, validación de grupos, aprendizaje no supervisado.

Abstract—MEData is a Medellín's town hall proposal consists of 237 data sets about health, population, safety, etc. In order to be one step closer to become Smart City; but some information stored there is not in optimal conditions (data with noise, missing data, non-relevant information) for database experts analyzes and makes the most convenient decisions. 10 databases were taken to which an unsupervised learning process was carried out, that is, transformation of categorical variables through

label-encoder, filling of missing values using the simple imputer strategy, outliers or atypical data management, database scaling through the MinMax method between 0 and 1, and finally, in the characteristics engineering stage, a principal component analysis (PCA) was made choosing the variables that explain at least 95% of the data variance for each database; then, the clustering algorithms were executed: K-Means, DBSCAN and GK-Means, and in some other situations such as Spectral Clustering and minibatch K-Means. Validation indices were used for each algorithm to select the optimal number of clusters.

Key Words: Clustering, Data bases, feature engineering, clusters validation, unsupervised learning.

I. INTRODUCCIÓN

Clustering o agrupación es un aprendizaje no supervisado que establece la relación entre un conjunto de elementos o muestras (objeto) y un grupo. Cada grupo es separable y compacto con respecto a otro grupo[1].

La alcaldía de Medellín, desarrolló una iniciativa de datos abiertos llamada MEData[2], que contiene información sobre educación, movilidad, salud, entre otros. Un problema que presenta esta herramienta es que las bases de datos no están etiquetadas, por lo que no es posible realizar tareas de clasificación o de predicción. Por lo tanto, en este proyecto se pretende etiquetar un total de diez bases de datos de MEData, haciendo uso de algoritmos de *clustering*; para esto, primero hay que hacer una limpieza de los datos, llamada ingeniería de característica[3], es decir, codificación de las variables categóricas, llenado de datos faltantes, manejo de

outliers o datos extraños, reducción de dimensionalidad y normalización de los datos. Posterior a esta etapa, se aplican los algoritmos de *clustering*: k-Means, DBSCAN, GK-means, en algunos problemas otros algoritmos adicionales, como spectral clustering y mini batch k-Means, y se corroboran los resultados con algunos índices de validación para conocer el número óptimo de cluster en cada problema.

En la sección 1 se hace un repaso de las investigaciones referentes al clustering desde el año 2016 al año 2019, en la sección 2 se definen los conceptos y métodos usados en este artículo y, además, se describen las bases de datos usadas, en la sección 3 se describe la metodología llevada propuesta para el tratamiento del problema, en la sección 4 se ilustran los resultados más relevantes para cada base de datos, en la sección 5 se realiza un análisis de los resultados encontrados con los diferentes métodos propuestos, en la sección 6 se dan los agradecimientos y en la sección 7 se finaliza con las conclusiones.

II. ESTADO DEL ARTE

Debido a los grandes avances, en la última década, con respecto al aprendizaje de máquina. Han ido apareciendo más métodos y algoritmos que nos permiten hacer esto de manera efectiva y eficiente. Como aseguran Qi, Yu, Wang, & Liu. (2016), a través de su artículo "*K*-Means: An Effective and Efficient K-Means Clustering Algorithm*" [4], en el cual proponen 3 principios de optimización para el algoritmo tradicional K-Means. El primero de los principios ellos proponen un principio de optimización jerárquica en el cual buscan el riesgo de selección aleatoria de semillas, y luego buscan utilizar el método top-n propuesto para fusionar los grupos más cercanos asociados con los n bordes más cortos en cada ronda hasta que se alcanzan los k grupos. Lo segundo que ellos proponen es una estrategia de poda de conglomerados para mejorar la eficiencia de k-means al omitir los conglomerados más lejanos para reducir el espacio de búsqueda ajustable para cada punto en cada iteración. Y por último implementaron una optimización al momento en el que el algoritmo realiza las actualizaciones en cada iteración. Al probar estos 3 principios en datasets de UCI Machine Learning Repository

obtuvieron un rendimiento que supera los métodos actuales.

Un segundo trabajo corresponde al de Nurhayati et al.(2018) en su estudio "*Big Data Technology for Comparative Study of K-Means and Fuzzy C-Means Algorithms Performance*"[5]. En el cual utilizando *big data*, es decir una gran cantidad de datos, comparar el rendimiento de 2 diferentes algoritmos de *clustering*, *Fuzzy C Means* y *K Means*. Los investigadores compararon los algoritmos en términos de precisión, tiempo de ejecución, y complejidad de los algoritmos. Esto se realizó construyendo una aplicación en Java, Hadoop y Hive, en la que se realizó un filtro de datos y los resultados fueron los siguientes. En promedio la precisión fue un 8.03% mejor para *K Means*, 718.58 ms mas rapido *K means* que *Fuzzy C Means* en tiempos de ejecución y en términos de complejidad los dos son de orden n^2 , al utilizar la ecuación *Big O* manda un resultado de una diferencia de 93.568 con el valor más pequeño para *K means*. Por lo tanto, concluyen que *K means* es mejor que *Fuzzy C Means* en los términos planteados.

Otro estudio, es un presentado por Surbhi Sharma, Arvind K Sharma y Dinesh Soni (2018) en "*Enhancing DBSCAN algorithm for data mining*". En el cual su objetivo fue proponer una metodología para la mejora del algoritmo DBSCAN en términos de precisión de a la hora de realizar la agrupación de los datos. Esta mejora se basa en el algoritmo de *BackPropagation* para calcular la distancia euclidiana de manera dinámica. En trabajo también muestran los resultados obtenidos de los métodos implementados propuestos y existentes y compara los resultados en términos de su tiempo de ejecución y precisión. En el estudio los resultados que se presentan es que tienen un precision de mas del 92.5% con respeto a los algoritmos actuales que es de un 85.5%, en tiempos de ejecución representa una mejora de aproximadamente 1.6 ms[6].

Un último trabajo que fue revisado es el de Chaomurilige, Yu, Yang.(2015) en el que abordan el problema de seleccionar los parametros optimos para el algoritmo de GK-Means. Ellos en el artículo "*Analysis of Parameter Selection for Gustafson-Kessel Fuzzy Clustering Using Jacobian Matrix*"[7], revelan cuál es la relación existente entre los centroides del algoritmo y los conjuntos

de datos mediante el análisis de matriz jacobiana, y luego proporcionan una base teórica para seleccionar el índice de difuminación en el algoritmo GK.

La inteligencia artificial (IA) [8] es uno de los componentes de la llamada cuarta revolución industrial [9] y se define como la simulación de procesos de inteligencia humana por parte de las máquinas. La IA es ampliamente usada hoy en día tanto para la industria como para uso cotidiano de las personas. Por ejemplo los bancos usan IA para predecir el valor de las acciones, muchas empresas están implementando chatbots para dar soporte al cliente, también se usa para reconocimiento facial, rellenado de imágenes y ahora con big data es necesaria la IA para análisis de datos.

Dentro de la inteligencia artificial hay un campo llamado *machine learning* (aprendizaje de máquina)[10] que es una disciplina científica que crea sistemas que aprenden automáticamente y es fundamental para el análisis de datos. Hay tres tipos:

- Aprendizaje supervisado (*supervised learning*) cuándo los datos poseen etiqueta, su objetivo es clasificar datos nuevos gracias al entrenamiento recibido con los datos existentes,
- Aprendizaje reforzado (*reinforcement learning*) se basa en que la máquina aprenda por la experiencia
- Aprendizaje no supervisado (*unsupervised learning*) cuando los datos no poseen etiquetas, su objetivo es agrupar los datos de la mejor manera para darles una etiqueta óptima.

En este proyecto se trabaja el aprendizaje no supervisado [11], en el que problema que se presenta es que la mayoría de datos no están etiquetados para realizar tareas posteriores como predicción o análisis de datos. Por esta razón, el proyecto integrador pretende crear soluciones basados en *clustering* para agrupar los datos en *clusters*, de tal forma que facilite la etiquetación de los datos.

En el trabajo se observará el proceso que seguimos para cumplir con nuestro objetivo principal el cual es encontrar las etiquetas, que nos permitirán facilitar el análisis y las decisiones a los expertos en bases de datos. Esto va a ser posible implementando una limpieza de datos o ingeniería de características que nos permitirá aplicar los algoritmos de *clustering* y por último aplicar algoritmos

de validación interna que nos permitirán encontrar el número óptimo de *clusters*.

III. PRELIMINARES

Clustering o Agrupamiento: es un aprendizaje no supervisado que establece la relación entre un conjunto de elementos o muestras (objeto) y un grupo. Cada grupo es separable y compacto con respecto a otro grupo.[1]

Algoritmos de clustering: En el transcurso de este proyecto se utilizaron diferentes algoritmos de aprendizaje no supervisado que nos permitieron agrupar de mejor manera las diferentes bases de datos con las que contamos.

- **K-Means:** Este algoritmo fue utilizado por nosotros en este proyecto debido a la fácil interpretación de los resultados e implementación. Este algoritmo de *clustering* consta de dividir las muestras en k conjuntos. Para lograr este objetivo, se designan k centros aleatorios, luego, a cada centro se le asocian las muestras más cercanas que tenga. Una vez que las muestras tengan un centro asociado, se calcula un nuevo centro a partir de todas las muestras que pertenezcan a ese mismo conjunto para nuevamente calcular las distancias que hay entre cada centro y cada muestra. Este proceso se repite una y otra vez hasta que los centros dejen de variar.[12]

- **DBSCAN:**

Es el primer algoritmo de agrupamiento basado en la densidad y fue diseñado para agrupar datos de formas arbitrarias en presencia de ruido en bases de datos de alta dimensión espaciales y no espaciales. Lo que este hace es clasificar cada punto como punto central, punto fronterizo o punto ruidoso, para esto se definen dos parámetros: eps es la distancia entre dos muestras para que una se considere en la vecindad de otra y muestras mínimas es el número de muestras (o peso total) en un vecindario para que un punto sea considerado como un punto central, los puntos central son los puntos al interior de un *cluster*, posee más de un número mínimo de puntos (muestras mínimas) dentro del radio específico (eps). Los puntos fronterizos tienen menos puntos que el muestras mínimas dentro de su radio eps, sin

embargo, están cerca de un punto central. Los puntos ruidosos son las muestras que no son ni punto central ni punto fronterizo. Los pasos que hace este algoritmo son:

- 1) Etiquetar todos los puntos como central, fronterizo o ruidoso.
- 2) Eliminar los puntos ruidosos.
- 3) Poner un borde entre todos los puntos central que están dentro del eps de otros puntos.
- 4) Hacer que cada grupo de puntos central conectados sea un cluster distinto.
- 5) Asignar cada punto fronterizo a uno de los *clusters* de sus puntos central asociados.

Se debe definir un vector de valores para los parámetros eps y muestras mínimas para cada una de las bases de datos.[13]

- **Spectral Clustering:** Es un método de agrupación popular que utiliza vectores propios de una matriz derivada de los datos. En este proyecto se utilizó cuando se presentaban problemas de desbalance, y se tuvimos la necesidad de corroborar resultados con este algoritmo y observar si se generaban mejores *clusters*[14].
- **GK-Means:** Este algoritmo como una variante o mejora del algoritmo clásico *k-means*, nos enseña una manera para encontrar los mejores y más efectivos centros iniciales a la hora de implementar el algoritmo de agrupamiento esto lo hace gracias a que se combina una estructura de cuadrículas y un indexamiento espacial con el algoritmo *k-means*[15].
- **Mini-Batch K-Means:** Este algoritmo fue utilizado debido a su gran eficiencia. Ya que este es una versión modificada del algoritmo *k-means*. Usa mini lotes para reducir el tiempo de cálculo en grandes conjuntos de datos. Además, intenta optimizar el resultado del agrupamiento. Para lograr esto, toma mini lotes como entrada, que son subconjuntos del conjunto de datos completo, al azar. El *Mini-Batch K-Means* se considera más rápido que *k-means* y normalmente se usa para los grandes conjuntos de datos[16].

Índices de Validación

Para la validación y la selección de los resultados que se generan al aplicar los algoritmos de clustering, se utilizaron los siguientes índices de vali-

dación. Estos índices de validación nos pueden dar cuenta de que tan bueno son los grupos o *clusters*, es decir, posee una alta homogeneidad dentro de los grupos, los grupos están bien separados y los puntos vecinos están altamente conectados. Estos índices se utilizaron en los algoritmos de *k-means*, *spectral clustering* y *mini-batch k-means*.

- **Inercia o Within-cluster-sum-of-squares(SSW)**[17]: La inercia nos permite reconocer o medir la coherencia interna de los grupos. Pero esta tiene dos inconvenientes, primeramente debido a que toma la suposición de que los clusters son convexos e isotrópicos, responde de una manera no deseada a clusters con formas irregulares o elongados. El segundo inconveniente se presenta debido a que no es métrica normalizada cuando tenemos un problema de alto espacio dimensional se suele inflar el valor. Se calcula de la siguiente manera.

$$Inercia = \sum_{i=0}^n \|x_i - \mu\|^2$$

Donde x_i es una muestra i en el grupo y μ es la media del grupo.

- **Coficiente de Silhouette (SC)**[17]: Es un índice que se determina para cada muestra y mientras más alto sea este valor es mejor. Se calcula por medio de la siguiente fórmula matemática.

$$SC = \frac{b - a}{\max(a, b)}$$

Donde a es la distancia media entre una muestra y las demás muestras en la misma clase, y b la distancia media entre una muestra y los puntos del proximo grupo más cercano.

- **Índice De Calinski and Harabasz**[17]: Este índice también conocido como criterio de relación de varianza, nos permite evaluar el modelo, que nos permite observar en puntaje como la relación entre la dispersión dentro del grupo y la dispersión entre grupos. Un puntaje alto nos demuestra un modelo con *clusters* mejor definidos. Para k grupos el índice s sera:

$$s(k) = \frac{Tr(B_k)}{Tr(W_k)} * \frac{N - k}{k - 1}$$

donde B_k es la matriz de dispersión entre grupos y W_k es la matriz de dispersión en los grupos.

$$W_k = \sum_{q=1}^k \sum_{x \in C_q} (x - c_q)(x - c_q)^T$$

$$B_k = \sum_{q=1}^k n_q (c_q - c)(c_q - c)^T$$

En este trabajo también se utilizaron diferentes índices de validación para validar el resultado generado por el algoritmo de *GK-Means* estos son los siguientes:

- **PC:** También conocido como coeficiente de partición.

$$PC = \frac{1}{n} \sum_{i=1}^c \sum_{j=1}^n \mu_{ij}^2$$

- **NPC :** Una modificación del índice PC el cual reduce la tendencia monótona.

$$NPC = 1 - \frac{c}{c-1} (1 - PC)$$

- **FS:** Es una función que mide la compacidad y separabilidad.

$$FS = \sum_{i=1}^c \sum_{j=1}^n \mu_{ij}^m \|x_j - a_i\|$$

- **FHV:** Es un índice que utiliza una matriz difusa F_i , una matriz de covarianza del cluster i . Un extremo en este índice nos indica una buena partición.

$$FHV = \sum_{i=1}^c [\det(F_i)]^{1/2}$$

Donde F_i esta determinada por

$$F_i = \frac{\sum_{j=1}^n \mu_{ij}^m (x_j - a_i)^T}{\sum_{j=1}^n \mu_{ij}^m}$$

- **XB:** Es una medida de compacidad y separabilidad.

IV. METODOLOGIA

Tabla 1. Descripción de las bases de datos (E = Entero, C = Categórico, R = Real)

Nombre base de datos	Datos originales				Datos procesados	
	Muestras	Características	Tipo de datos	Datos faltantes	Muestras	Características
Beneficiarios Discapacidad	29691	16	E, C	No	22327	10
Defunciones 2015	12967	60	E, C	Si	6350	13
Dignatarios	13093	9	E, C	No	10118	5
Encuesta juventud	8001	290	R, E, C	Si	5831	147
Individuos que participan	3459	166	E, C	Si	3033	101
Inversión por comunas y corregimientos Medellín 2016	3786	9	R, E, C	No	2375	5
Estado nutricional de menores de 6 años programa de crecimiento y desarrollo 2016	83861	17	R, E, C	No	55816	9
Población indígena	486	16	R, E, C	Si	461	7
Varicela individual	52843	15	E, C	Si	49471	8
Afiliados al régimen subsidiado de Medellín 2015	614626	13	E, C	Si	501163	4

Descripción de las bases de datos: En la *tabla 1* se detallan las características de cada base de datos, como lo es su nombre, las muestras y las características de los datos originales, el tipo de datos que tiene la BD (E = Entero, C = Categórico, R = Real), si tiene datos faltantes o no y las muestras y características resultantes del procesamiento de los datos, que son con las que se realizará los algoritmos de *clustering*. Por facilidad para explicar la metodología, se usarán diagramas de flujo diferenciados por su nombre: el apéndice 1 describe la ingeniería de características realizada a cada base de datos para dejar estas lo más limpias y óptimas posible para luego realizar el *clustering*. El apéndice 2, exhibe el proceso realizado para el algoritmo K-Means. El apéndice 3, muestra el desarrollo del algoritmo DBSCAN. Por último, el apéndice 4, expone el funcionamiento del algoritmo GK-Means.

El apéndice 1 representa el diagrama de flujo seguido para desarrollar la limpieza de datos por medio de la ingeniería de características. Para comenzar, se debe cargar una base de datos al espacio de trabajo, cómo se utiliza el lenguaje de programación Python, se hace uso de la librería Pandas la cual permite esta acción, este paso es realizado para poder disponer de los datos y trabajar con ellos. Posteriormente, se hace un análisis de los datos que nos permite tomar decisiones para realizar la ingeniería de características.

Una vez realizado el análisis, si se encuentran variables categóricas, por ejemplo: hombre o mujer, comuna 1 o comuna 2. Se procede a ejecutar label-encoder, esta técnica permite convertir las variables categóricas a numéricas con el fin de que la máquina

pueda operar sobre ellas, por ejemplo: Si y No, lo transforma a 0 y 1. En caso de que la base de datos no tenga variables categóricas, se inicia con el siguiente paso: revisar datos faltantes o nulos. Para poder continuar, cada dato que tenía algún diferenciable de ser un dato nulo (null, -1, “xxx”, entre otros) era eliminado dejando el espacio vacío, esto permite que al cargar la base de datos se le diera el valor de “NaN” que significa Not a Number. Sabiendo que la base de datos cuenta con los datos faltantes, los cuales tiene el valor de “NaN”, se ejecuta un método para completarlos; por convención, se decide ejecutar Simple imputer, que es un transformador de imputación para valores faltantes, con la estrategia *most-frequent*, es decir, reemplaza los valores faltantes por el dato más frecuente en cada característica, se usa esta estrategia debido a que casi la totalidad de datos faltantes eran de tipo categórico y el objetivo era asignar estos valores a una categoría existente.

Después, se eliminan datos extraños (outliers) suponiendo que los datos tienen una distribución Gaussiana, los datos que están por fuera de intervalo de varianza ± 0.05 son tomados como outliers y se eliminan las muestras con estos datos.

El siguiente, es uno de los pasos más importantes: escalar los datos. Se deben escalar los datos, porque si una característica tiene valores muy grandes con respecto a otra, por ejemplo: precio de una casa contra años de antigüedad, el precio de la casa tendrá un impacto mucho mayor en el algoritmo, dando resultados poco óptimos. Para esto se usó el método min max, con rango entre 0 y 1, es decir, escala los datos en un intervalo de 0 a 1.

Por último, se hace un análisis de componentes principales que consiste en eliminar las características que no aportan mucha información a la varianza para explicar el modelo, en este caso se puso el umbral del 0.95 de varianza, es decir, se dejan las variables con las que se obtiene el 95% de la varianza para explicar el modelo, las demás se pueden quitar. Luego se hace una reconstrucción de los datos con el fin de visualizar qué tanta información se perdió; en algunos casos fue necesario usar otras variaciones de PCA tales como kernel-PCA y *sparse-PCA*. Estos datos son los que se usarán en las siguientes etapas.

Una vez que se haya realizado la ingeniería de

características para cada base de datos, procedemos con ejecutar los algoritmos de *clustering*.

El apéndice 2 representa el diagrama de flujo del algoritmo *K-Means* y su validación por medio de la inercia, el coeficiente de silhouette y el índice de Calinski-Harabasz para conocer los resultados arrojados del modelo con los parámetros seleccionados, en este caso se realizaron pruebas con cantidad de 2 a 10 *clusters*. Un ejemplo de estas métricas se puede observar en los apéndices 3, 4 y 5.

Al ejecutarse las 9 veces el algoritmo, por medio de los índices de validación antes mencionados, se escoge el mejor modelo para cada base de datos. En la base de datos Dignatarios, se elige el número óptimo de *cluster* en 4 con *K-Means*. Es importante aclarar que para la base de datos “Afiliados al régimen subsidiado de Medellín 2015” la ejecución de este algoritmo no fue posible por lo que se usó una variante del *K-Means*, el *minibatch K-Means* con el mismo rango de parámetros.

La Figura 6. representa el funcionamiento del algoritmo DBSCAN. En este método el parámetro eps fue establecido en un rango desde 0.1 hasta 0.99 y se varían los valores dependiendo de los resultados obtenidos, y el parámetro muestras mínimas fue seleccionado con respecto a la cantidad de muestras de la base de datos.

En el diagrama de flujo de este modelo, *epss* y *mins* son los vectores que contienen todos los valores con los que se realizarán las pruebas. El algoritmo termina cuando haya evaluado cada valor eps del arreglo *epss* con todos los datos del vector *mins*.

Por último la Figura 7, expone el diagrama de flujo del algoritmo GK-Means. Este algoritmo al ser una variante del K-Means, su funcionamiento general es muy similar a este. Para validar los resultados de este modelo se implementaron 7 índices: pc, npc, fhv, fs, xb. Y se selecciona la mejor cantidad de cluster por medio de la moda de los resultados de los índices.

V. RESULTADOS

Ver apéndices 8 y 9.

VI. ANALISIS Y DISCUSIÓN

Antes de comenzar la primera etapa del proceso, se tuvo que hacer una limpieza manual de valores en algunas bases de datos, debido a la poca

claridad de algunas características, por ejemplo, En la base de datos “Beneficiarios Discapacidad” en la característica “tipo_discapacidad” una sola muestra repetía valores, es decir, aparecía “discapacidad mental, discapacidad mental”. Esto es un problema puesto que “discapacidad mental, discapacidad mental” para el algoritmo de codificación es diferente a “discapacidad mental”, creando así muchas más categorías para esta característica. La solución fue usar las herramientas que dispone excel para buscar y reemplazar estos valores. Además, manualmente, se retiraron las tildes, las letras “ñ” y letras extrañas (que no pertenecen al abecedario inglés), pues los algoritmos pueden tener problemas con estos caracteres. Luego, en la primera etapa, en la sección de reducción de dimensionalidad, para todas las bases de datos se utilizó la reconstrucción PCA, esta reconstrucción funcionaba muy bien para las bases de datos pequeñas y medianas, el problema se dio con las bases de datos más grandes, para estas últimas, la solución fue ejecutar reconstrucción Kernel PCA y sparse PCA, a pesar de que se tienen resultados un poco mejores, son algoritmos que tienen un alto costo computacional.

En la segunda etapa, aunque el K-Means sea el algoritmo más básico, fue el que en general dio mejores resultados; los peores resultados obtenidos por el K-Means fueron con la base de datos “Inversión por comunas y corregimientos Medellín 2016”, puesto que hay desbalance muy grande entre los dos grupos: un grupo con 176 muestras y el otro con 2199 muestras. Aunque la separabilidad entre clusters es buena. Con la base de datos “Estado nutricional de menores de 6 años programa de crecimiento y desarrollo 2016”, que aunque el balance de grupos es bueno (para un grupo 28303 muestras y para el otro 27513 muestras), la separabilidad entre clases es mínima. Y con la base de datos “Afiliados al régimen subsidiado de Medellín 2015” (ver gráfica 5), dado que la cantidad de muestras sobrepasan la capacidad de hardware disponible; el único algoritmo que fue capaz de arrojar un resultado fue el mini batch K-Means con 2 clusters, el cual fue favorable: dos grupos, uno con 273990 muestras y el otro con 227173 muestras.

Utilizar el índice silhouette era pertinente para bases de datos que no fueran muy grandes, si llegaba el caso de que la base de datos tuviera límites

extensos, este fallaba. Para estos casos, se dio más peso en los índices de Inercia y Calinski-Harabasz.

Con el algoritmo DBSCAN, en 5 de las 10 bases de datos se obtuvieron resultados similares en cantidad de clusters y total de muestras por cluster comparado con los resultados del K-Means, en 3 bases de datos no se obtuvieron resultados significativos, es decir, la gran mayoría de muestras se clasificaron como ruido, se forma un solo *cluster* o ningún *cluster*; y en 2 bases de datos dieron resultados diferentes a los arrojados por K-Means. A las dos bases de datos que se les aplicó spectral clustering, se generaron resultados buenos, muy similares a los dados por K-Means, aunque en la BD “Inversión por comunas y corregimientos Medellín 2016” se siguió teniendo el mismo problema de desbalanceo entre clases.

Por último, el *GK-Means*, dio mejores resultados con la base de datos “Estado nutricional de menores de 6 años programa de crecimiento y desarrollo 2016” pero generó peores resultados que el *K-Means* con las bases de datos “Encuesta juventud”, “Individuos que participan” y “Varicela individual”. Estos últimos dos algoritmos, tuvieron problemas de convergencia con algunas bases de datos, GK-Means debido a que no se pudo resolver la distancia de la matriz de covarianza inversa y spectral clustering pues el requerimiento de memoria para la culminación de este algoritmo fue superior al hardware disponible.

VII. AGRADECIMIENTOS

Le damos las gracias a la alcaldía de Medellín y a la gobernación de Antioquia por facilitarnos el acceso a las bases de datos de MEData y permitirnos tener la oportunidad de desarrollar el proyecto integrador 1.

VIII. CONCLUSIONES

Es indispensable realizar un preprocesamiento de los datos para obtener resultados óptimos en los algoritmos de *clustering*, pues dada la gran variedad de bases de datos tratadas, fue necesario aplicar diferentes técnicas y métodos diferentes para cada una de estas. También es esencial saber que un modelo no se ajusta a todos los problemas, por esto, se varían los parámetros de cada modelo para

cada base de datos. Por el momento, se ejecutaron los algoritmos K-Means, DBSCAN y GK-Means y, en algunos casos, spectral clustering y mini batch K-Means generando, en su mayoría, resultados muy positivos. Además, es importante disponer de índices de validación que den una métrica numérica del resultado para la selección de la mejor configuración para cada modelo.

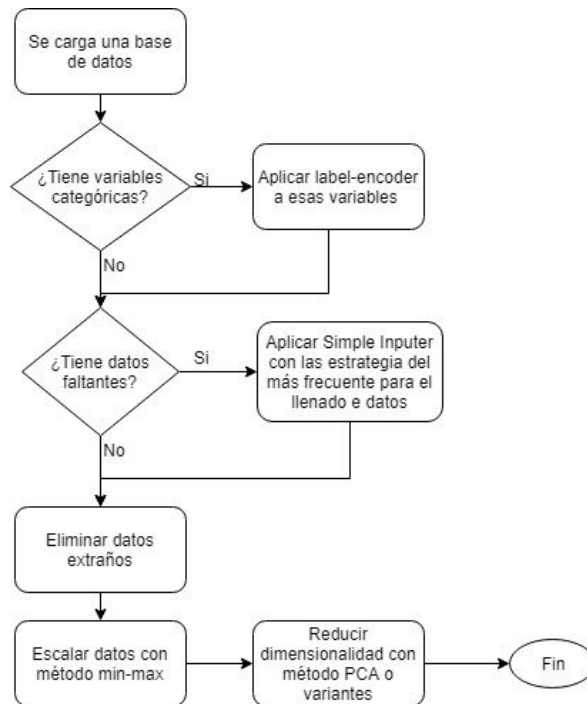
Para trabajos futuros, se pretende seguir probando diferentes algoritmos e implementar el modelo de collaborative clustering con las mismas diez bases de datos usadas en este artículo.

REFERENCIAS

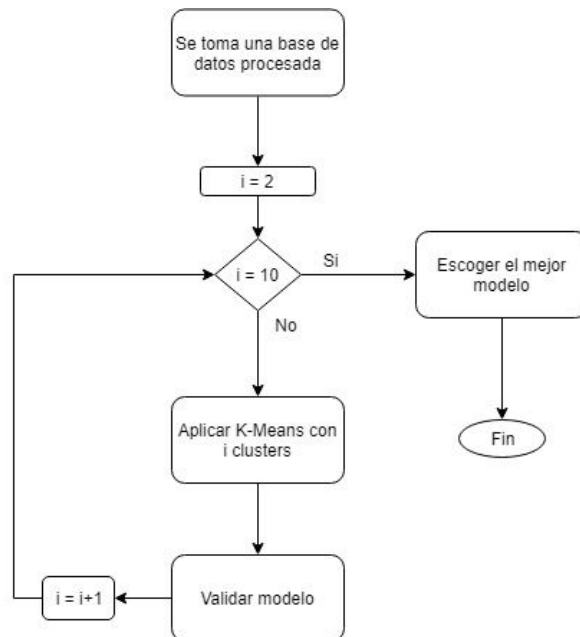
- [1] J. F. Botía Valderrama y D. J. L. Botía Valderrama, “On LAMDA clustering method based on typicality degree and intuitionistic fuzzy sets”, *Expert Syst. Appl.*, vol. 107, pp. 196–221, oct. 2018.
- [2] “MEData”, 2019. [En línea]. Disponible en: <http://medata.gov.co/>. [Consultado: 04-sep-2019].
- [3] A. Casari y A. Zheng, *Feature Engineering for Machine Learning*.
- [4] J. Qi, Y. Yu, L. Wang, y J. Liu, “K*-Means: An Effective and Efficient K-Means Clustering Algorithm”, en *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom) (BDCloud-SocialCom-SustainCom)*, 2016, pp. 242–249.
- [5] Nurhayati, T. S. Kania, L. K. Wardhani, N. Hakiem, Busman, y H. Maarif, “Big Data Technology for Comparative Study of K-Means and Fuzzy C-Means Algorithms Performance”, en *2018 7th International Conference on Computer and Communication Engineering (ICCCE)*, 2018, pp. 202–207.
- [6] S. Sharma, A. K. Sharma, y D. Soni, “Enhancing DBSCAN algorithm for data mining”, en *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, 2017, pp. 1634–1638.
- [7] Chaomurilige, J. Yu, y M.-S. Yang, “Analysis of Parameter Selection for Gustafson–Kessel Fuzzy Clustering Using Jacobian Matrix”, *IEEE Trans. Fuzzy Syst.*, vol. 23, núm. 6, pp. 2329–2342, dic. 2015.
- [8] R. Pino Diez, A. Gomez Gomez, y N. de. Abajo Martinez, *Introduccion a la inteligencia artificial: sistemas expertos, redes neuronales artificiales y computacion evolutiva*. Servicio de Publicaciones, Universidad de Oviedo, 2001.
- [9] K. Schwab, *The fourth industrial revolution*. 2017.
- [10] A. González, “¿Qué es Machine Learning?”, 2014. [En línea]. Disponible en: <https://cleverdata.io/que-es-machine-learning-big-data/>. [Consultado: 17-jun-2019].
- [11] J. C. Tello, “Reconocimiento de patrones y el aprendizaje no supervisado”, 2014.
- [12] E. W. Forgy, “Cluster Analysis of Multivariate data: efficiency vs. Interpretability of Classifications,” *Biometrics*, Vol. 21, 1965, pp. 768–769.
- [13] Ester, Martin; Kriegel, Hans-Peter; Sander, Jörg; Xu, Xiaowei (1996). “A density-based algorithm for discovering clusters in large spatial databases with noise”. Simoudis, Evangelos; Han, Jiawei; Fayyad, Usama M., eds. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*. AAAI Press. pp. 226–231.
- [14] A. Y. Ng, M. I. Jordan, y Y. Weiss, “On Spectral Clustering: Analysis and an algorithm”, 2001.
- [15] X. Chen, Y. Su, Y. Chen, y G. Liu, “GK-means: an Efficient K-means Clustering Algorithm Based on Grid”, en *2009 International Symposium on Computer Network and Multimedia Technology*, 2009, pp. 1–4.
- [16] K. Peng, V. C. M. Leung, y Q. Huang, “Clustering Approach Based on Mini Batch Kmeans for Intrusion Detection System Over Big Data”, *IEEE Access*, vol. 6, pp. 11897–11906, 2018.
- [17] Q. Zhao, “Cluster Validity in Clustering Methods”, 2012.
- [18] K.-L. Wu y M.-S. Yang, “A cluster validity index for fuzzy clustering”, *Pattern Recognit. Lett.*, vol. 26, núm. 9, pp. 1275–1291, jul. 2005.

APÉNDICES

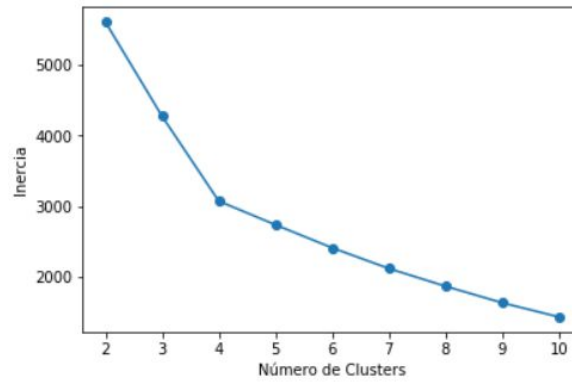
Apéndice 1. Diagrama de flujo de ingeniería de características.



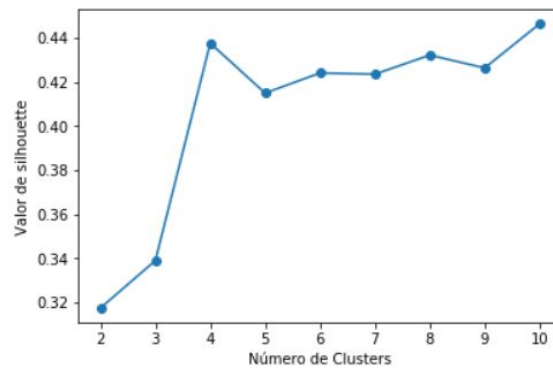
Apéndice 2. Diagrama de flujo del algoritmo K-Means



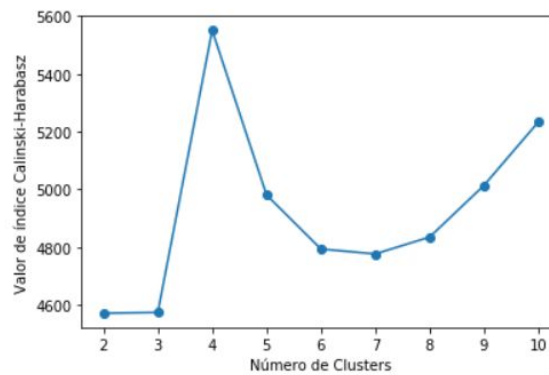
Apéndice 3. Gráfica de la inercia para la base de datos Dignatarios con el método K-Means



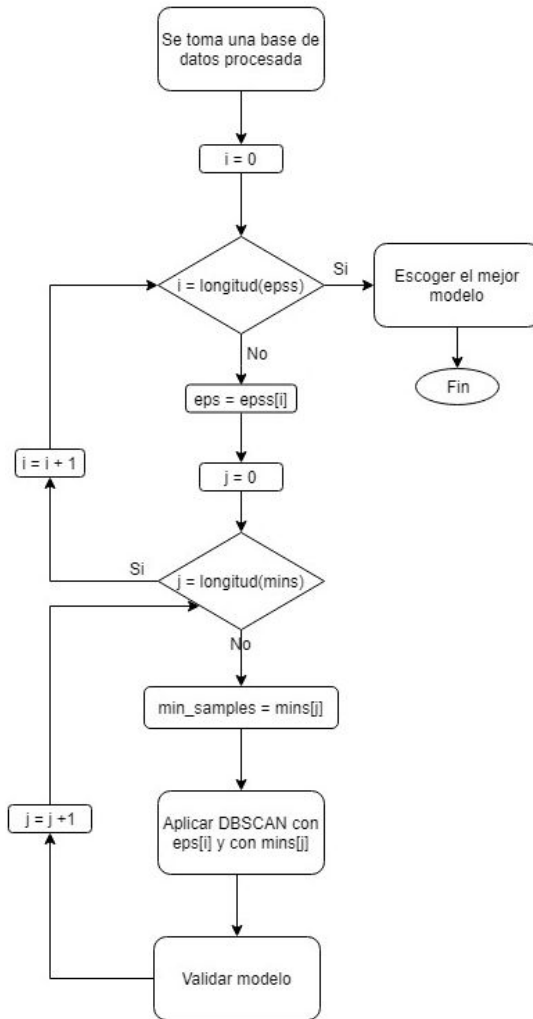
Apéndice 4. Gráfica del *silhouette_score* para la base de datos Dignatarios con el método K-Means



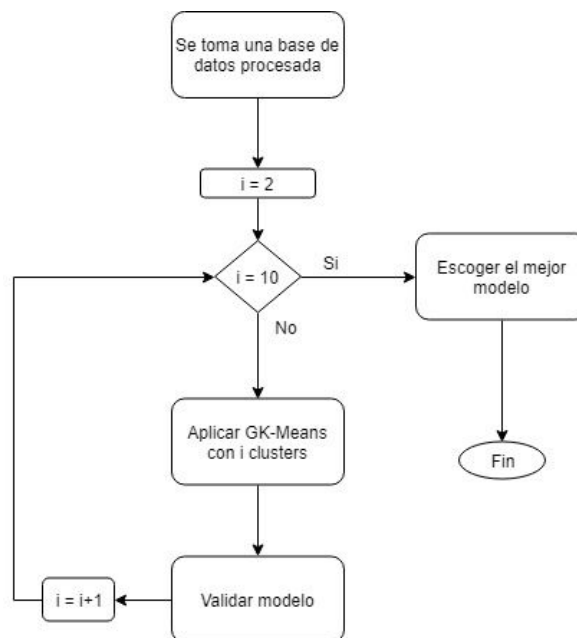
Apéndice 5. Gráfica del *calinski_harabasz_score* para la base de datos Dignatarios con el método K-Means



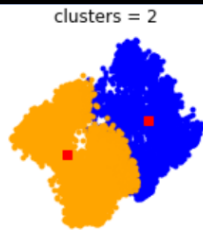
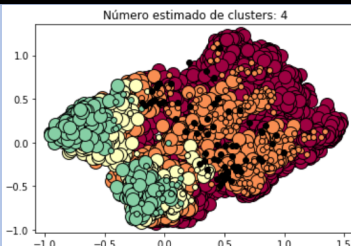

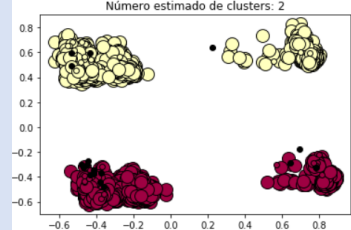
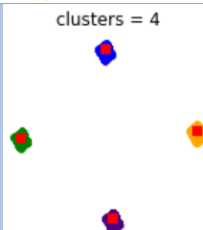


Apéndice 6. Diagrama de flujo del algoritmo DBSCAN

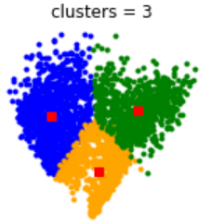


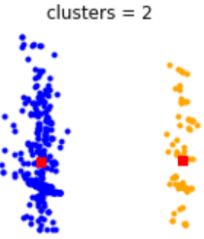
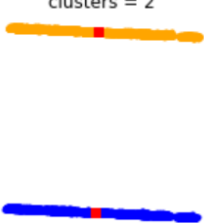


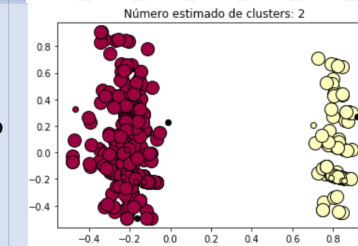
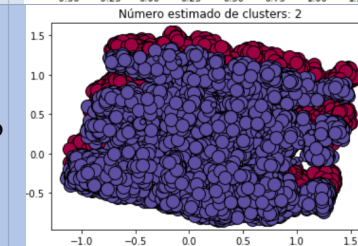
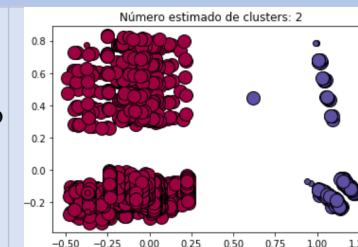
Apéndice 7. Diagrama de flujo del algoritmo GK-Means



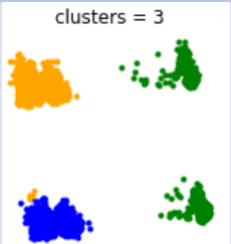
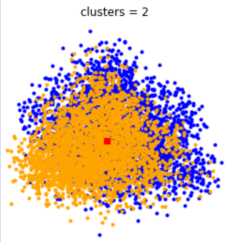
Apéndice 8. Tabla de resultado de los algoritmos K-Means y DBSCAN

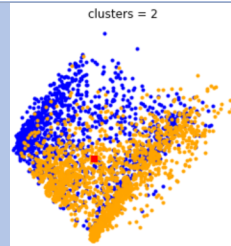

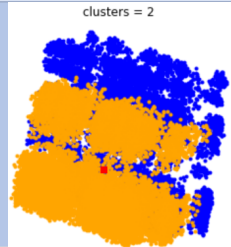
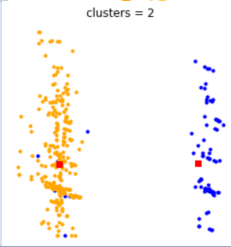
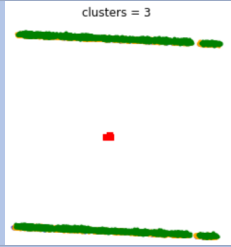
Nombre base de datos		K-Means	DBSCAN
Beneficiarios Discapacidad	Se realiza el experimento fijando 2 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.	clusters = 2 	Se realiza el experimento con un valor de e.p.s igual a 0.9, minimas muestras por cluster 50, y se obtuvo un resultado con 70 datos con ruido y una estimacion de 4 clusters. 
Defunciones 2015	Se realiza el experimento fijando 3 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.	clusters = 3 	Se realiza el experimento con un valor de e.p.s igual a 0.99, minimas muestras por cluster 3, y se obtuvo un resultado con 15 datos con ruido y una estimacion de 2 clusters. 
Dignatarios	Se realiza el experimento fijando 4 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.	clusters = 4 	Se realiza el experimento con un valor de e.p.s igual a 0.9, minimas muestras por cluster 30, y se obtuvo un resultado con 279 datos con ruido y una estimacion de 5 clusters. 
Encuesta juventud	Se realiza el experimento fijando 2 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.	clusters = 2 	No se obtuvieron resultados relevantes aplicando el DBSCAN a esta base de datos.

Individuos que participan	Se realiza el experimento fijando 2 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.		No se obtuvieron resultados relevantes aplicando el DBSCAN a esta base de datos.
Inversión por comunas y corregimientos Medellín 2016	Se realiza el experimento fijando 2 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.		Se realiza el experimento con un valor de e.p.s igual a 0.9, minimas muestras por cluster 30, y se obtuvo un resultado con 279 datos con ruido y una estimacion de 5 clusters.
Estado nutricional de menores de 6 años programa de crecimiento y desarrollo 2016	Se realiza el experimento fijando 2 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.		Se realiza el experimento con un valor de e.p.s igual a 0.9, minimas muestras por cluster 50, y se obtuvo un resultado con 0 ruido y 2 clusters diferentes.
Población indígena	Se realiza el experimento fijando 2 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.		Se realiza el experimento con un valor de e.p.s igual a 0.9, minimas muestras por cluster 50, y se obtuvo un resultado con 0 ruido y 2 clusters diferentes.
Varicela individual	Se realiza el experimento fijando 2 centros generados aleatoriamente, se hacen 1000 iteraciones, y con un error deseado (tol) de $1e-4$.		Se realiza el experimento con un valor de e.p.s igual a 0.5, minimas muestras por cluster 220, y se obtuvo un resultado con 2467 ruido y una estimacion 2 clusters diferentes.



<p>Afiliados al régimen subsidiado de Medellín 2015</p>	<p>Debido al tamaño de la base de datos se decidió usar una de las variaciones de KMeans que nos ofrece la librería de SKlearn llamada MiniBatchKMeans, esta utilizando parámetros con 2 centros generados aleatoriamente y con un error deseado (tol) de $1e-3$.</p>		<p>No se pudo ejecutar el algoritmo debido a que el tiempo de ejecución es muy alto.</p>
---	--	--	--

Nombre base de datos	Spectral	GK-Means
Beneficiarios Discapacidad	Como en los casos de K-Means y DBSCAN se obtuvieron buenos resultados, no fue necesaria la ejecución de este algoritmo.	No se pudo resolver la distancia de la matriz de covarianza inversa.
Defunciones 2015	<p data-bbox="533 552 904 679">Se realiza el experimento fijando 3 centros generados aleatoriamente, con una gamma igual a 3 y una afinidad en rbf.</p> 	No se pudo resolver la distancia de la matriz de covarianza inversa.
Dignatarios	Como en los casos de K-Means y DBScan se obtuvieron buenos resultados, no fue necesaria la ejecución de este algoritmo.	No se pudo resolver la distancia de la matriz de covarianza inversa.
Encuesta juventud	Como con K-Means se obtuvieron buenos resultados, no fue necesaria la ejecución de este algoritmo.	<p data-bbox="1326 1015 1697 1206">Se realiza el experimento con fijando 2 centros generados aleatoriamente, se hacen 300 iteraciones, con un error deseado (tol) de 1e-3 y un valor de e.p.s igual a 1e-18.</p> 

Individuos que participan	Como con K-Means se obtuvieron buenos resultados, no fue necesaria la ejecución de este algoritmo.	Se realiza el experimento con fijando 2 centros generados aleatoriamente, se hacen 300 iteraciones, con un error deseado (tol) de $1e-3$ y un valor de e.p.s igual a $1e-18$.	
Inversión por comunas y corregimientos Medellín 2016	Se realiza el experimento fijando 3 centros generados aleatoriamente, con una gamma igual a 3 y una afinidad en rbf.	No se pudo resolver la distancia de la matriz de covarianza inversa.	
Estado nutricional de menores de 6 años programa de crecimiento y desarrollo 2016	No se puede realizar la ejecución de este algoritmo para esta base de datos debido a limitaciones de hardware.	Se realiza el experimento con fijando 2 centros generados aleatoriamente, se hacen 300 iteraciones, con un error deseado (tol) de $1e-3$ y un valor de e.p.s igual a $1e-18$.	
Población indígena	Como en los tres casos restantes se obtuvieron buenos resultados, no fue necesaria la ejecución de este algoritmo.	Se realiza el experimento con fijando 2 centros generados aleatoriamente, se hacen 300 iteraciones, con un error deseado (tol) de $1e-3$ y un valor de e.p.s igual a $1e-18$.	
Varicela individual	Como en los algoritmos K-Means y DBSCAN se obtuvieron buenos resultados, no fue necesaria la ejecución de este algoritmo.	Se realiza el experimento con fijando 3 centros generados aleatoriamente, se hacen 300 iteraciones, con un error deseado (tol) de $1e-3$ y un valor de e.p.s igual a $1e-18$.	

Afiliados al régimen subsidiado de Medellín 2015	Como se obtuvo buen resultado con el MiniBatchKMeans, no fue necesaria la ejecución de este algoritmo.	No se pudo resolver la distancia de la matriz de covarianza inversa.
--	--	--