**Instituto Tecnológico de Costa Rica**

# Escuela de Ingeniería en Computación

**Programa de Maestría en Computación**

**Propuesta de algoritmo que combina el agrupamiento en subespacios basado en densidad y el agrupamiento basado en restricciones para la detección de grupos que incluyan atributos de interés en conjuntos de datos de alta dimensionalidad**

**Propuesta de Tesis sometida a consideración del Departamento de Computación, para optar por el grado de Magíster Scientiae en Computación, con énfasis en Ciencias de la Computación**

**Autor: Alonso Vallejos Peña**

**Tutor: Luis Alexander Calvo Valverde**

**Junio 2017**

# Titulo

Propuesta de algoritmo que combina el agrupamiento en subespacios basado en densidad SUBCLU y el agrupamiento basado en restricciones para la detección de grupos que incluyan atributos de interés en conjuntos de datos de alta dimensionalidad.

(Algorithm proposal that combines density based subspace clustering SUBCLU and constrained clustering for detecting clusters that include attributes of interest in high dimensional datasets)

# Resumen

El análisis de agrupamientos es una de las tareas más comunes en la minería de datos, utilizada a menudo en problemas de finanzas, biología, medicina y análisis de mercado entre otros [12].

Los datos de alta dimensionalidad plantean un desafío para los algoritmos de agrupamiento tradicionales, ya que las medidas de similitud convencionales utilizadas por estos no son significativas cuando se utilizan sobre el espacio completo de datos, afectando la calidad de los grupos. Ante esto, los algoritmos de agrupamiento de subespacios han sido propuestos como alternativa, buscando encontrar todos los grupos en todos los espacios del conjunto de datos [45].

Al detectar grupos en espacios de menor dimensionalidad, cada grupo detectado puede pertenecer a diferentes subespacios del conjunto de datos original [31]. Consecuentemente, atributos que el usuario considere de interés pueden ser excluidos en algunos o todos los grupos, perdiendo información importante y reduciendo el valor del resultado para los analistas.

Hoy en día, la mejora de los resultados en la detección de grupos más significativos, es considerada una de las áreas de mayor oportunidad en el análisis de agrupamientos de alta dimensionalidad. En particular, la capacidad de tomar en cuenta la relevancia de los atributos en la lógica de poda de los subespacios y la detección de los agrupamientos, es un campo abierto de estudio [30].

Para este trabajo de investigación, se estudió la posibilidad de dirigir la generación de agrupamientos en espacios de alta dimensionalidad, hacia subespacios de interés para el usuario. Para esto se extendió el algoritmo de agrupamiento de subespacios SUBCLU [1] con el algoritmo de agrupamiento por restricciones (Constrained clustering) [6], con el objetivo de dirigir la detección de grupos hacia espacios que incluyan estos atributos, y por ende generar grupos más significativos.

Usando este nuevo algoritmo (SUBCLU-R), se ejecutó un diseño de experimento para comparar el resultado del algoritmo SUBCLU y SUBCLU-R. En este diseño, primero se obtuvo el promedio de cohesión, separación e índice silueta de ambos algoritmos realizando múltiples pruebas en 3 muestras de un conjunto de datos, utilizando la misma configuración paramétrica tanto para

SUBCLU y SUBCLU-R. Luego, mediante una prueba estadística de hipótesis se compararon los promedios obtenidos por los algoritmos para determinar si las diferencias observadas son significativas. Finalmente, se realizó un análisis de los resultados, enfocado en comparar el desempeño del algoritmo propuesto contra el resultado del algoritmo original.

Los experimentos demostraron que es posible influir la detección de agrupamientos hacia aquellos que incluyan el atributo de interés, mediante la inclusión del concepto de restricciones para la poda de los subespacios. Con esta propuesta, *N-d* de los subespacios detectados (N siendo el total de subespacios, y d la cantidad de atributos del conjunto de datos) incluyen el atributo de interés del usuario. Al comparar los resultados de ambos algoritmos, se determinó que SUBCLU-R encuentra un porcentaje significativamente mayor agrupamientos que incluyen el atributo de interés del usuario, a la vez que no se encontraron diferencias estadísticamente significativas en la calidad interna de los agrupamientos detectados por ambos algoritmos.

# Abstract

Cluster analysis is one of data mining most common tasks, used frequently in finance, biology, medicine and market analysis problems [12].

High dimensional data poses a challenge to traditional clustering algorithms, where the similarity measures are not meaningful, affecting the quality of the groups. As a result, subspace clustering algorithms have been proposed as an alternative, aiming to find all groups in all spaces of the dataset [45].

By detecting groups on lower dimensional spaces, each group can belong to different subspaces of the original dataset [31]. Therefore, attributes the user may consider of interest can be excluded in some or all groups, decreasing the value of the result for the data analysts.

Currently, the improvement of the results and the detection of more significant groups, is considered one of the biggest opportunity areas in the cluster analysis of high dimensional data, particularly, the capability to consider the relevance of attributes on the subspace pruning logic and the group detection is an open research area [30].

For this project, a new algorithm is proposed, that combines SUBCLU [1] and the constraint clustering algorithms [6] that allows the users to identify variables as attributes of interest based on prior domain knowledge, targeting to direct group detection towards spaces that include users attributes of interest, thereafter, generating more meaningful groups.

Using this new algorithm (SUBCLU-R), an experiment was executed to compare the results from SUBCLU and SUBCLU-R. In this experiment, first, the average cohesion, separation and silhouette index was obtained for both algorithms by executing multiple tests in our dataset. Then, using a statistical hypothesis test we compared the obtained averages to find out if the observed differences were significant. Finally, a result analysis was performed, focused on comparing the performance of the proposed algorithm against the original SUBCLU.

The results indicate that it is possible to influence groupings towards those including attributes of interest, thanks to the inclusion of constrained clustering for subspace pruning. With this proposal, *N-d* detected subspaces (N is the total number of detected subspaces and d the number of attributes in the dataset) include the attribute of interest. After comparing both algorithm results, it was determined that SUBCLU-R detects a significantly higher percentage of groupings with the attribute of interest, while no significant statistical differences were found for the internal metrics of the groupings.

# Tabla de contenido

# Lista de Figuras

# Lista de Tablas

# Glosario

Agrupamientos: Conjunto de grupos resultado de un algoritmo de agrupamiento.

Dimensión: También llamado atributo, o característica. Describe los objetos de un conjunto de datos.

Épsilon: Para los algoritmos DBSCAN y SUBCLU, especifica el tamaño del vecindario.

Minpts: Para los algoritmos DBSCAN y SUBCLU, especifica la cantidad mínima de objetos para formar un grupo.

Subespacio: Subconjunto de dimensiones de un conjunto de datos.

Cohesión: Métrica que describe que tan similares son los objetos de un mismo grupo

Separación: Métrica que describe que tan distintos son los grupos entre sí

# Abreviaciones

ANOVA: Análisis de la varianza (Analysis of Variance)

BSS: Suma de cuadrados inter grupo (Between cluster sum of squares)

ELKI: Environment for Developing KDD-Applications Supported by Index-Structures

PCA: Principal Component Analysis

SUBCLU: density-connected Subspace Clustering

SUBCLU-R: density-connected Subspace Clustering – Restricted (propuesto en este trabajo)

WSS: Suma de cuadrados inter grupo (Within cluster sum of squares)

# 1. Introducción

El análisis de agrupamientos divide los datos en grupos que son significativos, basados únicamente en los datos que describen los objetos, es decir, los diferentes atributos del conjunto de datos. La finalidad de este tipo de análisis es que los objetos dentro de los grupos generados sean lo más similares posible entre sí, y lo más distintos posible con los objetos de otros grupos [19].

Comúnmente, en datos de alta dimensionalidad, muchos atributos son irrelevantes y pueden ocultar grupos existentes en datos ruidosos, dificultando la capacidad de los algoritmos de agrupamiento de encontrar grupos relevantes [2].

Técnicas de reducción de dimensionalidad como Principal Component Analysis (PCA), filtros de alta correlación y filtros de baja varianza son utilizadas comúnmente para eliminar atributos irrelevantes en los datos [38]. Aunque útiles para otras tareas de minería de datos [39], las técnicas de reducción no son muy prácticas para buscar grupos en datos de alta dimensionalidad, pues generan un subespacio único de los datos, y los grupos pueden ocultarse en distintos subespacios [31]. Ante esta deficiencia, surgen los algoritmos de agrupamiento de subespacios.

Los algoritmos de agrupamiento de subespacios son capaces de detectar grupos en diferentes espacios de menor dimensionalidad al conjunto de datos original [37]. Al poder encontrar grupos en distintos subespacios, cada grupo detectado puede poseer diferentes atributos del espacio original. Consecuentemente, atributos de interés pueden ser excluidos en algunos o todos los grupos identificados.

La exclusión de atributos de interés para el usuario, puede ocasionar un decremento en el valor de los grupos. La capacidad de utilizar conocimiento previo del dominio, en la generación de los subespacios, permitiría guiar la detección hacia espacios que incluyan los atributos de valor. Sin embargo, a pesar de la existencia de múltiples algoritmos para otorgar distintos pesos a los atributos [27][28][34], ninguno permite guiar la detección de agrupamientos en espacios que incluyan atributos de interés previamente identificados por el usuario [30].

Este proyecto se enfoca en mejorar los resultados en datos de alta dimensionalidad, al dar prioridad a grupos que incluyan atributos de interés para el usuario. Para este fin, se modificó el algoritmo de agrupamiento de subespacios SUBCLU [1] con el algoritmo de agrupamiento de restricciones (constrained clustering) [6]. A partir de este momento nos referiremos a este algoritmo como SUBCLU-R.

Para lograr influir la detección de los agrupamientos resultado y la poda de los subespacios, se propone el uso de la lógica del algoritmo de agrupamiento por restricción y la inclusión de un nuevo parámetro, "atributo de interés". El algoritmo de agrupamiento por restricción es utilizado para definir reglas sobre los grupos resultado, en este caso, en el contexto de los agrupamientos de subespacios, se propone su uso para definir restricciones sobre los subespacios a analizar, específicamente, sobre que atributos deben contener los subespacios.

Se encontró que con SUBCLU-R es posible guiar la detección de los subespacios y consecuentemente influir en los agrupamientos finales basado en el atributo de interés. Al comparar los resultados de ambos algoritmos, se determinó que SUBCLU-R encuentra un porcentaje significativamente mayor agrupamientos que incluyen el atributo de interés del usuario, a la vez que no se encontraron diferencias significativas en la calidad interna de los agrupamientos detectados por ambos algoritmos.

En la siguiente sección se presenta el marco teórico de este proyecto, como una breve introducción a la propuesta y al trabajo a realizar.

# 2. Marco teórico

En esta sección se presenta un marco teórico alrededor del problema y de la propuesta que se presenta en este proyecto.

## 2.1 Minería de datos

El presente trabajo se enmarca en el área de minería de datos, que en términos generales es la exploración y análisis de grandes conjuntos de datos para descubrir patrones y correlaciones ocultas [20]. Las tareas más comunes de minería de datos son: clasificación, estimación, predicción, afinidad de agrupamientos, análisis de agrupamientos y descripción de datos.

Este proyecto se desarrolla en el área de análisis de agrupamientos, a continuación, una breve descripción y una introducción a los conceptos y métodos más representativos.

## 2.2 Análisis de agrupamientos

El análisis de agrupamientos es el proceso de agrupar un conjunto de objetos en múltiples grupos, de manera que los objetos dentro del mismo grupo sean tan parecidos entre sí, y tan diferentes con respecto a los objetos de otros grupos como sea posible. La similitud entre los objetos es evaluada en base al valor de los atributos de los objetos y a menudo involucran medidas de distancia [19].

Los algoritmos de agrupamiento y los de clasificación son muy similares en naturaleza, la principal diferencia es que los algoritmos de agrupamiento no requieren de categorías o clases como datos de entrenamiento, es decir, no necesitan que se conozca de antemano el conjunto de grupos resultado y el tipo de objetos que se incluirá en cada uno.

El análisis de agrupamientos ha sido ampliamente utilizado en muchas aplicaciones como inteligencia de negocios, reconocimiento de imágenes, biología y seguridad. En reconocimiento de imágenes, por ejemplo, los algoritmos de agrupamiento pueden ser utilizados para descubrir caracteres en sistemas de reconocimiento de escritura.

En la actualidad existen muchos algoritmos de agrupamiento, estudios como [12][19], hacen un excelente análisis y resumen del estado actual del arte. En general, los métodos más relevantes en la literatura pueden ser clasificados en las categorías descritas a continuación.

## 2.3 Métodos de análisis de agrupamientos

En esta sección se presentan distintos métodos de agrupamiento.

### 2.3.1 Métodos de particiones:

Es considerado el método más simple y fundamental. Dado un conjunto de *n* objetos, un método de partición construye *k* particiones de los datos, donde cada partición representa un grupo.

La mayoría de los métodos de partición son basados en distancia. Dado *k* número de particiones a construir, el método crea las particiones iniciales. Luego usa una técnica de reubicación iterativa que intenta mejorar las particiones moviendo objetos entre grupo [19].

En general, los algoritmos de reubicación iterativa convergen localmente, y la solución óptima global no puede garantizarse. Dado que el número de objetos en cualquier conjunto de datos es finito, el problema puede solucionarse utilizando métodos de búsqueda exhaustiva. Sin embargo, encontrar la partición global óptima es un problema NP-complejo [46].

Una representación gráfica del proceso iterativo de los métodos de particiones puede observarse en la Figura 1, donde el algoritmo crea unos grupos iniciales en la Figura 1.a, y luego los objetos pueden moverse a grupos más similares en cada iteración como lo representa la Figura 1.b hasta encontrar los grupos finales.

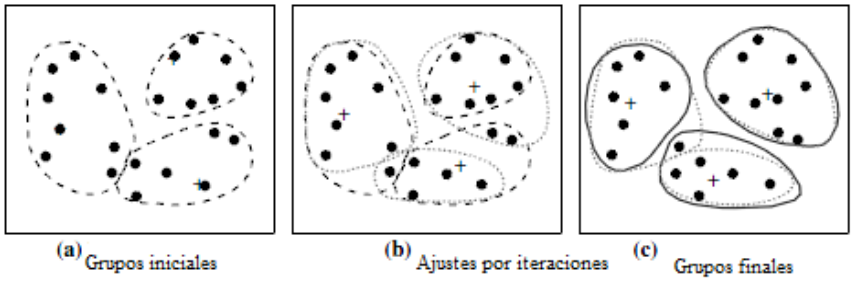



Figura 1. Representación gráfica de agrupamiento basado en particiones [19]

### 2.3.2 Métodos jerárquicos:

Los métodos jerárquicos crean una descomposición del conjunto de objetos, agrupando los datos en una jerarquía o árbol de grupos, generando de esta manera grupos en diferentes niveles [19], lo que es muy útil para resumir y visualizar los agrupamientos.

Los algoritmos jerárquicos pueden ser basados en distancia, densidad o basados en continuidad. Además, estos se subdividen en algoritmos divisivos y aglomerativos. La diferencia entre ambos enfoques es la estrategia de agrupamiento, el enfoque divisivo detecta los grupos de abajo hacia arriba (bottom-up), permitiendo que cada objeto tenga su propio grupo al inicio e iterativamente unir los grupos en grupos de mayor tamaño. El enfoque divisivo forma los grupos de arriba hacia abajo (top-down), creando un único grupo inicial con todos los objetos, e iterativamente particionar los grupos en subconjuntos más pequeños. En ambas estrategias, el usuario puede determinar el número deseado de grupos como una condición de terminación.

En la Figura 2 se puede apreciar la representación gráfica de las iteraciones de un algoritmo aglomerativo (línea superior) y las iteraciones de uno divisivo (línea inferior)

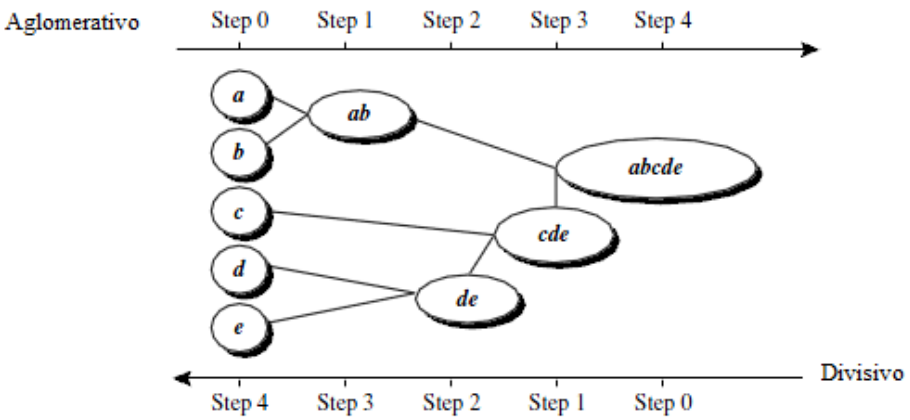



Figura 2. Representación gráfica de algoritmo jerárquico aglomerativo y divisivo [19]

### 2.3.3 Métodos basados en densidad

En los métodos basados en densidad, los grupos se califican como regiones densas de objetos en el espacio de datos, separado por regiones de menor densidad [4]. La idea general es que un grupo puede continuar creciendo siempre y cuando la densidad (número de objetos en un espacio determinado) en el vecindario, sobrepase un límite determinado (Minpts).

Este método a diferencia de los otros enfoques, es capaz de detectar grupos no esféricos, como los representados en la Figura 3.

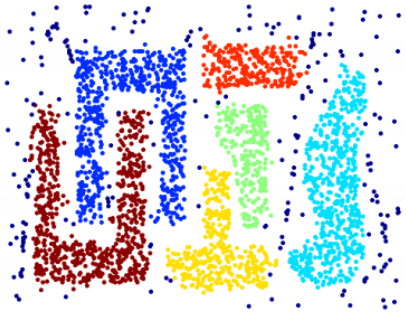


Figura 3. Representación gráfica de grupos no esféricos [47].

Uno de los algoritmos más representativos de los métodos de agrupamiento basados en densidad es DBSCAN.

**DBSCAN**

DBSCAN es uno de los algoritmos de agrupamiento más relevantes y más citados en la literatura. Este algoritmo requiere dos parámetros: e*ps* (tamaño del vecindario) y *minpts* (cantidad mínima de objetos para formar un grupo).

El algoritmo inicia en un punto arbitrario no visitado, la vecindad-*eps* de este punto es visitada, y si contiene suficientes puntos, se inicia un grupo sobre el mismo. De lo contrario, el punto es etiquetado como ruido (aún puede pertenecer a otra vecindad que lo incluya en el grupo).

Si un punto se incluye en la parte densa de un grupo, su vecindad-*eps* también forma parte del grupo. De esta forma, todos los puntos de esta vecindad-*eps* se agregan al grupo, al igual que las vecindades-*eps* de estos objetos que sean lo suficientemente densas. De esta forma el grupo continua expandiéndose hasta que forma un grupo completamente conectado, o las densidades no alcancen el límite mínimo [12].

Las principales ventajas de DBSCAN son la capacidad de detectar grupos con formas arbitrarias y que no necesitan conocimiento previo del número de grupos como si lo requieren algoritmos como los basados en particiones k-means y k-medoids.

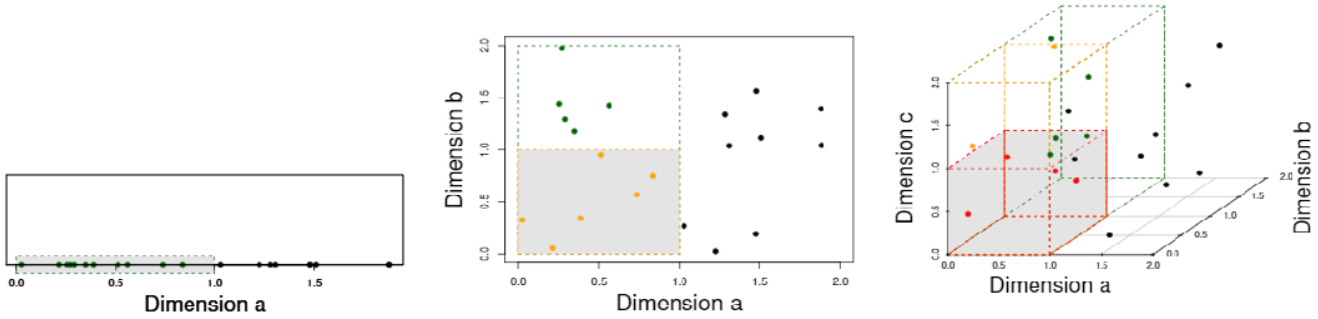
Las principales desventajas de este algoritmo son: los problemas en datos donde los grupos tengan grandes diferencias en las densidades [48], pero más relevante a este proyecto, es la dependencia de la noción de distancia, que se vuelve muy poco eficaz en datos de alta dimensionalidad [19].

## 2.4. Agrupamiento en conjuntos de datos de alta dimensionalidad

Los métodos y algoritmos mencionados en la sección 2.3, funcionan bien mientras los datos no tengan más de 10-15 atributos [19][3]. En datos de alta dimensionalidad, los algoritmos tradicionales luchan por detectar grupos relevantes, pues estos analizan el espacio completo de datos y los grupos pueden estar ocultos en un subconjunto de atributos [31].

Otra razón por la que los algoritmos de agrupamiento luchan con datos con gran cantidad de atributos, es por la maldición de la dimensionalidad [5]. A medida que la cantidad de variables aumenta en un conjunto de datos, las medidas de distancia se vuelven insignificantes. Es decir, en datos de alta dimensionalidad, las distancias entre un objeto y su vecino más lejano y más cercano son casi iguales.

En la figura 4 se representa gráficamente la maldición de la dimensionalidad. En la Figura 4.a se puede observar como los objetos proyectados en un eje x se encuentran dentro de un mismo cajón, en la Figura 4.b agregando una segunda dimensión se puede apreciar como los datos se dispersan y solamente 6 objetos se encuentran dentro de un mismo cajón, en la Figura 4.c agregando una tercera dimensión, los datos se dispersan aún más y solamente 4 objetos se encuentran en el mismo cajón.




(a) 11 objetos en 1 cajón (b) 6 objetos en 1 cajón (c) 4 objetos en 1 cajón

Figura 4. Representación gráfica de la maldición de la dimensionalidad [2].

En general, los métodos de reducción de dimensionalidad como PCA, Filtros de alta correlación y filtros de baja varianza [33] son utilizados para construir espacios con menor cantidad de variables.

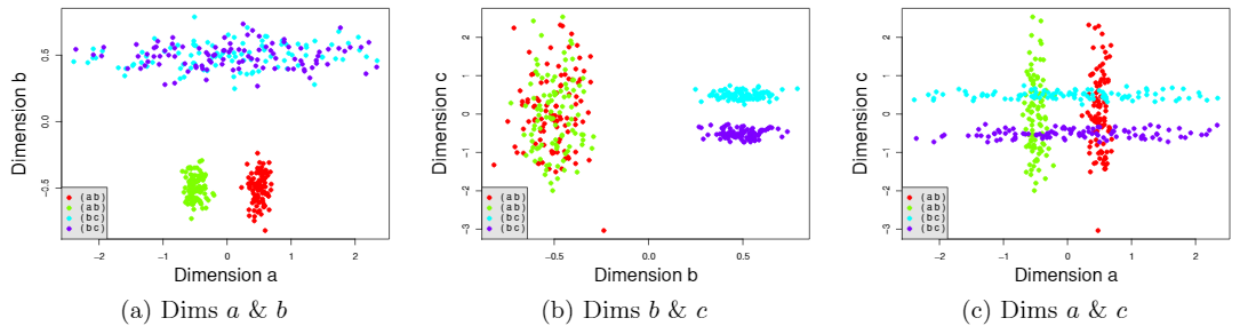
Sin embargo, estos métodos son técnicas de reducción global, pues generan un solo subespacio del espacio original de datos. Ya que distintos grupos pueden existir bajo diferentes subespacios, se descarta el uso de los algoritmos de reducción como una opción para los algoritmos de agrupamiento [31].

Los algoritmos de agrupamiento de subespacios surgen como respuesta ante el problema de la dimensionalidad.

### 2.4.1 Algoritmos de Agrupamiento de Subespacios

Los métodos de agrupamiento de subespacios son una extensión de los algoritmos de agrupamiento tradicionales, que intentan encontrar grupos en distintos subespacios del mismo conjunto de datos. Estos métodos adicionalmente a la organización de los objetos en distintos grupos, también deben definir el subconjunto de atributos relevantes para cada grupo [3].

En la Figura 5 se puede observar la importancia de la identificación de los atributos relevantes para cada grupo en un conjunto de datos de 3 dimensiones. En la Figura 5.a y 5.b solo 2 grupos pueden ser detectados en los subespacios "a, b" y "b,c" respectivamente. Pero si se analizan los datos utilizando las variables "a,c", como lo representa la Figura 5.c es posible detectar los 4 grupos ocultos




(a) Dims $a$ & $b$ (b) Dims $b$ & $c$ (c) Dims $a$ & $c$

Figura 5. Representación gráfica de 4 agrupamientos en distintos subespacios [3]

A pesar de ser un área que surgió en 1998, a la fecha existen múltiples implementaciones y variables de algoritmos de agrupamiento de subespacios, una lista exhaustiva puede encontrarse en [11][15]. Este proyecto extiende el algoritmo SUBCLU, el cual es presentado a continuación.

**SUBCLU**

Este método utiliza el concepto de objetos conectados por densidad que utiliza el algoritmo DBSCAN, gracias a esto, es capaz de detectar grupos de forma no esférica. SUBCLU es eficiente en la poda de subespacios gracias a la propiedad de monotonicidad, a la vez que detecta los mismos grupos que DBSCAN hubiese encontrado si se aplicara por separado en cada subespacio [1].

SUBCLU está basado en un algoritmo voraz de abajo hacia arriba para detectar los grupos conectados por densidad en todos los subespacios. El pseudocódigo del algoritmo es presentado en la Figura 6. Como puede observarse, el algoritmo en general construye una envoltura alrededor del algoritmo DBSCAN, ejecutando la lógica original del algoritmo por cada subespacio considerado relevante.

```
SUBCLU(SetOfObjects DB, Real ε, Integer m)
    /* STEP 1 Generate all 1-D clusters */
    S_1 := ∅    // set of 1-D subspaces containing clusters
    C_1 := ∅    // set of all sets of clusters in 1-D subspaces
    FOR each a_i ∈ A DO
        C^{a_i} := DBSCAN(DB, {a_i}, ε, m)    // set of all clusters in subspace a_i;
        IF C^{a_i} ≠ ∅ THEN    // at least one cluster in subspace {a_i} found
            S_1 := S_1 ∪ {a_i};
            C_1 := C_1 ∪ C^{a_i};
        END IF
    END FOR
    /* STEP 2 Generate (k+1)-D clusters from k-D clusters */
    k := 1;
    WHILE C_k ≠ ∅
        /* STEP 2.1 Generate (k+1)-dimensional candidate subspaces */
        CandS_{k+1} := GenerateCandidateSubspaces(S_k);
        /* STEP 2.2 Test candidates and generate (k+1)-dimensional clusters */
        FOR EACH cand ∈ CandS_{k+1} DO
            // Search k-dim subspace of cand with minimal number of objects in the clusters
            bestSubspace := min_{s∈S_k ∧ s⊆cand} Σ_{C_i∈C^s} |C_i|
            C^{cand} := ∅;
            FOR EACH cluster cl ∈ C^{bestSubspace} DO
                C^{cand} = C^{cand} ∪ DBSCAN(cl, cand, ε, m);
                IF C^{cand} ≠ ∅ THEN
                    S_{k+1} := S_{k+1} ∪ cand;
                    C_{k+1} := C_{k+1} ∪ C^{cand};
                END IF
            END FOR
        END FOR
        k := k + 1
    END WHILE
```

Figura 6. Pseudocódigo de algoritmo SUBCLU [1].

## 2.5. Agrupamiento con restricciones

Los usuarios a menudo poseen conocimiento previo de dominio que les gustaría integrar en el análisis de agrupamientos, los métodos de agrupamiento con restricciones permiten la inclusión de requerimientos adicionales que permiten guiar la detección de los grupos. Existen tres tipos diferentes de restricciones: restricciones a nivel de instancia, a nivel de grupo, y a nivel de medida de similitud.

**Restricciones a instancias:**

Especifican como un conjunto de objetos debe ser agrupado. Los tipos más comunes de restricciones de esta categoría son las restricciones *must-link* y *cannot-link.* Las restricciones *must-link* permiten definir cuando dos objetos (x,y) deben pertenecer al mismo grupo, similarmente, las restricciones *cannot-link* permiten definir cuando dos objetos (x,y) no pueden pertenecer al mismo grupo [20].

**Restricciones a grupos:**

Este tipo de restricción permite definir requerimientos específicos a los grupos, normalmente utilizando atributos que lo definen. Por ejemplo, es posible definir restricciones definiendo un diámetro máximo del grupo, o que la forma del grupo deba ser convexa [20].

**Restricciones a medidas de similitud**

Estas restricciones especifican requerimientos que el cálculo de similitud debe respetar. Por ejemplo: agrupar personas moviéndose en un parque podría hacerse utilizando la distancia Euclidiana para calcular la distancia entre dos puntos. Se podría implementar una restricción que especifique que la trayectoria implementando la distancia más corta no puede atravesar una pared [20].

## 2.6. Métodos de evaluación de agrupamientos

La validación de los algoritmos es un paso esencial en el análisis de agrupamientos, ya que cualquier algoritmo puede generar grupos como resultado de su ejecución, sin embargo, esto no significa que los grupos detectados tengan un porcentaje aceptable de exactitud. Existen dos tipos principales de validaciones de agrupamientos, basados en la disponibilidad o ausencia de *verdad base* (ground truth en inglés): métodos externos y métodos internos.

**Medidas de evaluación externa**

Estas métricas son basadas en comparaciones entre el resultado del sistema de agrupamiento y un *estándar dorado* o *verdad base*, que usualmente es construido manualmente por expertos del dominio [51]. La tarea de un método extrínseco, es asignar un valor $Q\ (C, C_g)$ a un agrupamiento $C$, dada una verdad base $C_g$. En general se considera que una medida extrínseca es efectiva, si toma en cuenta los siguientes cuatro criterios [20].

- **Homogeneidad del agrupamiento**: Este criterio requiere que entre más puros sean los grupos, mejor sea la calificación del método de agrupamiento. Por ejemplo, si al evaluar dos algoritmos de agrupamiento C1 y C2, C1 agrupa más objetos en el grupo correcto que el algoritmo C2, de acuerdo a las categorías (agrupamientos de la verdad base), entonces el algoritmo C1 debe tener un mayor puntaje [20].
- **Completitud del grupo**: Este criterio es la contraparte de la homogeneidad de grupo. Si dos objetos pertenecen a la misma categoría, entonces los objetos deben pertenecer al mismo grupo.
- **Bolsa de trapos (Rag bag)**: Una noción general de las tareas de agrupamiento establece que es más dañino introducir desorden a un grupo desordenado, que a un grupo ordenado [51]. Este criterio establece que ubicar objetos erróneos en un grupo puro, debe ser penalizado más que si se ubicara en un grupo de objetos misceláneos [20].
- **Preservación de agrupamiento pequeño**: Si una categoría pequeña es dividida en grupos más pequeños, es más probable que esos grupos pequeños se conviertan en ruido, y que por lo tanto esa categoría no sea descubierta por el algoritmo. Este criterio establece que partir categorías pequeñas en grupos más pequeños, es más dañino que partir categorías grandes en grupos más pequeños [20].

Para utilizar cualquier métrica externa es necesario contar con una verdad base, sin embargo, ya que no se cuenta con este tipo de datos para ninguno de los conjuntos de datos seleccionados para este experimento (sección 5.1), las métricas externas no se utilizarán en este proyecto.

**Evaluación interna:**

Cuando no se cuenta con una verdad base, es necesario utilizar métodos internos para evaluar la calidad de los agrupamientos. En general, este tipo de métodos evalúan qué tan bien agrupados y compactos son los grupos [20]. Ejemplos de métricas de métodos de evaluación interna son las siguientes:

- **Cohesión de los grupos:** Mide qué tan compacto es un grupo, o qué tan cercanos son todos los objetos dentro de un grupo [19].
- **Separación de los grupos:** Mide qué tan separados se encuentran los grupos entre sí [19].
- **Índice de silueta:** Mide qué tan similar es un objeto a su propio grupo y comparado a los demás grupos [21]. Esta métrica es un cálculo que evalúa los agrupamientos basado en las métricas de cohesión y separación.

En este proyecto se propone utilizar las 3 métricas expuestas en esta sección. Más detalles sobre las métricas a desarrollar en la sección 3.5 .

# 3 Propuesta del proyecto

En esta sección se define el problema en el que el proyecto se enfocó, la propuesta del trabajo desarrollada así como los trabajos relacionados.

## 3.1 Definición del problema de investigación

El análisis de grupos es una de las tareas más comunes en minería de datos, ideal para casos donde se necesitan formar grupos de objetos, bajo algún concepto de similitud de forma no supervisada. Este tipo de algoritmos ha sido aplicado con éxito en muchos dominios, por ejemplo: biología, finanzas, mercadeo, ente otros [12][32].

Avances en la recolección y manejo de datos ha ocasionado un aumento en la cantidad de dimensiones a analizar. Esto ha generado problemas para los algoritmos de agrupamiento tradicionales, que trabajan sobre el espacio completo de datos, ya que los grupos a detectar pueden estar ocultos en un subconjunto de atributos, e inclusive, distintos grupos pueden existir en diferentes subconjuntos de atributos [31].

Técnicas de selección de atributos son utilizadas a menudo para reducir la dimensionalidad, tales como PCA, Filtros de alta correlación y filtros de baja varianza [33]. Estos métodos son técnicas de reducción global, pues generan un solo subespacio del espacio original de datos. Ya que distintos grupos pueden existir bajo diferentes subespacios, los algoritmos de reducción no son una opción para los algoritmos de agrupamiento [31].

Los algoritmos de agrupamiento de subespacios surgen como respuesta ante el problema de la dimensionalidad, buscando encontrar los grupos en todos los subespacios. Aunque eficientes para resolver este problema, dos limitaciones han sido expuestas: la necesidad de manejo de datos más complejos (por ejemplo: datos de streaming infinitos, datos con ruido o datos categóricos), y la necesidad de mejores resultados de agrupamiento (por ejemplo: reducción de grupos, eliminación de grupos traslapados en diferentes subespacios o grupos más significativos) [30].

Este proyecto se concentra en la segunda limitación, específicamente en la detección de grupos más significativos, que incluyan atributos de interés para el usuario. Dada la naturaleza de este tipo de algoritmos, donde la detección de grupos se lleva a cabo en espacios de menor dimensionalidad, atributos de interés pueden ser excluidos en algunos o todos los grupos identificados. La detección de grupos que no incluyan atributos valiosos para el usuario, aunque optimizando medidas de calidad de grupos como cohesión y separación, reduce el valor de los agrupamientos y del análisis que se pueda realizar basado en estos.

Para ilustrar este problema, se puede considerar el caso de negocio de la corporación Intel en el departamento de ventas y mercadeo:

En Intel®, se consolidan datos de consumo de dispositivos con tecnología Intel, provenientes de distintas compañías de análisis de mercado. Los datos recolectados representan las ventas analizadas por más de 230 atributos en total, detallando distintos aspectos del consumo tales como: el lugar de venta, fabricante del dispositivo, unidades vendidas, ganancias y características del procesador entre otros.

Los datos son analizados en busca de oportunidades o patrones en los datos, que permitan guiar las estrategias de venta y mercadeo; lo anterior en un intento por maximizar las ventas. Actualmente, mucho trabajo manual es necesario por parte de los expertos de dominio para poder encontrar dichos patrones de consumo.

El uso de tareas de clasificación no supervisada permitiría encontrar patrones iguales o de mayor calidad en escenarios como el anterior. Sin embargo, en casos como el de Intel, los agrupamientos carecen de valor si son detectados en subespacios que omiten atributos de interés para el análisis, por ejemplo: nombre de procesador, país, entre otros.

Este trabajo pretende atacar precisamente este tipo de problemas, agrupamientos sobre conjunto de datos de alta dimensionalidad, donde los grupos detectados deban incluir grupos de atributos de interés para el usuario.

En la siguiente sección se describe la propuesta general del proyecto a desarrollar.

## 3.2 Propuesta del proyecto

Considerando el problema expuesto en la sección anterior, este proyecto propone un nuevo algoritmo de análisis de agrupamiento en conjuntos de alta dimensionalidad y capaz de detectar grupos en subespacios con atributos de interés.

Para esto, se extendió la funcionalidad del algoritmo de agrupamiento SUBCLU [1], mediante la inclusión de teoría de agrupamiento por restricciones [6]. Se escogió modificar este algoritmo, ya que es un método basado en densidades, lo que permite detección de grupos de formas irregulares (no elípticas), además, es un algoritmo popular con múltiples implementaciones de código abierto.

Este nuevo algoritmo permite guiar la poda de subespacios, hacia aquellos que el usuario considere de interés para su análisis, esto por medio de la inclusión de un nuevo parámetro: “atributo de interés”. Al lograr esto, los agrupamientos detectados incluirían más atributos de interés; con poco impacto en la calidad de los agrupamientos en términos de separación, cohesión e índice silueta cuando son comparados contra los generados por el algoritmo original.

A continuación, se presenta el seudocódigo del algoritmo SUBCLU original.

**SUBCLU**$(DB, eps, MinPts)$

$S_1 := \emptyset$
$C_1 := \emptyset$
**for each** $a \in Attributes$
  $C^{\{a\}} =$ **DBSCAN**$(DB, \{a\}, eps, MinPts)$
  **if**$(C^{\{a\}} \neq \emptyset)$
    $S_1 := S_1 \cup \{a\}$
    $C_1 := C_1 \cup C^{\{a\}}$
  **end if**
**end for**
// In a second step, $k+1$-dimensional clusters are built from $k$-dimensional ones:
$k := 1$
**while**$(C_k \neq \emptyset)$
  **CandS**$_{k+1}$ := **GenerateCandidateSubspaces**$(S_k)$
  **for each** $cand \in$ **CandS**$_{k+1}$
    **bestSubspace :=** $\min_{s \in S_k \wedge s \subset cand} \sum_{C_i \in C^s} |C_i|$
    $C^{cand} := \emptyset$
    **for each cluster** $cl \in C^{\text{bestSubspace}}$
      $C^{cand} := C^{cand} \cup$ **DBSCAN**$(cl, cand, eps, MinPts)$
      **if** $(C^{cand} \neq \emptyset)$
        $S_{k+1} := S_{k+1} \cup cand$
        $C_{k+1} := C_{k+1} \cup C^{cand}$
      **end if**
    **end for**
  **end for**
  $k := k + 1$
**end while**
**end**

La lógica del algoritmo de agrupación por restricciones se incluyó en el método GenerateCandidateSubspaces($S_k$), el cambio propuesto en este trabajo encuentra resaltado en verde.

```
GenerateCandidateSubspaces(S_k)
  CandS_{k+1} := ∅
  for each s_1 ∈ S_k
    for each s_2 ∈ S_k
      if (s_1 and s_2 differ in exactely one attribute)
        CandS_{k+1} := CandS_{k+1} ∪ {s_1 ∪ s_2}
      end if
    end for
  end for
  // Pruning of irrelevant candidate subspaces
  for each cand ∈ CandS_{k+1}
    for each k-element s ⊂ cand
      if (s ∉ S_k and validSubspace( s, attributeOfInterest))
        CandS_{k+1} = CandSa_{k+1} \ {cand}
      end if
    end for
  end for
end
```

La lógica del algoritmo de agrupamiento por restricciones se implementa en el paso de generación de subespacios en toman forma de filtro durante la poda de los subespacios irrelevantes. Este paso se ejecuta k veces (k= número de dimensiones), primero para generar subespacios de 1 dimensiones, 2 dimensiones, y así sucesivamente.

## 3.3 Trabajos relacionados

Desde que Aggrawal et al [23] propusieron los algoritmos de agrupamiento de subespacios, como solución para el problema de clasificación no supervisada en datos de alta dimensionalidad, mucho trabajo se ha realizado [24][25]26]. Entre los trabajos más relevantes para este proyecto, en [1] se propone SUBCLU, algoritmo que adopta la definición de grupos conectados por densidad propuesto en DBSCAN[25], la ventaja principal de este método es la detección de agrupamientos de distintas formas y tamaños (grupos de forma irregular).

En el área de agrupamiento semi-supervisado, Kailing et al [6] proponen el algoritmo de agrupamiento por restricciones, que permite a los usuarios especificar qué instancias pueden o no pueden agruparse en el mismo grupo. En este trabajo, logran extender con éxito el algoritmo COBWEB, y subsecuentemente, en [7] el K-means. Estos algoritmos, sin embargo, no son aplicables a datos de alta dimensionalidad.

En algoritmos de agrupamiento de subespacios, muy poco trabajo se ha hecho en el área de aprendizaje semi-supervisado, inclusive, ha sido identificado como una de las áreas con más oportunidades de crecimiento en [30]. En [26] Elisa et al. integran con éxito los algoritmos de agrupamiento de subespacios CLIQUE (basado en celdas) y NCLUSTER (basado en ventanas); con el algoritmo restricción de agrupamientos a nivel de instancia. Este trabajo se enfoca principalmente en estudiar cómo la introducción de restricciones de agrupamientos a los algoritmos de subespacio, influyen en el proceso y el desempeño de estos, pero no se realiza ningún tipo de análisis en la guía de elección de subespacios.

Otro enfoque utilizado para lidiar con atributos irrelevantes o ruidosos en conjuntos de alta dimensionalidad, es el uso de pesos para distintos atributos. En [27] se propone SYNCLUS, un método de agrupamiento que utiliza peso para grupos de atributos en el proceso de agrupamiento. SYNCLUS calcula el peso de cada atributo y los pesos de los grupos de atributos son dados por el usuario. Este método, sin embargo, es computacionalmente costoso, por lo que se consideró que no puede utilizarse para datos de alta dimensionalidad o bases de datos grandes (publicado en 1984), hoy en día podría revisitarse con implementaciones basadas en frameworks de *Big Data*, como Cloudera. Huang et. al proponen W-K-means [28], que calcula automáticamente el peso de

los atributos, donde el peso por atributo es calculado con base en la suma de la varianza intra-grupo para cada una de las dimensiones, esto permite identificar los atributos ruido y su efecto en el resultado puede ser minimizado. En [30], sin embargo, se expone que W-K-means pierde las relaciones directas entre la escala de los valores de los atributos y los pesos de los atributos, y ante esto extienden el algoritmo con la métrica de Minkowski, ya que el exponente de Minknowski coincide con el exponente que es asignado al peso de los atributos, sin embargo, el determinar el valor correcto del exponente para cada caso sigue siendo un reto. En [31][32] se proponen implementaciones similares con uso de vectores de pesos locales para cada uno. En todos los casos anteriores, de uso de pesos para distintos atributos, las modificaciones se realizan sobre el algoritmo K-means, las implementaciones no son escalables, y además ninguno implementa el concepto de múltiples espacios, por lo que no son factibles para datos de alta dimensionalidad.

En las siguientes secciones se define la hipótesis de investigación, las métricas a utilizar para la evaluación y la justificación de la propuesta del proyecto.

## 3.4 Hipótesis

Es posible combinar el algoritmo de agrupamiento de subespacios basado en densidad SUBCLU [4] y el algoritmo de agrupamiento de restricciones [6,7] para guiar el proceso de detección de los grupos, en subespacios que incluyan atributos de interés para el analista sin afectar negativamente las métricas de calidad interna de cohesión, compacidad e índice silueta, en comparación con los resultados de SUBCLU.

## 3.5 Métricas

La calidad de los agrupamientos es comúnmente medida en términos de exactitud, por ejemplo, la proporción de objetos agrupados correctamente. Sin embargo, en los métodos de clasificación no supervisada, la verdadera estructura de grupos es en la mayoría de los casos desconocida y no se tiene un grupo de datos de entrenamiento o una verdad base.

Adicionalmente, dado el número de distintos tipos de agrupamientos, donde cada algoritmo define de cierta forma su propio tipo de agrupamiento, puede parecer que cada método requiere una métrica de evaluación distinta. Por ejemplo, K-means puede ser evaluado en términos de MSE (error cuadrado medio), pero para los grupos generados basados en densidad que no necesitan ser esféricos, esta métrica no funcionaría bien para medir la calidad [19]. Como consecuencia de esto, los algoritmos de agrupamiento son a menudo evaluados manualmente con la ayuda de expertos del dominio. No obstante, esto conlleva una serie de problemas. Los expertos no siempre están disponibles y no siempre coincidirán con la calidad de los resultados; la valoración de los agrupamientos es necesaria, pero no puede ser comparada con el óptimo ya que es desconocido y, por último, la evaluación manual no escala a bases de datos de gran tamaño o a agrupamientos de gran tamaño [37].

Dado los puntos anteriores, para este trabajo se analizaron cuatro métricas que pueden medir el valor del algoritmo desarrollado, 3 métricas internas y 1 métrica heurística externa.

### 3.5.1 Métricas internas:

Este tipo de métricas son recomendadas cuando no se tiene una "verdad base" (ground truth). En general, este tipo de métodos evalúan los agrupamientos examinando qué tan compactos y qué tan separados entre sí están los agrupamientos [20].

**1. Cohesión de los grupos**: Mide qué tan compacto es un grupo, o qué tan cercanos son todos los objetos dentro de un grupo [19]. La Figura 7 describe la métrica gráficamente.

Figura 7. Representación gráfica de estimación de cohesión [35]

Para evaluar la cohesión entre los algoritmos se utilizará el cálculo estadístico Within cluster sum of squares (WSS). Este método mide por medio de distancia Euclidiana qué tan cerca se encuentran los objetos de grupo.

Fórmula:

En la Figura 8 se observa la fórmula de cohesión de grupos, donde, sea *i* el grupo actual a analizar, *x* cada objeto dentro del grupo $C_i$, y $m_i$ la distancia media entre objetos del grupo. WSS evalúa la variación de distancias de cada grupo, en relación a la variación de distancias de todos los grupos.

$$WSS = \sum_{i} \sum_{x \in C_i} \|x - m_i\|^2$$

Figura 8. Formula de Within Sum of Squares. [35]

**2. Separación de los grupos**: Mide qué tan separados se encuentran los grupos entre sí [19]. La figura 9 describe la métrica gráficamente.

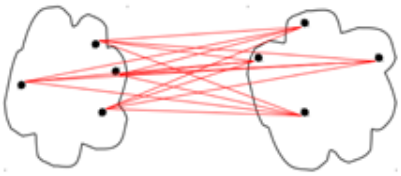


Figura 9. Representación gráfica de estimación de separación [35].

Para esta evaluación usaremos Sum of Squares Between (BSS), utilizada estadísticamente para medir la distancia entre cada grupo.

Fórmula:

En la figura 10 se observa la fórmula de separación de grupos, donde, sea *i* el grupo a evaluar, *m* la media total de distancias y $m_i$ la distancia media entre objetos del grupo y $|Ci|$ la cantidad de objetos en el grupo, entonces:

$$BSS = \sum_{i} |C_i| \|m - m_i\|^2$$

Figura 10. Fórmula de estimación de separación inter grupos [35]

**3. Índice de silueta:** Mide qué tan similar es un objeto a su propio grupo y comparado a los demás grupos. Los valores varían de -1 a 1, donde un valor alto indica que el objeto está bien ubicado en su propio grupo. Si la mayoría de los objetos tienen un valor alto, entonces el resultado se puede considerar inadecuado [21]. Esta métrica es un cálculo que evalúa los agrupamientos basado en las métricas de cohesión y separación.

Se calculará el índice de silueta del algoritmo original y el extendido para realizar un análisis comparativo entre ambos. Como se puede observar en la Figura 11, la métrica considera tanto la cohesión como la separación.

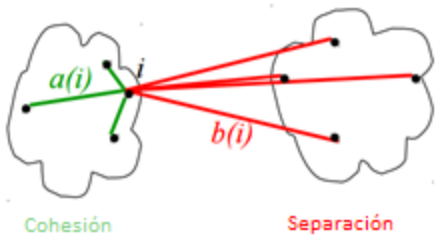



Figura 11. Representación gráfica de estimación de índice de silueta [35]

Fórmula:

En la Figura 12 se puede observar la fórmula de índice de silueta, donde para cada objeto *i*, sea *a(i)* la disimilitud promedio de *i* con todos los otros miembros del mismo grupo, y *b(i)* la disimilitud promedio menor de *i* con cualquier otro grupo al que *i* no pertenezca.

Podemos interpretar *a(i)* como que tan bien *i* está asignado en su propio grupo y *b(i)* como que tan diferente es *i* al resto de los grupos.

La fórmula para calcular el índice de la silueta puede definirse de dos formas:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i),\ b(i)\}}. \qquad s(i) = \begin{cases} 1 - a(i)/b(i) & \text{if } a(i) < b(i), \\ 0 & \text{if } a(i) = b(i), \\ b(i)/a(i) - 1 & \text{if } a(i) > b(i). \end{cases}$$

(a) (b)

Figura 12. Fórmula para estimación del índice de silueta [35]

De la definición en Figura 12:  -1 <= s(i) <= 1, donde:

- *s(i)* cercano a 1: el dato está bien agrupado.
- *s(i)* cercano a -1: el objeto estaría mejor agrupado en su grupo vecino
- *s(i) cercano a 0*: el objeto está en el borde de dos grupos naturales

El promedio *s(i)* de todos los objetos de un grupo, mide qué tan similares son todos los objetos dentro del grupo, por lo tanto, un promedio *s(i)* sobre el conjunto de datos que tan bien se agruparon todos los datos. Es por esto que utilizaremos el promedio total *s(i)* de todos los objetos para comparar el resultado de algoritmo SUBCLU y SUBCLU-R.

Adicionalmente, la representación gráfica y análisis de *s(i)* por instancia y por grupo ayudará a analizar el resultado a un nivel más detallado.

En la Figura 13, se puede observar la representación gráfica del índice silueta de 2 grupos. Gracias a esta visualización, se puede deducir que la mayoría de los objetos están bien agrupados, excepto unos cuantos objetos del grupo 2, que se encuentran en bordes de más de un grupo natural.

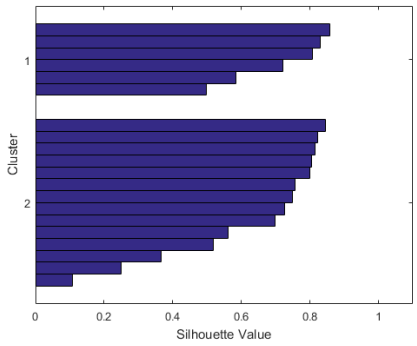



Figura 13. Representación gráfica de un índice de silueta para dos grupos [36].

### 3.5.2 Evaluación heurística:

Para la evaluación heurística, se realizó un estudio comparativo de los grupos generados por el algoritmo SUBCLU y el algoritmo SUBCLU-R y un análisis a fondo de los resultados. Como parte de la evaluación heurística, se evaluaron los siguientes aspectos de los agrupamientos:

- Grupos que incluyen el atributo de interés
- Subespacios únicos
- Subespacios únicos que incluyen el atributo de interés
- Grupos que no incluyen el atributo de interés
- Grupos en común
- Grupos en común con el atributo de interés
- Grupos en común sin el atributo de interés
- Grupos detectado por algoritmo y no detectados por el otro
- Grupos que incluyen el atributo de interés detectado por algoritmo y no detectados por el otro

## 3.6 Justificación del proyecto

Basado en las secciones anteriores, es evidente el valor de los algoritmos de agrupamiento de subespacios para datos de alta dimensionalidad. Sin embargo, la exclusión de atributos de interés por la reducción de los espacios, es un problema en casos donde ciertos atributos son fundamentales para el análisis de los grupos.

El área de algoritmos de agrupamiento de subespacios es un área relativamente nueva, donde el aprendizaje semi-supervisado para este tipo de algoritmos es un área de oportunidades [30]. Particularmente en el área de selección de espacios por medio de listas de atributos de interés, no se encontró ninguna publicación.

En [26] Fromont implementa algoritmos de restricción para algoritmos de subespacios. El trabajo se basa en el algoritmo de subespacios CLIQUE, que, debido a que es un algoritmo basado en celdas, tiene problemas con la identificación de grupos de forma irregular. Además, las restricciones presentadas por el algoritmo, solamente permiten especificar pares de instancias que deben o no deben pertenecer al mismo grupo, de ninguna forma permite guiar la selección de espacios por relevancia de atributos.

En la actualidad, no hay ningún método que nos permita usar el poder de los algoritmos de agrupamiento para datos de alta dimensionalidad y utilizar una lista de atributos de relevancia para guiar la poda de subespacios.

# 4. Objetivos, alcance y limitaciones

En esta sección se definen el objetivo general y objetivos específicos del proyecto.

## 4.1. Objetivo general

El objetivo general de este proyecto es:
Proponer un algoritmo que combine el agrupamiento de subespacios basado en densidad SUBCLU [1] y el algoritmo basado en restricciones (constrained clustering) [6] para la detección de grupos que incluyan atributos de interés en conjuntos de datos de alta dimensionalidad.

## 4.2. Objetivos específicos

Los objetivos específicos de este trabajo son:

- Proponer un algoritmo que combine el algoritmo basado en densidad SUBCLU y el algoritmo basado en restricciones (constrained clustering).
- Evaluar el algoritmo propuesto comparándolo con el algoritmo SUBCLU
- Analizar los resultados obtenidos en miras a comprobar si la inclusión de las restricciones aumenta la presencia de atributos de interés y su impacto en las métricas de calidad (cohesión, compacidad, índice silueta).

## 4.4. Entregables

Los entregables del presente trabajo son:

- Programa que implementa la combinación de algoritmo de subespacios basado en densidad SUBCLU y el algoritmo basado en restricciones (constrained clustering).
- Conjuntos de datos pre-procesados.
- Resultados de la ejecución del diseño de experimentos.
- Documento de análisis de los resultados obtenidos.

# 5. Metodología

En esta sección se detalla el diseño de experimentos que se utilizó para corroborar la veracidad de la hipótesis planteada.

## 5.1 Diseño de experimentos

En este proyecto se diseñó una serie de experimentos que permiten determinar el impacto de la inclusión de algoritmos de restricciones en el algoritmo SUBCLU. A continuación, detalles que fueron considerados en el diseño de los experimentos.

### 5.1.1. Declaración del problema

Determinar si mediante la combinación del algoritmo de agrupamiento por restricciones y el algoritmo de agrupamiento de subespacios SUBCLU, es posible guiar la detección de agrupamientos hacia espacios que incluyan atributos de interés definidos por el usuario.

### 5.1.2. Factores y niveles

De acuerdo al conocimiento y experiencia adquirida durante la etapa de investigación, los factores y niveles identificados como relevantes en el diseño de experimentos de este proyecto son:

1. Medidas de distancia
    a. Distancia Euclidiana: es la distancia entre dos puntos en cualquier número de dimensiones [49], se puede observar la representación gráfica en la Figura 14.a.
    b. Distancia Manhattan: es la distancia entre dos puntos si se tuviera que recorrer el camino a lo largo de las líneas de una matriz [49], se puede observar la representación gráfica en la Figura 14.b

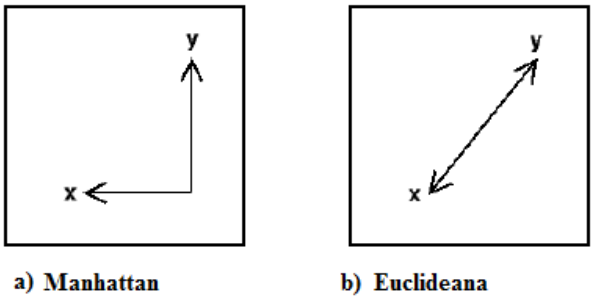



Figura 14. Representación gráfica de métrica de distancia Manhattan y Euclideana [49].

2. Parámetros a utilizar durante experimentos
    a. Épsilon: especifica el tamaño del vecindario [50].
    b. Minpts: parámetro especifica el número mínimo de objetos que forman un grupo [50].
    c. Atributo de interés: especifica nombre del atributo de interés para el usuario.

3. Conjunto de datos

Censo de 1990: Datos de censo de 1990 en Estados Unidos. Este conjunto de datos tiene 68 atributos y más de 4.500.000 objetos.

De este conjunto de datos, se obtuvieron 3 distintas muestras aleatorias de 10000 objetos cada una utilizando R Studio. Seguidamente, se compraron las muestras entre sí para verificar que eran diferentes entre sí.

a. Muestra 1
b. Muestra 2
c. Muestra 3

El porcentaje de promedio similitud de las 3 muestras fue de 38 objetos de 10000. Esto es 0.4066%. El código R utilizado para este análisis se puede ver en la sección 7.5.

El tamaño de las muestras y la dimensionalidad fue escogido en base a numerosas pruebas de tiempo de ejecución con distintos tamaños de datos y dimensionalidad. Lista de pruebas relizadas se incluyen en el Apéndice 7.1.

### 5.1.3. Variables de respuesta

Las variables a considerar para evaluar los experimentos corresponden a las métricas detalladas en la sección 3.5:

1. Cohesión
2. Separación
3. Índice de Silueta
4. Heurística

Para obtener los valores de cohesión, separación, y valores heurísticos se escribieron scripts en Spark que se ejecutaron en un sistema Hadoop, para el análisis de los archivos ".txt" en los que ELKI presenta los resultados. Los valores para el índice silueta se obtuvieron por medio del evaluador incluido en ELKI.

### 5.1.4. Recolección de datos

Los datos a utilizar en los experimentos, fueron obtenidos de repositorios públicos. Para la limpieza y pre-procesamiento de datos, se utilizó R studio [54].

Para la obtención de todas las variables de respuesta se utilizó un sistema con una distribución Hadoop de Cloudera, y se escribieron scripts para Spark que analizaron los archivos resultados de la ejecución de los experimentos.

### 5.1.5. Análisis de datos

Una vez concluida la etapa de ejecución de los experimentos, se realizó un análisis de los resultados para comparar los algoritmos SUBCLU y SUBCLU-R.

Como paso inicial, se realizó un análisis de la distribución de las variables respuesta cohesión, separación e índice de silueta. Para realizar esta comparación se utilizó el conjunto de pruebas sobre 3 muestras de datos del mismo tamaño y evaluados bajo dos distintas funciones de distancia: Euclideana y Manhattan. Cada algoritmo se ejecutó sobre la muestra con las mismas configuraciones paramétricas y se obtuvo el valor de cohesión, separación e índice silueta de cada algoritmo.

Para la comparación estadística de los resultados, primero se hizo un análisis de normalidad para cada los resultados de cada métrica y para los que mostraron una distribución normal de la población se utilizó ANOVA[43], y para los datos no cumplían el supuesto de normalidad. Ante esto, se optó por utilizar el método de Kruskal-Wallis [42].

**ANOVA:**

El ANOVA permite asegurar que la variación en los resultados de un experimento, no es mayor a la suma de variaciones de los factores y un cierto grado de error en sus medidas.

Esta técnica tiene como objetivo analizar la relación entre una variable cuantitativa X, y una variable cualitativa Y de k atributos. Donde cada atributo *i* define una población dada por la variable cuantitativa [43].

En el análisis, se desea determinar si las métricas de cohesión, separación e índice silueta no varían dependiendo del algoritmo ejecutado (SUBCLU o SUBCLU-R). Esto se puede plantear usando las hipótesis:

- **Hipótesis nula $h_0$:** la metrica (cohesion, separación e índice silueta) no varía según el algoritmo
- **Hipótesis alternativa $h_1$:** el índice silueta varía según el algoritmo

**Prueba Kruskal Wallis:**

La prueba Kruskal Wallis (KW) es una prueba no paramétrica, que se realiza sobre muestras independientes que pueden provenir de poblaciones que no se relacionan entre sí. A diferencia de ANOVA, no asume que los datos siguen una distribución normal [44].

Esta técnica al igual que ANOVA, tiene como objetivo analizar la relación entre una variable cuantitativa X, y una variable cualitativa Y de k atributos. La diferencia es que en KW, cada atributo o valor cualitativo define una población, cuyo valor es dado por la variable cuantitativa.

En el análisis, se desea determinar si X no varía con respecto a los valores de Y. Esto se puede plantear usando las hipótesis:

- **Hipótesis nula $h_0$:** Las muestras pertenecen a poblaciones idénticas, es decir, la metrica (cohesion, separación e índice silueta) no varía según el algoritmo (SUBCLU o SUBCLU-R).

- **Hipótesis alternativa $h_1$**: Al menos una muestra pertenece a una población diferente que las demás, es decir, el valor de las métricas (cohesión, separación e índice silueta) varía según el algoritmo.

### 5.1.6. Ambiente de desarrollo

El pre-procesamiento de los datos, implementación, ejecución y validación de los experimentos se realizó con un conjunto de bibliotecas y herramientas, estas se listan a continuación :

- Sistema Operativo: Windows 7

- Sistema Operativo: Linux Distribución Cloudera (CDH), Máquina virtual. Esta distribución facilita la configuración de Hadoop para el manejo y análisis de datos no estructurados.

- VMWare Workout 12 Player: Administrador de máquinas virtuales, utilizado en este proyecto para el ambiente Linux. Versión Gratuita

- Lenguaje de programación: Java, versión 8.

- Ambiente de Programación (IDE): Netbeans/Eclipse

- R Studio: IDE para el lenguaje de programación R. Se utilizó para análisis de distribución de los datos y selección de los atributos a utilizar

- Framework de experimentos: ELKI, software de minería de datos especializado en algoritmos de agrupamiento [40]. Esta herramienta es diseñada para investigación y permite modificar o extender fácilmente algoritmos de agrupamiento.

- Apache Spark: Motor de procesamiento de datos que permite por medio de Python o Escala analizar los datos almacenados en el sistema de datos distribuido HDFS de Hadoop.

## 5.2 Resultados

A continuación, se presentan los resultados obtenidos en las diferentes etapas del experimento.

El tamaño de la muestra utilizada para la evaluación de los algoritmos es de 10 mil objetos y 10 atributos. Tamaños de muestra mayores y conjunto de mayor dimensionalidad fueron evaluados, sin embargo, ELKI no aprovecha el paralelismo durante su ejecución, por lo que pruebas de mayor tamaño de muestra o dimensiones fueron descartadas ya que los tiempos proyectados de ejecución ascendían a más de 3 semanas por experimento. En la sección 7.1 del apéndice se documentaron las distintas pruebas de tamaño de muestra y dimensionalidad realizadas.

El framework ELKI genera un archivo de extensión "txt" por cada agrupamiento detectado, con detalle de los objetos pertenecientes a cada grupo. En total, 1729 archivos fueron generados como resultado de la ejecución de ambos algoritmos.

El conjunto de archivos generados por ambos algoritmos ocupa un espacio en disco superior a los 900 mb, por lo que se decidió implementar scripts en PySpark (Python para Spark) sobre los archivos en un sistema HDFS de Hadoop para aprovechar el procesamiento en paralelo de los mismos. Los scripts implementados para Spark son reutilizables y escalables para los resultados de cualquier algoritmo de agrupamiento en ELKI, el código se encuentra en la sección 7.3.

### 5.2.1 Selección de parámetros

En la Tabla 2 se muestran los parámetros seleccionados para la ejecución de los algoritmos (SUBCLU original y SUBCLU-R). En la sección 7.2 del apéndice se incluyen las distintas configuraciones paramétricas analizadas.

| Parámetro | Valor |
|---|---|
| **Epsilon** | 3.0 |
| **MinPts** | 600 |
| **Dimensiones** | 10 |
| **Dimensión de interés** | Edad |

Tabla 2. Tabla de selección de parámetros para ejecución de experimentos.

### 5.2.2 Conjunto de datos y selección de atributos del conjunto

El conjunto de datos seleccionado para los experimentos es USCENSUS1990, un conjunto de datos del censo de Estados Unidos de 1990 almacenado y compartido por la Universidad de California (UCI). El conjunto de datos es público, posee un total de 3.100.000 objetos y 68 atributos categóricos con gran variedad de rangos de posibles valores. Cabe destacar que el conjunto de datos ya ha sido limpiado, discretizado, y que es una muestra aleatoria del total del censo. Para el análisis de datos se utilizó la herramienta R-studio[54], herramienta para el lenguaje R que permite análisis estadístico y grafico de los datos.

Para los experimentos, se utilizaron muestras de 10.000 objetos y 10 variables (las mismas variables en todas las muestras). Inicialmente, se había planeado realizar el análisis con un conjunto de datos de mucho mayor tamaño (cantidad de objetos) y de mayor dimensionalidad, sin embargo, pruebas en múltiples sistemas demostraron que el Framework Elki no aprovechaba el paralelismo, por lo que se decidió disminuir el tamaño de los experimentos. En el Apéndice 7.1 se listan los experimentos ejecutados y los tiempos estimados de ejecución.

A continuación, en la Tabla 3 se presentan los atributos seleccionados para las pruebas. Los atributos a utilizar fueron seleccionados de acuerdo a la relevancia de los atributos, la cantidad de posibles valores a tomar y la distribución de los datos. De los atributos que se excluyeron, muchos atributos eran debido a lo específico, y el bajo porcentaje de objetos con valores datos para el atributo, por ejemplo, la cantidad de años que una persona sirvió en la guerra de Vietnam, o la cantidad de personas que con las que viaja en el mismo vehículo al trabajo.

| **Nombre de atributo** | **Descripción** | Resumen de los datos | |
|---|---|---|---|
| **dAge** | Edad | Min: 0<br>Mediana: 2<br>Media: 3.857<br>Max: 7 | IF edad = 0 Then 0<br>IF edad < 13 Then 1<br>IF edad < 20 Then 2<br>IF edad < 30 Then 3<br>IF edad < 40 THEN 4<br>IF edad < 50 THEN 5<br>IF edad < 65 THEN 6<br>ELSE 7 |
| **iCitizen** | Tipo de ciudadanía | Min: 0<br>Mediana: 0<br>Media: N/A (Dato categórico)<br>Max: 4 | 0= nacido en US<br>1= nacido en territorio de US<br>2= nacido en extranjero de padres de US<br>3=ciudadano americano por naturalización<br>4 = No ciudadano de US |
| **iFertil** | Número de hijos | Min: 0<br>Mediana: 0<br>Media: 1.209<br>Max: 13 | 0= (N/A) menor de 15 años<br>1= 0 hijos<br>2= 1 hijo<br>..<br>13=12 o más hijos |
| **dIncome1** | Ingresos | Min: 0<br>Mediana: 0<br>Media: 0.8859<br>Max: 4 | IF income = 0 THEN 0<br>IF income < 15000 THEN 1 |

| | | | |
|---|---|---|---|
| | | | IF income < 30000<br>THEN 2<br>IF income < 60000<br>Then 3<br>ELSE 4 |
| **dIndustry** | Clasificación industrial de área de trabajo | Min: 0<br>Mediana: 4<br>Media: N/A (Dato categórico)<br>Max: 12 | Posición de industria de trabajo de acuerdo a promedio salarial (0-12) |
| **dPOB** | Lugar de nacimiento | Min: 0<br>Mediana: 0<br>Media: N/A (Dato categórico)<br>Max: 6 | 0=Fuera de US<br>1=US estado continental<br>2=Isla de US<br>3=Territorios americanos<br>4=Nacido en US de padres extranjeros<br>5=Nacido en exterior de padres de US<br>6=Nacido en extranjero de padres extranjeros |
| **dPwgt1** | Peso de la persona | Min: 0<br>Mediana: 1<br>Media: N/A (Dato categórico)<br>Max: 3 | IF peso < 50<br>THEN 0<br>IF peso < 125<br>THEN 1<br>IF peso < 200<br>THEN 2<br>ELSE 3 |
| **iRspouse** | Estado civil | Min: 0<br>Mediana: 1<br>Media: N/A (Dato categórico)<br>Max: 6 | |
| **iSex** | Genero | Min: 0<br>Mediana: 1<br>Media: N/A (Dato categórico) | |

| | | | |
|---|---|---|---|
| | | Max: 1 | |
| **iYearsch** | Grado de estudio | Min: 0<br>Mediana: 10<br>Media: 8.443<br>Max: 17 | |

Tabla 3. Tabla de atributos seleccionados para la ejecución de los experimentos.

En general, la evaluación y comparación de resultados de agrupamiento es una tarea compleja pues no se tiene una verdad base, o se conocen los resultados esperados de antemano. Por otro lado, existe gran cantidad de métricas de calidad interna que reflejan diferentes aspectos de los agrupamientos, sin embargo, el uso de estos no es estandarizado ni universal, y se utiliza en cada investigación o propuesta de algoritmo a conveniencia [5]. Es por esto que se decidió evaluar con métricas externas que reflejan lo que consideramos importante de los agrupamientos detectados, y un conjunto de métricas internas más utilizadas y entendidas de la literatura [37].

Las métricas externas e internas de los experimentos se detallan en la sección 5.2.3 y 5.2.4 respectivamente.

### 5.2.3 Métricas externas

En esta sección se presenta un análisis de calidad externa de los grupos generados por ambos algoritmos. En la Tabla 4 se muestran los hallazgos más importantes de los grupos generados por ambos algoritmos.

| | **SUBCLU** | **SUBCLU-R** |
|---|---|---|
| **Total de grupos generados** | 1130 | 599 |
| **Grupos que incluyen el atributo de interés** | 595 | 590 |
| **Subespacios únicos** | 1023 | 521 |
| **Subespacios únicos que incluyen el atributo de interés** | 413 | 513 |
| **Grupos que no incluyen el atributo de interés** | 535 | 9 |
| **Grupos en común** | 549 | 549 |
| **Grupos en común con el atributo de interés** | 539 | 539 |

| | | |
|---|---|---|
| **Grupos en común sin el atributo de interés** | 9 | 9 |
| **Grupos detectado por algoritmo y no detectados por el otro** | 581 | 50 |
| **Grupos que incluyen el atributo de interés detectado por algoritmo y no detectados por el otro** | 56 | 51 |

Tabla 4. Tabla de comparación de métricas externas de los experimentos.

Cabe destacar que las métricas externas se mantuvieron idénticas para todos los experimentos de los algoritmos. Es decir, los 6 experimentos de SUBCLU dieron los mismos resultados en cantidad de grupos generados, grupos con atributos de interés, y el resto de métricas incluidas en la Tabla 4. Esto puede explicarse a que, aunque las muestras fueran distintas entre sí (0.038% de similitud), los bordes de los grupos se encontraban lo suficientemente separados entre sí dada la configuración paramétrica utilizada.

El análisis realizado en la Tabla 4, describe características de los agrupamientos de ambos algoritmos en términos de relevancia e interpretabilidad para el usuario. Por ejemplo, es más fácil interpretar 10 grupos, de los cuales todos incluyen el atributo de interés, que analizar 50 grupos de los cuales no se sabe cuántos incluyan el mismo atributo.

Los resultados de los experimentos generaron un total de 1130 grupos para el algoritmo SUBCLU, y 599 para SUBCLU-R, de los cuales 595 y 589 grupos incluían el atributo de interés respectivamente. Esto significa que de los grupos que SUBCLU detectó, un 52.65% incluían la dimensión deseada, mientras que el SUBCLU-R detectó un 98.33%; gracias a la implementación de reglas fuertes en la poda de los subespacios. La cantidad de grupos detectados por el algoritmo SUBCLU-R es mucho menor al algoritmo original, sin embargo, el 98.33% demuestra que es capaz de excluir eficazmente aquellos grupos que no sean de interés para el usuario.

En las pruebas iniciales se detectó que si la poda de subespacios iniciaba a partir del espacio de 2 dimensiones, el resultado era mucho mejor en cuanto a la cantidad de subespacios detectados e igualaba las generadas por el algoritmo original (siempre y cuando el atributo de interés se encuentre al inicio del conjunto de datos), esto explica el 1.77% de grupos del algoritmo propuesto que no incluyen el atributo de interés.

De los 1130 grupos que SUBCLU detectó, 1023 son subespacios únicos, mientras que SUBCLU-R encontró 521 de 599, esto representa un 90.53% y un 88.45%. Ante estos valores tan altos, se realizó un análisis de tamaño de los grupos, se encontró que para ambos algoritmos el tamaño medio de los grupos era de 7470 objetos. Tanto el tamaño de los grupos como el porcentaje tan alto de subespacios únicos son fuertes indicativos de que la densidad dada por los parámetros

combinados de eps y minPts fue muy baja. En DBSCAN un valor muy alto para este parámetro genera grupos excesivamente grandes (con la mayoría de los objetos); por transitividad en SUBCLU esto se traduce a pocos grupos por cada subespacio.

En total, ambos algoritmos coincidieron en 539 grupos, es decir, un 89.98% de los grupos detectados por SUBCLU-R son grupos que SUBCLU encontró, esto es un buen indicador de que los cambios realizados al algoritmo no afectaron el funcionamiento del algoritmo original, y a su vez un indicador de la eficacia del algoritmo para la realización de la poda de subespacios.

Ambos algoritmos encontraron grupos con el atributo de interés que el otro no detectó, SUBCLU encontró 56 grupos nuevos mientras que el SUBCLU restringido por reglas detectó 51 agrupamientos. Análisis de los grupos demostró que los 56 grupos detectados por SUBCLU eran variaciones de los 51 agrupamientos detectados por SUBCLU-R, es decir, contenían los mismos atributos, pero los grupos fueron generados por caminos distintos. Por ejemplo, el grupo X de SUBCLU fue construido por las columnas a,b,c,…. f; mientras que el grupo Y del subespacio del algoritmo propuesto era c, a, b,…., f siendo c el atributo de interés. La razón de esta diferencia de los subespacios es debido a la sensibilidad de SUBCLU-R con respecto a la posición de la columna dentro del conjunto de datos, dada la construcción de los subespacios de abajo hacia arriba, SUBCLU generara múltiples combinaciones del mismo subespacio, mientras que SUBCLU-R solo tomará en cuenta aquellos que tienen el atributo en la base del subespacio (primera o segunda posición en el conjunto de datos).

El efecto de los grupos incluidos/excluidos en la calidad de los agrupamientos puede ser entendido de forma más integral al observar las métricas de calidad internas, estas se presentan en la sección 5.2.4.

### 5.2.4 Métricas internas

Métricas internas promedio de los agrupamientos obtenidos por cada algoritmo se resumen en la Tabla 5. Aquí se observan los valores promedio de cohesión, separación e índice silueta para SUBCLU y SUBCLU-R sobre las 3 muestras analizadas.

| Parámetro | Función de distancia | SUBCLU | SUBCLU R |
|---|---|---|---|
| **Cohesión** | Euclideana | 4.113 | 3.8832 |
| **Cohesión** | Manhattan | 2.9312 | 2.8813 |
| **Subtotal Promedio** | | 3.5221 | 3.38225 |
| **Separación** | Euclideana | 17.7541 | 17.721 |
| **Separación** | Manhattan | 17.5425 | 17.5101 |
| **Subtotal Promedio** | | 17.6483 | 17.6155 |
| **Silueta** | Euclideana | 1.0065 | 1. 0061 |
| **Silueta** | Manhattan | 1.0055 | 1.0058 |
| **Subtotal Promedio** | | 1.006 | 1.006 |
| **Tiempo de ejecución (horas)** | Euclideana | 31:05 | 36:56 |

| | | | |
|---|---|---|---|
| **Tiempo de ejecución (horas)** | Manhattan | 29:45 | 36:11 |
| **Subtotal Promedio** | | 30:41 | 36:34 |

Tabla 5. Promedio de métricas internas de los experimentos.

En la Tabla 6, se pueden observar los valores desglosados para las pruebas realizadas para cada conjunto de datos prueba, aquí se observa la cohesión y separación de cada muestra tanto para SUBCLU como SUBCLU-R:

| ***Conjunto de datos*** | **Métrica** | **Función de distancia** | **SUBCLU** | **SUBCLU-R** |
|---|---|---|---|---|
| ***Muestra 1*** | Cohesión | Euclideana | 4.0973 | 4.0112 |
| | | Manhattan | 2.7588 | 3.1003 |
| | Separación | Euclideana | 16.8538 | 17.3103 |
| | | Manhattan | 17.4163 | 16.9851 |
| | Silueta | Euclideana | 1.0125 | 1.0127 |
| | | Manhattan | 1.0058 | 1.0074 |
| ***Muestra 2*** | Cohesión | Euclideana | 4.1615 | 3.9203 |
| | | Manhattan | 3.0646 | 2.8905 |
| | Separación | Euclideana | 17.9557 | 17.7421 |
| | | Manhattan | 17.3811 | 17.3443 |
| | Silueta | Euclideana | 1.0032 | 1.0024 |
| | | Manhattan | 1.0059 | 1.0088 |
| ***Muestra 3*** | Cohesión | Euclideana | 4.081 | 3.7202 |
| | | Manhattan | 2.9701 | 2.6531 |
| | Separación | Euclideana | 18.4533 | 18.1128 |
| | | Manhattan | 17.8303 | 18.2007 |
| | Silueta | Euclideana | 1.0040 | 1.0033 |
| | | Manhattan | 1.0049 | 1.00131 |

Tabla 6. Métricas obtenidas por los algoritmos en cada conjunto de datos

Como se puede observar en la Tabla 5 los resultados son muy similares para ambos algoritmos en las 3 métricas propuestas, sin embargo, si se observó una gran caída en tiempo de ejecución del algoritmo SUBCLU-R con respecto al algoritmo original.

Se observaron mejoras leves de los valores cohesión para SUBCLU-R. El excluir los grupos que no contenían la dimensión de interés mejoró la calidad interna de los grupos. Dado el alto porcentaje de coincidencias de grupos entre ambos algoritmos presentado en la sección 5.2.3, se puede inferir que la calidad de los grupos mejoró no por haber encontrado grupos nuevos de mayor calidad, sino porque se excluyeron grupos de menor calidad al no tener el atributo de interés. Esta hipótesis se refuerza al analizar el atributo de interés seleccionado para la ejecución del experimento de SUBCLU-R (edad), pues es un parámetro que en términos de análisis de población

tiene mucha inferencia en los agrupamientos, por lo que es muy probable que los grupos generados que no incluían el atributo edad tenían menor cohesión.

La calidad de los grupos en términos de separación es levemente inferior para el algoritmo propuesto, aunque la diferencia es solamente de 0.18%, el análisis de los grupos muestra que esta diferencia en separación se debe a que todos los grupos tienen un mayor porcentaje de atributos en común (la mayoría ahora tiene como mínimo el atributo de interés), en este caso todos los grupos tienen al menos un atributo en común (edad), por lo que la separación de los grupos tenderá a ser menor pues los grupos se encontrarán en espacios menos dispersos.

Al observar los resultados de cohesión y separación del algoritmo expuesto en este trabajo, y analizar el efecto de la restricción en los agrupamientos, se deduce que al comparar SUBCLU y SUBCLU-R la cohesión mejore y la separación empeore. El grado de cambio entre ambos resultados dependerá de distintas variables, como la cantidad de atributos del conjunto de datos, la cantidad de atributos de interés, la relevancia del atributo en los grupos naturales de los datos o la cantidad de veces en las que el atributo de interés aparece en los distintos agrupamientos. En cuanto al índice silueta, no se puede predecir como cambiará la métrica entre ambos algoritmos, pues el índice no se encuentra atado a los valores promedio de cohesión o separación de los grupos, sino a los valores individuales de cada grupo.

Hay una caída significativa de desempeño en el algoritmo SUBCLU-R. Aunque la cantidad de subespacios y grupos a analizar en cada iteración es menor que a los generados por el algoritmo original, el tiempo de pre-evaluación de los grupos para comprobar la existencia del atributo de interés termina representando un aumento significativo de tiempo de ejecución. Esto sucede pues en cada iteración de generación de algoritmo, todos los subespacios se evalúan, inclusive los subespacios evaluados en iteraciones anteriores.

## 5.3 Prueba de hipótesis

Una vez que se concluyó la ejecución del experimento, se estudiaron los resultados para realizar la comparación de los distintos algoritmos. Basado en los resultados presentados en la sección 5.2.3, se demuestra que al combinar el algoritmo SUBCLU con el algoritmo de agrupamiento de restricciones es posible guiar la detección de subespacios hacia aquellos que contengan atributos de interés para el usuario y podar subespacios que no los incluyen. El algoritmo SUBCLU-R generó grupos de los cuales un 98% incluyen el atributo de interés, mientras que solo un 53% de los grupos detectados por el algoritmo SUBCLU utilizando los mismos parámetros lo incluía.

De igual manera, de acuerdo al análisis de las métricas de cohesión, separación e índice silueta presentado en la sección 5.2.4, se demuestra que la calidad interna de los agrupamientos detectados se mantiene, con una mejora de 5.6% para la cohesión y una diferencia negativa de 0.186% para la separación de los grupos. Hubo una leve mejoría en cuanto a la cohesión de los grupos en particular, sin embargo, esta mejora puede ser atribuible al atributo seleccionado para el experimento y su relevancia en el conjunto de datos, por lo que se requieren más experimentos para poder asegurar que la calidad interna de los agrupamientos siempre va a tender a la mejora con la disminución de subespacios generados. Los experimentos detectaron que el valor del índice silueta promedio se mantuvo muy similar en ambos algoritmos.

A continuación, se presenta el análisis estadístico para las 3 métricas internas:

**Cohesión:**

En la Figura 15 se muestra el resultado del gráfico QQ (también conocido como gráfico cuantil-cuantil) de las mediciones de cohesión obtenidos por ambos algoritmos contra una distribución normal.

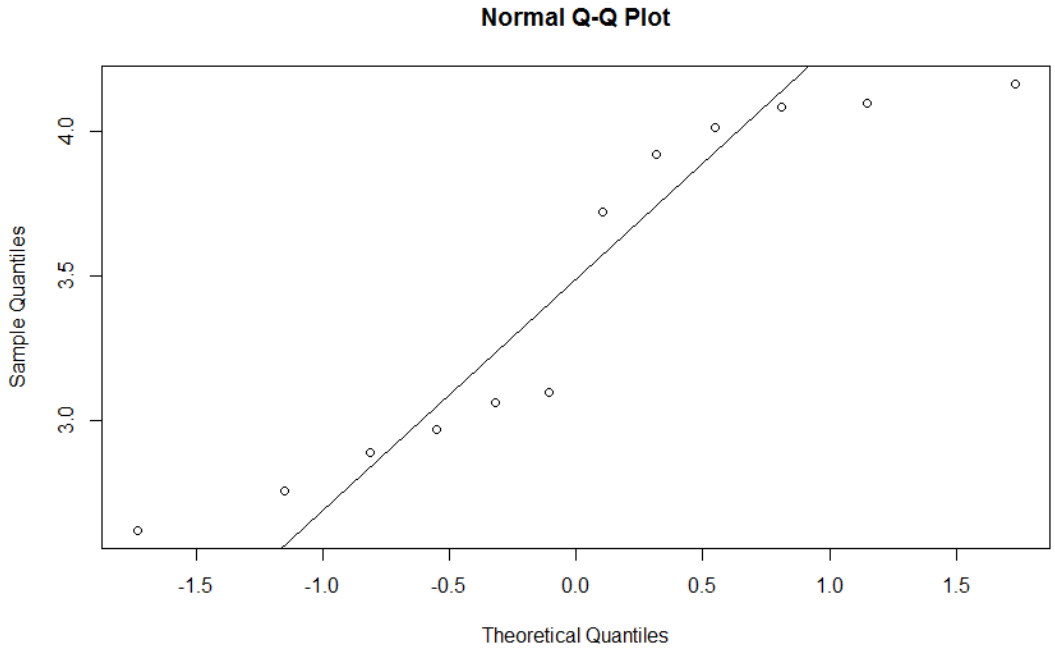



Figura 15. Gráfico QQ de la métrica cohesión obtenida contra una distribución normal.

```
        Shapiro-Wilk normality test

data:  mydata$Q
W = 0.85674, p-value = 0.0445
```

Figura 16. Prueba de normalidad Shapiro-Wilk para Cohesión

La Figura 15 muestra la distribución de las mediciones de cohesión con respecto a la línea que representa la distribución normal. La figura 16 muestra el resultado de la prueba de normalidad Shapiro-Wilk, considerando el nivel de significancia 0.05 se demuestra que las mediciones se desvían de la distribución normal con un valor-p 0.0445. Como no se cumplió esta suposición del análisis estadístico ANOVA, se procedió a utilizar el análisis de Kruskal-Wallis para la prueba de hipótesis.

```
        Kruskal-Wallis rank sum test

data:  Q by Algorithm
Kruskal-Wallis chi-squared = 0.64103, df = 1, p-value = 0.4233
```

Figura 17. Resultado de la prueba Kruska-Wallis para las observaciones de Cohesión

El análisis en la Figura 17 pone a prueba la hipótesis de que las distribuciones de las mediciones de cohesión obtenidas por los algoritmos son iguales. Como el valor p 0.4233 es mayor al nivel de significancia 0.05, se acepta la hipótesis nula. Es decir, se puede asegurar con una certeza del 95%, que no hay evidencia suficiente para determinar que los experimentos de SUBCLU-R obtuvieron diferentes valores de cohesión que los ejecutados con SUBCLU.

## Separación

En la Figura 18 se muestra el resultado del gráfico QQ de las mediciones de separación obtenidos contra la distribución normal.

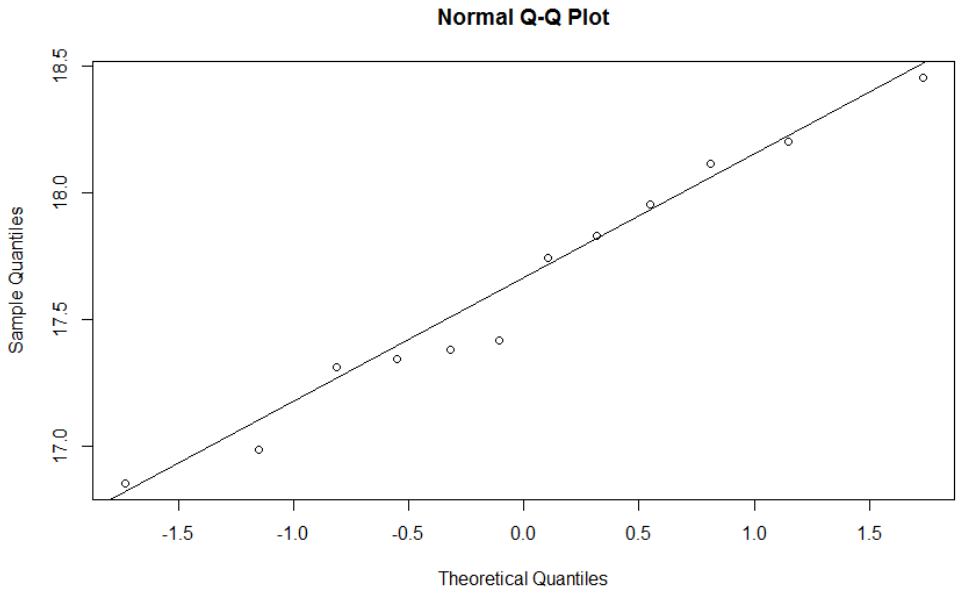


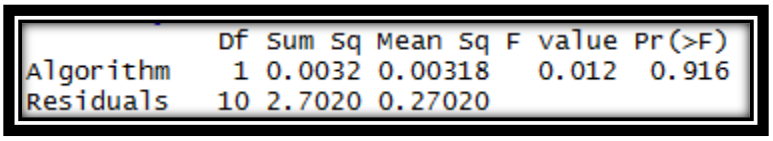

Figura 18. Gráfico QQ de la métrica separación obtenida contra una distribución normal.

```
        Shapiro-Wilk normality test

data:  mydata$Q
W = 0.96614, p-value = 0.8665
```

Figura 19. Prueba de normalidad Shapiro-Wilk para Separación

La Figura 18 muestra la distribución de las mediciones de separación con respecto a la línea que representa la distribución normal. La Figura 19 muestra el resultado de la prueba de normalidad Shapiro-Wilk, considerando el nivel de significancia 0.05 se demuestra que las mediciones siguen una distribución normal con un valor-p 0.8665. Dada la distribución normal de los datos, se prosigue con el análisis estadístico ANOVA para la métrica de separación, el resultado se muestra en la Figura 20.



| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| Algorithm | 1 | 0.0032 | 0.00318 | 0.012 | 0.916 |
| Residuals | 10 | 2.7020 | 0.27020 | | |

Figura 20. Resultado de la prueba de Anova para Separación

El análisis en la Figura 20 pone a prueba la hipótesis de que las distribuciones de las mediciones de separación obtenidas por los algoritmos son iguales. Como el valor p 0.916 es mayor al nivel de significancia 0.05, se acepta la hipótesis nula. Es decir, estadísticamente, se puede asegurar con una certeza del 95%, que no hay evidencia suficiente para determinar que los experimentos de SUBCLU-R obtuvieron diferentes valores de separación que los ejecutados con SUBCLU.

## Índice Silueta

En la Figura 21 se muestra el resultado del gráfico QQ de las mediciones de índice silueta contra la distribución normal.

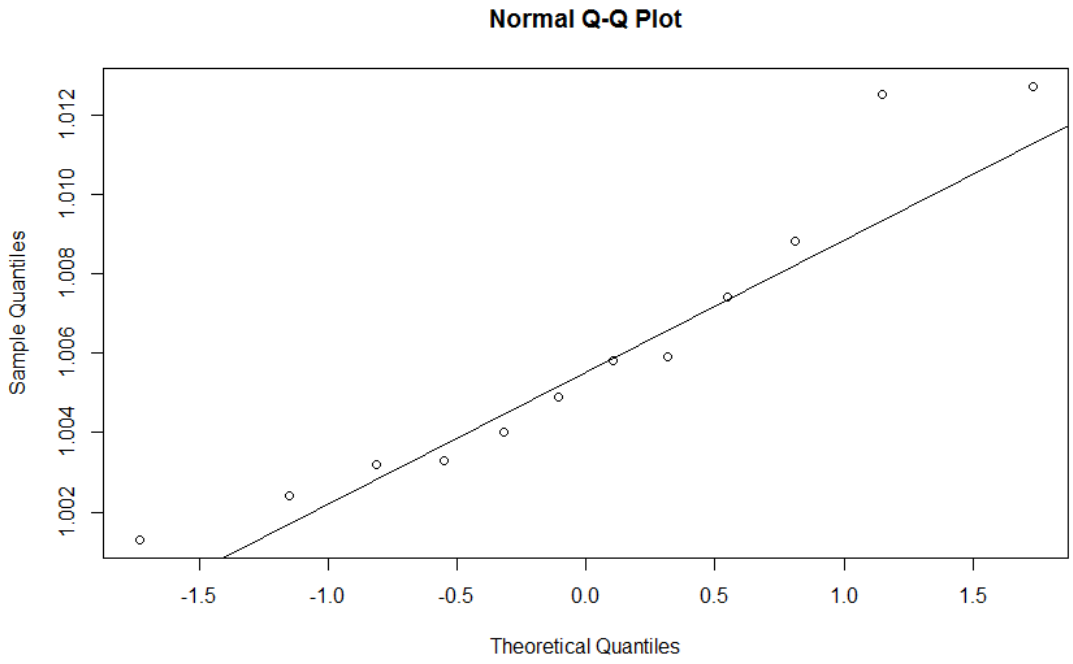



Figura 21. Gráfico QQ de la métrica índice silueta obtenida contra una distribución normal.

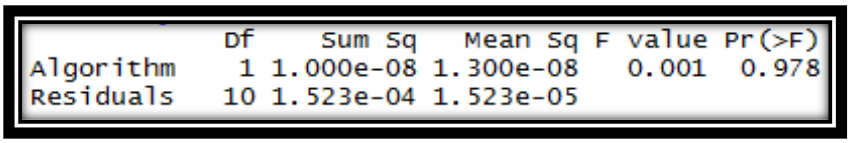
```
        Shapiro-Wilk normality test

data:  mydata$Q
W = 0.90847, p-value = 0.2039
```

Figura 22. Prueba de normalidad Shapiro-Wilk para índice silueta

La Figura 21 muestra la distribución de las observaciones de Índice Silueta con respecto a la línea de la distribución normal. La Figura 22 muestra el resultado de la prueba de normalidad Shapiro-Wilk, donde con un valor-p 0.2039 se demuestra que las muestras siguen una distribución normal. Dada la distribución normal de las observaciones, se prosigue con el análisis estadístico ANOVA para la prueba de hipótesis de las métricas del índice silueta.

En la Figura 23 se muestra el resultado de la prueba ANOVA sobre la métrica de índice silueta.



| | Df | Sum Sq | Mean Sq | F value | Pr(>F) |
|---|---|---|---|---|---|
| Algorithm | 1 | 1.000e-08 | 1.300e-08 | 0.001 | 0.978 |
| Residuals | 10 | 1.523e-04 | 1.523e-05 | | |

Figura 23. Resultado de la prueba de ANOVA para Índice Silueta

El análisis en la Figura 23 pone a prueba la hipótesis de que las distribuciones de las mediciones de índice silueta obtenidas por los algoritmos son iguales. Como el valor p 0.978 es mayor al nivel de significancia 0.05, se acepta la hipótesis nula. Es decir, estadísticamente, se puede asegurar con una certeza del 95%, que no hay evidencia suficiente para determinar que los experimentos de SUBCLU-R obtuvieron valores diferentes de índice silueta que los ejecutados con SUBCLU.

Finalizado el análisis estadístico de las observaciones se puede asegurar que las poblaciones de SUBCLU y SUBCLU-R son estadísticamente iguales para las 3 métricas analizadas (cohesión, separación e índice silueta), es decir, la calidad interna de los grupos para las métricas analizadas se mantiene estadísticamente igual. Ante esto, podemos deducir que la calidad interna de los grupos no mejora ni empeora con el algoritmo SUBCLU-R.

Por lo anterior y el análisis de la sección 5.2.4, donde se demuestra que es posible guiar el proceso de detección de grupos hacia subespacios de interés con el algoritmo SUBCLU-R, se aprueba la hipótesis planteada en este proyecto en la sección 3.4. Es decir, se puede afirmar que: es posible combinar el algoritmo de agrupamiento de subespacios basado en densidad SUBCLU y el algoritmo de agrupamiento de restricciones para guiar el proceso de detección de los grupos, en subespacios que incluyan atributos de interés para el analista sin que empeoren las métricas de cohesión, separación e índice silueta cuando se compara con los resultados de SUBCLU.

## 5.4 Conclusiones y trabajo futuro

En esta sección se presentan las conclusiones y el trabajo futuro que se propone para darle continuidad a los resultados de este trabajo.

### 5.4.1 Conclusiones

Para este trabajo de investigación, se estudiaron los algoritmos de agrupamiento SUBCLU y el algoritmo de agrupamiento por restricciones, y la posibilidad de combinarlos para guiar la generación de espacios hacia aquellos que contengan los atributos de interés, y consecuentemente la detección de agrupamientos de mayor valor para el usuario. Con este fin:

- Se realizó un análisis de distribución de datos utilizando R Studio, para la selección de los atributos a utilizar para el análisis.

- Se modificó la implementación del algoritmo SUBCLU del framework ELKI con la lógica del algoritmo de agrupamiento por restricciones. La lógica del algoritmo de restricciones es utilizada por el algoritmo SUBCLU durante la evaluación de los subespacios generados.

- Se ejecutaron experimentos con el algoritmo SUBCLU original y el algoritmo SUBCLU-R utilizando el conjunto de datos descrito en la sección 5.2.2 con la configuración paramétrica descrita en 5.2.1

- Se escribieron scripts en Spark para analizar los archivos texto generados por ambos experimentos, para obtener las mediciones internas y externas de los agrupamientos. Estos scripts sirven para analizar los resultados de cualquier algoritmo de agrupamiento obtenidos en ELKI.

- Se comparó el resultado de las métricas de calidad internas y externas de los agrupamientos obtenidos por ambos algoritmos.

- Se hizo un análisis estadístico de los resultados para aceptar o rechazar la hipótesis

- Con los resultados obtenidos se realizó un análisis en el que se buscó explicar los resultados obtenidos.

Como resultado de este procedimiento, fue posible guiar la detección de subespacios hacia aquellos que incluyen atributos de interés para el usuario, especificado como parámetro de ejecución.

Se observó que, aunque el algoritmo original tiene un mejor desempeño, el algoritmo modificado mantuvo valores muy similares de calidad interna, e incluso para algunos atributos presentó una

leve mejoría. Esto demuestra que, aunque el algoritmo propuesto excluya gran cantidad de grupos e incluso subespacios completos, la calidad general de los agrupamientos se mantiene.

A partir de los resultados y su posterior análisis se obtuvieron las siguientes conclusiones:

- Es posible extender el algoritmo SUBCLU con el algoritmo de agrupamiento por restricciones para guiar la poda de subespacios.

- La cantidad de agrupamientos detectados con atributo de interés por el algoritmo propuesto en este trabajo corresponde a N – d. Donde N es el número de agrupamientos y D la cantidad de atributos del conjunto de datos.

- La modificación propuesta es basado en restricciones fuertes, es decir, la poda tomará en cuenta únicamente aquellos subespacios que contengan el o los atributos de interés. La única excepción a la poda son los subespacios de una dimensión, estos no se podan pues reduce drásticamente las posibles combinaciones de subespacios generados subsecuentes.

- La posición del atributo dentro del conjunto de datos (número de columna) influye en la generación de los subespacios dada la naturaleza "de abajo hacia arriba" del algoritmo SUBCLU. La cantidad de subespacios generados es mayor si el atributo de interés (columna) se coloca al inicio del conjunto de datos. Esta observación se realizó en etapas tempranas de los experimentos, durante la ejecución se notó que los subespacios siendo generados era mucho menor cuando la columna no se encontraba en la primera posición del conjunto de datos. Esto se debe a que el algoritmo SUBCLU genera los subespacios de abajo hacia arriba (subespacios de 1 dimensión hacia subespacios de mayor dimensionalidad), por lo que, si el subespacio base no tiene el atributo de interés, la cantidad de combinaciones a generar decrece notablemente.

- La cantidad de agrupamientos detectados por el algoritmo modificado es mucho menor que el algoritmo original. Esto se debe a que a pesar de que el algoritmo poda más subespacios que el algoritmo SUBCLU original, el criterio de calidad interno del algoritmo SUBCLU para la detección de los grupos en los subespacios se mantiene.

- El tiempo de ejecución aumentaba exponencialmente con cada atributo adicional, razón por la que las pruebas se limitaron a 10 atributos. Se identificó que la razón del pobre desempeño era el framework ELKI y el pobre uso de paralelismo, se concluye que la actual implementación en ELKI no es apta para pruebas de alta dimensionalidad a menos que se cuente con un sistema que pueda correr de forma ininterrumpida por varios días. Otra razón del pobre desempeño es la implementación del método "ValidateSubspaces", pues los subespacios de menor tamaño son validados varias veces en las distintas iteraciones de la generación de subespacios, ya que valida subespacio k+1, k, k-1, k-2, … 1. Una variante puede ser el uso de una lista de subespacios ya validados para mejorar el desempeño de ejecución.

### 5.4.2 Trabajo futuro

Como se observó en los resultados y se reiteró en las conclusiones, hay varias áreas en las que se puede extender la investigación, o en las que el algoritmo propuesto se puede cambiar o mejorar. En esta sección se presentan algunas ideas para trabajo futuro.

Dados los problemas de desempeño con conjuntos grandes de datos con alta dimensionalidad señalados en la sección anterior, se sugiere implementar el algoritmo en un sistema con mayores capacidades de aprovechar el paralelismo para realizar pruebas con conjuntos de datos de mayor tamaño y dimensionalidad. Se recomienda el uso de Spark Apache [52], framework de código libre de computación distribuida para manejo y análisis de conjuntos de datos grandes (big data). Este framework fue utilizado en este proyecto para el análisis de resultados de los grupos, y demostró grandes capacidades de escalabilidad al permitir crear clústeres con computadoras de escritorio e incluso máquinas virtuales.

En la misma línea de mejora de desempeño, se encontró que la implementación del método "ValidateSubspaces" genera trabajo innecsario, pues los subespacios de menor tamaño son validados varias veces en las distintas iteraciones de la generación de subespacios, ya que valida subespacio k+1, k, k-1, k-2, … 1. Una variante puede ser el uso de una lista de subespacios ya validados para mejorar el desempeño de ejecución.

Otra posible línea de trabajo consiste en estudiar el problema de la posición de la columna dentro del conjunto de datos y la incapacidad del algoritmo de encontrar subespacios si la columna no se encuentra al inicio del conjunto de datos. Una posible opción es modificar la lógica de generación de subespacios de SUBCLU y generar un subespacio s0, el cual incluya el atributo de interés desde el inicio, y luego continúe la ejecución de la misma forma que SUBCLU-R.

En este trabajo se concluyó que era posible añadir restricciones de subespacios al algoritmo SUBCLU. La cantidad de agrupamientos generados no se puede anticipar, sin embargo, si se conoce de antemano que la cantidad de grupos con el atributo de interés es N – d. La generación de los subespacios y agrupamientos que se detectan por el algoritmo propuesto es por restricciones fuertes (a excepción de los d subespacios de una dimensión), y aunque las métricas internas y externas demuestran que la calidad general de los agrupamientos se mantiene, podrían estarse excluyendo agrupamientos de gran calidad con cohesión superior. Se propone modificar el algoritmo de restricciones fuertes a restricciones basadas en pesos, donde, por medio un nuevo parámetro se le pueda otorgar un peso al atributo de interés para guiar la poda de los subespacios. Esto permitiría mayor flexibilidad al algoritmo actual, pues permitiría la priorización de atributos sin que se excluyan grupos de gran calidad que no incluyan el atributo.

Una de las suposiciones basadas en los resultados, es que la cohesión de SUBCLU-R será mejor y la separación peor en la mayoría de las ocasiones comparándolo con los resultados de SUBCLU.

Trabajo futuro podría enfocarse en la ejecución de más experimentos para poder confirmar o negar esta hipótesis.

En este proyecto solo se validó la inclusión de único atributo de interés, sin embargo, se puede extender la implementación, el análisis y las pruebas para que trabaje con listas de atributos de interés.

Finalmente, este trabajo se centró en la capacidad de crear agrupamientos de mayor interés para el usuario, aunque este trabajó se concentró en SUBCLU, dados los resultados de este proyecto, intuitivamente se puede decir, que aplicar la misma lógica de restricciones para la generación, selección o evaluación de los subespacios podría generar resultados similares. Por lo que un trabajo de investigación similar podría ser llevado a cabo para otros algoritmos de agrupamiento de subespacios.

# 6. Bibliografia

[1] Kailing K, Kriegel H, Kroger P, et al. Density connected subspace clustering for high dimensional data -

[2] Subspace clustering for high dimensional data: A review. Lancer Parsons, Ehtesham Haque, Huan liu , 2004

[3] K. Beyer, J.Goldstein, R. Ramakrishnan, and U.Shaft. When is nearest neighbors meaningful. In IDBT, pages 213-235,1999

[4] Singh Vijendra. Efficient Clustering for High Dimensional Data: Subspace Based Clustering and Density Based Clustering, 2011

[5] Emmanuel Muller, Stephan Gunnemman, Ira Assent, Thomas Seidl. Evaluating Clustering in Subspace Projections of High Dimensional Data

[6] K Wagstaff, C Cardie. Clustering with instance level constraints. 2000

[7] K Wagstaff, C Cardie. Constrained k-means clustering with background knowledge. 2001

[8] Viping Kumar, Sonajharia Minz. Feature Selection: A literature Review. 2014

[9] Huan Liu, Motoda, Hiroshi. Feature selection for Knowledge Discovery and Data mining. 1998

[10] Avril L Blum. Selection of Relevant Features and Examples in Machine Learning. 1997

[11] Lance Parsons Subspace Clustering for High Dimensional Data: A Review

[12] P. Berkhin. Survey of clustering data mining techniques. Technical report. 2002

[13] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications.

[14] Chun-Hung Cheng, Ada Waichee Fu, Yi Zhang. Entropy based subspace clustering for mining numeric data. 1999

[15] Lance Parsons, Ehtesham Haque, Huan Liu. Evaluating Subspace Clustering algorithms

[16] Sanjay Goil, Harsha Nagesh, Alok Choudhary. MAFIA: Efficiente and Scalable Subspace Clsutering for Very Large Datasets. 1999

[17] K.G Woo and J. H. Lee. FINDINT: a Fast and Intelligent Subspace Clustering Algorithm using Dimension Voting. 2002

[18] Singh Vijendra. Efficient Clustering for High Dimensional Data: Subspace Based Clustering and Density Based Clustering. 2011

[19] PN Tan, M Steinbach, V Kumar. Data mining cluster analysis: Basic concepts and algorithms, 2013.

[20] Jiawei Han, Micheline Kamber, Jian Pei, HAN Data mining concepts and techniques. 2011. 3rd Edition

[21] Peter J. Rousseeuw. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis, 1987

[22] Richard Ernest Bellman, Dynamic Programming, 1957.

[23] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunupulos, Prabhakar Raghavan, Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. 1998

[24] C.-H. Cheng, A. W. Fu, Y. Zhan, Entropy-based subspace clustering for mining numerical data.1999

[25] M. Ester, H.P, Krigel, J. Sander, X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, 1996.

[26] Elisa Fromont, Adriana Prado, Celine Robardet. Constraint-based Subspace Clustering, 2009

[27] W. DeSarbo, J. Carroll, L. Clark, P. Green. Synthesized clustering:A method for amalgamating alternative clustering bases with differential weighting of variables. 1984

[28] Z. Huang, M. Ng. H. Rong, Z. Li Automated variable weighting in k-means type clustering, 2005

[29] Xiaojun Chen, Yunming Ye, Xiaofei Xu, Joshua Zhexue Huang. A feature grop weighting method for subspace clustering of high-dimensional data.

[30] Kelvin Sim, Vivekanand Gopalkrishnan, Arthur Zimek, Gao Cong, A Survey on enchanced subspace clustering, 2012.

[31] Hans-Peter Kriegel, Peer Kroger, Arthur Zimek. Clustering High Dimensional Data: A Survery on subspace clustering, pattern-based clustering, and correlation clustering

[32] Pavel Berkhin. Survey of Clustering Data mining Techniques. 2006

[33] Joliffe, Principal Component Analysis. 1986. Springer-Verlag

[34] Karin Kailing, Hans-Peter Kriegel, Peer Kroger, Stefanie Wanka. Ranking for Interesting subspaces for Clustering High Dimensional Data. Knowledge Discovery in Databases: PKKD. 2003

[35] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, 2005. Introduction to Data Mining. Chapter 8 Cluster Analysis: Basic Concepts and Algorithms. Page 488 – 554

[36] Ejemplo de graficación de indice silueta. Recuperado el 20 de Julio, 2016, de http://www.mathworks.com/help/stats/silhouette.html

[37] Lifei Chen, Qingshan Jiang, Shengrui Wang. Cluster Validation for Subspace Clustering on High Dimensional Data.2008

[38] Rosaria Silipo, Iris Adae, Aaron Hart, Michael Berthold. Seven Techniques for Dimensionality Reduction. Knime. 2014

[39] Larens van der Maaten, Eric Postma. Dimensionality Reduction: A Comparative Reduction. 2009

[40] ELKI: Environment for Developing KDD-Applications Supported by Index-Structures. (n.d.). Retrieved August 02, 2016, from http://elki.dbs.ifi.lmu.de/

[41] StatSoft, Inc., Electronic Statistics Textbook, Tulsa, OK: StatSoft, 2013.

[42] Kruskal-Wallis H Test using SPSS Statistics. Recuperado el 04 de agosto del 2016, de https://statistics.laerd.com/spss-tutorials/kruskal-wallis-h-test-using-spss-statistics.php

[43] R.E. Walpole, R.H. Myers, S.L Myers. Probability and Statistics for Engineers and Scientists. Pearson Education.

[44] Handbook of Biological Statistics. Recuperado el 04 de agosto del 2016, de http://www.biostathandbook.com/kruskalwallis.html

[45] Guanhua Chen, Dongqing Yang, Shiwei Tang, Meng Shuai. Mining Representative Subspace Clus ters in High-Dimensional Data. Sixth International Conference on Fuzzy Systems and Knowledge Discover. 2009

[46] Introduction to partitioning-based clustering methods with. Recuperado el 7 de agosto de 2016 de http://users.jyu.fi/~samiayr/pdf/introtoclustering_report.pdf

[47] J. G. (n.d.). Density-based Methods. Retrieved from http://www.cse.buffalo.edu/~jing/cse601/fa12/materials/clustering_density.pdf

[48] Gabriela Moise. Finding non-redundant statistically significant regions in high dimensional data. Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining

[49] Euclidean and Euclidean Squared. Recuperado el 12 de agosto de 2016, de http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Euclidean_and_Euclidean_Squared_Distance_Metrics.htm

[50] DBSCAN (RapidMiner Studio Core). Recuperado el 12 de agosto de 2016, de http://docs.rapidminer.com/studio/operators/modeling/segmentation/dbscan.htm

[51] Enrique Amigo, Julio Gonzalo, Javier Artiles. A comparison of Extrinsic Clustering Evaluation Metrics based on Formal Constraints Technical Report. 2008

[52] http://spark.apache.org/documentation.html. Recuperado el 13 de mayo de 2017

[53] https://archive.ics.uci.edu/ml/datasets/US+Census+Data+(1990) . Recuperado el 09 de septiembre de 2016

[54] https://www.rstudio.com/. Recuperado el 10 de mayo de 2017.

# 7. Apéndice

En esta sección se incluye información que puede considerarse relevante o de contexto para este proyecto.

## 7.1 Tamaños de muestra y dimensionalidad analizada

Se experimentó con distintos tamaños de muestra y diferentes dimensionalidades, esto para encontrar la combinación que permitiese realizar múltiples experimentos dado el tiempo que había para desarrollar el proyecto sin comprometer el concepto de alta dimensionalidad bajo el que opera SUBCLU. Cabe destacar que en estos experimentos no se utilizaron las mismas configuraciones paramétricas (épsilon, minpts y dim) que, en los experimentos formales, pues en este punto solo interesaba entender el impacto de la dimensionalidad y conjunto de datos en el tiempo de ejecución. Las pruebas se realizaron con el algoritmo SUBCLU.

Basado en este análisis, se concluyó que la dimensionalidad tiene mayor influencia en el tiempo de ejecución que el tamaño del conjunto de datos. Este descubrimiento es sumamente importante al pensar en implementaciones más escalables de SUBCLU o SUBCLU-R, una solución que aproveche el paralelismo, donde distintos subespacios son procesados por distintos nodos.

| **Tamaño de conjunto de datos** | **Dimensionalidad** | **Notas sobre experimento** |
|---|---|---|
| **2.500.000** | 26 | Se canceló después de 26 horas de ejecución, apenas había procesado 25.000 objetos (1%) en el subespacio de 1 dimensión |
| **500.000** | 19 | Se canceló después de 7 horas pues únicamente había procesado 10.500 objetos en el subespacio de 1 dimensión |
| **125.000** | 19 | Se canceló después de 52 horas pues únicamente había procesado 53.000 objetos en el subespacio de 3 dimensiones |
| **10.000** | 19 | Esta prueba era para determinar si la dimensionalidad o el tamaño del conjunto de datos influía más en el tiempo de ejecución. Se canceló después de 36 horas pues aún se encontraba analizando el subespacio de 3 dimensiones. |
| **125.000** | 12 | Esta prueba era para comprobar el efecto en tiempo de ejecución de |

| | | |
|---|---|---|
| | | menor dimensionalidad. Se canceló después de 40 horas, el algoritmo ya estaba trabajando sobre el espacio de 5 dimensiones, por lo que fue notable la mejora en ejecución a menor dimensionalidad |
| **25.000** | 12 | Apagón canceló el experimento después de 63 horas de ejecución, aun se estaba analizando el subespacio de 8 atributos. Se tenía un tiempo de ejecución proyectado de alrededor de 115 horas. |
| **25.000** | 11 | Experimento completó después de 81 horas de ejecución |
| **10.000** | 11 | Experimento completó después de 59 horas de ejecución |
| **25.000** | 10 | Experimento completó después de 42 horas de ejecución |
| **10.000** | 10 | Experimento completó después de 32 horas de ejecución |

## 7.2 Parámetros de SUBCLU

Las configuraciones en la Tabla, son los valores utilizados en experimentos, en la búsqueda de valores paramétricos que detectaran patrones ocultos en los datos en forma de agrupaciones.

| **Parámetro** | **Descripción** | **Rango de Valores analizados** | **Espacio paramétrico** |
|---|---|---|---|
| **Epsilon ε** | Especifica que tan cerca deben encontrarse objetos entre si para ser considerados parte de la misma agrupación. Si el valor ε es muy pequeño una gran parte de los datos no será agrupada, y un valor ε muy alto hace que la mayoría de los objetos sean agrupados en la misma agrupación. En general, valores pequeños son preferibles. | 3.0, 5, 7, 11 | Entero mayor a 0 |
| **MinPts** | Cantidad de vecinos necesarios en un espacio para ser considerado un grupo. Un minPts bajo significa que puede construir grupos de ruido, por lo que no debe ser un valor muy bajo. | 200, 500, 600, 900 ,1500 | Entero mayor a 0 |

| | | | |
|---|---|---|---|
| **Dimensiones** | Una cantidad alta de dimensiones afecta considerablemente el tiempo de ejecución del algoritmo dado que la plataforma ELKI no paraleliza la ejecución. | 10, 12, 15, 19, 24 | Entero mayor a 0 |

Tabla 7. Configuraciones paramétricas utilizadas en experimentos iniciales

## 7.3 Scripts Spark para análisis de agrupamientos resultado de framework ELKI

En esta sección se incluyen los scripts en pySpark utilizados para obtener las métricas de cohesión y separación. Los scripts se ejecutaron en Spark Hadoop sobre una distribución Linux CDH de Cloudera. Los archivos de los experimentos se subieron a las carpetas de HDFS SUBCLUOriginal y SUBCLUAlonso sobre los que operan los scripts de esta sección.

--SCRIPTS usados para obtener metricas de cohesión y separación

```
def flattened(x):
        return (x[3],x[5],x[6])


def flattenedGetClusters(x):
        x.pop(0) #remove clusterID
        x.pop(0) #remove ClusterName
        x.pop(0) #remove clusterNoiseFlag
        x.pop(0) #remove clusterSize
        x.pop(0) #remove clusterModel Class
        x.pop(1) #remove subspace information of the cluster
        return x

#checks if a dimension exists in a list of dimensions
def containsSubstring(listOfDims, dim):
        list = listOfDims.replace("[","").replace("]","").split(',')
        return dim in list

def calculateAverage(listOfNumbers):
        numberCount = len(listOfNumbers)
        sumOfNumbers = 0
        for index in range(0,numberCount):
                sumOfNumbers = sumOfNumbers+ float(listOfNumbers[index])
        return sumOfNumbers/numberCount
```

```
#testing modified
def calculateCohesion(listX, listOfMeans):
	listX.pop(0)	#remove the clusterMeans first item on the list (easier to remove on this function than on map)
	listX.pop()	#there is an empty space at the end of every cluster, delete
	objCount = len(listX)
	means = listOfMeans.replace("# Cluster Mean: ","").replace(",","").split(' ') #assuming it only comes with whats inside the squared brackets
	subspaceSize = len(means)
	splitObject = 0
	cohesionValue = 0

	for clusterObjectLineIndex in range(0,objCount): #each line corresponds to an object of the cluster
		clusterObject = listX[clusterObjectLineIndex]
		clusterObject = clusterObject.split(' ')
		clusterObject.pop(0) #first item corresponds to ID of the object in cluster, so remove it
		clusterObject.pop()  #last item also corresponds to  ID of the object in cluster, so remove it
		for clusterColumn in range(0, subspaceSize ): #iterate for each column in the object and compare against the mean
			objectAttributeValue = float(clusterObject[clusterColumn])
			cohesionValue = cohesionValue + ((float(means[clusterColumn]) - objectAttributeValue)**2)

	return cohesionValue/objCount #divide the sum of squares between the number of objects

#testing modified
def calculateCohesion2(listX, listOfMeans):
	listX.pop(0)	#remove the clusterMeans first item on the list (easier to remove on this function than on map)
	listX.pop()	#there is an empty space at the end of every cluster, delete
	objCount = len(listX)
	means = listOfMeans.replace("# Cluster Mean: ","").replace(",","").split(' ') #assuming it only comes with whats inside the squared brackets
	subspaceSize = len(means)
	splitObject = 0
	cohesionValue = 0
	aggregatedCohesion = 0
	for clusterObjectLineIndex in range(0,objCount): #each line corresponds to an object of the cluster
		clusterObject = listX[clusterObjectLineIndex]
		clusterObject = clusterObject.split(' ')
		clusterObject.pop(0) #first item corresponds to ID of the object in cluster, so remove it
		clusterObject.pop()  #last item also corresponds to  ID of the object in cluster, so remove it
		cohesionValue = 0
		for clusterColumn in range(0, subspaceSize ): #iterate for each column in the object and compare against the mean
			objectAttributeValue = float(clusterObject[clusterColumn])
			cohesionValue = cohesionValue + ((float(means[clusterColumn]) - objectAttributeValue)**2)
		aggregatedCohesion = aggregatedCohesion + (cohesionValue/subspaceSize)

	return aggregatedCohesion/objCount #divide the sum of squares between the number of objects


#calculate Total Sum of squares, with this we can calculate
```

```
def calculateTotalSum(listX, listOfMeans):
	listX.pop(0)	#remove the clusterMeans first item on the list (easier to remove on this function than on map)
	listX.pop()	#there is an empty space at the end of every cluster, delete
	objCount = len(listX)
	means = listOfMeans.replace("# Cluster Mean: ","").replace(",","").split(' ') #assuming it only comes with whats inside the squared brackets
	subspaceSize = len(means)
	splitObject = 0
	sumValue = 0

	for clusterObjectLineIndex in range(0,objCount): #each line corresponds to an object of the cluster
		clusterObject = listX[clusterObjectLineIndex]
		clusterObject = clusterObject.split(' ')
		clusterObject.pop(0) #first item corresponds to ID of the object in cluster, so remove it
		clusterObject.pop()  #last item also corresponds to  ID of the object in cluster, so remove it
		for clusterColumn in range(0, subspaceSize ): #iterate for each column and sum
			objectAttributeValue = float(clusterObject[clusterColumn])
			sumValue = sumValue +  objectAttributeValue

	return sumValue/objCount #divide the sum of squares between the number of objects
```

```
--create result files for modified algorithm
clusterFiles = sc.wholeTextFiles("/SubcluAlonso/",minPartitions=None, use_unicode=True)
filteredClusterFiles = clusterFiles.filter(lambda (fname, content): 'cluster' in fname)
filesRDD = filteredClusterFiles.map(lambda (fName,content): content)
splitRDD = filesRDD.map(lambda f: f.split('\n'))

clusterInfoHighLvl = splitRDD.map(flattened)
keyPair = clusterInfoHighLvl.map(lambda linea: (linea[2],(linea)))
OrderedByKey = keyPair.sortByKey(ascending=True)
OrderedByKey.saveAsTextFile("/SubcluAlonsoResultsClusterAnalysis")
```

```
--create result files for original algorithm
clusterFiles = sc.wholeTextFiles("/SubcluAlonso/",minPartitions=None, use_unicode=True)
filteredClusterFiles = clusterFiles.filter(lambda (fname, content): 'cluster' in fname)
filesRDD = filteredClusterFiles.map(lambda (fName,content): content)
splitRDD = filesRDD.map(lambda f: f.split('\n'))
clusterInfoHighLvl = splitRDD.map(flattened)
keyPair = clusterInfoHighLvl.map(lambda linea: (linea[2],(linea)))
OrderedByKey = keyPair.sortByKey(ascending=True)
OrderedByKey.saveAsTextFile("/SubcluAlonsoResultsClusterAnalysis")
```

```
clusterFiles = sc.wholeTextFiles("/SubcluOriginal/",minPartitions=None, use_unicode=True)
filteredClusterFiles = clusterFiles.filter(lambda (fname, content): 'cluster' in fname)
filesRDD = filteredClusterFiles.map(lambda (fName,content): content)
splitRDD = filesRDD.map(lambda f: f.split('\n'))
clusterInfoHighLvl = splitRDD.map(flattened)
keyPair = clusterInfoHighLvl.map(lambda linea: (linea[2],(linea)))
OrderedByKey = keyPair.sortByKey(ascending=True)
```

```
OrderedByKey.saveAsTextFile("/SubcluAlonsoResultsClusterAnalysis")


-- Calculo de cohesión y separación para SUBCLU-R
clusterFiles = sc.wholeTextFiles("/SubcluAlonso/",minPartitions=None, use_unicode=True)
filteredClusterFiles = clusterFiles.filter(lambda (fname, content): 'cluster' in fname)
filesRDD = filteredClusterFiles.map(lambda (fName,content): content)
splitRDD = filesRDD.map(lambda f: f.split('\n'))
clusterInfoHighLvl = splitRDD.map(flattenedGetClusters)
cohesionAlonso = clusterInfoHighLvl.map(lambda cluster: calculateCohesion2(cluster, cluster[0])) #calculates cohesion of each cluster
cohesionAlonso = cohesionAlonso.collect()
cohesionAlonsoTotal = calculateAverage(cohesionAlonso)
totalSUMAlonso = clusterInfoHighLvl.map(lambda cluster: calculateTotalSum(cluster, cluster[0])) #calculates cohesion of each cluster
totalSUMAlonso = totalSUMAlonso.collect()
totalSUMAlonso = calculateAverage(totalSUMAlonso)
BSSAlonso = totalSUMAlonso - cohesionAlonsoTotal



-- Cálculo de cohesión y separación para SUBCLU
clusterFiles = sc.wholeTextFiles("/SubcluOriginal/",minPartitions=None, use_unicode=True)
filteredClusterFiles = clusterFiles.filter(lambda (fname, content): 'cluster' in fname)
filesRDD = filteredClusterFiles.map(lambda (fName,content): content)
splitRDD = filesRDD.map(lambda f: f.split('\n'))
clusterInfoHighLvl = splitRDD.map(flattenedGetClusters)
cohesionOriginal = clusterInfoHighLvl.map(lambda cluster: calculateCohesion2(cluster, cluster[0])) #calculates cohesion of each cluster
cohesionOriginal = cohesionOriginal.collect()
cohesionOriginalTotal = calculateAverage(cohesionOriginal)

totalSUMOriginal = clusterInfoHighLvl.map(lambda cluster: calculateTotalSum(cluster, cluster[0])) #calculates cohesion of each cluster
totalSUMOriginal = totalSUMOriginal.collect()
totalSUMOriginal = calculateAverage(totalSUMOriginal)
BSSOriginal = totalSUMOriginal – cohesionOriginalTotal
```

## 7.4 Archivos de agrupamiento generados por ELKI

A continuación, un agrupamiento generado por el framework ELKI. Cada uno de estos archivos fue analizado por el código Spark adjunto en la sección 7.3

- Cluster Model: Id del grupo (autogenerado)
- Cluster Name: Nombre del grupo (autogenerado, el mismo que el ID)
- Cluster Size: Cantidad de objetos dentro del grupo
- Model Class: Subclase de agrupamiento (categorías en las que se subdividen los algoritmos en el framework Elki).
- Cluster Mean: Vector que representa la media del grupo para cada atributo
- Subspace Dimensions: Lista de atributos parte del grupo (subespacio). Por ejemplo, la muestra que se presenta a continuación es de un grupo que tiene 7 atributos de los 10 del conjunto de datos analizado).
- Resto del archivo : El resto del archivo es la lista de todos los objetos y el valor de cada atributo.

```
# Cluster: cluster_114
# Cluster name: cluster_114
# Cluster size: 444
# Model class: de.lmu.ifi.dbs.elki.data.model.SubspaceModel
# Cluster Mean: 6.087837837837838, 0.2882882882882883, 2.6666666666666665, 0.009009009009009009, 0.02702702702702703, 0.25900900900900903, 1.0495495495495495, 1.8018018018018018, 0.9054054054054054, 10.268018018018019
# Subspace: Dimensions: [3, 4, 5, 7, 8, 9, 10]
ID=26909 3.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6909
ID=22678 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2678
ID=29429 6.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9429
ID=21144 6.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1144
ID=27488 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7488
ID=24008 6.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 4008
ID=22523 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2523
ID=20322 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 322
ID=25363 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5363
ID=25353 6.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5353
ID=28111 5.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 8111
ID=29837 5.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9837
ID=27928 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7928
ID=22630 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2630
ID=22501 6.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2501
ID=29034 7.0 3.0 2.0 0.0 0.0 2.0 1.0 1.0 1.0 10.0 9034
ID=25569 5.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5569
ID=21715 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1715
ID=27389 7.0 0.0 2.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 7389
```

ID=21978 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1978
ID=27925 5.0 0.0 2.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 7925
ID=21786 7.0 0.0 2.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 1786
ID=26683 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6683
ID=26927 3.0 2.0 3.0 0.0 0.0 6.0 1.0 1.0 1.0 10.0 6927
ID=25500 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5500
ID=26901 7.0 3.0 3.0 0.0 0.0 2.0 1.0 1.0 1.0 10.0 6901
ID=27502 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7502
ID=23535 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3535
ID=25508 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5508
ID=27970 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7970
ID=23370 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3370
ID=28676 7.0 3.0 2.0 0.0 0.0 3.0 2.0 1.0 1.0 10.0 8676
ID=27795 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7795
ID=20606 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 606
ID=23219 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3219
ID=29480 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9480
ID=22507 7.0 0.0 2.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 2507
ID=26257 3.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 6257
ID=23876 7.0 0.0 2.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 3876
ID=22396 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2396
ID=28038 7.0 0.0 2.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 8038
ID=20009 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9
ID=20628 4.0 4.0 3.0 0.0 0.0 3.0 1.0 1.0 1.0 10.0 628
ID=27620 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7620
ID=22325 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 2325
ID=25856 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5856
ID=22252 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2252
ID=20912 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 912
ID=27807 3.0 0.0 2.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 7807
ID=25577 4.0 0.0 2.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 5577
ID=21939 7.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 1939
ID=25922 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5922
ID=26441 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6441
ID=21604 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1604
ID=26238 7.0 0.0 2.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 6238
ID=22361 4.0 4.0 2.0 0.0 0.0 6.0 1.0 2.0 1.0 10.0 2361
ID=25064 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5064
ID=27530 6.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 7530
ID=25137 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5137
ID=26770 7.0 0.0 2.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 6770
ID=28632 3.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 8632
ID=24977 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 4977
ID=29183 4.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 9183
ID=24973 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 4973
ID=23263 4.0 0.0 2.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 3263

ID=26130 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6130
ID=20221 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 221
ID=27336 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7336
ID=26237 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6237
ID=29010 7.0 0.0 2.0 0.0 0.0 0.0 1.0 2.0 1.0 10.0 9010
ID=29059 3.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9059
ID=21078 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1078
ID=27643 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7643
ID=29104 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9104
ID=28316 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 8316
ID=29329 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9329
ID=25659 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5659
ID=22829 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2829
ID=22936 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2936
ID=24658 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 4658
ID=21487 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1487
ID=20188 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 188
ID=22267 4.0 0.0 3.0 0.0 0.0 0.0 1.0 2.0 1.0 10.0 2267
ID=22693 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 2693
ID=23751 6.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 3751
ID=29824 5.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 9824
ID=29565 6.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 9565
ID=21098 4.0 4.0 3.0 0.0 0.0 2.0 2.0 1.0 1.0 10.0 1098
ID=27292 7.0 3.0 3.0 0.0 0.0 2.0 1.0 1.0 1.0 11.0 7292
ID=23187 7.0 0.0 3.0 0.0 0.0 0.0 1.0 2.0 1.0 10.0 3187
ID=20692 7.0 2.0 3.0 0.0 0.0 4.0 1.0 1.0 1.0 11.0 692
ID=20956 6.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 956
ID=29805 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 9805
ID=23446 7.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 3446
ID=26349 3.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 6349
ID=27438 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 7438
ID=28724 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 8724
ID=24844 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 4844
ID=22685 4.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 2685
ID=21381 7.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 1381
ID=24824 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 4824
ID=20698 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 698
ID=26105 3.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 6105
ID=26867 5.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 6867
ID=25192 6.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 5192
ID=28426 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 8426
ID=25288 7.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 5288
ID=22147 4.0 0.0 3.0 0.0 1.0 0.0 1.0 1.0 1.0 10.0 2147
ID=28735 4.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 8735
ID=28302 5.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 8302
ID=29845 5.0 4.0 1.0 0.0 0.0 4.0 0.0 1.0 1.0 10.0 9845

ID=23035 3.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 3035
ID=22682 4.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 2682
ID=24322 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 4322
ID=24051 7.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 4051
ID=22849 5.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 2849
ID=26199 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 6199
ID=23987 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 3987
ID=26140 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 6140
ID=20166 7.0 0.0 1.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 166
ID=25380 7.0 0.0 1.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 5380
ID=23664 7.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 3664
ID=20268 7.0 0.0 3.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 268
ID=23887 5.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 3887
ID=23421 7.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 3421
ID=23942 4.0 0.0 2.0 0.0 0.0 0.0 0.0 1.0 1.0 11.0 3942
ID=23238 5.0 0.0 2.0 0.0 0.0 0.0 2.0 1.0 1.0 11.0 3238
ID=22584 7.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 2584
ID=26993 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 6993
ID=25558 3.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 5558
ID=21211 5.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 1211
ID=28489 6.0 4.0 3.0 0.0 0.0 2.0 2.0 1.0 1.0 10.0 8489
ID=24727 7.0 0.0 1.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 4727
ID=28707 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 8707
ID=28215 6.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 8215
ID=27791 6.0 3.0 3.0 0.0 0.0 3.0 2.0 1.0 1.0 10.0 7791
ID=25135 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 5135
ID=26242 4.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 6242
ID=29990 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 9990
ID=28171 4.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 8171
ID=29315 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 9315
ID=26512 4.0 4.0 1.0 0.0 0.0 3.0 2.0 1.0 1.0 10.0 6512
ID=25298 6.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 5298
ID=28731 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 8731
ID=23494 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 11.0 3494
ID=26411 3.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 11.0 6411
ID=24866 5.0 0.0 3.0 0.0 1.0 0.0 0.0 1.0 1.0 10.0 4866
ID=24129 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 11.0 4129
ID=21699 7.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 11.0 1699
ID=26307 4.0 3.0 3.0 0.0 0.0 4.0 1.0 2.0 1.0 11.0 6307
ID=27127 5.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 11.0 7127
ID=23182 7.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 11.0 3182
ID=22013 7.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 11.0 2013
ID=24306 7.0 3.0 2.0 0.0 0.0 2.0 1.0 1.0 1.0 12.0 4306
ID=28020 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 8020
ID=25481 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5481
ID=23660 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3660

ID=25350 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5350
ID=28841 6.0 0.0 3.0 1.0 1.0 0.0 0.0 1.0 1.0 10.0 8841
ID=22225 7.0 3.0 2.0 0.0 0.0 2.0 1.0 3.0 1.0 10.0 2225
ID=23538 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3538
ID=24199 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 4199
ID=27974 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7974
ID=26329 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 6329
ID=22345 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 2345
ID=28910 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 8910
ID=20790 6.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 790
ID=26605 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6605
ID=29940 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 9940
ID=28062 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 8062
ID=24093 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 4093
ID=21780 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1780
ID=28349 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 8349
ID=25262 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5262
ID=26536 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6536
ID=21763 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1763
ID=23861 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3861
ID=25909 7.0 3.0 2.0 0.0 0.0 2.0 1.0 3.0 1.0 10.0 5909
ID=22178 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 2178
ID=28467 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 8467
ID=21276 6.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 1276
ID=29782 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9782
ID=25958 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 5958
ID=25127 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 5127
ID=24992 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 4992
ID=23157 5.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3157
ID=21478 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 1478
ID=20015 4.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 15
ID=29324 4.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9324
ID=22937 4.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2937
ID=23222 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3222
ID=21622 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1622
ID=23257 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3257
ID=24431 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 4431
ID=26920 4.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6920
ID=27823 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7823
ID=26037 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6037
ID=21509 5.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1509
ID=26108 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 6108
ID=27565 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 7565
ID=24582 5.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 4582
ID=24778 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 4778
ID=24040 4.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 4040

ID=21923 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 1923
ID=26265 7.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 6265
ID=21985 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 1985
ID=22901 4.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 2901
ID=23823 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 3823
ID=24475 7.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 4475
ID=23550 5.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 3550
ID=26175 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 6175
ID=29273 4.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 9273
ID=27007 7.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 7007
ID=21403 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 1403
ID=20251 4.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 251
ID=20198 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 198
ID=20430 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 430
ID=20886 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 886
ID=29650 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 9.0 9650
ID=29890 4.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 9890
ID=23147 3.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 3147
ID=25312 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 5312
ID=23154 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 3154
ID=25011 5.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 5011
ID=21309 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 1309
ID=26082 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 6082
ID=27934 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 12.0 7934
ID=25078 7.0 3.0 1.0 0.0 0.0 3.0 1.0 3.0 1.0 10.0 5078
ID=24107 7.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 4107
ID=22797 3.0 4.0 4.0 0.0 0.0 4.0 0.0 1.0 1.0 10.0 2797
ID=22616 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 9.0 2616
ID=25834 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 5834
ID=21912 5.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 1912
ID=22745 5.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 2745
ID=21657 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 1657
ID=22277 3.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 2277
ID=23406 3.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 3406
ID=27742 7.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 10.0 7742
ID=28984 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 8984
ID=22297 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 2297
ID=24320 4.0 4.0 4.0 0.0 0.0 4.0 1.0 1.0 1.0 9.0 4320
ID=28570 4.0 3.0 4.0 0.0 0.0 2.0 1.0 1.0 1.0 9.0 8570
ID=22940 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 2940
ID=27611 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 7611
ID=26315 6.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 6315
ID=26714 5.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 6714
ID=28442 7.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 12.0 8442
ID=25488 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 5488
ID=28144 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 8144

ID=22660 4.0 0.0 3.0 0.0 0.0 0.0 1.0 1.0 1.0 12.0 2660
ID=26194 5.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 11.0 6194
ID=26142 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 6142
ID=28636 7.0 4.0 2.0 0.0 0.0 4.0 1.0 3.0 1.0 11.0 8636
ID=22802 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 2802
ID=23975 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 3975
ID=26098 7.0 0.0 2.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 6098
ID=23034 7.0 0.0 2.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 3034
ID=24140 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 4140
ID=24168 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 4168
ID=21634 7.0 0.0 2.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 1634
ID=29149 7.0 3.0 1.0 0.0 0.0 2.0 1.0 3.0 1.0 10.0 9149
ID=28101 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 8101
ID=20168 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 168
ID=26417 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 6417
ID=29318 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 9318
ID=28623 7.0 3.0 3.0 0.0 0.0 2.0 1.0 3.0 1.0 10.0 8623
ID=24514 6.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 4514
ID=27010 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 7010
ID=28122 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 8122
ID=20373 7.0 3.0 3.0 0.0 0.0 2.0 1.0 3.0 1.0 10.0 373
ID=26932 7.0 3.0 3.0 0.0 0.0 6.0 1.0 3.0 1.0 10.0 6932
ID=26646 7.0 0.0 2.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 6646
ID=20946 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 946
ID=26437 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 6437
ID=26773 6.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 6773
ID=25401 6.0 0.0 3.0 0.0 0.0 0.0 3.0 1.0 1.0 10.0 5401
ID=29226 7.0 0.0 2.0 0.0 0.0 0.0 0.0 3.0 1.0 10.0 9226
ID=23107 7.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 3107
ID=29911 6.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 9911
ID=25757 4.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 5757
ID=22453 7.0 0.0 1.0 0.0 0.0 0.0 3.0 1.0 1.0 10.0 2453
ID=25413 5.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 5413
ID=28776 4.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 8776
ID=20034 5.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 10.0 34
ID=26132 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 6132
ID=26008 7.0 0.0 3.0 0.0 0.0 0.0 3.0 2.0 1.0 10.0 6008
ID=20605 7.0 0.0 1.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 605
ID=22860 7.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 11.0 2860
ID=24143 5.0 0.0 3.0 0.0 0.0 0.0 2.0 1.0 1.0 12.0 4143
ID=22965 6.0 0.0 1.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 2965
ID=23145 7.0 0.0 3.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 3145
ID=25754 7.0 0.0 1.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 5754
ID=21235 7.0 0.0 1.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 1235
ID=27150 7.0 0.0 1.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 7150
ID=25100 7.0 0.0 3.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 5100

ID=24494 6.0 0.0 1.0 0.0 0.0 0.0 2.0 1.0 1.0 12.0 4494
ID=28810 7.0 0.0 3.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 8810
ID=26590 7.0 0.0 3.0 0.0 0.0 0.0 0.0 3.0 1.0 10.0 6590
ID=29153 6.0 3.0 3.0 0.0 0.0 4.0 1.0 3.0 1.0 11.0 9153
ID=26622 7.0 0.0 1.0 0.0 0.0 0.0 0.0 3.0 1.0 10.0 6622
ID=26228 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 6228
ID=29617 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 9617
ID=26642 7.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 9.0 6642
ID=23949 6.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 11.0 3949
ID=20681 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 681
ID=24173 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 4173
ID=26709 4.0 0.0 4.0 1.0 1.0 0.0 1.0 1.0 1.0 10.0 6709
ID=25748 4.0 0.0 4.0 0.0 0.0 0.0 0.0 1.0 1.0 11.0 5748
ID=29457 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 9457
ID=27112 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 9.0 7112
ID=22496 7.0 0.0 2.0 0.0 1.0 0.0 0.0 3.0 1.0 10.0 2496
ID=28779 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 8779
ID=21118 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 9.0 1118
ID=25180 4.0 4.0 4.0 0.0 0.0 4.0 1.0 2.0 1.0 9.0 5180
ID=23534 6.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 9.0 3534
ID=22485 6.0 0.0 4.0 0.0 1.0 0.0 1.0 1.0 1.0 11.0 2485
ID=22982 3.0 0.0 3.0 0.0 1.0 0.0 0.0 1.0 1.0 12.0 2982
ID=28519 3.0 1.0 3.0 0.0 0.0 1.0 3.0 2.0 1.0 9.0 8519
ID=29141 6.0 4.0 3.0 0.0 0.0 4.0 2.0 3.0 1.0 11.0 9141
ID=27469 7.0 4.0 3.0 0.0 0.0 2.0 2.0 3.0 1.0 11.0 7469
ID=21413 7.0 0.0 0.0 0.0 0.0 0.0 1.0 2.0 0.0 11.0 1413
ID=22770 7.0 0.0 0.0 0.0 0.0 0.0 2.0 1.0 0.0 11.0 2770
ID=20311 7.0 0.0 0.0 0.0 0.0 0.0 2.0 1.0 0.0 11.0 311
ID=21920 7.0 0.0 0.0 0.0 0.0 0.0 2.0 2.0 0.0 10.0 1920
ID=20010 6.0 0.0 0.0 0.0 0.0 0.0 2.0 1.0 0.0 11.0 10
ID=28315 4.0 0.0 4.0 1.0 1.0 0.0 0.0 1.0 1.0 10.0 8315
ID=23195 7.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 11.0 3195
ID=29381 7.0 0.0 3.0 0.0 0.0 0.0 0.0 3.0 1.0 11.0 9381
ID=29587 7.0 0.0 0.0 0.0 0.0 0.0 2.0 1.0 0.0 11.0 9587
ID=29685 7.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 11.0 9685
ID=28228 7.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 11.0 8228
ID=26306 7.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 11.0 6306
ID=28968 5.0 0.0 0.0 0.0 0.0 0.0 0.0 2.0 0.0 10.0 8968
ID=21033 7.0 0.0 3.0 0.0 0.0 0.0 0.0 3.0 1.0 9.0 1033
ID=27258 6.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 11.0 7258
ID=21394 7.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 11.0 1394
ID=28870 7.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 11.0 8870
ID=24822 6.0 4.0 4.0 0.0 0.0 2.0 1.0 3.0 1.0 10.0 4822
ID=24036 3.0 0.0 2.0 0.0 0.0 0.0 3.0 1.0 1.0 12.0 4036
ID=24745 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 4745
ID=29555 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 9555

ID=29303 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 9303
ID=25446 5.0 4.0 4.0 0.0 0.0 4.0 1.0 1.0 1.0 12.0 5446
ID=24324 4.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 12.0 4324
ID=21550 6.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 1550
ID=20381 7.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 12.0 381
ID=27117 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 7117
ID=29054 7.0 0.0 0.0 0.0 0.0 0.0 2.0 2.0 0.0 11.0 9054
ID=27401 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 7401
ID=26393 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 6393
ID=23645 6.0 0.0 4.0 0.0 2.0 0.0 1.0 1.0 1.0 10.0 3645
ID=27433 4.0 0.0 4.0 0.0 0.0 0.0 1.0 1.0 1.0 12.0 7433
ID=28007 6.0 0.0 4.0 1.0 1.0 0.0 0.0 1.0 1.0 11.0 8007
ID=20189 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 189
ID=22713 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 10.0 2713
ID=23100 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 3100
ID=21800 6.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1800
ID=23800 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 9.0 3800
ID=21552 6.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1552
ID=23577 6.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 3577
ID=25421 4.0 0.0 4.0 0.0 1.0 0.0 1.0 1.0 1.0 12.0 5421
ID=21026 4.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1026
ID=21821 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 1821
ID=21031 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 9.0 1031
ID=27422 7.0 0.0 4.0 0.0 0.0 0.0 0.0 3.0 1.0 10.0 7422
ID=26084 5.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6084
ID=24316 7.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 4316
ID=21658 7.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 1658
ID=20474 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 474
ID=20916 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 916
ID=20239 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 11.0 239
ID=23448 5.0 4.0 5.0 0.0 0.0 3.0 1.0 1.0 1.0 10.0 3448
ID=20437 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 437
ID=27741 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 7741
ID=26184 7.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 12.0 6184
ID=22112 5.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2112
ID=28154 7.0 0.0 4.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 8154
ID=22521 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 2521
ID=25444 7.0 0.0 4.0 0.0 0.0 0.0 2.0 3.0 1.0 10.0 5444
ID=24288 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 4288
ID=22272 7.0 0.0 4.0 0.0 0.0 0.0 2.0 1.0 1.0 12.0 2272
ID=22700 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 2700
ID=27652 4.0 4.0 4.0 0.0 0.0 2.0 2.0 1.0 1.0 12.0 7652
ID=23936 4.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 3936
ID=29281 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 9281
ID=25311 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 5311
ID=28643 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 8643

ID=22418 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 2418
ID=26761 6.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 6761
ID=25749 5.0 3.0 5.0 0.0 0.0 2.0 1.0 1.0 1.0 10.0 5749
ID=29540 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 9540
ID=22243 7.0 0.0 4.0 0.0 0.0 0.0 0.0 3.0 1.0 10.0 2243
ID=21201 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 10.0 1201
ID=25542 4.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 5542
ID=22814 6.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 2814
ID=29302 6.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 9302
ID=27257 7.0 0.0 3.0 0.0 0.0 0.0 1.0 3.0 1.0 12.0 7257
ID=29445 7.0 0.0 1.0 0.0 0.0 0.0 1.0 3.0 1.0 12.0 9445
ID=23734 7.0 0.0 2.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 3734
ID=23307 7.0 0.0 2.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 3307
ID=26045 7.0 0.0 2.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 6045
ID=27025 7.0 0.0 2.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 7025
ID=26731 4.0 0.0 2.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 6731
ID=28912 4.0 0.0 2.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 8912
ID=28741 6.0 0.0 2.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 8741
ID=25687 7.0 0.0 3.0 0.0 0.0 0.0 3.0 3.0 1.0 10.0 5687
ID=22119 7.0 0.0 1.0 0.0 0.0 0.0 3.0 3.0 1.0 10.0 2119
ID=27624 7.0 0.0 5.0 0.0 0.0 0.0 1.0 1.0 1.0 10.0 7624
ID=29935 7.0 0.0 0.0 0.0 0.0 0.0 3.0 1.0 0.0 10.0 9935
ID=20659 6.0 4.0 5.0 0.0 0.0 4.0 1.0 1.0 1.0 10.0 659
ID=26858 7.0 0.0 2.0 0.0 0.0 0.0 1.0 4.0 1.0 11.0 6858
ID=26680 7.0 0.0 3.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 6680
ID=29649 6.0 0.0 3.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 9649
ID=20006 7.0 0.0 1.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 6
ID=21182 4.0 0.0 3.0 0.0 0.0 0.0 1.0 4.0 1.0 11.0 1182
ID=25988 7.0 0.0 3.0 0.0 0.0 0.0 1.0 4.0 1.0 11.0 5988
ID=28281 7.0 0.0 1.0 0.0 0.0 0.0 1.0 4.0 1.0 11.0 8281
ID=20194 4.0 0.0 3.0 0.0 0.0 0.0 2.0 4.0 1.0 10.0 194
ID=24907 7.0 0.0 3.0 0.0 0.0 0.0 2.0 4.0 1.0 10.0 4907
ID=20997 4.0 0.0 3.0 0.0 0.0 0.0 1.0 4.0 1.0 9.0 997
ID=28662 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 11.0 8662
ID=26421 7.0 0.0 1.0 0.0 0.0 0.0 2.0 4.0 1.0 11.0 6421
ID=27894 7.0 0.0 4.0 0.0 0.0 0.0 0.0 3.0 1.0 11.0 7894
ID=23894 7.0 0.0 4.0 0.0 0.0 0.0 0.0 3.0 1.0 11.0 3894
ID=24540 7.0 0.0 0.0 0.0 0.0 0.0 2.0 3.0 0.0 10.0 4540
ID=20811 7.0 0.0 0.0 0.0 0.0 0.0 2.0 3.0 0.0 10.0 811
ID=29536 7.0 0.0 3.0 0.0 0.0 0.0 2.0 3.0 1.0 8.0 9536
ID=25947 4.0 0.0 3.0 0.0 0.0 0.0 0.0 4.0 1.0 11.0 5947
ID=24201 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 11.0 4201
ID=24142 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 11.0 4142
ID=23103 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 11.0 3103
ID=21895 7.0 0.0 0.0 0.0 0.0 0.0 1.0 3.0 0.0 11.0 1895
ID=28915 7.0 0.0 0.0 0.0 0.0 0.0 0.0 3.0 0.0 11.0 8915

ID=29404 7.0 0.0 0.0 0.0 0.0 0.0 2.0 3.0 0.0 11.0 9404
ID=22779 5.0 0.0 4.0 0.0 0.0 0.0 1.0 4.0 1.0 10.0 2779
ID=29244 7.0 0.0 2.0 0.0 0.0 0.0 0.0 4.0 1.0 8.0 9244
ID=27209 7.0 0.0 0.0 0.0 0.0 0.0 1.0 4.0 0.0 10.0 7209
ID=20023 6.0 0.0 0.0 0.0 0.0 0.0 1.0 4.0 0.0 10.0 23
ID=25782 6.0 0.0 0.0 0.0 0.0 0.0 1.0 4.0 0.0 10.0 5782
ID=23754 4.0 0.0 0.0 0.0 0.0 0.0 1.0 4.0 0.0 10.0 3754
ID=28088 7.0 4.0 2.0 0.0 0.0 2.0 1.0 5.0 1.0 10.0 8088
ID=24147 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 8.0 4147
ID=29612 7.0 0.0 4.0 0.0 0.0 0.0 1.0 3.0 1.0 8.0 9612
ID=29048 7.0 0.0 1.0 0.0 0.0 0.0 1.0 4.0 1.0 12.0 9048

## 7.5 Código SUBCLU-R

Las modificaciones descritas en este trabajo se encuentran en el repositorio GITHUB en https://github.com/alonvalle1788/SUBCLU-R.

Las modificaciones se realizaron solamente en la clase SUBCLU.java, para utilizar las modificaciones solo es necesario solo es necesario descargar el código fuente ELKI y sustituir la SUBCLU.java y compilar la solución.

## 7.6 Código R utilizado para el análisis de las muestras

```
--generar subset 1
set.seed(sample(1:100, 1, FALSE)) --valor semilla : 52
subset1 <- sample(nrow(USCensusData), 10000, replace=FALSE)
muestra1 <- USCensusData[subset1,]


--generar subset 2
set.seed(sample(1:100, 1, FALSE)) --resultado: --valor semilla : 24
subset2 <- sample(nrow(USCensusData), 10000, replace=FALSE)
muestra2 <- USCensusData[subset2,]


--generar subset 3
set.seed(sample(1:100, 1, FALSE)) --valor semilla : 44
subset3 <- sample(nrow(USCensusData), 10000, replace=FALSE)
muestra3 <- USCensusData[subset3,]


--identificar comunes entre subset1 y subset2
common.idsAandB <- (subset1$caseid %in% subset2$caseid)

--identificar comunes entre subset1 y subset3
common.idsAandC <- (subset1 %in% subset3)

--identificar comunes entre subset2 y subset3
common.idsBandC <- (subset2 %in% subset3)

--ver coincidencias
summary(common.idsAandB)
summary(common.idsAandC)
summary(common.idsBandB)
```

## 7.7 Tabla de similitud entre muestras del conjunto de datos

Las muestras obtenidas del conjunto de datos USCensus1990 se compararon entre sí para verificar que fuesen diferentes. Se encontró un promedio de similitud de un 0.406% entre las muestras. En la Tabla 8 se observa el desglose de las similitudes.

| | Muestra 1 | Muestra 2 | Muestra 3 |
|---|---|---|---|
| Muestra 1 | | 0.34% | 0.41% |
| Muestra 2 | 0.34% | | 0.47% |
| Muestra 3 | 0.41% | 0.47% | |

Tabla 8. Tabla de similitud entre muestras

## 7.8 Configuración ELKI para ejecución de algoritmos

Adicional a los parámetros del algoritmo SUBCLU utilizados, establecidos en la sección 5.2.1, el framework ELKI tiene muchos parámetros adicionales, por ejemplo, filtros sobre el conjunto de datos seleccionados, el evaluador del algoritmo seleccionado, entre otros. A continuación, se presentan todos los valores paramétricos utilizados, la mayoría son los valores por defecto del sistema.

| Parámetro | Valor |
|---|---|
| **verbose** | True |
| **enableDebug** | True |
| **db** | Default: StaticArrayDatabase |
| **dbc** | Default: FileBasedDatabaseConnection |
| **dbc.in** | C:\users\alonso\Downloads\muestra.csv |
| **dbc.parser** | Default: NumberVectorLabelParser |
| **parser.colsep** | Default: \s*[,;s]s* |
| **parser.quote** | Default: “ |
| **string.comment** | Default: ^\s*(#\|//\|;).*$ |
| **parser.labelIndices** | |
| **parser.vector-type** | Default: DoubleVector |
| **db.filter** | |
| **db.index** | |
| **time** | True |
| **algorithm** | clustering.subspace.SUBCLU |
| **Subclu.distancefunction *** | SubspaceManhattanDistanceFunction / SubspaceEuclideanDistanceFunction |
| **distance.dims** | 10 |
| **subclu.epsilon** | 3.0 |
| **subclu.minpts** | 600 |
| **evaluator** | Clustering.internal.EvaluateSillhouette |
| **silhouette.distance** | SubspaceManhattanDistanceFunction / SubspaceEuclideanDistanceFunction |
| **silhouette.noisehandling** | Default: TREAT_NOISE_AS_SINGLETONS |
| **resulthandler** | ResultWriter |
| **out** | C:\users\alonso\Downloads\SUBCLUExperiment |
| **out.gzip** | Default: false |
| **out.silentoverwrite** | Default: false |
| **out.filter** | |

Tabla 9. Configuración ELKI para ejecución de los experimentos.

Programa de Maestría
en Computación
Teléfono: 2550-2402

**APROBACIÓN DE LA TESIS**

# "Propuesta de algoritmo que combina el agrupamiento en subespacios basado en densidad y el agrupamiento basado en restricciones para la detección de grupos que incluyan atributos de interés en conjuntos de datos de alta dimensionalidad"

**TRIBUNAL EXAMINADOR**

Máster Luis Alexander Calvo Valverde
Profesor Asesor

Doctor José Enrique Araya Monge
Profesor Lector

Máster Jacqueline Tatiana Solís Céspedes
Profesor Externo

Doctor Roberto Cortés Morales
Coordinador del Programa
de Maestría en Computación

Junio, 2017