CIMAT

Centro de Investigación en Matemáticas, A.C.

Agrupamiento semi-supervisado con restricciones a nivel instancias mediante algoritmos evolutivos

T E S I S

Que para obtener el grado de

Maestra en Cómputo Estadístico

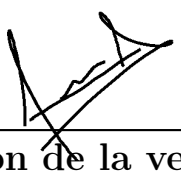
Presenta:

Erika Tatiana Rueda Santos

Directores de Tesis:

Dr. Alejandro Rosales Pérez

Dr. Norberto Alejandro Hernández Leandro



Autorización de la versión final

Monterrey, Nuevo León, 31 de Enero de 2022

Datos del jurado

Datos del alumno:
Erika Tatiana Rueda Santos
Centro de Investigación en Matemáticas, CIMAT
Unidad Monterrey
Maestría en Cómputo Estadístico

Datos del director de tesis:
Dr. Alejandro Rosales Pérez
CIMAT, Monterrey

Datos del co-asesor:
Dr. Norberto Alejandro Hernández Leandro
CIMAT, Monterrey

Datos del sinodal 1:
Dra. Martha Selene Casas Ramírez
CIMAT, Monterrey

Datos del sinodal 2:
Dr. Salvador García López
Universidad de Granada

Datos del trabajo escrito:
Agrupamiento semi-supervisado con restricciones a nivel instancias mediante algoritmos evolutivos.
77 páginas.
2022.

Dedicatoria

A mi familia,
mis axiomas.

... Vi el Aleph, desde todos los puntos,
vi en el Aleph la tierra,
y en la tierra otra vez el Aleph
y en el Aleph la tierra,
vi mi cara y mis vísceras,
vi tu cara, y sentí vértigo y lloré,
por que mis ojos habían visto ese objeto
secreto y conjetural,
cuyo nombre usurpan los hombres,
pero que ningún hombre ha mirado:
el inconcebible universo.

Jorge Luis Borges. El Aleph.

Agradecimientos

A mis padres Ana Santos y Atilano Rueda, pilares fundamentales en mi vida cuyo apoyo ha sido inconmensurable. Gracias por acompañarme en el viaje de este sueño.

A mis hermanos, sobrinos y cuñadas, por su gran apoyo, compañía y alegría.

A Amilcar, por cada palabra de aliento y por las alegrías.

A mi director de tesis, el Dr. Alejandro Rosales Pérez, por la paciencia, tiempo y gran apoyo brindados durante la realización de este trabajo. Gracias por estar siempre al pendiente de los avances, por su retroalimentación y por compartir su conocimiento.

A mi Co-asesor el Dr. Norberto Alejandro Hernández Leandro y a mis sinodales, la Dra. Martha Selene Casas Ramírez y el Dr. Salvador García López, por todo el tiempo brindado en las revisiones; sus valiosos comentarios mejoraron ampliamente este trabajo.

Al CONACyT por el apoyo económico otorgado a través de la beca 16934 y al Centro de Investigación en Matemáticas A. C. Monterrey y al Laboratorio de Supercómputo del Bajío (proyecto CONACyT 300832) por las facilidades otorgadas para la realización de la tesis.

Resumen

Existe una gran variedad de problemas en el mundo real en los que se desea aprender un modelo de clasificación a partir de un conjunto de datos, pero las etiquetas de clase con las que se dispone no son suficientes. El agrupamiento es una alternativa para aprender a partir de datos no etiquetados y, recientemente, se ha mostrado que produce mejores agrupamientos cuando es retroalimentado con información parcial del conjunto de datos, la cual puede ser representada en forma de restricciones. Este tipo de agrupamiento se le conoce como agrupamiento semi-supervisado con restricciones. El agrupamiento con restricciones es un problema desafiante, ya que su solución no sólo implica explorar el espacio de búsqueda de manera eficiente, sino que también debe asegurar que las particiones resultantes sean factibles. Diversos métodos del estado del arte se han propuesto para dar solución al problema de agrupamiento haciendo uso de restricciones a nivel de instancias. No obstante, estos métodos en general asumen que se conoce el número de grupos que contiene el conjunto de datos, lo cual es típicamente desconocido en problemas reales. Esta tesis propone MOECC: *Multi-Objective Evolutionary Constrained Clustering*. MOECC formula el problema de agrupamiento como un problema multi-objetivo, considerando la compacidad y la conectividad de los grupos como los objetivos a optimizar. Además, adapta un mecanismo basado en ϵ-restringido para el manejo de las restricciones a nivel instancias. MOECC implementa operadores de cruza y mutación especialmente diseñados para agrupamiento e incorpora una estrategia para ayudar a escoger de manera automática el número de grupos. La eficacia de MOECC es comparada con COP-K-Medias, LCVQE, RDPM y ME-MOEA/D. El estudio experimental considera conjuntos de datos de referencia y se evalúa el desempeño considerando diferentes niveles de restricciones. Los resultados experimentales dan evidencia que MOECC es capaz de encontrar agrupamientos de manera eficaz y eficiente en comparación con los métodos de referencia.

Abstract

In real-world problems, learning a classification model from data is often desired. Nevertheless, labeled data is not always available. Clustering techniques emerge as an alternative to learn from unlabeled data and have recently been shown to produce better groups when they are provided with partial information of the problem in the form of constraints. This type of clustering is known as semi-supervised clustering with constraints and is a challenging problem. The solution of this problem does not only involves exploring the search space efficiently but the feasibility of the resulting partition must also be ensured. Several methods have been proposed to deal with constrained clustering. However, they generally assume that the number of clusters is known, which is typically unknown in real-world problems. This thesis proposes MOECC: Multi-Objective Evolutionary Constrained Clustering. MOECC formulates the constrained clustering problem as a multi-objective optimization one, where compactness and connectivity of the groups are the objectives to be optimized. Furthermore, it adapts the ϵ-constrained-based mechanism for instance-level constraints handling. MOECC implements a cluster-based crossover and a neighborhood-based mutation to guide the search, and it incorporates a strategy for choosing the number of clusters. The effectiveness of MOECC is compared to COP-K-Means, LCVQE, RDPM, and ME-MOEA/D. The experimental study considered a set of benchmark problems under different levels of constraints. The experimental results provides evidence that MOECC can find a solution effectively and efficiently compared to the reference methods.

Índice general

Datos del jurado I

Dedicatoria II

Agradecimientos III

Resumen IV

Abstract V

Índice general VI

Índice de figuras IX

1. Introducción 1
- 1.1. Justificación . . . 3
- 1.2. Hipótesis de trabajo . . . 4
- 1.3. Objetivos . . . 4
- 1.4. Contribuciones . . . 4
- 1.5. Estructura del documento . . . 5

2. Fundamento teórico 6
- 2.1. Problemas de optimización . . . 6
  - 2.1.1. Problema de optimización multi-objetivo . . . 7
    - Óptimo de Pareto . . . 7
    - Medida del hiper-volumen . . . 8
- 2.2. Algoritmos evolutivos . . . 9
  - 2.2.1. ¿Qué es y cómo funciona un algoritmo evolutivo? . . . 10
- 2.3. Algoritmos evolutivos multi-objetivo . . . 11
  - 2.3.1. MOEA/D . . . 12
    - Aproximación de Tchebycheff . . . 12
- 2.4. Agrupamiento semi-supervisado con restricciones . . . 14
  - 2.4.1. Aplicaciones . . . 16
    - Agrupamiento en imágenes . . . 17
    - Agrupamiento en análisis de videos . . . 18
    - Datos de GPS . . . 18
- 2.5. Algoritmos de agrupamiento semi-supervisado con restricciones . . . 20
  - 2.5.1. K-Medias con restricciones (COP-K-Medias) . . . 20
  - 2.5.2. *Linear Constrained Vector Quantization Error* (LCVQE) . . . 22

2.5.3. *Relational Dirichlet Process-Means* (RDPM) . . . 23
2.5.4. *Improving Constrained Clustering Via Decomposition-based Multiobjective Optimization with Memetic Elitist* (ME-MOEA/D) 24
2.6. Resumen del capítulo . . . 25

3. Algoritmo evolutivo multi-objetivo para problemas de agrupamiento con restricciones, MOECC 27
3.1. Representación de los individuos . . . 28
3.2. Operadores evolutivos . . . 29
3.2.1. Cruza basada en grupos . . . 29
3.2.2. Mutación basada en vecindario . . . 31
3.3. Funciones objetivo . . . 32
3.3.1. Compacidad . . . 33
3.3.2. Conectividad . . . 34
3.4. Regla de factibilidad por ϵ-restringido . . . 34
3.5. Criterio de paro . . . 36
3.6. Selección y mejora del agrupamiento . . . 37

4. Diseño experimental y análisis de resultados 39
4.1. Diseño experimental . . . 39
4.1.1. Aproximación *bootstrap* para la búsqueda del tamaño de grupos K. . . . 39
4.1.2. COP-K-Medias . . . 40
4.1.3. LCVQE . . . 41
4.1.4. RDPM . . . 41
4.1.5. ME-MOEA/D . . . 42
4.1.6. MOECC . . . 43
4.1.7. Conjuntos de datos . . . 44
4.1.8. Generación de restricciones . . . 45
4.2. Análisis de resultados . . . 46
4.2.1. Desempeño general . . . 47
4.2.2. Desempeño por nivel de restricciones . . . 48
4.2.3. Desempeño por conjunto de datos . . . 52
4.2.4. Tamaño de grupos generados . . . 56
4.2.5. Proporción de restricciones violadas . . . 59
4.2.6. Tiempos de ejecución . . . 62

5. Conclusiones y trabajo futuro 64

Referencias 67

A. Mecanismo MOECC 71

B. Algoritmo *Constrained Vector Quantization Error (CVQE)* 74

C. Métricas de evaluación 76
C.1. Métricas de validación interna . . . 76
C.1.1. Índice Hartigan . . . 76
C.2. Métricas de validación externa . . . 77

C.2.1. ARI *(Adjusted Rand Index)* 77
C.2.2. AMI *(Adjusted Mutual Information)* 77
C.2.3. Homegeneidad 78
C.2.4. Medida de completitud 78
C.2.5. Medida V 79

Índice de figuras

2.1. Ejemplo de dominancia de Pareto. Imagen adaptada (Zhang, 2021). . 8
2.2. Ejemplo de frontera de Pareto. Imagen adaptada (Zhang, 2021). . . . 8
2.3. Hiper-volumen para el caso de dos funciones objetivo. Imagen adaptada (Pétrowski y Ben-Hamida, 2017). 9
2.4. Flujo de un algoritmo evolutivo. Imagen adaptada (Pétrowski y Ben-Hamida, 2017). 10
2.5. Ejemplo de MOEA/D con un tamaño de población igual a ocho. Imagen adaptada (Zhou *et al.*, 2012). 13
2.6. Problema de un agrupamiento con tres restricciones. Imagen adaptada (Davidson y Basu, 2007). 17
2.7. Muestra del agrupamiento de imágenes del conjunto de datos *CMU*. Imagen de Davidson y Basu (2007). 17
2.8. Ejemplo de restricciones *cannot-link*. Imagen de Davidson y Basu (2007). 18
2.9. Ejemplos de varios tipos de restricciones en vídeos. Imagen de Yan *et al.* (2006). 19

3.1. Ejemplo de representación basada en etiquetas con diez instancias y cuatro grupos. 29
3.2. Ejemplo de cruza basada en grupos con diez instancias. La figura **a)** muestra el individuo resultante p^{new} que hereda un grupo sin cambios de p^a y el resto de los grupos son heredados de p^b, ignorando los previamente agrupados. La figura **b)** muestra el individuo resultante p^{new}, que hereda un grupo sin cambios de p^a y el resto de los grupos forman un solo conjunto. 31
3.3. Ejemplo de mutación basada en vecindarios con diez instancias. Muestra el individuo p^{new}, resultado de la mutación de p^c. Las instancias 1, 4, 5 y 10 satisfacen la probabilidad de mutación y cada una se coloca en el mismo grupo que uno de sus vecinos más cercanos, el cual se selecciona aleatoriamente (4, 7, 2 y 5, respectivamente). 33
3.4. Ejemplo de conectividad en un conjunto de datos. Imagen adaptada de (Handl *et al.*, 2005). 35
3.5. Ejemplo de selección de la solución final. Las evaluaciones resultantes en la frontera de Pareto son ordenadas de manera ascendente con respecto al número de grupos de sus correspondientes soluciones en la frontera de Pareto. Al obtener la razón entre la compacidad y la conectividad, la máxima diferencia entre niveles jerárquicos se alcanza en el índice 2, por lo que su solución asociada representa el agrupamiento final, el cual contiene 2 grupos. 38

4.1. Desempeño general de los métodos, mediante las métricas ARI, AMI, homogeneidad, completitud y medida V. 47
4.2. ARI por nivel de restricciones y por modelo. 50
4.3. AMI por nivel de restricciones y por modelo. 51
4.4. Homogeneidad por nivel de restricciones y por modelo. 51
4.5. Completitud por nivel de restricciones y por modelo. 52
4.6. Medida V por nivel de restricciones y por modelo. 53
4.7. Proporción de restricciones violadas por escenario de restricciones. . . 60

Capítulo 1

Introducción

El aprendizaje máquina, también llamado aprendizaje automático, es una rama de la inteligencia artificial que en los últimos años ha ganado interés por su aplicación en la resolución de problemas del mundo real. Por ejemplo, en el área de la biología, se ha usado para la segmentación de imágenes de células (Chen *et al.*, 2006; Kan, 2017); en el área de la agricultura, ha permitido incrementar la productividad de las compañías al generar soluciones en las que se pueden aprovechar de una mejor manera los recursos naturales (Balducci *et al.*, 2018); en el área médica, el aprendizaje máquina posibilita soluciones a diversos problemas de diagnósticos o pronóstico de enfermedades, generando oportunidades para facilitar y mejorar el trabajo de los médicos, dando como resultado una mayor eficiencia y calidad de los servicios médicos (de Siqueira *et al.*, 2021; Arias *et al.*, 2019). Generalmente, existen dos paradigmas de aprendizaje máquina:

- **Aprendizaje supervisado**. Este tipo de aprendizaje se caracteriza por tener datos etiquetados; es decir, se conoce la respuesta asociada a cada instancia. Existen dos tipos principales de aprendizaje supervisado: clasificación y regresión. Los problemas de clasificación se caracterizan porque el valor de salida es un valor categórico; por otra parte, en los problemas de regresión, la salida es un valor continuo.

  La meta del aprendizaje supervisado es aprender una función que asocie una instancia de entrada a un valor de salida. Esta función es posteriormente usada para predecir el valor de salida de nuevas instancias.

- **Aprendizaje no supervisado**. Este tipo de aprendizaje se caracteriza porque los datos no se encuentran etiquetados. Por tanto, el aprendizaje no supervisado busca detectar patrones o estructuras de los datos. Generalmente, se identifican dos tipos de problemas de aprendizaje no supervisado: reducción de dimensiones y agrupamiento (Nasraoui y N'Cir, 2019).

  En la reducción de dimensiones, el objetivo es encontrar una representación compacta de los datos preservando su estructura. En contraste, el agrupamiento busca una forma en cómo dividir los datos, tal que las instancias pertenecientes a un mismo grupo sean lo más similares entre sí y desemejantes con las instancias pertenecientes a otros grupos.

  El aprendizaje no supervisado ha dado lugar al descubrimiento de conocimientos sobre grandes conjuntos de datos de diferentes disciplinas, tales como en

astronomía (Ball y Brunner, 2010), bioinformática (Tang *et al.*, 2019; Min *et al.*, 2017), meteorología (Gaffney *et al.*, 2007), entre otros.

A pesar del éxito de los algoritmos de aprendizaje máquina, su aplicación en problemas del mundo real puede verse limitada, ya que es necesario contar con una cantidad de datos suficientes para el aprendizaje. Además, es deseable que estos estén etiquetados, a fin de que ayuden a construir modelos más precisos y confiables. No obstante, el etiquetado de los datos no es una tarea trivial y depende mucho del conocimiento de un experto. Por ello, en los últimos años ha habido un interés creciente en un enfoque híbrido entre el aprendizaje supervisado y el no supervisado, que permita aprender un modelo a partir de la combinación de una porción de datos no etiquetados y una porción pequeña de datos etiquetados. A este enfoque se le conoce como **aprendizaje semi-supervisado**.

El agrupamiento semi-supervisado es un tipo de aprendizaje en el que se tiene acceso a información acerca de los grupos que se buscan. Esta información complementaria puede presentarse a través del conocimiento de las etiquetas de grupos para algunas instancias; de la similitud sobre ciertas instancias, indicando si deben o no estar en un mismo grupo; o de preferencias del usuario sobre cómo se deben agrupar las instancias. Por tanto, los algoritmos de agrupamiento semi-supervisado que logren explotar el uso de información parcial pueden identificar grupos de manera más precisa en comparación con métodos tradicionales de agrupamiento.

Un ejemplo de agrupamiento semi-supervisado es dado por Cohn *et al.* (2003) con el problema de *Yahoo*, descrito del modo siguiente:

> *"Imagine que se cuenta con 100,000 documentos de textos (artículos científicos, artículos de noticias, páginas web, etc.) y se desea agruparlos dentro de clases o jerarquías tal que los textos que están relacionados sean agrupados en la misma clase. De antemano no se sabe qué clases o jerarquías usar o qué documentos están relacionados; sin embargo, sí se tiene un criterio en mente de cómo son los elementos de cada grupo. La taxonomía que se quiere crear pretende que los documentos puedan ser encontrados y accedidos de manera eficiente, ya sea por uno mismo o por otra persona".*

A partir de este problema, Cohn *et al.* (2003) proponen una solución interactiva para mejorar la calidad de los grupos obtenidos. La metodología propuesta consiste en los siguientes pasos:

1. Emplear un algoritmo no supervisado para identificar los grupos sobre el conjunto de documentos.

2. Analizar un subconjunto de grupos resultantes y proveer retroalimentación sobre si un documento debe estar en otro grupo o si dos documentos deben estar en un mismo grupo.

3. Agrupar nuevamente permitiendo que el algoritmo trabaje con la información dada en la retroalimentación

4. Repetir iterativamente hasta que se encuentre un agrupamiento satisfactorio.

A partir del problema de *Yahoo* y la solución propuesta, puede observarse la importancia del agrupamiento semi-supervisado al permitir identificar grupos de mejor calidad a partir de la información parcial que se tiene y, al analizar una parte de los datos, el esfuerzo requerido por el experto es menor en constraste con etiquetar el conjunto completo de datos. Además, aunque en el agrupamiento semi-supervisado se asume que el usuario tiene en mente criterios que le permiten evaluar la calidad de los grupos, no se espera que escriba una función que defina dicho criterio por completo, sino que interactúe con el sistema de tal manera que intente aprender un criterio que arroje grupos con los que el usuario se sienta satisfecho.

1.1. Justificación

A pesar del éxito de los algoritmos de aprendizaje supervisado en diferentes problemas, su entrenamiento típicamente requiere de una gran cantidad de datos etiquetados, lo cual limita su aplicación en dominios donde no se cuenta con las etiquetas de clase de los datos completos, siendo necesario su etiquetado de manera manual. No obstante, etiquetar los datos de manera precisa es una tarea desafiante que requiere del conocimiento del experto en el dominio, lo que conlleva tiempo, esfuerzo y un costo proporcional al tamaño de los datos a etiquetar.

Por otro lado, aunque los algoritmos de agrupamiento no supervisado son una alternativa para aprender a partir de datos no etiquetados, los grupos resultantes pueden no ser de utilidad para el usuario. Además, analizar los grupos puede requerir de un experto para comprender los patrones o conceptos correlacionados de los grupos obtenidos.

El agrupamiento semi-supervisado es una alternativa para hacer frente a las problemáticas previamente mencionadas. Primero, no se requiere tener todos los datos etiquetados, sino contar con información sobre un subconjunto de instancias, la cual puede ser a través de las etiquetas de clase o de relaciones entre las instancias. Esto puede traducirse en menor esfuerzo y costo en introducir el conocimiento. Segundo, aun cuando no se conozcan de antemano los grupos finales, una matriz de estas posibles relaciones entre las instancias es más intuitiva y fácil de proveer que etiquetar todos los datos. Finalmente, el agrupamiento semi-supervisado da la facilidad de seleccionar aquellas instancias a retroalimentar, permitiendo una mejora en los grupos y que cumplan con la configuración deseada.

En la literatura existe una gran variedad de trabajos que abordan el problema de agrupamiento semi-supervisado que hacen uso de restricciones a nivel de instancias. Por ejemplo, Yan *et al.* (2006), Wagstaff y Cardie (2000) y Basu *et al.* (2004) dan evidencia que el uso de restricciones ayuda a mejorar el desempeño de los modelos de agrupamiento tradicionales. No obstante los avances en el área de desarrollo de algoritmos para agrupamiento con restricciones (Wagstaff *et al.*, 2001; Pelleg y Baras, 2007) tienen la desventaja que asumen que se conoce el número de grupos, lo cual es una limitante ya que en problemas reales este dato no suele conocerse.

Dentro de los algoritmos de agrupamiento semi-supervisado existe, además, el problema de factibilidad, el cual cuestiona la existencia de alguna solución que satisfaga todas las restricciones. Davidson y Ravi (2007) encontraron que el peor de los

escenarios al tratar de resolver el problema de agrupamiento bajo restricciones, involucra resolver un sub-problema que es ***NP-completo*** (intratable); es decir, requiere un tiempo de ejecución exponencial para resolver el agrupamiento. Un enfoque para hacer frente a esto es a través de métodos heurísticos, los cuales permiten obtener una solución aproximada en un tiempo polinómico. No obstante, estas heurísticas son susceptibles a quedarse atrapadas en óptimos locales. Por consiguiente, en esta tesis se explora el uso de técnicas de búsqueda global a través de los algoritmos evolutivos para abordar el problema de agrupamiento semi-supervisado. Se estudia una formulación multi-objetivo que permite generar soluciones con diferentes números de grupos de manera simultánea.

1.2. Hipótesis de trabajo

La hipótesis de esta tesis es que el problema de agrupamiento con restricciones a nivel de instancias puede ser resuelto con algoritmos evolutivos multi-objetivo y técnicas de manejo de restricciones, permitiendo encontrar grupos de manera eficaz y determinar de forma automática el número de grupos.

1.3. Objetivos

Estudiar y desarrollar un método que, basado en el uso de algoritmos evolutivos multi-objetivo y técnicas de manejo de restricciones, permita resolver el problema del agrupamiento semi-supervisado con restricciones a nivel de instancias.

Para lograr el objetivo general planteado, se proponen los siguientes objetivos específicos:

1. Diseñar e implementar un algoritmo evolutivo multi-objetivo que permita encontrar de manera no supervisada el número de grupos en problemas de agrupamiento.

2. Analizar las técnicas de manejo de restricciones para algoritmos evolutivos y adaptarlas al problema de agrupamiento semi-supervisado.

3. Evaluar y analizar el desempeño del algoritmo evolutivo multi-objetivo y las técnicas de manejo de restricciones en problemas de agrupamiento con restricciones a nivel instancia.

1.4. Contribuciones

Las principales contribuciones del trabajo realizado son las siguientes:

1. Se propone el algoritmo MOECC: *Multi-Objective Evolutionary Constrained Clustering*, como solución al problema de agrupamiento con restricciones a nivel de instancias basado en algoritmos evolutivos multi-objetivo.

2. Se sabe que es la primer propuesta basada en algoritmos evolutivos multi-objetivo que adapta un manejador de restricciones para las restricciones tipo *must-link* y *cannot-link*.

3. Se proponen operadores evolutivos específicos para el problema de agrupamiento.

4. MOECC permite obtener todo un conjunto de soluciones óptimas y además, propone una estrategia para escoger una solución de entre el conjunto de soluciones óptimas.

1.5. Estructura del documento

El documento de tesis se estructura de la siguiente forma:

- El Capítulo 2 presenta los principales fundamentos teóricos relacionados con la optimización multi-objetivo, los algoritmos evolutivos y el agrupamiento semi-supervisado.
- El Capítulo 3 presenta el algoritmo propuesto y la descripción de los elementos que lo componen, su propósito es dar solución al problema de agrupamiento semi-supervisado con restricciones a nivel instancias, mediante un enfoque de algoritmos evolutivos.
- En el Capítulo 4 se presenta el diseño experimental y la evaluación de los resultados obtenidos en la experimentación.
- En el Capítulo 5 se muestran las conclusiones y posibles extensiones a este trabajo.

Capítulo 2

Fundamento teórico

Este capítulo presenta los principales fundamentos teóricos asociados con la optimización multi-objetivo, los algoritmos evolutivos y el agrupamiento semi-supervisado. La Sección 2.1 describe los problemas de optimización de un objetivo y multi-objetivo y los principales conceptos asociados. La Sección 2.2 describe en detalle los algoritmos evolutivos, mientras que la Sección 2.3 describe la extensión de los algoritmos evolutivos para problemas multi-objetivo. Posteriormente, la Sección 2.4 describe el problema del agrupamiento semi-supervisado con restricciones y se listan ejemplos de aplicaciones reales. Finalmente, la Sección 2.5 describe en detalle métodos del estado del arte comúnmente usados para agrupamiento con restricciones a nivel instancia.

2.1. Problemas de optimización

Un problema de optimización tiene como propósito encontrar una solución que minimice o maximice una o varias funciones objetivo. Si el problema tiene una función, se dice que es *un problema de optimización de un objetivo*. De acuerdo con Coello Coello *et al.* (2007), se define como:

Definición: 2.1.1. Sea $f(\mathbf{x})$ una función con valores $\mathbf{x}$ en el espacio de búsqueda Ω. Un problema de optimización de un objetivo busca una solución $\mathbf{x} \in \Omega$ que minimice[1] la función $f(\mathbf{x})$ sujeta a un conjunto de restricciones, es decir:

$$\begin{aligned} &\arg\min_{\mathbf{x}} \ f(\mathbf{x}) \\ &\text{sujeto a: } \mathbf{x} \in \Omega \end{aligned} \tag{2.1}$$

donde $f : \Omega \to \mathbb{R}$ es un escalar, $\mathbf{x} = (x_1, \ldots, x_n)^T$ es un vector $n-$dimensional de variables de decisión y Ω es el espacio de soluciones factibles.

La solución a un problema de optimización de un objetivo puede ser única, sin embargo, también existe la posibilidad de encontrar más de una solución.

[1] Sin perdida de generalidad se asume minimizar. Un problema de maximización se puede convertir a uno de minimización usando el principio de dualidad, esto es, multiplicando la función objetivo por menos uno.

2.1.1. Problema de optimización multi-objetivo

Este problema de optimización consta de dos o más funciones objetivo. Formalmente, un problema de optimización multi-objetivo se define como (Coello Coello *et al.*, 2007; Zhang y Li, 2007):

Definición: 2.1.2. Sea $F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^T$ un vector con m funciones objetivo, se establece como:

$$\begin{aligned} &\arg\min_{\mathbf{x}} \; F(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^T \\ &\text{sujeto a: } \mathbf{x} \in \Omega \end{aligned} \tag{2.2}$$

donde $F : \Omega \to \mathbb{R}^m$ es un vector m-dimensional. El conjunto de objetivos alcanzable se define como el conjunto $\{F(x) \mid x \in \Omega\}$.

En general, las funciones objetivo en un problema de optimización multi-objetivo están en conflicto, por lo que no existe una solución $\mathbf{x} \in \Omega$ que minimice todas las funciones objetivo simultáneamente. El criterio comúnmente utilizado en optimización multi-objetivo para decidir si una solución es mejor que otra es mediante el óptimo de Pareto (Coello Coello *et al.*, 2007).

Óptimo de Pareto

La mayoría de los métodos modernos de optimización multi-objetivo usan el concepto de la dominancia de Pareto para determinar si una solución es mejor que otra. Más formalmente, la dominancia de Pareto se define como (Coello Coello *et al.*, 2007; Zhang y Li, 2007):

Definición: 2.1.3 (Dominancia de Pareto). Sean $a, b \in \mathcal{R}^m$ dos soluciones, se dice que a domina a b, denotado como $a \preceq b$, si a no es peor que b en ningún objetivo y existe al menos un objetivo en a que es mejor que b. Formalmente:

$$a \preceq b \iff \forall i \in \{1, \ldots, m\} : a_i \leq b_i \wedge \exists i \in \{1, \ldots, m\} : a_i < b_i. \tag{2.3}$$

La Figura 2.1 muestra un ejemplo en el cual $F(A)$ domina a $F(B)$, pues cumple con ambos puntos de la definición de dominancia. Además, observe que $F(A)$ y $F(C)$ no son comparables.

Un punto $\mathbf{x}^* \in \Omega$ es *óptimo de Pareto* del problema de la Ecuación 2.2 si no existe otro punto $\mathbf{x} \in \Omega$ tal que $F(\mathbf{x}) \preceq F(\mathbf{x}^*)$. $F(\mathbf{x}^*)$ se conoce como *vector objetivo óptimo de Pareto*. Cualquier mejora en un punto óptimo de Pareto en algún objetivo conduce al deterioro en al menos otro objetivo. Al conjunto de todos los puntos óptimos de Pareto se le conoce como *conjunto de Pareto (CP)* y al conjunto de todos los vectores objetivo óptimos de Pareto se le conoce como *frontera de Pareto (FP)*. La Figura 2.2 muestra estos elementos.

Una vez que se tienen las soluciones en la frontera de Pareto, es importante definir algún indicador que permita determinar cuando estos valores son mejores o peores con respecto a otras soluciones. Un indicador de calidad que cumple con este propósito es la medida del hiper-volumen.

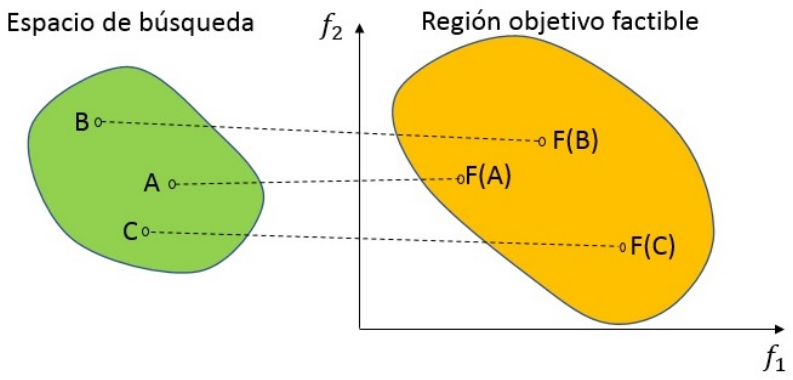


Figura 2.1: Ejemplo de dominancia de Pareto. Imagen adaptada (Zhang, 2021).

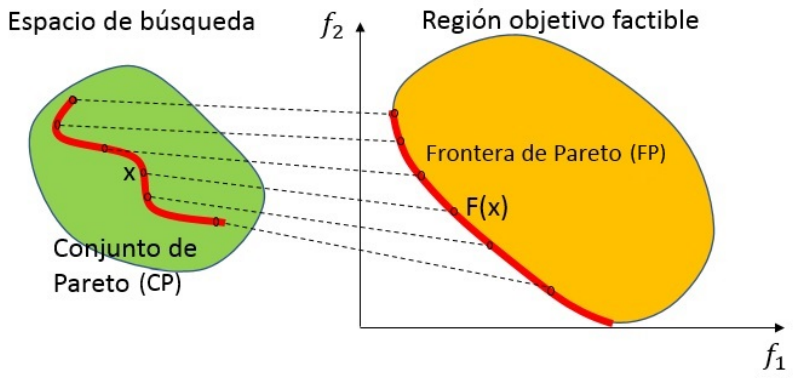


Figura 2.2: Ejemplo de frontera de Pareto. Imagen adaptada (Zhang, 2021).

Medida del hiper-volumen

De acuerdo con Pétrowski y Ben-Hamida (2017), el hiper-volumen se define de la siguiente manera:

Definición: 2.1.4. Sea $\rho = (\rho_1, \ldots, \rho_m)$ un punto de referencia en la región objetivo factible, y sea $a_i = (a_i^1, \ldots, a_i^m)$ un vector objetivo no dominado de la frontera de Pareto, FP, $i = 1, ..., |FP|$. Para el caso de funciones objetivo a minimizar, se debe cumplir que $a_i^j \leq \rho_j$ para cada $j = 1, ..., m$, es decir, cada una de las coordenadas de ρ es una cota superior de las coordenadas de todos los vectores contenidos en FP. Los vectores ρ y a_i definen el hiper-rectángulo cuyos bordes son paralelos a los ejes coordenados de la región objetivo. La expresión del hiper-volumen dados estos dos vectores es la siguiente:

$$\nu(a_i, \rho) = \prod_{j=1}^{m} (\rho_j - a_i^j), \text{ para cada } i = 1, .., |FP|. \tag{2.4}$$

Así que, dado el conjunto FP y el punto de referencia ρ, se define el **hiper-volumen** $\nu(FP, \rho)$ en la región objetivo por la unión de los hiper-rectángulos asociados con los elementos de FP, es decir:

$$\nu(FP, \rho) = \cup_{i=1}^{|FP|} \nu(a_i, \rho). \tag{2.5}$$

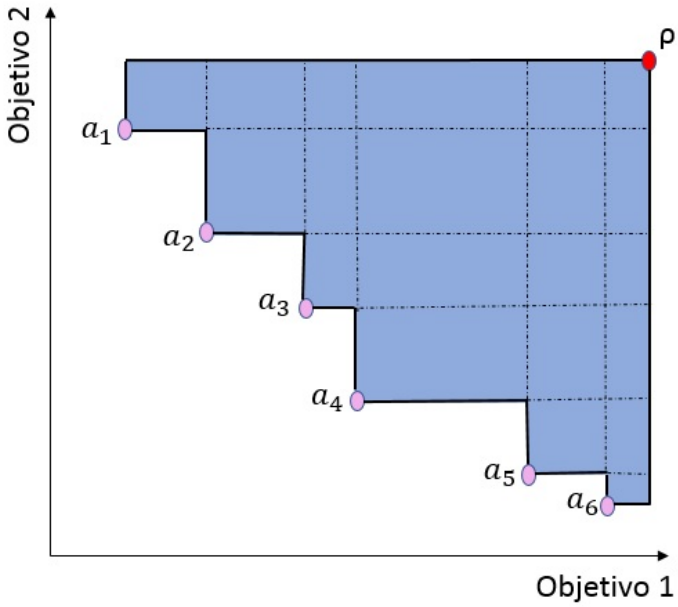


Figura 2.3: Hiper-volumen para el caso de dos funciones objetivo. Imagen adaptada (Pétrowski y Ben-Hamida, 2017).

La Figura 2.3 muestra un ejemplo de la medida de hiper-volumen en un problema con dos funciones objetivo. La medida del hiper-volumen $\nu(FP, \rho)$ es una métrica muy útil para comparar soluciones de conjuntos no dominados, esto debido a que dada la naturaleza de la dominancia de Pareto, su comportamiento será monótono. Por ejemplo, si se tienen dos conjuntos de soluciones no dominadas A y B, pero alguna de las soluciones del conjunto B es dominada por al menos alguna solución del conjunto A, entonces $\nu(A, \rho) > \nu(B, \rho)$ y con ello se concluiría que las soluciones en el conjunto A son mejores que las del conjunto B.

2.2. Algoritmos evolutivos

En esta sección se describen los elementos que constituyen a un algoritmo evolutivo. Su importancia radica en que existen problemas complejos para los cuales no es posible encontrar una solución haciendo uso de métodos exactos, en estos casos, el uso de algoritmos evolutivos permite encontrar soluciones de buena calidad en un tiempo de cómputo razonable, además de realizar una búsqueda global.

En 1859, Darwin publica su libro *El origen de las especies* (Darwin's, 1859), en el cual establece que en la evolución de los seres vivos ocurre un conjunto de variaciones en las características individuales entre los padres y los descendientes, muchas de las cuales son heredadas. Establece también que existe un tipo de competición entre los individuos, a partir de la cual se seleccionan aquellos que son más aptos con su entorno, siendo estos los que sobreviven y se reproducen. Esta competición es la que da pauta a la transmisión de variaciones genéticas hereditarias que benefician a los individuos, las cuales se van transmitiendo de generación en generación.

En la década de los sesenta surgen las primeras ideas del modelizado de procesos

evolutivos basados en la teoría de Darwin, habiendo principalmente tres aproximaciones que replican este mecanismo: las estrategias evolutivas, la programación evolutiva y los algoritmos genéticos. Muchas otras modificaciones han surgido a partir de esas tres aproximaciones, es por ello que en 1993, aparece el término de *computación evolutiva*, la cual se le designa principalmente a los métodos cuyo sustento es la teoría de la evolución biológica (Pétrowski y Ben-Hamida, 2017).

Una de las ventajas de los algoritmos evolutivos es que son universalmente aplicables, ya que todos los pasos involucrados en su desarrollo funcionan de manera independiente al problema que se desee resolver, salvo en la definición de la función de evaluación y en la forma en que se codifican las soluciones. A continuación se describe de manera genérica lo que es un algoritmo evolutivo y los elementos que lo componen.

2.2.1. ¿Qué es y cómo funciona un algoritmo evolutivo?

Un algoritmo evolutivo es un procedimiento iterativo que usualmente ocurre sobre una población de tamaño constante, la cual evoluciona a través de varias generaciones. De acuerdo con Affenzeller *et al.* (2009) y Pétrowski y Ben-Hamida (2017), la idea general en que ocurre este procedimiento se puede apreciar en la Figura 2.4 y se describe a continuación:

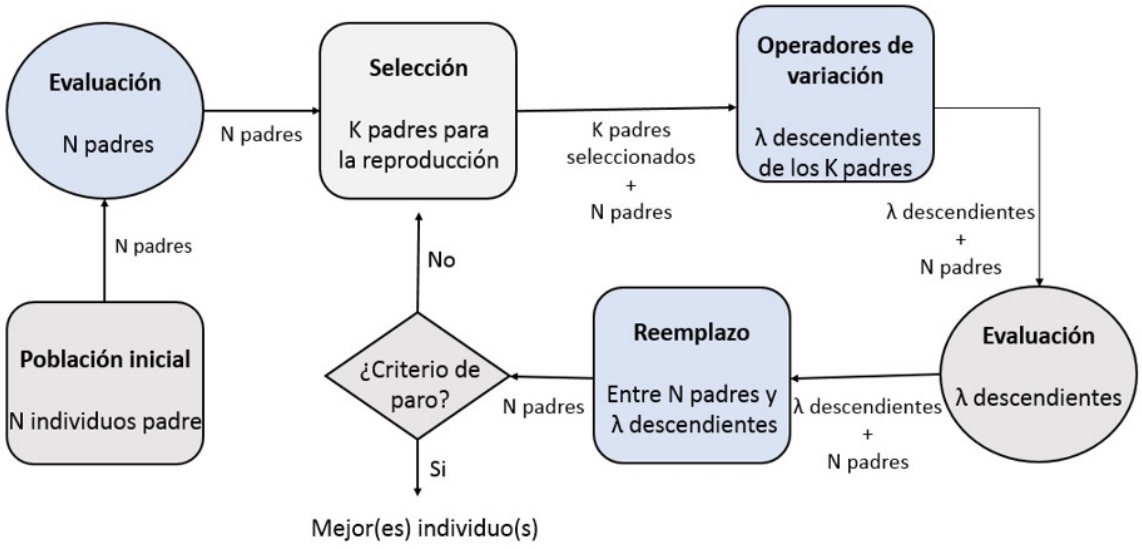


Figura 2.4: Flujo de un algoritmo evolutivo. Imagen adaptada (Pétrowski y Ben-Hamida, 2017).

1. Partir de una población inicial de individuos, que de manera usual se genera aleatoriamente.

2. A cada iteración del algoritmo se le conoce como *generación*. En cada generación, los individuos de la población actual son evaluados, asignándoles como resultado un **valor de aptitud**. Esta evaluación se realiza mediante una **función de aptitud**. En el contexto de optimización, esta función de aptitud juega el rol de la función objetivo.

3. Con la finalidad de formar una nueva población, se aplica el operador de selección sobre la población actual. Usualmente, esta selección es de acuerdo con una probabilidad proporcional a sus valores de aptitud relativos. Este paso tiene sentido pues cada individuo de la población contiene información necesaria para representar a una solución que tiene características deseables para una mejor evaluación, y se espera que los individuos mejor adaptados (aquellos con mayor valor de aptitud), sean los que se reproduzcan con el objetivo de que hereden las características antes mencionadas.

4. A partir de estos individuos, se producen los descendientes de esta generación mediante operadores evolutivos. Existen dos categorías de operadores:

   - **Cruza:** Juega el papel de operador genético primario y se involucra en el momento de la reproducción de los padres. Dados los padres (mínimo dos de ellos), se mezclan sus códigos genéticos para generar hijos que posean un código híbrido. Se debe establecer como parámetro una *tasa de cruza*, la cual determina la proporción de individuos que se cruzan dentro de la población.
   - **Mutación:** Es esencialmente una modificación arbitraria sobre un individuo, la cual ayuda a prevenir convergencias prematuras al muestrear de manera aleatoria nuevos puntos en el espacio de búsqueda. La alteración causada al individuo por una mutación es independiente al resto de los individuos. Debe establecerse la *tasa de mutación*, la cual determina la proporción de descendientes que sufren una mutación. Usualmente esta tasa suele ser baja comparada con la tasa de cruza.

5. Se evalúan los individuos descendientes con la misma función de aptitud empleada para evaluar a los padres.

6. Mediante un operador de reemplazo se escoge los individuos que sobreviven a la siguiente generación de entre los padres y los descendientes, de tal manera que al final de cada generación, algunos descendientes reemplazan a los padres y pasan a formar parte de la próxima generación de padres.

7. Este proceso se repite hasta que ocurra el criterio de paro, el cual se debe definir *a priori*, en caso contrario, comienza una nueva generación.

Durante la descripción de las etapas que se involucran en el algoritmo evolutivo, se ha hecho mención de la etapa de **evaluación de la población**, la cual asigna un valor de aptitud a cada individuo dentro de una generación. En esta evaluación, puede existir la necesidad de que cada solución satisfaga varias funciones objetivo de manera simultánea, lo cual nos lleva al caso de la optimización multi-objetivo. De esta manera, se tiene una extensión del algoritmo evolutivo simple (de un objetivo), al algoritmo evolutivo de optimización multi-objetivo (MOEA, por sus siglas inglés).

2.3. Algoritmos evolutivos multi-objetivo

En un problema de optimización multi-objetivo, el concepto de dominancia de Pareto (véase Sección 2.1.1) se emplea para la selección de las mejores soluciones al

problema en el proceso de búsqueda. Al término se obtiene un conjunto de soluciones óptimas de Pareto, que al ser evaluadas en las funciones objetivo conforman la frontera de Pareto. Existe una gran variedad de algoritmos evolutivos que dan solución a problemas de optimización multi-objetivo. Uno de estos es el propuesto por Zhang y Li (2007), denominado MOEA/D y el cual se describe a continuación.

2.3.1. MOEA/D

MOEA/D (*Multiobjective optimization evolutionary algorithm based on decomposition*) es un algoritmo propuesto por Zhang y Li (2007) para resolver problemas de optimización multi-objetivo. La idea consiste en descomponer el problema de optimización multi-objetivo en N subproblemas de optimización escalar (de un solo objetivo), los cuales son resueltos de manera simultánea. MOEA/D evoluciona una población de soluciones con el objetivo de encontrar la mejor aproximación de la frontera de Pareto en cualquier problema de optimización multi-objetivo.

En cada generación, la población se compone por las mejores soluciones encontradas hasta ese momento en cada subproblema. Cada subproblema i tiene un vector de pesos diferente $\lambda^i,\ i = 1, \ldots, N$ para cada una de las m funciones objetivo, es decir, cada $\lambda^i = (\lambda^i_1, \ldots, \lambda^i_m)^T$, con $\lambda^i_j \geq 0, \forall j = 1, \ldots, m$, además se debe de cumplir que, $\sum_{j=1}^{m} \lambda^i_j = 1, \forall i = 1, \ldots, N$. Estos vectores de pesos $\lambda^1, \ldots, \lambda^N$, están uniformemente distribuidos.

Cada subproblema i tiene asignado un conjunto $\beta(i) = \{i_1, \ldots, i_\tau\}$, el cual contiene los τ índices de los vectores de pesos más cercanos a su vector de pesos λ^i. Para obtener las $\beta(i)'s$, se calcula la distancia euclídea entre cada par de vectores de pesos. MOEA/D parte de la premisa que la solución óptima de dos vecinos cercanos debería de ser similar; por tanto, cada subproblema usa información proveniente de los subproblemas que son sus vecinos más cercanos.

Existen diversas maneras de llevar a cabo la descomposición de un problema de optimización multi-objetivo, como lo son la aproximación por sumas ponderadas, la aproximación de Tchebycheff o la aproximación por intersección de límites. En el presente trabajo, se hará uso de la aproximación de Tchebycheff (Zhang y Li, 2007).

Aproximación de Tchebycheff

Definición: 2.3.1. Sea $\mathbf{z}^* = (z^*_1, \ldots, z^*_m)$ un punto de referencia. La máxima separación entre las funciones objetivo $f_i(\mathbf{x})$, $i = 1, \ldots, m$, y el punto de referencia $\mathbf{z}^*$, multiplicada por el vector de pesos λ^j_i, se denota como $g^{te}(\mathbf{x} \mid \lambda^j, z^*)$, para cada $j = 1, \ldots, N$, y se define de la siguiente manera:

$$g^{te}(\mathbf{x} \mid \lambda^j, z^*) = \max_{1 \leq i \leq m} \{\lambda^j_i \mid f_i(\mathbf{x}) - z^*_i \mid\}. \tag{2.6}$$

En el caso de un problema de minimizar, $\mathbf{z}^*_i = \min\{f_i(\mathbf{x}) : \mathbf{x} \in \Omega\}$, para cada $i = 1, \ldots, m$. **La aproximación de Tchebycheff** busca minimizar la máxima separación.

Para cada punto óptimo de Pareto, $\mathbf{x}^*$, existe un vector de pesos λ^j tal que $\mathbf{x}^*$ es solución óptima de la Ecuación 2.6 y cada solución óptima de la Ecuación 2.6 es una solución óptima (Pareto) para el problema multi-objetivo del que provienen.

El algoritmo MOEA/D resuelve los N subproblemas en una corrida, es decir, minimiza las N funciones objetivo simultáneamente. En la Figura 2.5 se muestra un ejemplo que permite apreciar los elementos que constituyen a un MOEA/D con ocho subproblemas.

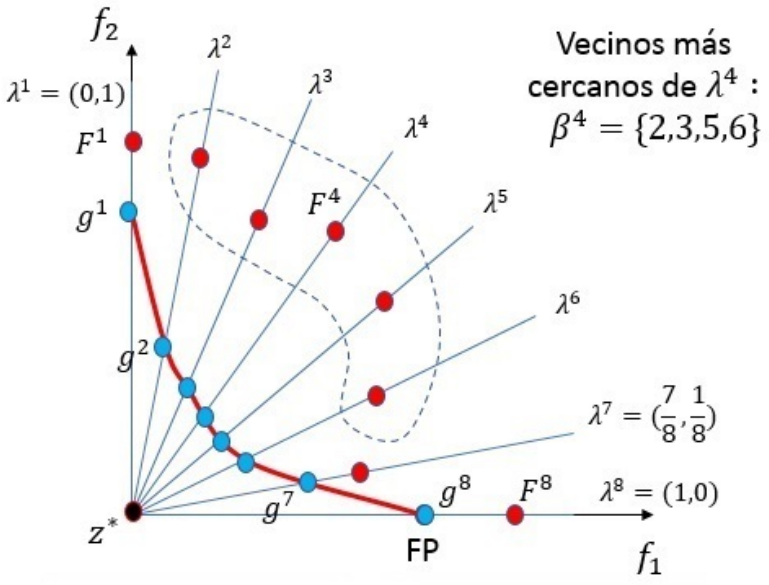


Figura 2.5: Ejemplo de MOEA/D con un tamaño de población igual a ocho. Imagen adaptada (Zhou *et al.*, 2012).

El Algoritmo 1 describe de manera general a MOEA/D. La manera como funciona es la siguiente: se inicializa el conjunto EP vacío (el cual guardará las soluciones y los valores tanto del conjunto de Pareto, como de la frontera de Pareto); se generan de manera aleatoria a los N individuos de la población inicial y se evalúan en las funciones objetivo que se hayan definido de acuerdo con el problema a resolver, finalmente, se inicializa el punto de referencia $\mathbf{z}^*$, el cual guardará los valores mínimos o máximos (para el caso de una función de minimizar o maximizar, respectivamente) de cada una de las funciones objetivo que se han evaluado previamente en los N individuos de la población. El algoritmo se ejecuta mientras que no se alcance el criterio de paro definido y cada iteración corresponde a una generación. En cada generación, por cada uno de los individuos de la población x^i, $i = 1, ..., N$, se **produce** un descendiente y a partir de la selección aleatoria de dos de sus vecinos con pesos más cercanos. Al descendiente y se le puede aplicar una **mejora** que produzca un nuevo descendiente, el cual reemplaza a y. Después, se realiza la **actualización del punto de referencia** $\mathbf{z}^*$. Posteriormente, se actualizan las soluciones colindantes en $\beta(i)$ haciendo uso de la aproximación de Tchebycheff. Finalmente, se actualiza EP, que guarda las soluciones y los valores contenidos en el conjunto de Pareto y la frontera de Pareto, respectivamente. Si se alcanza el criterio de paro, entonces se termina la ejecución del algoritmo y se devuelve EP.

Para el caso de este algoritmo evolutivo multi-objetivo, un posible criterio de paro puede ser que, entre dos generaciones consecutivas, la variación en las métricas del hipervolumen (Sección 2.1.1) sea menor a algún valor $\epsilon > 0$.

Algoritmo 1 MOEA/D (Zhang y Li, 2007)

Entrada: *Criterio de paro*; N: Número de subproblemas en MOEA/D; $\lambda^1, \ldots, \lambda^N$: Vectores de pesos; τ: Número de vectores de pesos colindantes.
Salida: *Población externa, EP.*

```
Función MOEAD(Criterio de paro, N, λ^1, ..., λ^N, τ)
    EP = ∅.
    Inicializar la población x^1, ..., x^N.
    Para cada i = 1, ..., N hacer:
        Encontrar β(i).
        Evaluar la población FV^i = F(x^i).
    Fin Para
    Inicializar z* = (z*_1, ..., z*_m)^T.
    Repetir:
        Para i = 1 hasta N hacer:
            Seleccionar k y l de β(i).
            Generar y a partir de x^k y x^l haciendo uso de operadores de variación.
            Mejorar la solución y para producir y'.
            Para j = 1, ..., m hacer:
                Si z*_j < f_j(y') entonces:
                    z*_j = f_j(y').
                Fin Si
            Fin Para
            Para cada j ∈ β(i) hacer:
                Si g^te(y' | λ^j, z*) < g^te(x^j | λ^j, z*) entonces:
                    x^j = y'; FV^j = F(y').
                Fin Si
            Fin Para
            Eliminar todos los vectores dominados por F(y') de EP.
            Agregar F(y') a EP si no hay algún vector en EP que lo domine.
        Fin Para
    Hasta que: Se alcance criterio de paro.
    Regresa EP.
Fin Función
```

2.4. Agrupamiento semi-supervisado con restricciones

De acuerdo con González Almagro *et al.* (2020), un problema de agrupamiento tradicional se puede definir de la siguiente manera:

Definición: 2.4.1 (Problema de agrupamiento). Sea $X = \{x_1, x_2, \ldots, x_n\}$ un conjunto de datos compuesto de n instancias, donde cada instancia x_i es descrita por u características, tales que $x_i = (x_{[i,1]}, \ldots, x_{[i,u]})$, $\forall i \in \{1, \ldots, n\}$. El problema de agrupamiento consiste en particionar las instancias del conjunto de datos en un número de grupos fijo k, asignando una etiqueta de clase l_i a cada instancia $x_i \in X$; es decir, se obtiene un conjunto de etiquetas $L = \{l_1, \ldots, l_n\}$, con $l_i \in \{1, 2, \ldots, k\}$. Como resultado, se obtiene una partición C de X en k grupos disjuntos denotados como c_i, tales que $C = \{c_1, \ldots, c_k\}$ con $\cap_{i=1}^{k} c_i = \emptyset$.

A diferencia del agrupamiento tradicional, el agrupamiento semi-supervisado se caracteriza en que el usuario tiene algún conocimiento previo acerca de la partición deseada C. Este conocimiento previo se puede incorporar a través de restricciones. Entre los tipos de agrupamiento con restricciones, se encuentran los siguientes cinco y se diferencian entre sí de acuerdo al nivel en que ocurren las restricciones:

- *Agrupamiento con restricciones a nivel de instancias.*

  De acuerdo con Davidson y Basu (2007), las restricciones se presentan por parejas de instancias y especifican si dos instancias deben de pertenecer al mismo grupo (*must-link*) o en diferentes grupos (*cannot-link*). Se busca que la solución satisfaga el número máximo de restricciones, para ello el algoritmo de agrupamiento se modifica haciendo uso de las restricciones disponibles para sesgar la búsqueda y obtener un agrupamiento apropiado de los datos. Existen dos modelos de métodos basados en restricciones a nivel de instancias:

  1. Aquellos que fuerzan el cumplimiento de las restricciones e intentan encontrar la mejor asignación posible que no infrinja alguna de ellas (Davidson y Ravi, 2005a; Wagstaff *et al.*, 2001).
  2. Aquellos que hacen una interpretación relajada de las restricciones, permitiéndose incumplir algunas de ellas para optimizar la función objetivo (Basu *et al.*, 2004; Davidson y Ravi, 2005b), sin embargo, en este caso surge un compromiso entre el número de restricciones incumplidas y el valor de la función objetivo.

- *Agrupamiento con restricciones basado en distancias.*

  De acuerdo con Davidson y Basu (2007), se entrena la función de distancia (para el caso de los algoritmos de agrupamiento que emplean esta función) con base en las restricciones, manteniendo cercanas en el plano a aquellas instancias que tengan una restricción *must-link* y separadas a aquellas instancias con una restricción *cannot-link*.

- *Agrupamiento con restricciones a nivel características.*

  Sun *et al.* (2010) proponen un algoritmo con restricciones basado en el orden de preferencia de las características de los datos. Se usan restricciones a nivel de atributo como parte de la función objetivo a optimizar, ya que asumen que una característica s es más importante que una característica t. Por otro lado, Seret *et al.* (2014) también realizan una propuesta de modelo de agrupamiento con restricciones a nivel de características en las que reflejan la importancia de las variables de acuerdo a la percepción del analista, introduciendo un sesgo que guía el algoritmo de agrupamiento.

- *Agrupamiento con restricciones a nivel modelo.*

  Zhong y Ghosh (2003) presentan un marco de referencia unificado para un agrupamiento basado en modelos probabilísticos. Se intenta aprender modelos generativos a partir de los datos y cada modelo representa un grupo particular. La relación entre los datos y los grupos se da a través de una gráfica bipartita formada por las conexiones entre los datos y el espacio de modelos. En este tipo de aproximación, se especifica *a priori* el tipo de modelo a usar.

- *Agrupamiento con restricciones a nivel grupo.*

  Bradley *et al.* (2000) proponen una variante del algoritmo K-Medias agregando restricciones a nivel de grupo, donde se asegura que no haya grupos vacíos o grupos con pocos elementos. Se agregan K restricciones al problema de optimización del agrupamiento, estableciendo que cada grupo contenga al menos un número mínimo de elementos.

Este trabajo se enfoca en el agrupamiento con restricciones a nivel de instancias y de manera formal, Wagstaff y Cardie (2000) definen las restricciones de la siguiente manera:

Definición: 2.4.2 (**Restricción *Must-link*, ML**(x_i, x_j))**.** Sean x_i, $x_j \in X$. Una restricción *must-link* establece que las instancias x_i y x_j se deben asignar en el mismo grupo.

Definición: 2.4.3 (**Restricción *Cannot-link*, CL**(x_i, x_j))**.** Sean x_i, $x_j \in X$. Una restricción *cannot-link* establece que las instancias x_i y x_j no deben ser asignadas en el mismo grupo.

A partir del conjunto de restricciones *must-link* y *cannot-link*, el problema de agrupamiento semi-supervisado puede ser definido como:

Definición: 2.4.4 (**Problema de agrupamiento con restricciones**)**.** Sea X un conjunto de datos compuesto de n instancias, donde cada instancia x_i, $\forall i \in \{1, \ldots, n\}$, es descrita por u características. El problema de agrupamiento semi-supervisado, consiste en particionar las instancias del conjunto de datos dentro de un número de grupos fijo k, que idealmente satisfaga todas las restricciones del conjunto de restricciones $R = ML \cup CL$, asignando una etiqueta de clase l_i a cada instancia $x_i \in X$, $l_i \in \{1, 2, \ldots, k\}$, obteniéndose el conjunto $L = \{l_1, \ldots, l_n\}$. Como resultado, se obtiene una partición C de X en k grupos disjuntos denotados como c_i, tales que $C = \{c_1, \ldots, c_k\}$ con $\cap_{i=1}^{k} c_i = \emptyset$ y $n = |X| = \sum_{i=1}^{k} |c_i|$.

En el agrupamiento semi-supervisado, sólo se dispone de un subconjunto de instancias etiquetadas, las cuales se pueden ver en forma de restricciones *must-link* y *cannot-link*. La incorporación de restricciones tiene la ventaja de que no se requiere que el usuario que aporta estas restricciones sea un experto en el tema. La Figura 2.6, muestra un ejemplo de varias instancias que se desea agrupar, así como algunas restricciones *must-link* y *cannot-link* que se deben satisfacer. La figura del lado izquierdo muestra una solución que no es factible, ya que los grupos formados violan dos restricciones (una *must-link* y una *cannot-link*), mientras que la figura del lado derecho corresponde a una solución de agrupamiento factible, ya que se cumplen todas las restricciones.

2.4.1. Aplicaciones

Se muestran algunos ejemplos reales abordados en la literatura donde se hace uso de restricciones en el proceso de agrupamiento. Se pretende evidenciar la importancia de este tipo de agrupamiento y la gran variedad de usos que tiene. Para cada ejemplo se muestra el proceso mediante el cual se generan las restricciones.

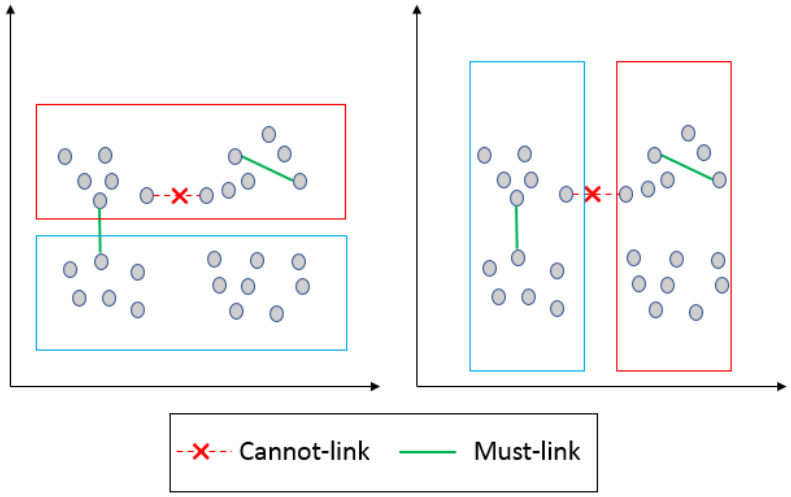


Figura 2.6: Problema de un agrupamiento con tres restricciones. Imagen adaptada (Davidson y Basu, 2007).

Agrupamiento en imágenes

Este ejemplo se obtiene de Davidson y Basu (2007). Se desea agrupar un conjunto de imágenes, tomando en consideración algunas de las características contenidas en ellas. Por ejemplo, en la Figura 2.7, se presenta una muestra de imágenes de rostros del conjunto de datos *CMU (Cargenie Mellon University)* y la tarea es agruparlas según su orientación, ya sea que la imagen muestre el rostro de perfil, de frente, mirando hacia arriba, etc. La orientación representa la etiqueta de clase a la que pertenece la imagen.

Figura 2.7: Muestra del agrupamiento de imágenes del conjunto de datos *CMU*. Imagen de Davidson y Basu (2007).

Las restricciones se generan con base en un subconjunto de instancias etiquetadas como sigue: si dos instancias tienen diferentes etiquetas de clase, se establece una restricción *cannot-link* entre ellas, en caso contrario se establece una restricción *must-link*. La Figura 2.8 muestra dos rostros seleccionados de manera aleatoria. El primero pertenece a la clase con *Rostro hacia arriba*, mientras que el segundo pertenece a la clase con *Rostro de perfil*. Siguiendo la metodología anterior, se les asigna una restricción del tipo *cannot-link*, ya que pertenecen a una clase diferente (tienen diferente orientación), aun cuando los rostros corresponden a la misma persona.

Figura 2.8: Ejemplo de restricciones *cannot-link*. Imagen de Davidson y Basu (2007).

Agrupamiento en análisis de videos

En la investigación presentada en Yan *et al.* (2006), se aborda el problema de clasificación de objetos en videos cuando los datos etiquetados son insuficientes. La solución que proponen para aumentar el desempeño del aprendizaje es emplear restricciones generadas a partir de aspectos espacio-temporales en secuencias de videos, esto es debido a que dentro de las características inherentes que tiene un video se encuentran la continuidad secuencial y multimodal que permiten obtener diversos tipos de restricciones. Hacen uso de restricciones en un problema de clasificación de identidades de las personas a partir de videos de vigilancia. La Figura 2.9 muestra diversas formas en que generan las restricciones. La imagen (a), corresponde a restricciones extraídas del seguimiento de una persona durante un periodo de tiempo y se pueden generar restricciones *must-link* entre grupos de píxeles en fotogramas sucesivos del video, si representan al mismo objeto, o bien, restricciones *cannot-link* entre dos segmentos de imágenes diferentes del mismo video cuando tienen una probabilidad baja de pertenecer al mismo objeto. La imagen (b), corresponde a restricciones espaciales que asocian dos objetos localizados en el mismo fotograma al mismo tiempo; la imagen (c), corresponde a restricciones obtenidas al comparar rostros y la imagen (d), corresponde a restricciones proporcionadas por la retroalimentación de un usuario.

Como conclusión, se encontró que los algoritmos de clasificación haciendo uso de restricciones, mejoran de manera considerable el desempeño de la clasificación comparado con el clasificador base que usa únicamente datos etiquetados.

Datos de GPS

Otra posible aplicación del uso de restricciones en el proceso de agrupamiento se aborda en el artículo Wagstaff *et al.* (2001), en el cual se pretende detectar de manera automática el carril por el que circula cada vehículo en una carretera. Cada vehículo se representa por la posición que ocupa en el carril, mediante coordenadas

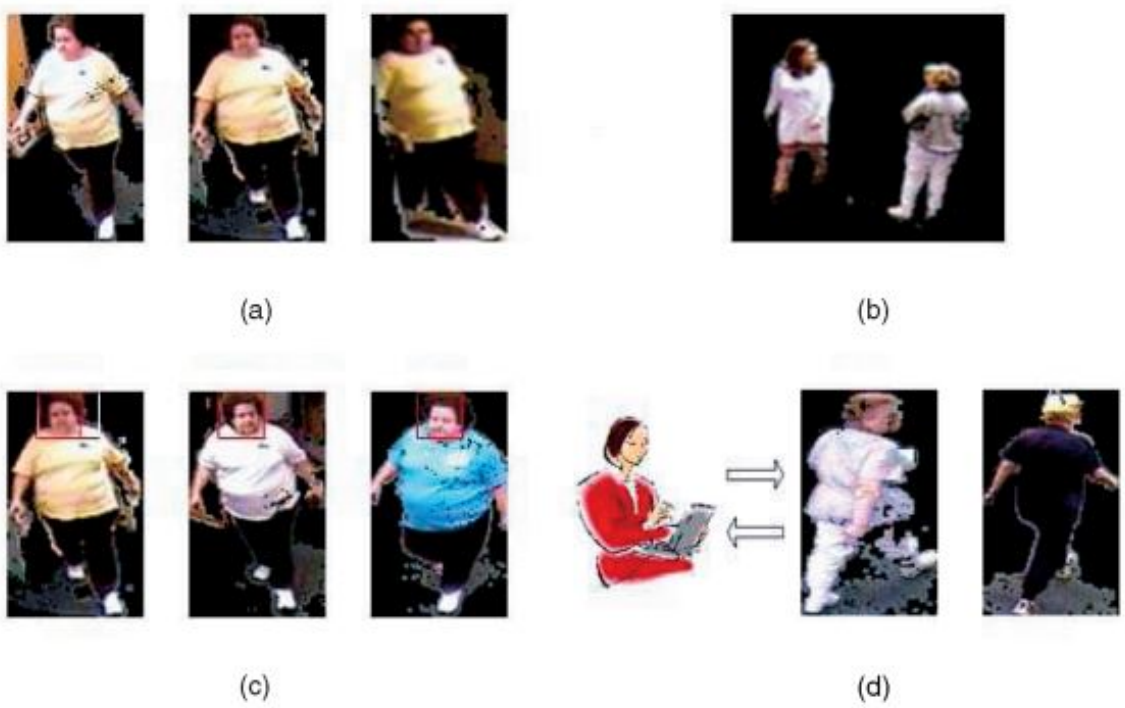


Figura 2.9: Ejemplos de varios tipos de restricciones en vídeos. Imagen de Yan *et al.* (2006).

cartesianas bidimensionales (x, y), las cuales se obtienen en función a los datos del GPS. La generación de restricciones es por parejas de instancias y se basan en los siguientes dos enfoques:

- *Restricciones de acuerdo a la traza de contigüidades*:

  Se imponen restricciones del tipo *must-link* entre pares de automóviles, al considerar una contigüidad en los puntos generados por ambos vehículos en diferentes posiciones del carril y, dado que ninguno de ellos hace cambio de carril, estos deben de permanecer en el mismo grupo.

- *Restricciones de acuerdo a la separación máxima*:

  Se imponen restricciones del tipo *cannot-link* en aquellas parejas de vehículos que se encuentran separados por más de 4 metros en dirección perpendicular al sentido de la trayectoria de uno de ellos. Esto debido a que el ancho máximo de un carril es de 4 metros, por lo tanto, los automóviles con una distancia mayor a esta deben estar en carriles diferentes y por tanto, en distintos grupos.

Dentro de las conclusiones encontradas, se prueba que este modelo de agrupamiento es muy útil para la navegación en tiempo real, permitiendo notificar al usuario cuándo debe de cambiar de carril o bien, cuándo permanecer en el mismo carril.

A continuación se describen algunos métodos del estado del arte propuestos en la literatura, en los que se aborda el problema de agrupamiento semi-supervisado con restricciones a nivel instancias.

2.5. Algoritmos de agrupamiento semi-supervisado con restricciones

Dentro de los algoritmos de agrupamiento semi-supervisado con restricciones existe el problema de factibilidad, que cuestiona la existencia de una solución factible que logre satisfacer todas las restricciones. En el peor de los escenarios, resolver este problema involucra resolver un problema que es **NP-Completo**, es decir, que el tiempo requerido para dar solución al problema de agrupamiento es exponencial. Dos enfoques que hacen frente a este problema son el basado en métodos heurísticos y el basado en algoritmos evolutivos.

El enfoque basado en métodos heurísticos es una buena alternativa para hacer frente a los problemas que son **NP-Completos**, ya que permite obtener soluciones factibles en un tiempo polinomial. Sin embargo, estas heurísticas son susceptibles a quedar atrapadas en óptimos locales, por lo que las soluciones que se obtienen son buenas, pero no necesariamente las óptimas, principalmente en problemas complejos. Estos métodos tienen la ventaja de ser sencillos de implementar. Por otro lado, los métodos que siguen un enfoque de algoritmos evolutivos tienen la ventaja de que llevan a cabo una búsqueda global en un tiempo eficiente.

En esta sección se describen métodos del estado del arte para la resolución de problemas de agrupamiento con restricciones. Particularmente, se presenta una revisión de los métodos de COP-K-Medias (Subsección 2.5.1), LCVQE (Subsección 2.5.2), RDPM (Subsección 2.5.3) y ME-MOEA/D (Subsección 2.5.4).

2.5.1. K-Medias con restricciones (COP-K-Medias)

Uno de los algoritmos básicos para realizar agrupamiento es el K-medias (Hartigan y Wong, 1979). Wagstaff *et al.* (2001) desarrollaron COP-K-Medias, el cual extiende el algoritmo K-Medias para trabajar con restricciones.

A diferencia de K-medias, que asigna una instancia al grupo con centro más cercano, COP-K-Medias asigna cada instancia $x_i \in X$ al grupo más cercano c_j, siempre y cuando esta asignación no viole alguna restricción dentro del conjunto de restricciones R. Si no se encuentra una asignación adecuada, se devuelve la partición vacía como resultado. El algoritmo COP-K-Medias proporciona una partición del conjunto X que cumple necesariamente todas las restricciones.

El Algoritmo 2 describe COP-K-Medias, el cual funciona de la siguiente manera: se generan k centros iniciales, a partir de estos, cada una de las instancias contenidas en el conjunto de datos se va asignando al grupo con el centro más cercano. Antes de realizarse la asignación, se hace uso de la función *VIOLARESTRICCION(x,c,R)*, la cual verifica que al agregar esta instancia en ese grupo, no habrá alguna restricción que se viole. En caso de que se viole alguna restricción, se regresa un conjunto vacío. Una vez que se han asignado todas las instancias en los diferentes grupos, se actualizan los valores de los centros. La iteración se detiene hasta que se encuentre una asignación adecuada para cada instancia.

COP-K-Medias es un algoritmo bastante restrictivo, ya que busca una solución que satisfaga el total de las restricciones. Además, tiene la propiedad de que el número

Algoritmo 2 COP-K-Medias (Wagstaff *et al.*, 2001)

Entrada: X: Conjunto de datos; R: Conjunto de restricciones; k: Número de grupos en el conjunto de datos.
Salida: C: Partición del conjunto de datos.
Función COPKM(X, R, k)
Cálculo de los k centros iniciales $V = \{v_1, ..., v_k\}$.
Repetir:
Para cada $i = 1, ..., N$, **hacer:**
Asignar $x_i \in X$, al grupo c_j con el centro más cercano v_j, $j = 1, ..., k$, tal que *VIOLARESTRICCION*(x_i, c_j, R)=Falso.
Si $\nexists\, c_j \in C$ tal que *VIOLARESTRICCION*(x_i, c_j, R) = Falso, **entonces:**
regresa $\emptyset$.
Fin Si
Fin Para
Para cada $j = 1, ..., k$, **hacer:**
Actualizar el centro v_j, correspondiente al grupo c_j, mediante el promedio de las instancias asignadas en c_j.
Fin Para
Hasta que: se alcance el criterio de convergencia.
Regresa C.
Fin Función

Entrada: Instancia x; Grupo c ; ML: Conjunto de restricciones *must-Link*; CL: Conjunto de restricciones *cannot-Link*.
Salida: *Verdadero:* En caso de que se viole alguna restricción - *Falso*: En caso contrario.
Función *VIOLARESTRICCION*(x, c, R)
Para cada $(x, x_{ML}) \in ML$ **hacer:**
Si $x_{ML} \notin c$ **entonces:**
regresa $Verdadero$.
Fin Si
Fin Para
Para cada $(x, x_{CL}) \in CL$ **hacer:**
Si $x_{CL} \in c$ **entonces:**
regresa $Verdadero$.
Fin Si
Fin Para
En otro caso, regresa Falso.
Fin Función

de grupos esperado $k \in \mathbb{Z}^+$ se tiene que especificar *a priori*. Finalmente, este algoritmo sigue una estrategia voraz, por lo que la solución encontrada no necesariamente será la mejor.

2.5.2. *Linear Constrained Vector Quantization Error* (LCVQE)

El algoritmo LCVQE también aborda el problema de agrupamiento con restricciones. A diferencia del COP-K-Medias, LCVQE es más flexible en el manejo de restricciones. Este algoritmo fue desarrollado por Pelleg y Baras (2007) tomando como base el algoritmo *Constrained vectorization quantization error* (CVQE) (véase Apéndice B para detalles de este algoritmo), que se basa en el algoritmo K-Medias y lo generaliza al incluir restricciones ML y CL por parejas de instancias. Una de las particularidades del algoritmo CVQE es que el orden en que se presentan las instancias que integran cada pareja del conjunto de restricciones es relevante, este hecho da como consecuencia un costo computacional alto.

El algoritmo LCVQE modifica al algoritmo CVQE en el sentido de que no sea relevante el orden de las instancias de cada pareja del conjunto de restricciones, lo cual trae una mejora en el costo computacional. Este cambio ocurre en la regla de actualización de los centros, la cual se presenta a continuación, y tomando en consideración la siguiente notación:

- Sea v_i, el centro del i-ésimo grupo c_i, $i = 1, ..., k$;
- Sea M, la función que devuelve el índice del grupo que contiene a la instancia $x \in X$, tal que, $M = \{j \mid x \in c_j\}$;
- Sea $L_j(i)$, la función que arroja la instancia entre $x_1(i)$ y $x_2(i)$ que más lejos se encuentra del centro v_j;
- Sea $MM(x)$, la función que devuelve el centro más cercano a x distinto de $v_{M(x)}$;
- Sean $g(i) = M(x_1(i))$ y $g'(i) = M(x_2(i))$;
- Sea $h(i)$, la función que devuelve el centro más cercano a v_i;
- Sea $vl(i)$, la variable que indica si la restricción i-ésima ha sido violada, de tal manera que, $vl(i) = 1$ si y sólo si $g(i) \neq g'(i)$ para $i = 1, ..., |ML|$. De forma análoga, $vl(i) = 1$ si y sólo si $g(i) = g'(i)$, para $i = |ML| + 1, ..., |R|$.

La regla de actualización de los centros se define por la siguiente expresión:

$$v_j = \frac{1}{N_j}\left[\sum_{x_i \in c_j} x_i + \frac{1}{2}S_1 + \frac{1}{2}S_2 + \frac{1}{2}S_3\right],$$

donde:

$$S_1 = \sum_{l=1,g(l)=j}^{|ML|} vl(l) \cdot x_2(l),$$
$$S_2 = \sum_{l=1,g'(l)=j}^{|ML|} vl(l) \cdot x_1(l),$$
$$S_3 = \sum_{l=|ML|+1,j=MM(L_{M(x_1(l))}(l))}^{|R|} vl(l) \cdot L_{M(x_1(l))}(l),$$
$$N_j = |c_j| + \frac{1}{2} \sum_{l=1,g(l)=j}^{|ML|} vl(l) + \frac{1}{2} \sum_{l=1,g'(l)=j}^{|ML|} vl(l) + \sum_{l=|ML|+1,j=MM(L_{M(x_1(l))}(l))}^{|R|} vl(l).$$

Finalmente, la función de error que se minimiza durante este proceso de actualización viene dado por la siguiente expresión :

$$E_j = \frac{1}{2} \sum_{x_i \in c_j} T_{j,1} + \frac{1}{2} \sum_{l=1,g(l)=j}^{ml} T_{j,2} + \frac{1}{2} \sum_{l=ml+1,g'(l)=j}^{ml} T_{j,3} + \frac{1}{2} \sum_{l=ml+1,j=MM(L_{M(x_1(l))}(l))}^{ml+cl} T_{j,4},$$

donde:

$$T_{j,1} = (v_j - x_i)^2,$$
$$T_{j,2} = \left[\frac{1}{2}(v_j - x_2(l))^2 \cdot vl(l)\right],$$
$$T_{j,3} = \left[\frac{1}{2}(v_j - x_1(l))^2 \cdot vl(l)\right],$$
$$T_{j,4} = \left[(v_j - L_{M(x_1(l))}(l))^2 \cdot vl(l)\right].$$

El algoritmo LCVQE minimiza una función similar a la que minimiza CVQE, a diferencia de que en cada asignación considera a lo más dos grupos (los dos más naturalmente apropiados). El método realiza $\mathcal{O}(p)$ operaciones en cada paso, con p igual al número de dimensiones, así que la complejidad del algoritmo se vuelve independiente de k. Por tanto, el algoritmo LCVQE es eficaz y eficiente para problemas con un gran número de restricciones y permite encontrar, casi siempre, una solución factible. Adicionalmente, este algoritmo tiene la propiedad de que el número de grupos esperado $k \in \mathbb{Z}^+$, se tiene que especificar *a priori*.

2.5.3. *Relational Dirichlet Process-Means* (RDPM)

Este algoritmo de agrupamiento fue propuesto por Khashabi *et al.* (2015) y se basa en el algoritmo *Dirichlet Process-Means* (DPM), propuesto por Jiang *et al.* (2012) y en el método de agrupamiento llamado *Two-Views Clustering* (TVClust).

El método de agrupamiento TVClust mezcla dos modelos probabilísticos. Estos se modelan de manera independiente mediante una estructura de agrupamiento latente. El primero se asocia con las instancias contenidas en el conjunto de datos y se modela de acuerdo a un proceso de Dirichlet, el cual es un proceso estocástico usado en modelos Bayesianos no paramétricos. El segundo modelo asocia información complementaria proveniente de un conjunto de restricciones, las cuales se generan a partir de un grafo aleatorio y por parejas de instancias. El método TVClust combina estos modelos, buscando un consenso entre ellos, dando como resultado la partición del conjunto de datos X.

TVClust es un método robusto ante errores presentes tanto en el conjunto de datos, como en el conjunto de restricciones y esta fortaleza ocurre al considerar un modelo por separado para ambos elementos. Además, la interpretación de las restricciones es más relajada, pues en este contexto una restricción $ML(x_i, x_j)$, donde $x_i, x_j \in X$, significa que las instancias x_i y x_j podrían estar en el mismo grupo. Mientras que una restricción $CL(x_i, x_j \in X)$, indica que las instancias x_i y x_j podrían no estar en el mismo grupo.

El modelo DPM genera un modelo determinista a partir de un modelo probabilístico al escalar su varianza. Por consiguiente, el algoritmo de agrupamiento RDPM se deriva a partir del escalamiento de la varianza del modelo probabilístico generado con TVClust. El algoritmo RDPM tiene la propiedad de que no se requiere especificar *a priori* el número de grupos esperado del conjunto de datos.

2.5.4. *Improving Constrained Clustering Via Decomposition-based Multiobjective Optimization with Memetic Elitist* (ME-MOEA/D)

ME-MOEA/D, propuesto por González-Almagro, Rosales-Pérez, *et al.* (2021), formula el problema de agrupamiento semi-supervisado con restricciones como un problema de optimización multi-objetivo. Además, sigue un enfoque basado en algoritmos evolutivos y una estrategia evolutiva memética elitista, que consiste en aplicar un procedimiento de búsqueda local sobre los individuos de mejor calidad.

ME-MOEA/D guía su método de solución con base en el mecanismo que sigue MOEA/D (véase Subsección 2.3.1) debido a la capacidad que muestra este último para descomponer un problema de optimización multi-objetivo en subproblemas de un objetivo. ME-MOEA/D hace uso de una búsqueda local para optimizar cada uno de los subproblemas.

Este algoritmo considera a las restricciones *must-link* y *cannot-link* por parejas de instancias y se puede catalogar como algoritmo flexible en el cumplimiento de las restricciones, ya que, a pesar de que este conjunto se usa como guía fuerte durante la evolución del algoritmo, en la partición final del conjunto de datos se permite que sean satisfechas de manera parcial, aunque el ideal es que sean satisfechas por completo.

A continuación se describen los elementos que caracterizan a este algoritmo:

- Con la finalidad de dar solución al problema de agrupamiento, los individuos siguen una **representación basada en etiquetas**.

- Con el propósito de dar solución al problema de agrupamiento semi-supervisado, este algoritmo propone las siguientes tres funciones objetivo, las cuales se desea minimizar:

  - ***Davies-Bouldin***: es una métrica que mide la razón entre la distancia promedio dentro de los grupos y la distancia entre grupos.

    Esta función pretende mantener, en la medida de lo posible, grupos compactos y separados de otros grupos.
  - **Conectividad**: se utiliza para tener bajo control el número de grupos de las soluciones generadas.
  - **No factibilidad**: se usa para integrar las restricciones dentro del proceso de agrupamiento y mide el número de restricciones que se violan en una partición dada.

- La generación de nuevos individuos es a partir de los operadores de variación: **cruza uniforme y mutación uniforme**. La selección de los individuos a cruzar se hace a partir de un operador de selección sesgado. Este toma en consideración tanto al conjunto de los individuos en la población P, como al conjunto de soluciones no dominadas EP. Esta selección se da con una probabilidad de elitismo.

- En el esquema de optimización, mediante el elitismo memético se introduce un sesgo hacia individuos de alta calidad a partir de un procedimiento de búsqueda local. En cada generación se elige a la población de *élite*, la cual se conforma de los ρ individuos menos dominados. Para ello, a cada individuo se le calcula su índice de dominancia, conformado por el número de individuos de la población que lo domina. Posteriormente, a cada individuo p^i de esta población se le aplica un procedimiento de búsqueda local con el objetivo de mejorarlo. Esta búsqueda consiste en seleccionar aleatoriamente el valor en la posición j del individuo, $j \in 1, ..., n$ e iterativamente asignarlo en diferentes grupos. Si se detecta una mejora en las funciones objetivo, se aplica el cambio en la solución. En caso de que se alcance un número máximo de búsquedas sin que se encuentre una mejora, la búsqueda local termina.

2.6. Resumen del capítulo

En este capítulo se revisaron los fundamentos principales de optimización y se definen formalmente los problemas de optimización de un objetivo y multi-objetivo. Uno de los conceptos más importantes en optimización multi-objetivo es la dominancia de Pareto, la cual permite comparar dos soluciones y determinar si una es mejor que otra o que ambas son igualmente buenas desde la perspectiva de Pareto. A partir de la dominancia de Pareto, se definen los conceptos de óptimo de Pareto, conjunto óptimo de Pareto y frontera de Pareto.

Este capítulo también presenta una introducción a los algoritmos evolutivos y su extensión para problemas multi-objetivo. Particularmente, se describe el algoritmo de

MOEA/D, el cual descompone el problema de optimización en N problemas escalares y permite obtener un aproximación a la frontera de Pareto de manera eficiente.

En una segunda parte, el capítulo aborda el problema de agrupamiento semi-supervisado, resaltando las diferencias y ventajas con respecto al problema de agrupamiento tradicional. Finalmente, se describen algunos métodos del estado del arte propuestos para el agrupamiento semi-supervisado, entre los que destacan COP-K-Medias, LCVQE, RDPM y ME-MOEA/D.

Los conceptos revisados en este capítulo forman la base de la descripción del método propuesto dada en el Capítulo 3 y el estudio experimental abordado en el Capítulo 4.

Capítulo 3

Algoritmo evolutivo multi-objetivo para problemas de agrupamiento con restricciones, MOECC

Este capítulo introduce MOECC: *Multi-Objective Evolutionary Constrained Clustering*, un método propuesto para resolver el problema de agrupamiento semi-supervisado con restricciones a nivel instancias. Para ello, MOECC adopta un algoritmo evolutivo multi-objetivo y técnicas de manejo de restricciones para explorar el espacio de soluciones; es decir, los posibles agrupamientos que satisfacen las restricciones de instancias. La formulación como un problema multi-objetivo requiere de la definición de los siguientes aspectos:

- ¿Qué algoritmo evolutivo multi-objetivo se debe adaptar?
- ¿Qué operadores evolutivos y manejador de restricciones serán adaptados al algoritmo evolutivo multi-objetivo para el problema de agrupamiento considerando restricciones del tipo *Must-link* y *Cannot-link*? (Sección 3.2, Sección 3.4)
- ¿Cómo medir la calidad del agrupamiento? (Sección 3.3, Sección 3.5)
- ¿Cómo escoger un agrupamiento de la aproximación a la frontera de Pareto resultante? (Sección 3.6)

La definición de los puntos anteriores no es trivial e impactarán en la eficacia y eficiencia de MOECC. Con respecto a la primera pregunta, MOECC adopta MOEA/D (Zhang y Li, 2007) (véase Subsección 2.3.1) como algoritmo de búsqueda base. MOEA/D tiene varias bondades que lo hacen una opción atractiva para el problema de agrupamiento con restricciones. Primero, MOEA/D descompone el problema multi-objetivo en N subproblemas de optimización escalares, lo que permite encontrar diversos tipos de agrupamientos. Además, gracias a la descomposición, el costo computacional de MOEA/D es inferior a otros algoritmos evolutivos multi-objetivo tradicionales. Segundo, obtiene una aproximación de la frontera de Pareto en una sola ejecución del algoritmo; por tanto, el tomador de decisión cuenta con un abanico de soluciones permitiéndole usar información adicional para escoger el "mejor" agrupamiento.

En el resto del capítulo se describen los elementos que componen a MOECC, en los que se detalla cada una de las cuestiones mencionadas anteriormente como son la

representación de los individuos (Sección 3.1), los operadores evolutivos (Sección 3.2), las funciones objetivo a optimizar (Sección 3.3), un manejador de restricciones (Sección 3.4) y el criterio de paro (Sección 3.5). Finalmente, se describe el método para la selección y mejora del agrupamiento final (Sección 3.6).

3.1. Representación de los individuos

La representación de los individuos juega un papel importante en el diseño de un algoritmo evolutivo para agrupamiento. Una buena representación debe permitir codificar los posibles agrupamientos y, combinada con operadores de evolución adecuados, explorar de manera eficiente el espacio de búsqueda. En la literatura, se identifican tres tipos principales de representación para agrupamiento con algoritmos evolutivos (José-García y Gómez-Flores, 2016):

- **Representación basada en etiquetas**. En este tipo de representación, un individuo está definido como un vector n-dimensional, donde n es el número de instancias en el conjunto de datos. Cada posición del individuo representa una instancia del conjunto de datos y toma un valor entero en el dominio $\{1, 2, \ldots, k\}$, para una partición en k grupos.
- **Representación basada en centros**. Los individuos representan directamente los centros de cada grupo. Existen dos codificaciones principales que se han usado para representar los centros: codificación basada en enteros y codificación basada en reales. En la codificación basada en enteros, un individuo es de longitud k, para un agrupamiento con k particiones, y cada posición del individuo representa el índice de un punto del conjunto de datos que es tomado como el elemento prototipo del grupo. En contraste, en una representación basada en reales, un individuo es de tamaño $k \times u$, donde u es la dimensión de una instancia del conjunto de datos; por tanto, el individuo representa las posiciones de los centros que definen el agrupamiento.
- **Representación basada en grafos**. Este tipo de representación, al igual que la basada en etiquetas, usa una codificación de enteros. Sin embargo, cada posición del individuo puede tomar un valor j en el conjunto $\{1, 2, \ldots, n\}$, tal que si la i-ésima posición toma el valor de j indica que existe un enlace entre las instancias del conjunto de datos i y j; es decir, estos dos elementos están en el mismo grupo. El agrupamiento final se obtiene al identificar todos los elementos conectados y construir el grafo dirigido correspondiente.

En este trabajo, se adopta una representación basada en etiquetas de clase, ya que tiene la ventaja de capturar cualquier forma de agrupamiento y permite representar los grupos de una manera sencilla. Además, la construcción de los grupos a partir de las etiquetas se realiza en tiempo lineal. Sin embargo, a diferencia de la representación tradicional basada en etiquetas, en este trabajo no se establece *a priori* un número máximo de grupos, dando la habilidad de representar de manera natural soluciones con diferentes número de particiones. La Figura 3.1 muestra un ejemplo que permite apreciar esta representación en un individuo p^i de la población, su relación con las instancias del conjunto de datos X y la partición generada.

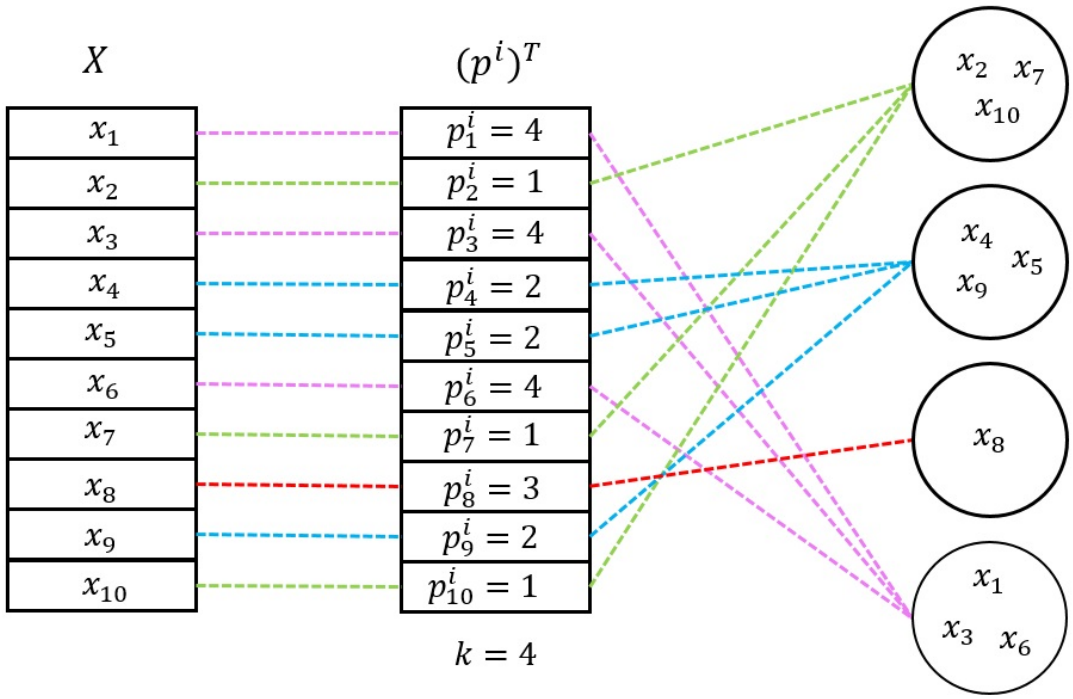


Figura 3.1: Ejemplo de representación basada en etiquetas con diez instancias y cuatro grupos.

La población inicial se crea de manera aleatoria. Cada individuo de la población se crea siguiendo dos pasos principales. Primero, se genera al azar un valor $K \in \{2, \ldots, 20\}$ que corresponde al número de grupos en los que se partirán los datos. Segundo, a cada instancia del conjunto de datos se le asigna una etiqueta aleatoria, es decir, toma un valor en el rango de 1 a K. La inicialización aleatoria tiene la ventaja de ser eficiente y producir una población inicial diversa. Aunque esta población inicialmente considera un tamaño máximo de grupos igual a 20, los operadores evolutivos de cruza y mutación propuestos en esta tesis permiten producir nuevas soluciones con tamaños de agrupamiento diferentes, explorando el espacio de los diferentes posibles agrupamientos.

3.2. Operadores evolutivos

Se ha mencionado anteriormente que la información que contienen los individuos de la población determina la manera en que las instancias de un conjunto de datos se agrupan, por lo que es de gran importancia que los operadores evolutivos, útiles para la reproducción, permitan encontrar individuos cada vez mejores cuyas soluciones arrojen buenos agrupamientos. A continuación se describen los operadores de cruza y mutación adaptados en el algoritmo.

3.2.1. Cruza basada en grupos

El operador de cruza tiene por objetivo generar un nuevo individuo hijo a partir de la información contenida en los padres. En esta tesis, se propone un operador de cruza

que crea un nuevo hijo a partir de los grupos definidos por los padres. La cruza, como usualmente sucede en los algoritmos evolutivos, ocurre con una cierta probabilidad denominada tasa o probabilidad de cruza.

El Algoritmo 3 describe de manera general la cruza basada en grupos. Su funcionamiento es el siguiente: dado los padres p^a, que divide el conjunto de datos X en los grupos $C_{p^a} = \{c_{p^a 1}, \ldots, c_{p^a k_a}\}$, y p^b con grupos $C_{p^b} = \{c_{p^b 1}, \ldots, c_{p^b k_b}\}$, el operador de cruza propuesto crea una solución descendiente p^{new} con grupos $C_{p^{new}} = \{c_{p^{new} 1}, \ldots, c_{p^{new} k_d}\}$. Para ello, primero se selecciona de manera aleatoria un subconjunto de grupos de p^a que pasarán al hijo p^{new} sin cambios. Esta selección se realiza con una distribución de probabilidad uniforme, es decir, cada grupo tiene la misma probabilidad de ser o no seleccionado. La premisa detrás de ello es tratar de conservar la información de los grupos potenciales proveniente del padre. Las instancias de los grupos no seleccionados de p^a son agrupadas en una de dos formas posibles. La primera consiste en agrupar esas instancias con base en la información dada por p^b, ignorando aquellas previamente agrupadas. En la segunda, las instancias restantes son agrupadas en conjunto, formando un grupo. De manera similar, la selección de una de estas dos maneras se da con una distribución de probabilidad uniforme.

Algoritmo 3 Cruza basada en grupos

Entrada: p^a: Padre 1; p^b: Padre 2; *rate*: Valor en el intervalo (0,1).
Salida: p^{new}: Solución descendiente.

Función CLUSTERING_BASED(p^a, p^b, *rate*)
  Obtener los k_a grupos de p^a: $C_{p^a} = \{c_{p^a 1}, \ldots, c_{p^a k_a}\}$.
  Obtener los k_b grupos de p^b: $C_{p^b} = \{c_{p^b 1}, \ldots, c_{p^b k_b}\}$.
  Para cada $i = 1, ..., k_a$, **hacer:**
    Generar valor uniforme aleatorio, α.
    Si $\alpha < rate$ **entonces:**
      El grupo $c_{p^a i}$ pasa a formar parte $C_{p^{new}}$.
    Fin Si
  Fin Para
  Generar valor uniforme aleatorio, ρ.
  Si $\rho < rate$ **entonces:**
    Las instancias de los grupos no seleccionados de p^a se agrupan de acuerdo a los grupos en C_{p^b}, sin considerar las ya agrupadas.
  si no:
    Las instancias de los grupos no seleccionados de p^a conforman un único grupo en p^{new}.
  Fin Si
  Regresa p^{new}.
Fin Función

La Figura 3.2 muestra un ejemplo de este operador con un conjunto de datos con diez instancias. Inicialmente, se tienen los individuos p^a y p^b con soluciones con tres y cuatro grupos, respectivamente. En la figura a) se muestra el hijo producido al preservar el grupo formado por las instancias 2, 3 y 8 de p^a y el resto de las instancias son agrupadas de acuerdo con la información dada en p^b ignorando las previamente agrupadas. La figura b) representa el hijo resultante de preservar el grupo formado por las instancias 2, 3 y 8 de p^a y el resto de las instancias se unen en un nuevo grupo.

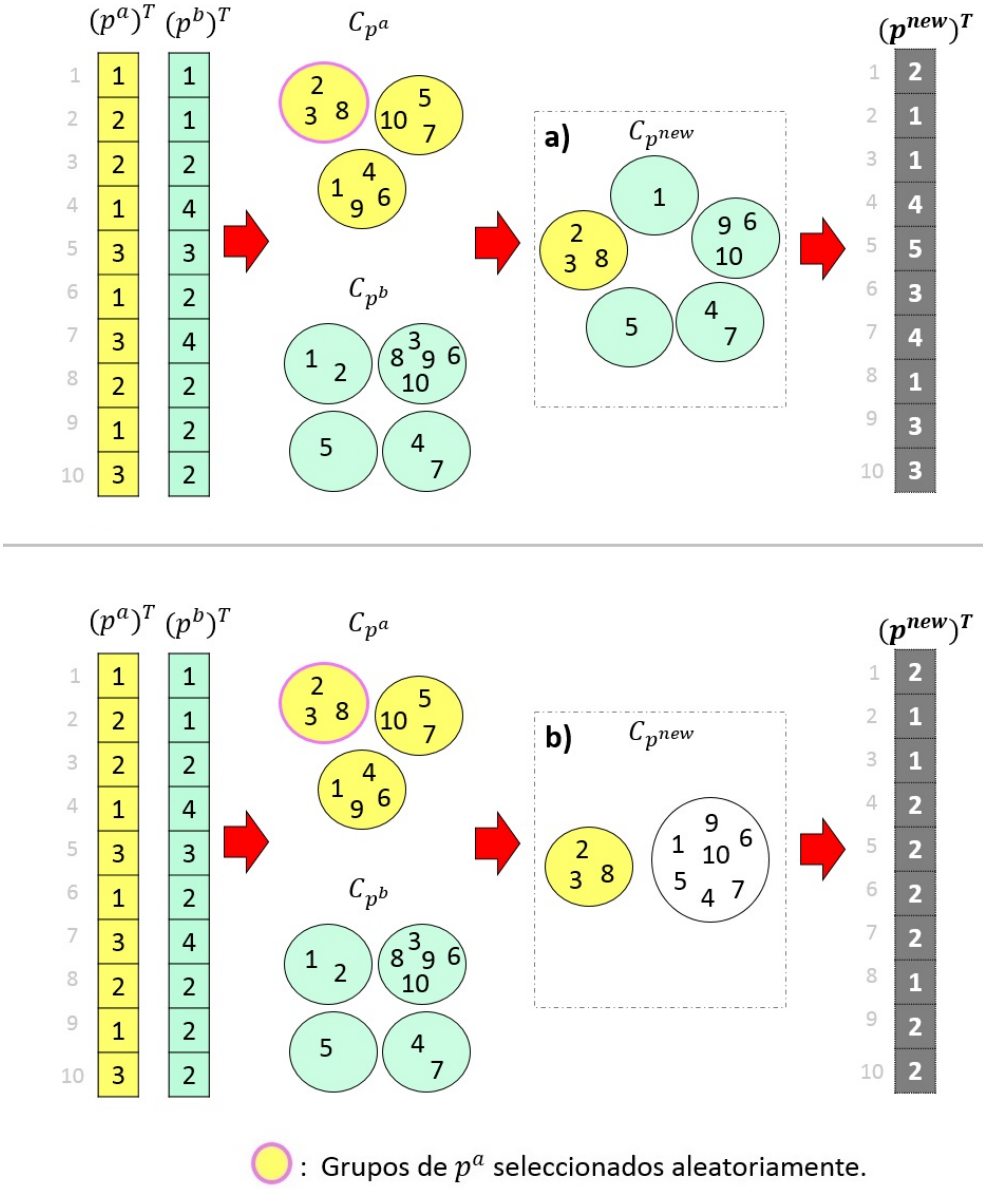


Figura 3.2: Ejemplo de cruza basada en grupos con diez instancias. La figura **a)** muestra el individuo resultante p^{new} que hereda un grupo sin cambios de p^a y el resto de los grupos son heredados de p^b, ignorando los previamente agrupados. La figura **b)** muestra el individuo resultante p^{new}, que hereda un grupo sin cambios de p^a y el resto de los grupos forman un solo conjunto.

3.2.2. Mutación basada en vecindario

MOECC implementa la mutación basada en vecindario. La premisa básica de este operador es que instancias que son vecinas tienen mayor posibilidad de pertenecer al mismo grupo. De esta manera, la mutación favorece la conectividad del agrupamiento.

El Algoritmo 4 describe de manera general la mutación basada en vecindario. Su funcionamiento es el siguiente: dada una solución candidata p^c, el operador de mutación determina si la etiqueta de grupo p_i^c para la instancia i es mutada o no. Para ello, se mantiene una lista de los L vecinos más cercanos de cada instancia $\mathbf{x}_i \in X \setminus \mathbf{x}_i$. Posteriormente, se genera un número aleatorio con distribución uniforme

para cada instancia y se verifica si cumple o no con la probabilidad de mutación. Aquellas instancias que satisfacen la probabilidad de mutación son colocadas en el mismo grupo de uno de sus L vecinos más cercanos seleccionado de manera aleatoria.

Algoritmo 4 Mutación basada en vecindario

Entrada: p^c: Solución candidata; $neighbors\_list$: L vecinos más cercanos de cada instancia $\mathbf{x}_i \in X \setminus \mathbf{x}_i$; $mutation\_rate$: Tasa de mutación.
Salida: p^{new}: Solución mutada.

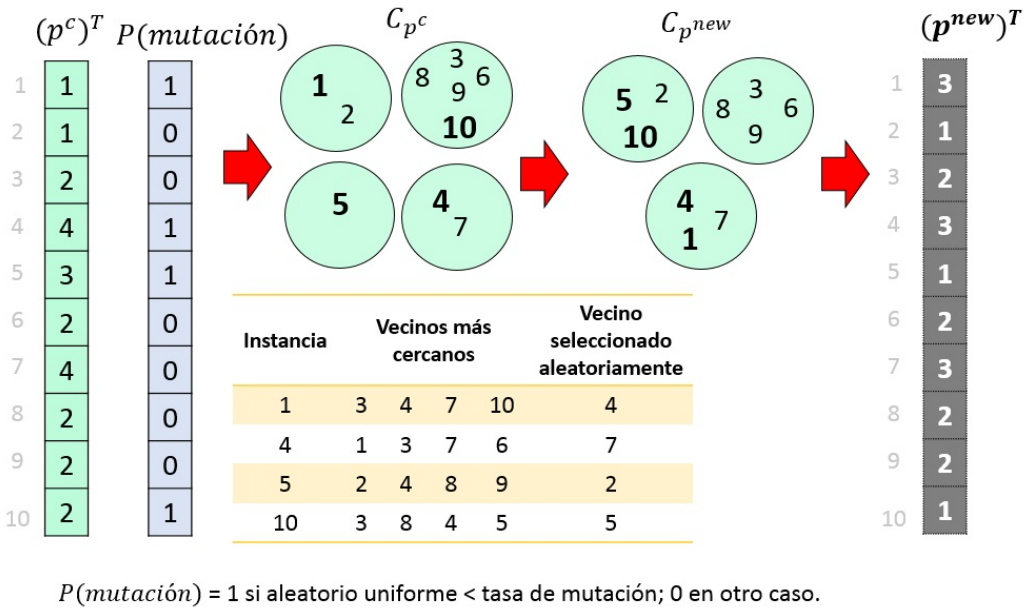
```
Función NEAREST_MUTATION(p^c, neighbors_list, mutation_rate)
    Para cada i = 1, ..., |p^c| hacer:
        Generar valor uniforme aleatorio, α.
        Si α < mutation_rate entonces:
            Selección aleatoria de uno de los vecinos más cercanos de x_i, x_nearest.
            Intercambio de la instancia x_i al grupo de x_nearest.
        Fin Si
    Fin Para
    Regresa p^new.
Fin Función
```

La Figura 3.3 muestra un ejemplo de este operador. Se tiene un individuo que representa una solución con diez instancias y cuatro grupos. A cada instancia se determina de manera independiente si entra al proceso de mutación o no. En el ejemplo, las instancias 1, 4, 5 y 10 son seleccionadas. Para cada una de ellas, se selecciona uno de sus vecinos más cercanos de manera aleatoria, siendo la instancias 4, 7, 2 y 5 respectivamente. Como resultado, en el individuo mutado, las instancias 1, 4 y 7 son colocadas en el mismo grupo, la instancia 5 se mueve al grupo de la instancia 2 y la instancia 10 se mueve al grupo de la instancia 2 y 5.

3.3. Funciones objetivo

Un buen agrupamiento busca, idealmente, que las instancias más semejantes entre sí se encuentren en el mismo grupo, mientras que las instancias que están en diferentes grupos sean desemejantes. Una medida típica para estimar la similitud entre dos instancias es la distancia euclídea, de tal manera que entre menor sea esta distancia, más similares son. Por tanto, en un buen agrupamiento, la distancia promedio entre los pares de instancias del mismo grupo es menor en comparación con la distancia promedio con pares de instancias de diferentes grupos.

Una manera de validar la calidad en el agrupamiento de un conjunto de instancias es mediante métricas de validación interna. En esta tesis se ha adoptado la compacidad y la conectividad para medir la calidad del agrupamiento. Estas dos métricas son complementarias, ya que la compacidad busca soluciones que tengan un número grande de grupos, mientras que la conectividad busca soluciones con pocos grupos. Estos dos criterios son optimizados a través del algoritmo evolutivo multi-objetivo, dando a MOECC la habilidad de balancear el número de particiones de manera automática.



Instancia	Vecinos más cercanos				Vecino seleccionado aleatoriamente
1	3	4	7	10	4
4	1	3	7	6	7
5	2	4	8	9	2
10	3	8	4	5	5

Figura 3.3: Ejemplo de mutación basada en vecindarios con diez instancias. Muestra el individuo p^{new}, resultado de la mutación de p^c. Las instancias 1, 4, 5 y 10 satisfacen la probabilidad de mutación y cada una se coloca en el mismo grupo que uno de sus vecinos más cercanos, el cual se selecciona aleatoriamente (4, 7, 2 y 5, respectivamente).

3.3.1. Compacidad

La compacidad está muy relacionada con la varianza interna de los grupos, también conocida como inercia, y nos indica qué tan cercanas se encuentran las instancias contenidas en un mismo grupo.

Dado un individuo p^i de la población que agrupa al conjunto de datos X en las particiones $C = \{c_1, \ldots, c_{k_i}\}$, la varianza interna del grupo c_j se calcula como la suma de las distancias euclídeas al cuadrado entre cada una de las instancias $x_i \in c_j$ y el centro μ_j, esto es:

$$\text{WCSS}(c_j) = \sum_{x_i \in c_j} (x_i - \mu_j)^2. \tag{3.1}$$

La compacidad asociada al individuo p^i es la suma de las varianzas internas de cada grupo, es decir:

$$\text{Compacidad}(p^i) = \sum_{c_j \in C} \text{WCSS}(c_j). \tag{3.2}$$

Un agrupamiento que tiene una suma de cuadrados menor es más compacto que un agrupamiento con una suma de cuadrados mayor. Esto es, valores muy cercanos a cero en la compacidad indicarán que la calidad de los grupos es buena y, por consiguiente, se desea minimizar este valor.

3.3.2. Conectividad

Esta medida se basa en la idea de que las instancias que están cercanas entre sí deben de estar asignadas en el mismo grupo.

Dado un individuo p^i de la población que agrupa el conjunto de datos X en las particiones $C = \{c_1, \ldots, c_{k_i}\}$, la conectividad se calcula como el grado en el cual instancias vecinas han sido colocadas en el mismo grupo, esto es:

$$\text{Conectividad}(p^i) = \frac{1}{n}\sum_{j=1}^{n}\left(\sum_{l=1}^{L}\delta\left(x_j, x_{nn_{jl}}\right)\right), \tag{3.3}$$

donde L es el tamaño del vecindario y $\delta\left(x_j, x_{nn_{jl}}\right)$ es una función que penaliza que la instancia x_j y su l-ésimo vecino $x_{nn_{jl}}$ no sean asignados al mismo grupo, y se calcula como:

$$\delta\left(x_j, x_{nn_{jl}}\right) = \begin{cases} \frac{1}{l} & \text{si } p^i_j \neq p^i_{nn_{jl}} \\ 0 & \text{en caso contrario.} \end{cases} \tag{3.4}$$

donde p^i_j y $p^i_{nn_{jl}}$ son las etiquetas de grupo que el individuo p^i asigna a las instancias x_j y $x_{nn_{jl}}$, respectivamente.

Nótese que la función δ enfatiza la cercanía de las instancias, es decir, entre más cercanas estén dos instancias, mayor será la penalización. El mejor escenario ocurre cuando $p^i_j = p^i_{nn_{jl}}$, para todo $j \in \{1, \ldots, n\}$ y todo $l \in \{1, \ldots, L\}$, obteniéndose una conectividad de cero. Esto es, un agrupamiento se considera como bueno si su conectividad es muy cercana a cero y, por esta razón, se desea minimizar esta función.

La Figura 3.4 ilustra dos grupos con buena compacidad (imagen lado izquierdo) y dos grupos altamente conectados (imagen lado derecho). Finalmente, la compacidad tiene su mejor valor cuando cada instancia del conjunto de datos forma un grupo, aunque esto causaría el peor valor en la conectividad. La conectividad, por su parte, tiene su mejor valor cuando todos los vecinos son agrupados en conjunto, incrementando el valor de la compacidad. Por tanto, la optimización simultánea de ambos criterios ayuda a compensar los sesgos individuales de cada uno de ellos y a buscar soluciones diversas tanto en particiones como en la agrupación de las instancias.

3.4. Regla de factibilidad por ϵ-restringido

MOECC adopta la técnica de ϵ-restringido (Takahama y Sakai, 2006) para trabajar de manera apropiada con las restricciones *Must-link* y *Cannot-link* a fin de obtener soluciones factibles, es decir, que cumplan con las restricciones. Básicamente, el método de ϵ-restringido permite comparar dos soluciones p^a y p^b y determinar cuál es mejor considerando los valores tanto de las funciones objetivo como de las violaciones de las restricciones[1].

El método de ϵ-restringido consiste de dos pasos: la relajación de las restricciones y la comparación de las soluciones. La relajación permite considerar como factibles

[1]Una restricción se dice que es violada cuando la solución no satisface la condición impuesta por la restricción.

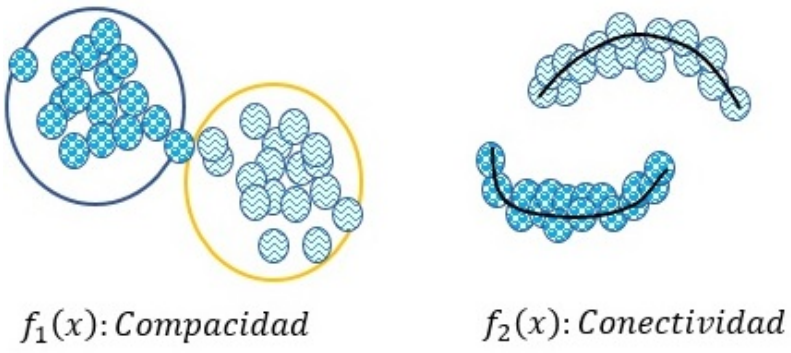


Figura 3.4: Ejemplo de conectividad en un conjunto de datos. Imagen adaptada de (Handl *et al.*, 2005).

a soluciones que tengan un nivel de violación de restricciones menor que un umbral $\epsilon \geq 0$. Este umbral se ajusta durante la búsqueda del modo siguiente:

$$\epsilon = \begin{cases} \epsilon(0)\left(1 - \frac{k}{T_c}\right)^{cp}, & \text{si } k < T_c, \\ 0, & \text{en otro caso.} \end{cases} \tag{3.5}$$

donde T_c es el parámetro de control de generaciones, cp es un parámetro que controla la velocidad de reducción de la relajación y se define en el rango de 1 a 5, k es el contador de la generación actual y $\epsilon(0)$ es el número de restricciones no satisfechas de la θ-ésima mejor solución de la población inicial ordenada con base en el número de restricciones violadas.

Por otra parte, la comparación de soluciones se realiza de manera lexicográfica, verificando primero la violación de las restricciones y después el valor de aptitud. Lo anterior se basa en la idea que una solución factible es preferible a una solución no factible con mejor valor en las funciones objetivo. La comparación por ϵ-restringido indica que la solución p^a es mejor que p^b si y sólo si se cumple alguno de los siguientes casos:

$$\begin{cases} g^{te}(p^a \mid \lambda_i, \mathbf{z}^*) < g^{te}\left(p^b \mid \lambda_i, \mathbf{z}^*\right) & \text{si } \phi(p^a), \phi\left(p^b\right) \leq \epsilon; \\ g^{te}(p^a \mid \lambda_i, \mathbf{z}^*) < g^{te}\left(p^b \mid \lambda_i, \mathbf{z}^*\right) & \text{si } \phi(p^a) = \phi\left(p^b\right); \\ \phi(p^a) < \phi\left(p^b\right) & \text{en otro caso.} \end{cases} \tag{3.6}$$

donde $\phi(p^a)$ y $\phi\left(p^b\right)$ representan la proporción de restricciones violadas por las soluciones p^a y p^b, $g^{te}(p^a \mid \lambda, \mathbf{z}^*)$ y $g^{te}\left(p^b \mid \lambda, \mathbf{z}^*\right)$ son los valores obtenidos con la función de escalamiento de Tchebycheff para p^a y p^b, λ_i es el i-ésimo vector de pesos y $\mathbf{z}^*$ es el vector objetivo ideal.

Para el cálculo de las aproximaciones de Tchebycheff, MOECC normaliza los valores de compacidad y conectividad, tal que estos valores estén en el rango de cero a uno. Para hacer esto, primero se calculan los valores máximos y mínimos que se pueden alcanzar para cada función objetivo, considerando lo siguiente:

- La compacidad toma su valor mínimo cuando el número de grupos es igual al número de instancias en el conjunto de datos, esto es, cada instancia es un grupo y es el centro del mismo. En este caso, la distancia entre cada instancia y su centro es de cero, dando como resultado una compacidad mínima de cero.

- La compacidad toma su valor máximo cuando el número de grupos es igual a uno. En este caso, el centro está definido como el punto medio de todas las instancias. En consecuencia, la distancia de cada una de ellas al centro es mayor o igual que cero. El valor máximo de la compacidad depende del conjunto de datos, por lo que se calcula asumiendo una solución con un único grupo.

- La conectividad toma su valor mínimo cuando cada instancia está agrupada con sus L vecinos más cercanos. En este caso, la función δ de la Ecuación 3.4 toma el valor de cero. Por tanto, el valor mínimo de la conectividad es de cero.

- La conectividad toma su valor máximo cuando cada instancia está en un grupo distinto al de sus L vecinos más cercanos. En esta caso, el valor máximo de la conectividad está dado por $\sum_{l=1}^{L} \frac{1}{l}$.

Nótese que para MOECC, el vector objetivo ideal $\mathbf{z}^*$ corresponde a los valores mínimos en compacidad y conectividad. Sea $\mathbf{z}^w$ el vector con los valores máximos de compacidad y conectividad, y $\mathbf{f}(p^i)$ el vector con los valores de compacidad y conectividad de la solución p^i, los valores normalizados se calculan como:

$$\hat{\mathbf{f}}\left(p^i\right) = \frac{\mathbf{f}(p^i) - \mathbf{z}^*}{\mathbf{z}^w - \mathbf{z}^*} \tag{3.7}$$

3.5. Criterio de paro

MOECC explora el espacio de agrupamientos, buscando soluciones que satisfagan un compromiso entre los criterios optimizados: la compacidad y la conectividad. Este proceso se da de manera iterativa hasta que una condición de parada se cumpla. MOECC implementa dos criterios: convergencia y un máximo número de evaluaciones de las funciones objetivo.

En esta tesis, se considera que se ha alcanzado la convergencia cuando la mejora en la aproximación a la frontera de Pareto no es significativa durante un número fijo de iteraciones o generaciones consecutivas. Para medir la calidad de la aproximación a la frontera de Pareto, se adopta la medida del hiper-volumen (véase Sección 2.1.1 para más detalles).

Sea EP_t el conjunto de soluciones no dominadas al término de la generación t, $\nu(EP_t, \rho)$ y $\nu(EP_{t-1}, \rho)$ los valores de hiper-volumen de dos generaciones consecutivas, $\rho = \mathbf{1}_2$ el punto de referencia para el cálculo del hiper-volumen y $\sigma > 0$ el umbral de significatividad. Se considera que existe una mejora en la aproximación a la frontera de Pareto si se cumple lo siguiente:

$$\frac{\nu(EP_t, \rho)}{\nu(EP_{t-1}, \rho)} - 1 \geq \sigma. \tag{3.8}$$

Para evitar un sesgo en el cálculo del hiper-volumen debido a la diferencia de escalas de los objetivos, estos primero son escalados usando la Ecuación 3.7.

Una vez que alguna de las condiciones de parada se ha cumplido, se obtiene el conjunto de soluciones que mejor aproxima la frontera de Pareto. Esta es una ventaja del algoritmo, pues permite obtener más de una solución óptima al problema. Sin embargo, pese a que todas esas soluciones son buenas desde el punto de vista

multi-objetivo, es necesario seleccionar una de ellas como el agrupamiento solución. A continuación, se describe la estrategia propuesta para escoger una solución dentro del conjunto óptimo de Pareto.

3.6. Selección y mejora del agrupamiento

Como se ha mencionado, el resultado de la optimización multi-objetivo no es un único agrupamiento, sino un conjunto de agrupamientos. Cada una de estas particiones corresponde a una solución que satisface en diferente grado un compromiso entre la compacidad y la conectividad. Además, estas particiones pueden corresponder a soluciones con diferente número de grupos. Por tanto, un experto en el dominio del problema puede analizar los diferentes agrupamientos resultantes y escoger uno con base en su experiencia.

Cuando el conocimiento del experto no está disponible, es deseable contar con una estrategia que permita seleccionar de manera automática la solución dentro del conjunto de soluciones. Afortunadamente, MOECC provee información sobre cada posible partición a través de sus valores de compacidad y conectividad, los cuales pueden ser analizados para la toma de decisión.

Sea K_i el número de grupos de la i-ésima solución. Las evaluaciones resultantes en la frontera de Pareto son ordenadas de manera ascendente con respecto al número de grupos. Si K_i es bajo, se espera tener un valor alto en la compacidad y un valor bajo en la conectividad; por el contrario, si K_i es alto, entonces se espera un valor bajo de compacidad y un valor alto de conectividad. A partir de esto, la razón entre la compacidad y la conectividad será alta para grupos pequeños e irá decreciendo conforme K_i se incrementa. Siguiendo el criterio propuesto por Milligan y Cooper (1985), la máxima diferencia entre niveles jerárquicos es tomada como indicativo del agrupamiento correcto de los datos y, por tanto, la solución asociada es seleccionada dentro del conjunto de agrupamientos.

La Figura 3.5 ilustra un ejemplo de la estrategia de toma de decisión para el agrupamiento final. Se tiene cinco soluciones resultantes en el conjunto de Pareto, cada una asociada por sus valores de compacidad y conectividad escalados y ordenadas de manera ascendente de acuerdo al número de grupos en su solución. La máxima diferencia entre niveles jerárquicos es 15.83 y corresponde a la solución del índice 2, siendo la solución seleccionada.

Una vez seleccionada la solución p^* de la frontera de Pareto, entra en una fase de mejora. Esta fase consiste en escoger los dos grupos más cercanos y unirlos siempre y cuando no exista una degradación significativa en las funciones objetivo y que no añada una violación de las restricciones. Se considera que no existe una degradación significativa si la razón de la compacidad entre el agrupamiento K y el $K-1$ es superior a 0.95.

Índice	Soluciones - conjunto de Pareto								*K*
1	6	4	3	2	1	2	5	3	6
2	1	1	2	2	2	1	2	1	2
3	1	3	2	2	1	4	1	5	5
4	1	4	2	3	1	4	2	3	4
5	1	1	2	3	2	1	1	3	3

Ordenamiento, *K*

Índice	Soluciones del conjunto de Pareto								*K*
2	1	1	2	2	2	1	2	1	2
5	1	1	2	3	2	1	1	3	3
4	1	4	2	3	1	4	2	3	4
3	1	3	2	2	1	4	1	5	5
1	6	4	3	2	1	2	5	3	6

Índice	Evaluaciones - frontera de Pareto	
	Compacidad	Conectividad
1	0.01	0.83
2	0.9	0.03
3	0.22	0.69
4	0.69	0.19
5	0.85	0.06

Ordenamiento, *K*

Índice	Evaluaciones - frontera de Pareto		Razón	Diferencia
	Compacidad	Conectividad		
2	0.9	0.03	**30.00**	**15.83**
5	0.85	0.06	14.17	10.54
4	0.69	0.19	3.63	3.31
3	0.22	0.69	0.32	0.31
1	0.01	0.83	0.01	

Agrupamiento final: 1 1 2 2 2 1 2 1 **Número de grupos: 2**

Figura 3.5: Ejemplo de selección de la solución final. Las evaluaciones resultantes en la frontera de Pareto son ordenadas de manera ascendente con respecto al número de grupos de sus correspondientes soluciones en la frontera de Pareto. Al obtener la razón entre la compacidad y la conectividad, la máxima diferencia entre niveles jerárquicos se alcanza en el índice 2, por lo que su solución asociada representa el agrupamiento final, el cual contiene 2 grupos.

Capítulo 4

Diseño experimental y análisis de resultados

En este capítulo se presenta el diseño experimental (Sección 4.1) y se analizan los resultados obtenidos en la experimentación (Sección 4.2).

4.1. Diseño experimental

En esta sección se describe la configuración usada en las implementaciones para cada algoritmo de referencia y para MOECC, las cuales se realizan en *Python*. Para los métodos COP-K-Medias, LVCQE y RDPM se toman como referencia las implementaciones generadas por González Almagro (2018), mientras que para el método ME-MOEA/D se toma como referencia la implementación proporcionada por el autor González Almagro *et al.* (2020).

Para medir el desempeño en los grupos generados por cada algoritmo, se hace uso de las métricas de validación externa: ARI *(Adjusted Rand Index)*, AMI *(Adjusted Mutual Information)*, la homegeneidad, la medida de completitud y la medida V (véase Sección C.2).

Los algoritmos RDPM y MOECC detectan el número de grupos de manera automática, mientras que COP-K-Medias, LCVQE y ME-MOEA/D requieren conocer *a priori* el número de grupos. Con el propósito de hacer comparables a los métodos, en esta tesis se adopta una metodología que permite encontrar el número de grupos de manera automática en los métodos COP-K-Medias, LCVQE y ME-MOEA/D, la cual se describe a continuación.

4.1.1. Aproximación *bootstrap* para la búsqueda del tamaño de grupos K.

Se hace uso de la aproximación *bootstrap* para el análisis de grupos propuesta por Jain y Moreau (1987). A través del *bootstrap*, se mide la estabilidad en los grupos generados de un conjunto de datos, lo cual permite determinar el número de grupos adecuado.

El muestreo *bootstrap* consiste en generar un conjunto de datos mediante un muestreo con reemplazo de las instancias contenidas en el conjunto de datos original. La

aproximación *bootstrap* consiste en realizar q muestreos *bootstrap* y por cada uno de ellos, aplicar un método de agrupamiento, variando el tamaño de grupos K.

Posteriormente, se aplica un criterio de validación de grupos, que consiste en comparar las variaciones de alguna métrica que mida la calidad de los grupos, R_k, sobre cada una de las muestras y por cada valor de K. Grandes variaciones en R_k, para un número fijo de grupos K, indica una inestabilidad en los agrupamientos de las muestras y por tanto, malos agrupamientos en los conjuntos de datos original. Por lo que variaciones pequeñas en R_k, para un número fijo K, será indicador de un buen agrupamiento en el conjunto de datos original.

Dentro de la experimentación de estos métodos de referencia, se considera a la variable K en el conjunto $\{2, \ldots, 20\}$ y para cada valor se realizan 25 muestreos *Bootstrap*. Además, la métrica de validación que se propone para medir la calidad del agrupamiento es el índice Hartigan (véase Sección C.1).

A continuación se describe la configuración empleada para cada método de referencia.

4.1.2. COP-K-Medias

Este algoritmo devuelve como salida la partición del conjunto de datos considerando el tamaño predefinido de grupos K. El procedimiento está definido por:

```
def COPKM(X, K, constraint_mat, max_iter, tol, init,
random_state)
```

- X: conjunto de datos de tamaño $n \times m$, con n instancias y m características;
- K: número de grupos de la partición final.
  $K \in \{2, ..., 20\}$;
- *constraint_mat*: matriz de restricciones de tamaño $n \times n$, donde la entrada $(i, j) = 1$ si se trata de una restricción ML entre las instancias x_i y x_j, o bien, $(i, j) = -1$ si se trata de una restricción CL;
- *max_iter*: máximo número de iteraciones que puede realizar el algoritmo.
  $max\_iter = 300$;
- *tol*: umbral para criterio de paro del algoritmo. Sirve para determinar el movimiento mínimo permitido en los centros.
  $tol = 0.0001$;
- *init*: modelo de inicialización de los centros.
  $init = rand$, es decir, que se generan de manera aleatoria;
- *random_state*: semilla para la generación de los centros.
  $random\_state = 2^i$, $i = 1, ..., 30$.

4.1.3. LCVQE

Este algoritmo devuelve como salida la partición del conjunto de datos considerando el tamaño predefinido de grupos K. Su procedimiento está definido por:

```
def LCVQE(X, K, constraint_mat, max_iter, random_state)
```

- X: conjunto de datos de tamaño $n \times m$, con n instancias y m características;
- K: número de grupos de la partición final.
  $K \in \{2, ..., 20\}$;
- $constraint\_mat$: matriz de restricciones de tamaño $n \times n$, donde la entrada $(i, j) = 1$ si se trata de una restricción ML entre las instancias x_i y x_j, o bien, $(i, j) = -1$ si se trata de una restricción CL;
- max_iter: máximo número de iteraciones que puede realizar el algoritmo.
  $max\_iter = 300$;
- $random\_state$: semilla para la generación de los centros.
  $random\_state = 2^i$, $i = 1, ..., 30$.

4.1.4. RDPM

Este algoritmo devuelve como salida la partición del conjunto de datos. Su procedimiento está definido por:

```
def RDPM(X, lamb, constraint_mat, max_iter, xi_0, xi_rate)
```

- X: conjunto de datos de tamaño $n \times m$, con n instancias y m características;
- $lamb$: distancia para considerar si una instancia pertenece a un grupo, tomando como referencia el centro de grupo.
  $lamb = 6$;
- $constraint\_mat$: matriz de restricciones de tamaño $n \times n$, donde la entrada $(i, j) = 1$ si se trata de una restricción ML entre las instancias x_i y x_j, o bien, $(i, j) = -1$ si se trata de una restricción CL;
- max_iter: máximo número de iteraciones que puede realizar el algoritmo.
  $max\_iter = 300$;
- xi_0: parámetro que controla el compromiso entre los pesos del conjunto de datos y el de restricciones.
  $xi\_0 = 0.001$;
- xi_rate: Parámetro que controla la proporción en que se incrementa xi_0.
  $xi\_rate = 2$.

4.1.5. ME-MOEA/D

Este algoritmo devuelve como salida la partición del conjunto de datos considerando el tamaño predefinido de grupos K. Su procedimiento está definido por:

```
def MEMOEAD_LB_GECCO(X, K, ml_const, cl_const, N, ref_z,
pbt_mutation, pbt_inherit, pbt_elite_crossover,
lambda_neigh_size, elite_percent, fail_percent,
connect_neigh_size, km_percent, random_state)
```

- X: conjunto de datos de tamaño $n \times m$, con n instancias y m características;
- K: número de grupos de la partición final.
  $K \in \{2, ..., 20\}$;
- ml_const: matriz de restricciones $must-link$ de tamaño $r \times 3$. Cada renglón representa una pareja de instancias, las primeras dos columnas indican los índices de las instancias y la columna 3 el valor 1;
- cl_const: matriz de restricciones $cannot - link$ de tamaño $r \times 3$. Cada renglón representa una pareja de instancias, las primeras dos columnas indican los índices de las instancias y la columna 3 el valor -1;
- N: Tamaño de la población.
  $N = 200$;
- ref_z: Punto de referencia.
  $ref\_z = [0, 0, 0]$;
- $pbt\_mutation$: tasa de mutación.
  $pbt\_mutation = 0.1$;
- $pbt\_inherit$: tasa de cruza.
  $pbt\_inherit = 0.1$;
- $pbt\_elite\_crossover$: tasa para selección de padre 2 para la cruza.
  $pbt\_elite\_crossover$=0.25;
- $lambda\_neigh\_size$: número de vecinos más cercanos para los vectores de pesos λ.
  $lambda\_neigh\_size = 10$;
- $elite\_percent$: porcentaje de individuos de la población que conforman el grupo élite.
  $elite\_percent = 0.25$;
- $fail\_percent$: máxima proporción de fallos permitidos en la búsqueda local.
  $fail\_percent \in (0, 1)$.
  $fail\_percent = 0.1$;
- $connect\_neigh\_size$: número de vecinos más cercanos para cada instancia del conjunto de datos.
  $connect\_neigh\_size = 10$;

- *km_percent*: porcentaje de individuos considerados para la generación de la población inicial.
  $km\_percent = 0.25$;
- *random_state*: semilla para la generación de los vectores de pesos λ, la población inicial y la generación de números aleatorios para la mutación y la cruza.
  $random\_state = 2^i$, $i = 1, ..., 30$.

4.1.6. MOECC

Este algoritmo devuelve como salida la partición del conjunto de datos. Su procedimiento está definido por:

```
def MOECC(X, constraints_mat, N, max_evaluations,
neighborhood_size, n_neighbors, crossover_rate,
mutation_rate, delta_early_stopping,
patience_early_stopping, random_state)
```

- X: conjunto de datos de tamaño $n \times m$, con n instancias y m características;
- *constraint_mat*: Matriz de restricciones de tamaño $n \times n$, donde la entrada $(i, j) = 1$ si se trata de una restricción ML entre las instancias x_i y x_j, o bien, $(i, j) = -1$ si se trata de una restricción CL;
- N: Tamaño de la población.
  $N = 200$;
- *max_evaluations*: Número máximo de evaluaciones.
  $max\_evaluations = 100,000$;
- *neighborhood_size*: Número de vectores colindantes para las instancias.
  $neighborhood\_size = 25$;
- *n_neighbors*: Número de vectores colindantes para los vectores de pesos λ_i.
  $n\_neighbors = 5$;
- *crossover_rate*: Tasa de cruza.
  $crossover\_rate = 1.0$;
- *mutation_rate*: Tasa de mutación.
  $mutation\_rate = 0.10$;
- *delta_early_stopping*: Cambio mínimo porcentual en el hiper-volumen entre dos generaciones consecutivas.
  $delta\_early\_stopping = 0.05$;
- *patience_early_stopping*: Máximo número de generaciones consecutivas en que el cambio en el hiper-volumen es menor que *delta_early_stopping*.
  $patience\_early\_stopping = 100$;
- *random_state*: semilla para la generación de los vectores de pesos λ, la población inicial y la generación de números aleatorios para la mutación y la cruza.
  $random\_state = 2^i$, $i = 1, ..., 30$.

4.1.7. Conjuntos de datos

En la Tabla 4.1 se describen los conjuntos de datos que se usan en la experimentación. Los conjuntos de datos son diversos en el número de instancias, de variables y de clases que contienen. Estos conjuntos de datos son previamente preprocesados para tener media cero y desviación estándar unitaria y los atributos categóricos son convertidos a numéricos.

Tabla 4.1: Conjuntos de datos usados en la experimentación.

Datos	Descripción	Instancias	Variables	Clases
Accent voice	Reconocimiento de acentos en tonos de voz. Personas de distintas ciudades	329	12	6
BankNote	Características de imágenes de billetes falsos y verdaderos	1,372	4	2
Bench	Señales del sonido de rebote proveniente de un cilindro de metal y de una roca aproximadamente cilíndrica.	208	60	2
Breast cancer	Diagnósticos de cáncer de mama	569	30	2
Diabetes	Persona con o sin diabetes	768	8	2
Digits	Características de imágenes de dígitos entre el 0 y el 9.	1,797	63	10
Divorce	Predictores de divorcio	170	54	2
Heart failure	Detección de muertes por insuficiencia cardiaca	299	12	2
Ionosphere	Detección de electrones libres en la ionosfera buenos o malos	351	32	2
Iris	Conjunto de plantas iris	150	4	3
Qsar biodeg	Descriptores moleculares utilizados para clasificar productos químicos en 2 clases, listo y no listo biodegradable.	1,055	41	2
Steel plates	Indicadores para medir si hay defectos superficiales en placas de acero inoxidable	1,941	33	2
Wine	Reconocimiento de vinos	178	13	3
Zoo	Conjuntos de animales	101	16	7

4.1.8. Generación de restricciones

Para la generación de restricciones se define *a priori* una tasa de restricciones, que indica el número de restricciones que se crearán en función al número de instancias del conjunto de datos.

Para automatizar el proceso de generación de restricciones, se hace uso de las etiquetas de clase de las instancias del conjunto de datos. Además, las instancias a las que se les impone restricción se obtienen de manera aleatoria. Cada restricción ocurre entre pares de instancias diferentes. Para cada restricción, se compara la etiqueta de clase de ambas instancias seleccionadas; si las etiquetas son iguales, entonces se les asigna la restricción *must-link*. Por el contrario, si las etiquetas son diferentes, entonces se les asigna la restricción *cannot-link*.

La generación de restricciones se realiza de manera automática a través de un método en Python definido como:

```
def generate_constraints(X, y, rate, noise, random_state)
```

- X: conjunto de datos de tamaño $n \times m$, con n instancias y m características;
- y: etiquetas de clase de las instancias del conjunto de datos X;
- $rate$: porcentaje de restricciones.
  $rate \in \{0.5, 0.10, 0.15, 0.20, 0.25, 0.30\}$;
- $noise$: porcentaje de restricciones inválidas.
  $noise = 0.0$;
- $random\_state$: semilla para la selección aleatoria de instancias con restricción.
  $random\_state = 2^i$, $i = 1, ..., 30$.

La Tabla 4.2 muestra el número de restricciones generadas por cada conjunto de datos y de acuerdo al nivel de restricciones. Las restricciones se generan a partir de las instancias de las que se conoce su etiqueta de clase.

Sea $r = n \cdot rate$ el número de instancias de las que se sabe su etiqueta de clase, n es el número de instancias del conjunto de datos. A partir de la información de las r instancias, el número de restricciones será el número total de pares que se forman entre ellas y se calcula como:

$$restricciones = \frac{r(r-1)}{2}. \tag{4.1}$$

Tabla 4.2: Número de restricciones generadas por conjunto de datos y por nivel de restricciones.

Datos	Instancias	5 %	10 %	15 %	20 %	25 %	30 %
Accent voice	329	120	496	1,176	2,080	3,321	4,753
BankNote	1,372	2,278	9,316	20,910	37,401	58,653	84,255
Bench	208	45	190	465	820	1,326	1,891
Breast cancer	569	378	1,540	3,570	6,328	10,011	14,365
Diabetes	768	703	2,850	6,555	11,628	18,336	26,335
Digits	1,797	3,916	15,931	36,046	64,261	100,576	144,991
Divorce	170	28	136	300	561	861	1,275
Heart failure	299	91	406	946	1,711	2,701	3,916
Ionosphere	351	136	595	1,326	2,415	3,741	5,460
Iris	150	21	105	231	435	666	990
Qsar biodeg	1,055	1,326	5,460	12,403	22,155	34,453	49,770
Steel plates	1,941	4,656	18,721	42,195	75,078	117,370	169,071
Wine	178	28	136	325	595	946	1,378
Zoo	101	10	45	105	190	300	435

4.2. Análisis de resultados

En esta sección se muestran los resultados reportados por COP-K-Medias, LCV-QE, RDPM y ME-MOEA/D y MOECC. Las métricas de validación externa que se usan para medir el desempeño en la calidad de los grupos generados por cada algoritmo son el índice ARI *(Adjusted Rand Index)*, el índice AMI *(Adjusted Mutual Information)*, la homogeneidad, la medida de completitud y la medida V (véase Sección C.2 para más detalles de estas métricas).

Para comparar los desempeños de los métodos se hace uso de gráficas de caja y, de acuerdo con la naturaleza de cada una de las métricas de validación, en general, el comportamiento ideal es aquel en que el rango de valores de la caja se encuentra más próximo al valor 1, que es el mejor desempeño en cada una de las métricas e indica que el método logra una adecuada agrupación de los datos; se busca además que las cajas sean compactas, lo que significa que hay poca dispersión y gran estabilidad en el desempeño del método. Aunado a lo anterior, también se buscan cajas con asimetría negativa ya que de esta manera los desempeños tendrán una mayor concentración en los valores altos. Por el contrario, los métodos cuyas cajas tienen una asimetría positiva y además presentan un rango con valores bajos, se pueden considerar como métodos que no están logrando agrupar de manera adecuada a los datos.

Con respecto al comparativo en el tiempo y en el porcentaje de restricciones no satisfechas, se espera un comportamiento contrario al descrito anteriormente, ya que

en estos casos lo que se busca son métodos con un tiempo de ejecución y un porcentaje de restricciones no satisfechas que sean lo más próximas a cero.

4.2.1. Desempeño general

La Figura 4.1 muestra el desempeño general de los métodos, los cuales se obtienen al comparar las etiquetas predichas contra las etiquetas de los grupos reales. En la gráfica correspondiente a la medida ARI, MOECC muestra el mejor desempeño, con un ARI promedio de 0.54, mientras que el método COP-K-Medias, LCVQE, RDPM y ME-MOEA/D tienen un ARI promedio de 0.04, 0.31, 0.25 y 0.32, respectivamente. El método con el desempeño más bajo es COP-K-Medias.

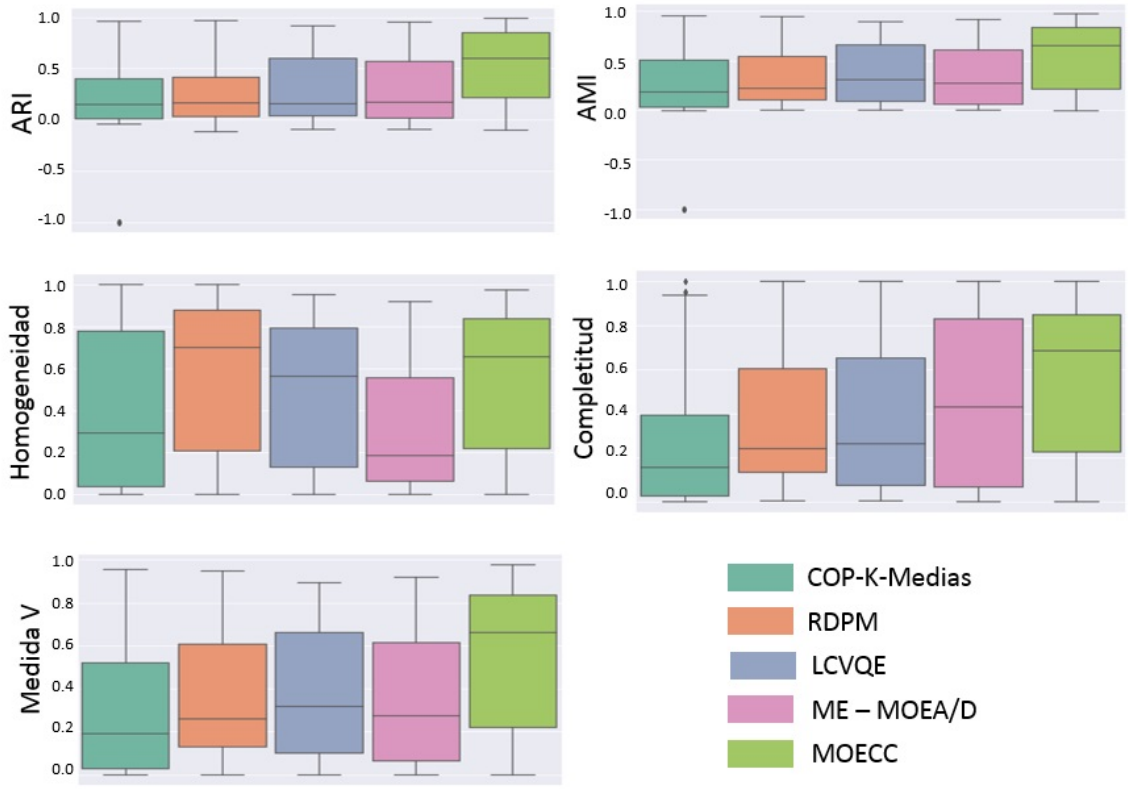


Figura 4.1: Desempeño general de los métodos, mediante las métricas ARI, AMI, homogeneidad, completitud y medida V.

Con respecto a la medida AMI, MOECC presenta el mejor desempeño con un AMI promedio de 0.53, mientras que los métodos COP-K-Medias, LCVQE, RDPM y ME-MOEA/D tienen un AMI promedio de 0.08, 0.37, 0.32 y 0.35, respectivamente. El método con menor desempeño en AMI es, nuevamente, COP-K-Medias.

En cuanto a la homogeneidad, la medida de completitud y la medida V, estas métricas se basan en medidas de entropía condicional normalizadas. En homogeneidad, RDPM muestra el mejor desempeño con un valor promedio de 0.57, mientras que COP-K-Medias, LCVQE, ME-MOEA/D y MOECC presentan una homogeneidad promedio de 0.41, 0.47, 0.33 y 0.53, respectivamente. En la medida de completitud, MOECC tiene el mejor desempeño con un valor de 0.54, en tanto COP-K-Medias,

LCVQE, RDPM y ME-MOEA/D obtienen un valor promedio de 0.25, 0.37, 0.39 y 0.43, respectivamente. Finalmente, en la medida V, al ser esta la media armónica entre la homogeneidad y la completitud, MOECC presenta el mejor desempeño con un valor promedio de 0.53; COP-K-Medias, LCVQE, RDPM y ME-MOEA/D obtienen un valor promedio de 0.29, 0.38, 0.35 y 0.36, respectivamente.

4.2.2. Desempeño por nivel de restricciones

Hasta este momento, en una vista general, la propuesta MOECC es la que mejor desempeño muestra en las métricas de validación. La Tabla 4.3 muestra un desglose por nivel de restricciones. Lo que se observa es que, independientemente de la proporción de restricciones que se generen, el desempeño promedio en todas las métricas tiende a ser mejor en MOECC, salvo en la homogeneidad de los escenarios de restricciones 5 %, 10 % y 15 %, donde RDPM muestra un mejor desempeño.

Además, se puede observar que a medida que crece la proporción de restricciones consideradas, el desempeño de los métodos tiende a mejorar. MOECC alcanza un desempeño promedio por arriba de 0.60 en todas las métricas de evaluación, al considerar el 30 % de restricciones, lo cual es bueno, pues sólo fue necesario generar restricciones para un 30 % de las instancias contenidas en los conjuntos de datos, en lugar de haber tenido que etiquetar a todas las instancias, lo cual ahorra tiempo, esfuerzo y costos. Además, el cambio en el desempeño de MOECC entre los escenarios de restricciones del 25 % y 30 % es de 0.03. En todos los niveles de restricciones ME-MOEA/D obtuvo el segundo lugar en las métricas de ARI y completitud y la tercer mejor posición en las métricas de AMI y medida V.

Tabla 4.3: Desempeño promedio de los métodos por nivel de restricciones.

Restric.	Método	ARI	AMI	Homogeneidad	Completitud	Medida V
5 %	COPKM	0.13	0.19	0.40	0.25	0.29
	LCVQE	0.26	0.32	0.42	0.31	0.33
	RDPM	0.18	0.26	**0.50**	0.34	0.29
	MEMOEAD	0.26	0.29	0.27	0.37	0.30
	MOECC	**0.38**	**0.39**	0.38	**0.43**	**0.39**
10 %	COPKM	0.06	0.10	0.38	0.24	0.28
	LCVQE	0.27	0.34	0.43	0.33	0.34
	RDPM	0.20	0.29	**0.55**	0.33	0.32
	MEMOEAD	0.27	0.31	0.29	0.39	0.32
	MOECC	**0.48**	**0.47**	0.47	**0.48**	**0.47**
15 %	COPKM	-0.02	0.02	0.38	0.23	0.28
	LCVQE	0.32	0.39	0.48	0.39	0.40
	RDPM	0.26	0.33	**0.57**	0.39	0.36
	MEMOEAD	0.33	0.37	0.34	0.45	0.37

(Continúa en la página siguiente)

Tabla 4.3: Desempeño promedio de los métodos por nivel de restricciones (continuación).

Restric.	Método	ARI	AMI	Homogeneidad	Completitud	Medida V
	MOECC	**0.56**	**0.55**	0.55	**0.55**	**0.55**
20 %	COPKM	-0.04	-0.01	0.38	0.24	0.28
	LCVQE	0.34	0.41	0.50	0.41	0.42
	RDPM	0.28	0.35	0.60	0.42	0.39
	MEMOEAD	0.34	0.38	0.36	0.46	0.38
	MOECC	**0.62**	**0.60**	**0.61**	**0.61**	**0.61**
25 %	COPKM	-0.02	0.01	0.44	0.26	0.31
	LCVQE	0.36	0.43	0.54	0.42	0.44
	RDPM	0.32	0.38	0.62	0.46	0.41
	MEMOEAD	0.38	0.41	0.38	0.51	0.42
	MOECC	**0.67**	**0.65**	**0.66**	**0.65**	**0.66**
30 %	COPKM	0.11	0.15	0.50	0.29	0.35
	LCVQE	0.32	0.39	0.51	0.38	0.40
	RDPM	0.31	0.35	0.60	0.43	0.38
	MEMOEAD	0.38	0.40	0.37	0.46	0.40
	MOECC	**0.65**	**0.63**	**0.65**	**0.62**	**0.63**

La Figura 4.2 muestra los diagramas de caja con base en la métrica ARI, por nivel de restricciones. De manera general, el método COP-K-Medias es el más disperso, además, este método obtiene los valores más bajos en desempeño en todos los niveles de restricciones, su rango de valores se concentra principalmente en (0.00, 0.30). RDPM siempre mantiene una asimetría positiva, concentrando sus valores en la parte inferior y su rango de valores se concentra principalmente en (0.00, 0.60). LCVQE y ME-MOEA/D muestran una distribución un poco similar, haciendo una transición de asimetría positiva a negativa conforme aumenta el número de restricciones, es decir, que mejora su desempeño. Para ambos métodos, su rango de valores se concentra principalmente en (0.00, 0.60). Finalmente, MOECC con pocas restricciones tiene simetría y a medida que aumenta el número de restricciones hace una transición a una asimetría negativa, es decir que su desempeño se va concentrando en la parte positiva, su rango de valores también se recorre hacia la parte superior, concentrándose principalmente en (0.25, 0.90).

La Figura 4.3 muestra los diagramas de caja con base en la métrica AMI, por nivel de restricciones. De manera general, COP-K-Medias es el más disperso, y obtiene los valores más bajos en desempeño en todos los niveles de restricciones y su rango de valores se concentra principalmente en (0.10, 0.50). En RDPM predomina la asimetría positiva, concentrando sus valores principalmente en la parte inferior y su rango de valores se concentra principalmente en (0.10, 0.60). LCVQE y ME-MOEA/D muestran una transición parecida en sus distribuciones a medida que aumenta el número de

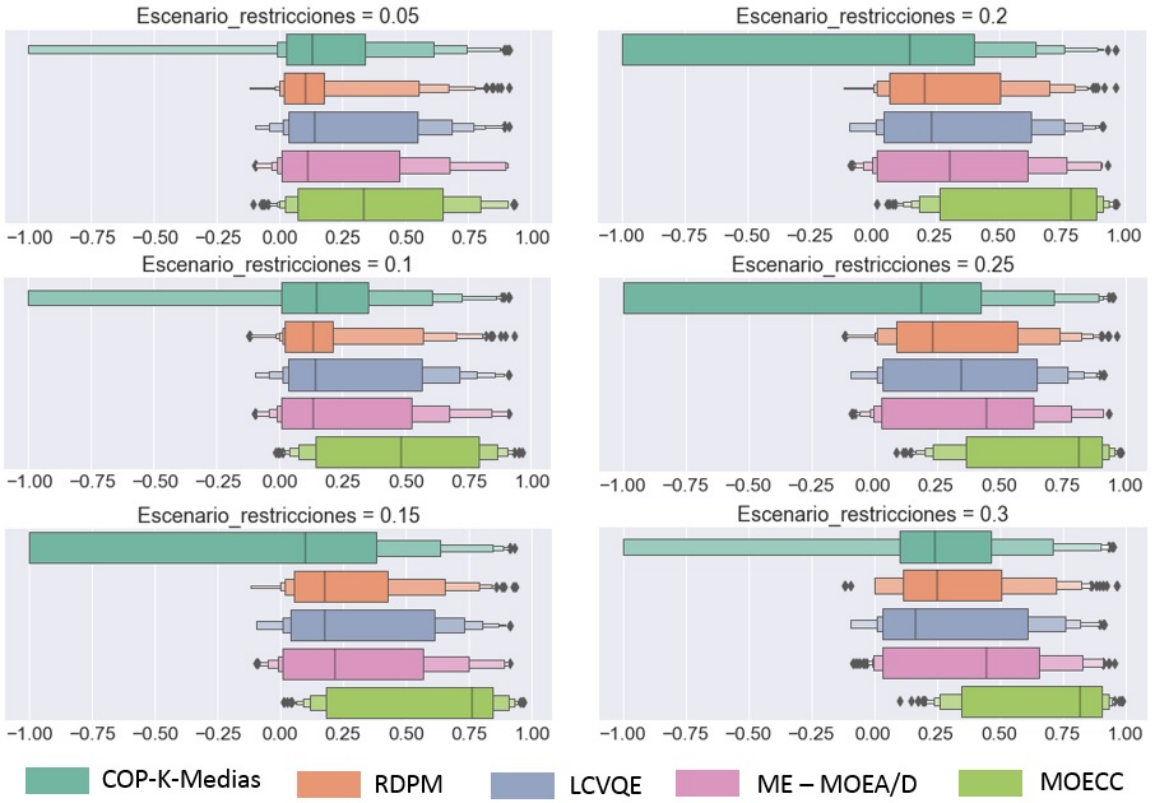


Figura 4.2: ARI por nivel de restricciones y por modelo.

restricciones, pasando de una asimetría positiva a una negativa y el rango de valores se concentra principalmente en (0.10, 0.70). MOECC también muestra una transición de asimetría positiva a negativa a medida que aumenta el número de restricciones y su rango se concentra principalmente en (0.25, 0.90).

La Figura 4.4 muestra los diagramas de caja con base en la métrica homogeneidad, por nivel de restricciones. En esta métrica, los cinco métodos muestran un rango de desempeño muy disperso, lo que significa que puede haber conjuntos de datos donde los grupos generados exhiben buena homogeneidad, en tanto otros exhiben baja homogeneidad. Conforme aumenta el número de restricciones, COP-K-Medias pasa de una asimetría positiva a simétrica, es decir, que logra buen desempeño en la mitad de los conjuntos de datos a medida que tiene más información. RDPM en todos los escenarios de restricciones mantiene una asimetría negativa, haciéndose cada vez más negativa a medida que aumenta el nivel de restricciones, además, el 50 % de sus desempeños están entre el 0.80 y 0.90. LCVQE y ME-MOEA/D también en esta métrica presentan el mismo comportamiento en transición de asimetrías, yendo de una asimetría positiva a una negativa a medida que crece el número de restricciones, aunque el desempeño de LCVQE es mejor que el de ME-MOEA/D. Finalmente, MOECC pasa de una simetría a una asimetría negativa a medida que aumentan las restricciones y, al igual que RDPM, el 50 % de sus desempeños están entre 0.80 y 0.90. A diferencia de RDPM, a medida que aumentan las restricciones el rango de MOECC tiende a compactarse en la parte superior.

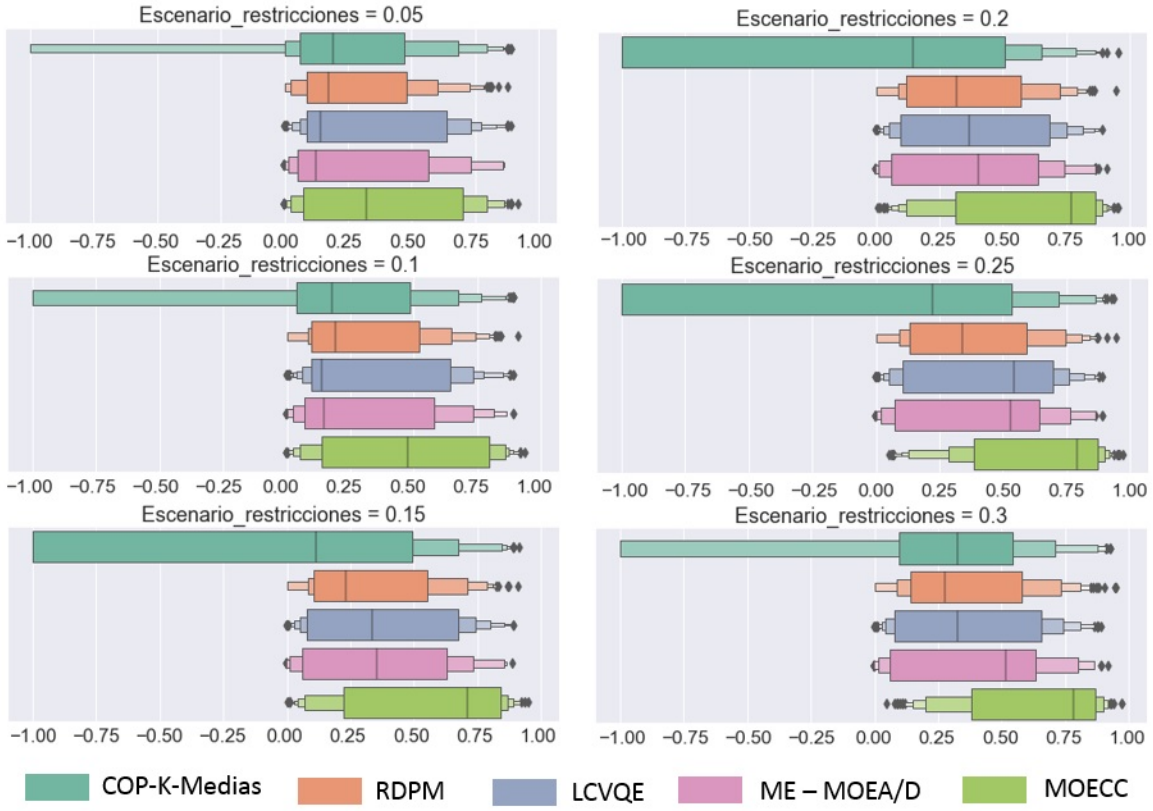


Figura 4.3: AMI por nivel de restricciones y por modelo.

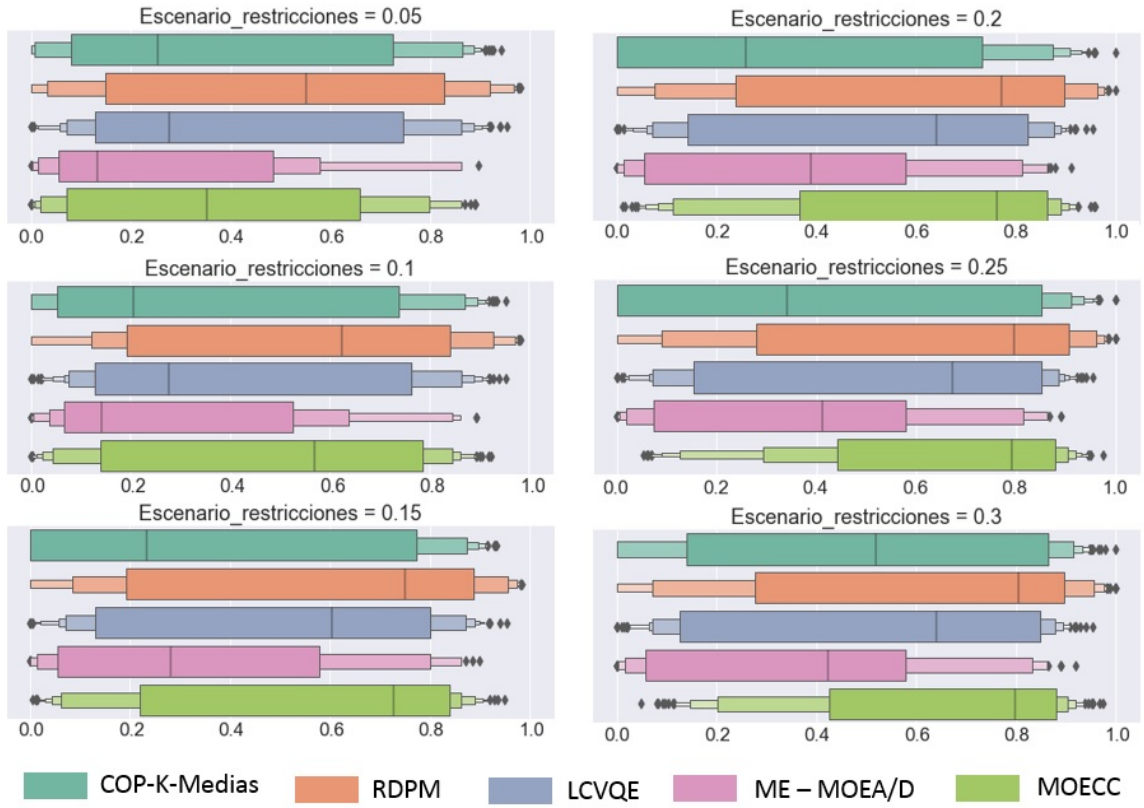


Figura 4.4: Homogeneidad por nivel de restricciones y por modelo.

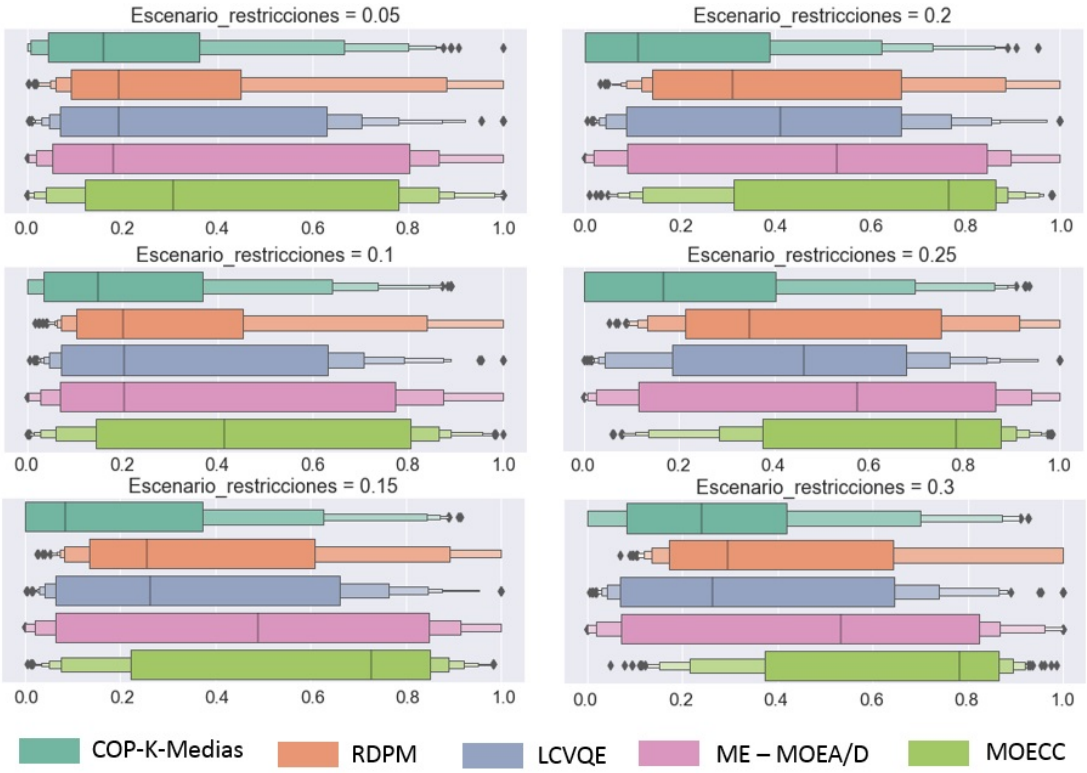


Figura 4.5: Completitud por nivel de restricciones y por modelo.

La Figura 4.5 muestra los diagramas de caja con base en la medida de completitud, por nivel de restricciones. COP-K-Medias en los diferentes escenarios tiene un desempeño bajo, el cual está en el rango (0.00, 0.40), presentando algunos valores atípicos en la parte superior. RDPM de manera general permanece en una asimetría positiva, aunque su rango mejora ligeramente a medida que crece el número de restricciones y se concentra en el rango (0.20, 0.70). LCVQE, ME-MOEAD y MOECC muestran una transición de asimetría positiva a negativa a medida que crece el número de restricciones. Este cambio se ve aún más marcado en MOECC, ya que no sólo se hace más compacto su rango, sino que además este se recorre hacia la parte superior y el 50 % de las ocasiones su desempeño ronda el 0.80.

La Figura 4.6 muestra los diagramas de caja con base en la media V, por nivel de restricciones. Se ha dicho anteriormente que RDPM es el método con mejor desempeño en cuanto a homogeneidad, sin embargo, su nivel en la medida de completitud es baja. Por otro lado, MOECC muestra el mejor desempeño en la medida de completitud y además ocupa el segundo lugar en desempeño con base en la homogeneidad. Al ser la medida V la media armónica entre la homogeneidad y la completitud, MOECC es el método con mejor desempeño de manera general.

4.2.3. Desempeño por conjunto de datos

La Tabla 4.4 muestra el desempeño promedio de los métodos al considerar todos los escenarios de restricciones (5 %, 10 %, 15 %, 20 %, 25 % y 30 %) por cada uno de los conjuntos de datos. El desempeño de los modelos varía según el conjunto de datos,

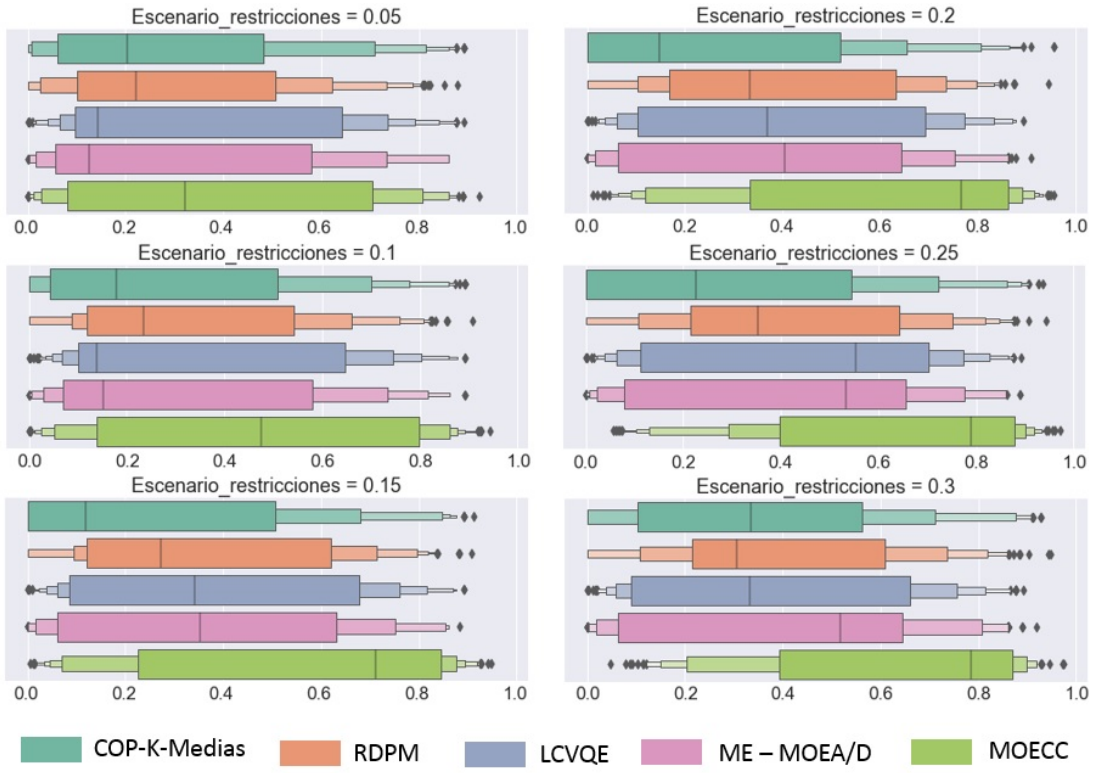


Figura 4.6: Medida V por nivel de restricciones y por modelo.

habiendo casos en los que los agrupamientos generados son buenos y, por consiguiente, el valor en las métricas de validación son altas. Sin embargo, también hay conjuntos de datos donde ninguno de los métodos logra un buen desempeño.

Para el caso de los conjuntos de datos donde se obtiene un desempeño en los modelos menor a 0.70 se encuentran *Accent voice* (329 instancias; 12 variables; 6 clases), *BankNote* (1,372 instancias; 4 variables; 2 clases), *Bench* (208 instancias; 60 variables; 2 clases), *Diabetes* (768 instancias; 8 variables; 2 clases) , *Heart failure* (299 instancias; 12 variables; 2 clases), *Ionosphere* (351 instancias; 32 variables; 2 clases), *Qsar biodeg* (1,055 instancias; 41 variables; 2 clases) y *Steel plates* (1,941 instancias; 33 variables; 2 clases), estos en su mayoría son de clases pequeñas. MOECC es el método que mejor desempeño muestra en 5 de estos conjuntos de datos (*Accent voice*, *BankNote*, *Bench*, *Ionosphere* y *Steel plates*), mientras que RDPM separa mejor en 2 (*Diabetes* y *Heart failure*) y COP-K-Medias también en 2 (*Heart failure* y *Qsar biodeg*).

El resto de los conjuntos de datos en su mayoría tienen la particularidad de poseer un alto número de variables y de clases, *Breast cancer* (569 instancias; 30 variable; 2 clases), *Digits* (1,797 instancias; 63 variables; 10 clases), *Divorce* (170 instancias; 54 variables; 2 clases), *Iris* (150 instancias; 4 variables; 3 clases), *Wine* (178 instancias; 13 variables; 3 clases) y *Zoo* (101 instancias, 16 variables y 7 clases). MOECC es el que obtiene los mejores agrupamientos pues su desempeño en las métricas es superior a 0.75, a excepción de la métrica de homogeneidad, donde RDPM es el que mejor los agrupa.

Tabla 4.4: Desempeño promedio de los métodos por conjunto de datos.

Datos	Método	ARI	AMI	Homogeneidad	Completitud	Medida V
Accent voice	COPKM	-0.89	-0.87	0.03	0.02	0.03
	LCVQE	-0.06	0.13	0.13	0.27	0.14
	RDPM	0.04	0.19	0.20	0.25	0.21
	MEMOEAD	-0.02	0.10	0.09	0.19	0.11
	MOECC	**0.20**	**0.31**	**0.35**	**0.33**	**0.34**
BankNote	COPKM	0.17	0.26	**0.50**	0.18	0.26
	LCVQE	0.11	0.19	0.38	0.13	0.19
	RDPM	0.07	0.05	0.05	**0.43**	0.05
	MEMOEAD	0.04	0.04	0.06	0.03	0.04
	MOECC	**0.30**	**0.34**	**0.50**	0.26	**0.34**
Bench	COPKM	0.07	0.09	0.16	0.07	0.10
	LCVQE	0.02	0.05	0.10	0.05	0.06
	RDPM	0.03	0.11	**0.82**	0.14	**0.24**
	MEMOEAD	0.01	0.02	0.03	0.03	0.03
	MOECC	**0.25**	**0.22**	0.22	**0.24**	0.22
Breast cancer	COPKM	0.36	0.42	0.71	0.32	0.42
	LCVQE	0.16	0.32	0.75	0.21	0.33
	RDPM	0.20	0.32	**0.82**	0.22	0.34
	MEMOEAD	0.62	0.51	0.53	0.51	0.51
	MOECC	**0.78**	**0.70**	0.69	**0.71**	**0.70**
Diabetes	COPKM	0.10	0.10	**0.20**	0.07	0.10
	LCVQE	0.06	0.07	0.15	**0.08**	0.07
	RDPM	**0.17**	**0.11**	0.18	**0.08**	**0.11**
	MEMOEAD	0.12	0.07	0.08	0.07	0.07
	MOECC	0.13	0.07	0.07	**0.08**	0.07
Digits	COPKM	-1.00	-1.00	0.00	0.00	0.00
	LCVQE	0.45	0.59	0.56	0.65	0.59
	RDPM	0.19	0.55	**0.98**	0.46	0.62
	MEMOEAD	0.16	0.29	0.21	0.56	0.30
	MOECC	**0.75**	**0.78**	0.79	**0.79**	**0.79**
Divorce	COPKM	0.37	0.48	0.84	0.36	0.49
	LCVQE	0.65	0.66	0.88	0.54	0.66
	RDPM	0.49	0.51	**0.93**	0.38	0.54
	MEMOEAD	0.83	0.80	0.85	0.77	0.80
	MOECC	**0.92**	**0.88**	0.88	**0.88**	**0.88**

Tabla 4.4: Desempeño promedio de los métodos por conjunto de datos (continuación).

Datos	Método	ARI	AMI	Homogeneidad	Completitud	Medida V
Heart failure	COPKM	0.13	**0.11**	0.15	**0.09**	**0.11**
	LCVQE	0.03	0.04	0.11	0.04	0.05
	RDPM	**0.14**	0.10	**0.17**	0.08	**0.11**
	MEMOEAD	0.01	0.01	0.01	0.01	0.01
	MOECC	0.11	0.06	0.06	0.08	0.06
Ionosphere	COPKM	0.16	0.22	0.45	0.16	0.23
	LCVQE	0.16	0.12	0.13	0.12	0.13
	RDPM	0.17	0.20	**0.69**	0.16	0.26
	MEMOEAD	0.17	0.13	0.15	0.12	0.14
	MOECC	**0.33**	**0.26**	0.26	**0.28**	**0.27**
Iris	COPKM	0.48	0.59	0.74	0.54	0.60
	LCVQE	0.57	0.64	0.65	0.67	0.65
	RDPM	0.06	0.07	0.06	**1.00**	0.07
	MEMOEAD	0.55	0.68	0.59	0.86	0.68
	MOECC	**0.78**	**0.80**	**0.80**	0.82	**0.81**
Qsar biodeg	COPKM	0.17	**0.17**	0.19	**0.16**	**0.17**
	LCVQE	0.04	0.11	0.21	0.09	0.12
	RDPM	0.02	0.10	**0.40**	0.07	0.12
	MEMOEAD	0.09	0.12	0.13	0.11	0.12
	MOECC	**0.20**	0.13	0.13	0.13	0.13
Steel plates	COPKM	0.01	0.06	0.09	0.05	0.06
	LCVQE	0.02	0.10	0.18	0.08	0.11
	RDPM	0.06	0.18	**0.56**	0.11	0.19
	MEMOEAD	-0.02	0.06	0.06	0.07	0.06
	MOECC	**0.46**	**0.44**	0.54	**0.37**	**0.44**
Wine	COPKM	0.78	0.80	0.87	0.75	0.80
	LCVQE	0.76	0.78	0.87	0.72	0.78
	RDPM	0.72	0.75	**0.89**	0.66	0.76
	MEMOEAD	0.60	0.67	0.61	0.82	0.68
	MOECC	**0.83**	**0.83**	0.82	**0.84**	**0.83**

Tabla 4.4: Desempeño promedio de los métodos por conjunto de datos (continuación).

Datos	Método	ARI	AMI	Homogeneidad	Completitud	Medida V
Zoo	COPKM	-0.29	-0.24	0.36	0.32	0.34
	LCVQE	0.68	0.73	0.78	0.77	0.76
	RDPM	0.68	0.70	0.64	0.84	0.72
	MEMOEAD	0.52	0.60	0.49	0.91	0.62
	MOECC	**0.82**	**0.84**	**0.80**	**0.92**	**0.85**

4.2.4. Tamaño de grupos generados

Con respecto al número de grupos generados, la Tabla 4.5 muestra el valor obtenido en cada modelo por cada escenario de restricciones y por cada uno de los conjuntos de datos. Estos resultados corresponden al promedio obtenido de las 30 inicializaciones por cada escenario de restricciones.

A partir de los resultados, es posible observar que existen conjuntos de datos para los que se infiere el número esperado de grupos; sin embargo, su desempeño en las métricas de validación es bajo. Esto sugiere que las instancias no fueron agrupadas de una manera esperada. Un ejemplo de esta situación es RDPM en el conjunto de datos *BankNote* ya que a pesar de que en todos los escenarios acierta el número de grupos esperados (2), sus métricas de desempeño promedio son de 0.13.

Por otro lado, se observa que al comparar los grupos reales contra los predichos en los métodos con mayor desempeño (MOECC y RDPM), RDPM tiende a generar soluciones con un número elevado de grupos, alejándose mucho del valor real. Para el caso de MOECC, en los conjuntos de datos en los que obtiene medidas de desempeño inferior a 0.7 no obtiene el número exacto de grupos, aunque el número de grupos generalmente se encuentra a una unidad de distancia. Además, para los conjuntos de datos en que MOECC tiene un desempeño superior a 0.75 se encuentran soluciones que corresponden al número esperado de grupos.

Finalmente, se observa que para conjuntos de datos con un número alto de grupos y con un alto número de restricciones, COP-K-Medias no logra encontrar alguna solución. Como ejemplo los conjuntos de datos *Accent voice*, *Digits* y *Zoo*.

Tabla 4.5: Grupos generados promedio en cada conjunto de datos y por nivel de restricciones.

Datos	Modelo / Restr.	5 %	10 %	15 %	20 %	25 %	30 %	Valor real
Accent voice	COPKM	**6**	- -	- -	- -	- -	- -	6
	LCVQE	4	4	4	4	5	5	
	RDPM	5	**6**	**6**	**6**	**6**	7	
	MEMOEAD	2	3	2	2	2	2	
	MOECC	**6**	7	7	8	8	8	
BankNote	COPKM	8	6	10	7	5	8	2
	LCVQE	7	10	10	10	**2**	6	
	RDPM	**2**	**2**	**2**	**2**	**2**	**2**	
	MEMOEAD	3	**2**	**2**	5	**2**	3	
	MOECC	4	4	5	4	6	6	
Bench	COPKM	4	4	5	6	7	7	2
	LCVQE	7	6	6	6	6	7	
	RDPM	107	106	104	103	100	95	
	MEMOEAD	**2**	3	**2**	**2**	**2**	3	
	MOECC	**2**	**2**	**2**	**2**	**2**	**2**	
Breast cancer	COPKM	7	7	8	7	8	9	2
	LCVQE	14	14	14	14	14	14	
	RDPM	39	38	37	36	35	34	
	MEMOEAD	**2**	3	**2**	**2**	**2**	**2**	
	MOECC	**2**	**2**	**2**	**2**	**2**	**2**	
Diabetes	COPKM	8	9	12	12	10	11	2
	LCVQE	12	12	11	10	8	10	
	RDPM	8	8	8	8	8	8	
	MEMOEAD	3	**2**	**2**	**2**	**2**	**2**	
	MOECC	**2**	**2**	**2**	**2**	**2**	**2**	
Digits	COPKM	- -	- -	- -	- -	- -	- -	10
	LCVQE	12	12	12	12	11	13	
	RDPM	265	261	256	252	247	246	
	MEMOEAD	3	2	3	3	2	3	
	MOECC	**10**	**10**	**10**	**10**	**10**	**10**	
Divorce	COPKM	10	9	8	9	9	9	2
	LCVQE	5	5	5	4	4	5	
	RDPM	18	18	17	17	16	16	
	MEMOEAD	**2**	**2**	3	3	**2**	**2**	

Tabla 4.5: Grupos generados promedio en cada conjunto de datos y por nivel de restricciones (continuación).

Datos	Modelo / Restr.	5 %	10 %	15 %	20 %	25 %	30 %	Valor real
	MOECC	**2**	**2**	**2**	**2**	**2**	**2**	
Heart failure	COPKM	3	4	3	3	7	5	2
	LCVQE	10	10	9	9	10	10	
	RDPM	7	7	7	7	7	7	
	MEMOEAD	**2**	**2**	**2**	**2**	**2**	**2**	
	MOECC	**2**	**2**	**2**	**2**	**2**	**2**	
Ionosphere	COPKM	10	10	10	10	17	16	2
	LCVQE	**2**	3	**2**	**2**	**2**	**2**	
	RDPM	75	71	71	65	63	59	
	MEMOEAD	**2**	3	**2**	**2**	**2**	**2**	
	MOECC	**2**	**2**	**2**	**2**	3	3	
Iris	COPKM	6	6	7	6	10	11	3
	LCVQE	**3**	**3**	**3**	**3**	4	**3**	
	RDPM	1	1	1	1	1	1	
	MEMOEAD	2	2	2	**3**	**3**	**3**	
	MOECC	**3**	**3**	**3**	**3**	**3**	**3**	
Qsar biodeg	COPKM	**2**	**2**	13	15	14	19	2
	LCVQE	9	10	3	12	12	11	
	RDPM	100	98	101	99	95	95	
	MEMOEAD	**2**	**2**	6	**2**	3	5	
	MOECC	**2**	**2**	**2**	**2**	**2**	**2**	
Steel plates	COPKM	4	4	5	4	5	4	2
	LCVQE	7	7	7	7	6	5	
	RDPM	50	50	50	50	49	47	
	MEMOEAD	**2**	**2**	3	**2**	**2**	**2**	
	MOECC	3	3	3	3	3	3	
Wine	COPKM	4	4	4	5	5	5	3
	LCVQE	5	5	5	4	5	5	
	RDPM	6	6	6	6	6	6	
	MEMOEAD	2	2	**3**	**3**	2	2	
	MOECC	**3**	**3**	**3**	**3**	**3**	**3**	

Tabla 4.5: Grupos generados promedio en cada conjunto de datos y por nivel de restricciones (continuación).

Datos	Modelo / Restr.	5 %	10 %	15 %	20 %	25 %	30 %	Valor real
Zoo	COPKM	8	**7**	2	1	- -	- -	7
	LCVQE	8	8	8	8	8	8	
	RDPM	4	4	5	5	5	6	
	MEMOEAD	3	2	3	3	3	3	
	MOECC	3	5	6	6	6	**7**	

4.2.5. Proporción de restricciones violadas

Los métodos de agrupamiento semi-supervisado con restricciones buscan generar las particiones adecuadas en los conjuntos de datos incumpliendo el menor número de restricciones. La Figura 4.7 muestra las gráficas de caja asociadas con la proporción de restricciones que se incumplen en cada modelo, por cada nivel de restricciones. Para aquellos casos en que es posible encontrar soluciones factibles, COP-K-Medias presenta siempre una proporción promedio de cero debido a su propia naturaleza; RDPM muestra una asimetría muy positiva por lo que las restricciones no satisfechas se concentran en el cero, reporta un promedio de restricciones violadas de 8 %. LCVQE también muestra una asimetría positiva y su proporción de restricciones no satisfechas promedio es del 11 %. Por otra parte, ME-MOEA/D muestra una asimetría más cargada hacia la parte negativa y reporta un promedio de restricciones no satisfechas del 19 %. Finalmente, MOECC obtiene un promedio de restricciones no satisfechas de cero y es el segundo mejor método en cuanto al cumplimiento de las restricciones, después de COP-K-Medias.

Nótese que COP-K-Medias es muy estricto en el manejo de las restricciones, tal que si no es capaz de encontrar un agrupamiento que las satisfaga en su totalidad, entonces regresa una partición vacía. En cambio, el resto de los métodos relajan el manejo de las restricciones, permitiendo obtener agrupamientos que satisfacen el mayor número posible de estas restricciones.

En la Tabla 4.6 se muestra en detalle el porcentaje de restricciones a nivel de conjunto de datos y por escenario de restricciones.

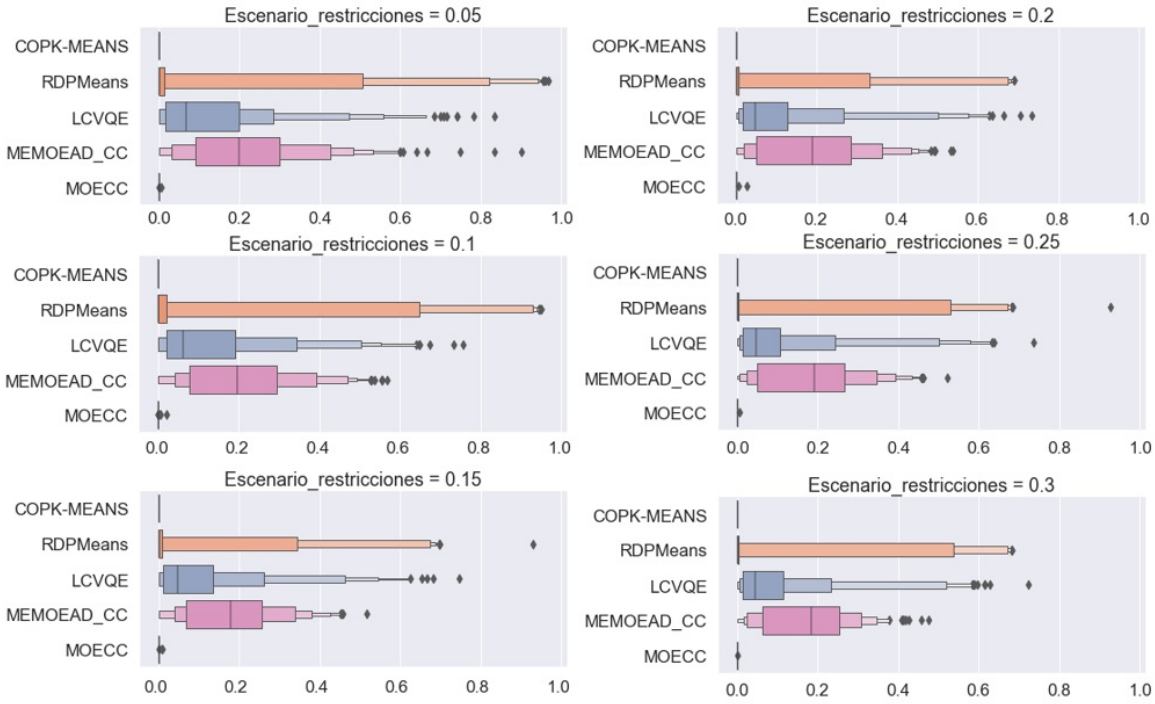


Figura 4.7: Proporción de restricciones violadas por escenario de restricciones.

Tabla 4.6: Proporción de restricciones que no son satisfechas en promedio por cada conjunto de datos y por nivel de restricciones.

Datos	Modelo / Restr.	5 %	10 %	15 %	20 %	25 %	30 %
Accent voice	COPKM	0.00	- -	- -	- -	- -	- -
	LCVQE	0.48	0.47	0.45	0.45	0.41	0.40
	RDPM	0.18	0.15	0.15	0.16	0.11	0.06
	MEMOEAD	0.41	0.35	0.34	0.34	0.34	0.31
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
BankNote	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.09	0.02	0.02	0.02	0.24	0.10
	RDPM	0.22	0.00	0.00	0.00	0.00	0.17
	MEMOEAD	0.13	0.22	0.24	0.06	0.23	0.20
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Bench	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.14	0.15	0.15	0.15	0.15	0.14
	RDPM	0.01	0.00	0.00	0.00	0.00	0.00
	MEMOEAD	0.33	0.28	0.29	0.29	0.26	0.27
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00

Tabla 4.6: Proporción de restricciones que no son satisfechas en promedio por cada conjunto de datos y por nivel de restricciones (continuación).

Datos	Modelo / Restr.	5 %	10 %	15 %	20 %	25 %	30 %
Breast cancer	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.01	0.01	0.01	0.01	0.01	0.01
	RDPM	0.00	0.00	0.00	0.00	0.00	0.00
	MEMOEAD	0.06	0.06	0.06	0.06	0.06	0.06
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Diabetes	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.06	0.05	0.07	0.10	0.08	0.10
	RDPM	0.00	0.00	0.00	0.00	0.00	0.00
	MEMOEAD	0.10	0.10	0.18	0.20	0.17	0.17
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Digits	COPKM	- -	- -	- -	- -	- -	- -
	LCVQE	0.11	0.11	0.10	0.11	0.12	0.07
	RDPM	0.00	0.00	0.00	0.00	0.00	0.00
	MEMOEAD	0.37	0.39	0.33	0.37	0.41	0.33
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Divorce	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.01	0.02	0.01	0.01	0.02	0.01
	RDPM	0.01	0.01	0.00	0.00	0.00	0.00
	MEMOEAD	0.02	0.02	0.02	0.01	0.02	0.02
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Heart failure	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.08	0.08	0.10	0.09	0.08	0.08
	RDPM	0.01	0.00	0.00	0.00	0.00	0.00
	MEMOEAD	0.20	0.20	0.21	0.19	0.19	0.21
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Ionosphere	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.19	0.18	0.19	0.19	0.19	0.18
	RDPM	0.01	0.00	0.00	0.00	0.00	0.00
	MEMOEAD	0.17	0.15	0.17	0.14	0.19	0.18
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00

Tabla 4.6: Proporción de restricciones que no son satisfechas en promedio por cada conjunto de datos y por nivel de restricciones (continuación).

Datos	Modelo / Restr.	5 %	10 %	15 %	20 %	25 %	30 %
Iris	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.10	0.10	0.10	0.10	0.09	0.10
	RDPM	0.61	0.62	0.63	0.63	0.62	0.63
	MEMOEAD	0.18	0.16	0.17	0.14	0.15	0.15
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Qsar biodeg	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.13	0.12	0.25	0.05	0.06	0.06
	RDPM	0.00	0.00	0.00	0.00	0.00	0.00
	MEMOEAD	0.13	0.17	0.10	0.16	0.15	0.14
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Steel plates	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.13	0.13	0.16	0.14	0.20	0.17
	RDPM	0.00	0.00	0.00	0.00	0.00	0.00
	MEMOEAD	0.19	0.26	0.24	0.29	0.27	0.31
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Wine	COPKM	0.00	0.00	0.00	0.00	0.00	0.00
	LCVQE	0.02	0.02	0.02	0.02	0.02	0.02
	RDPM	0.11	0.08	0.03	0.02	0.00	0.00
	MEMOEAD	0.16	0.17	0.15	0.14	0.17	0.16
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00
Zoo	COPKM	0.00	0.00	0.00	0.00	- -	- -
	LCVQE	0.04	0.04	0.05	0.05	0.05	0.06
	RDPM	0.13	0.11	0.08	0.03	0.04	0.03
	MEMOEAD	0.22	0.26	0.19	0.20	0.19	0.21
	MOECC	0.00	0.00	0.00	0.00	0.00	0.00

4.2.6. Tiempos de ejecución

La Tabla 4.2.6 muestra la mediana y el rango intercuartil (RIC) del tiempo de ejecución en los diferentes escenarios. Los métodos más eficientes en general son el RDPM, MOECC, COP-K-Medias y LCVQE, con una mediana general de 0.04, 0.14, 0.15 y 0.21 minutos, respectivamente; en general presentan una mediana pequeña, independientemente del número de restricciones. Además, RDPM y MOECC tienen rango intercuartil de 0.15, siendo el más bajo con respecto al resto de los métodos, mostrando compacidad en los tiempos de ejecución en el 50 % de los casos.

Por otro lado, ME-MOEA/D es el que más tiempo requiere, con una mediana en tiempo de 30.48 minutos y un RIC general de 42.34 minutos.

Tabla 4.7: Mediana y RIC de los tiempos de ejecución en minutos, por método y por nivel de restricciones.

Porcentaje	Medida	COPKM	LCVQE	RDPM	MEMOEAD	MOECC
5 %	Mediana	0.08	0.23	0.06	34.29	0.13
	RIC	0.63	0.37	0.28	87.56	0.17
10 %	Mediana	0.12	0.22	0.05	33.46	0.14
	RIC	0.70	0.45	0.21	57.29	0.21
15 %	Mediana	0.14	0.20	0.03	28.87	0.13
	RIC	0.65	0.27	0.12	37.55	0.11
20 %	Mediana	0.15	0.21	0.03	28.34	0.14
	RIC	0.65	0.32	0.11	32.70	0.12
25 %	Mediana	0.18	0.19	0.02	28.14	0.15
	RIC	0.57	0.29	0.09	18.51	0.12
30 %	Mediana	0.24	0.22	0.03	29.8	0.16
	RIC	0.59	0.23	0.10	20.44	0.15

Capítulo 5

Conclusiones y trabajo futuro

Con base en el algoritmo de búsqueda MOEA/D propuesto por Zhang y Li (2007) se ha hecho una adaptación que ha permitido dar solución al problema de agrupamiento semi-supervisado con restricciones a nivel instancias, tal propuesta recibe el nombre de MOECC (*Multi-Objective Evolutionary Constrained Clustering*).

MOECC considera una representación de los individuos de la población basada en etiquetas. Además, se adaptaron dos tipos de operadores evolutivos: una cruza basada en grupos, que busca conservar la información de los grupos potenciales provenientes de los padres, y una mutación basada en vecindario, dando la posibilidad de que instancias vecinas pertenezcan al mismo grupo. MOECC adopta una técnica de factibilidad por ϵ-restringido, la cual considera restricciones *Must-link* y *Cannot-link* y permite encontrar soluciones factibles al problema de agrupamiento, es decir, encontrar soluciones que satisfagan las restricciones.

Otro elemento clave en MOECC son las funciones objetivo, estas se basan en la compacidad y conectividad, las cuales permiten validar la calidad en los agrupamientos. Debido a que estas funciones son complementarias, le dan la ventaja a MOECC de balancear el número de particiones generadas de manera automática, la compacidad busca soluciones que contengan un número grande de grupos, mientras que la conectividad busca soluciones con pocos grupos. Adicionalmente, MOECC considera dos criterios de parada, por convergencia y de acuerdo a un número máximo de evaluaciones de las funciones objetivo. Se consideró que la convergencia se alcanza cuando no hay una mejora significativa en la aproximación a la frontera de Pareto. La medición de esta mejora se realiza a través de la medida del hiper-volumen.

Una de las primeras ventajas de MOECC es que al realizar una optimización multi-objetivo, una vez que se ha encontrado la mejor aproximación a la frontera de Pareto, se obtiene un conjunto de particiones óptimas, las cuales poseen diversos niveles de compromiso entre la compacidad y conectividad. MOECC explota la información de estas funciones e implementa una estrategia para escoger una solución de entre ese conjunto.

Con la finalidad de probar la eficacia de MOECC, se comparó su desempeño con cuatro métodos en el estado del arte, los cuales son COP-K-Medias, LCVQE, RDPM y ME-MOEA/D. Su desempeño se mide a partir de las métricas de validación externa AMI, ARI, homogeneidad, completitud y la medida V. La evaluación experimental consideró 14 conjuntos de datos comúnmente usados en la literatura especializada. Además, con el objetivo de medir el impacto de las restricciones en la calidad de los

agrupamientos, se han generado 6 escenarios considerando un 5 %, 10 %, 15 %, 20 %, 25 % y 30 % de restricciones por conjunto de datos.

De manera general, MOECC tiene el mejor desempeño promedio con respecto al resto de los métodos de referencia. En una vista a nivel de restricciones, MOECC conserva el mejor desempeño independientemente del porcentaje de restricciones, a excepción de tres escenarios en los cuales RDPM presenta el mejor desempeño en homogeneidad. Se logró observar que a mayor número de restricciones mejora el desempeño de los métodos en general.

Por otro lado, en una vista a nivel conjunto de datos, de los 14 conjuntos considerados, MOECC obtiene los desempeños más altos en 11 de ellos; en 6 de los conjuntos su desempeño general es superior a 0.75, lo cual indica que está generando particiones de calidad. En cuanto al nivel de restricciones no satisfechas en los 14 conjuntos de datos considerados y por cada escenario de restricciones, se observan casos atípicos donde el valor máximo de incumplimiento es de 0.03, sin embargo, al obtener la mediana de todos los conjuntos de datos por escenario, existe en promedio un 0 % de incumplimiento, independientemente del nivel de restricciones, es decir que se encuentra una solución factible para casi todos los escenarios. Finalmente, el número de grupos promedio en las soluciones igualan el número de clases originales para la mayoría de los conjuntos de datos, por lo que MOECC es muy robusto en estos casos. Los conjuntos de datos mejor agrupados poseen un alto número de clases y de variables.

Con respecto al costo computacional, MOECC ocupa la segunda posición, siendo RDPM el método más eficiente, sin embargo, la diferencia entre uno y otro es pequeña, además, a pesar de que RDPM es más rápido tiene un mayor nivel de infactibilidad.

En conjuntos de datos con un alto número de instancias y bajo número de clases, a pesar de que se obtiene el mejor desempeño en MOECC, la calidad en los grupos generados es baja pues en promedio es de 0.32. En estos conjuntos de datos RDPM se ve favorecido en la métrica de homogeneidad con desempeño promedio de 0.80. La desventaja que presenta RDPM es que en el resto de las métricas su desempeño suele ser muy bajo, así como su nivel de restricciones no satisfechas en general es alto y, en varias ocasiones, las soluciones que genera no se aproximan al número de clases del conjunto de datos, llegando incluso a generar particiones con un número elevado de grupos.

A partir del trabajo desarrollado, se han identificado algunas líneas de estudio futuro, las cuales se listan a continuación.

- En esta tesis se consideró un nivel de restricciones máximo del 30 %, observándose una mejoría al incrementar el número de restricciones. Este comportamiento tiene sentido, ya que el agrupamiento recibe más información acerca de si determinadas instancias deben o no estar juntas. No obstante, la ganancia en las métricas de evaluación tiende a ser menor conforme se incrementa el nivel de restricciones, así como la dificultad para encontrar una solución factible se incrementa. Por tanto, un camino interesante a explorar consiste en analizar el desempeño de los métodos propuestos con un porcentaje mayor de restricciones.

- En el estudio experimental realizado, las restricciones se generaron a partir de la información de las etiquetas de clase que se tiene de las instancias, definiéndose restricciones del tipo *Must-link* cuando dos intancias tienen la misma etiqueta

de clase y una restricción *Cannot-link* en caso contrario. No obstante, se ha asumido que las instancias están correctamente etiquetadas y, por tanto, las restricciones están libres de ruido. En este sentido, una línea a estudiar es el análisis y adaptación de MOECC a escenarios cuando puede existir ruido en las restricciones.

- En la evaluación experimental se consideró 14 conjuntos de datos de referencia. No obstante, en problemas del mundo real puede presentarse un desequilibrio entre los tamaños de grupos distintos. Los criterios de optimización adoptados en MOECC no consideran de manera directa la asimetría que puede existir en los grupos. Por tanto, evaluar la efectividad de MOECC a problemas donde la proporción de ejemplos pertenecientes a diferentes grupos es dispar es otro camino de estudio futuro.

- MOECC requiere de la definición de diferentes hiper-parámetros para su funcionamiento, entre los que destacan las tasas de cruza y mutación, el tamaño de la población, el tamaño de vecindario para la evaluación de la conectividad y la mutación. Analizar el impacto de los hiper-parámetros en la efectividad de MOECC es otra línea a explorar en trabajo futuro.

- Extender el estudio comparativo con otros enfoques meta-heurísticos para agrupamiento con restricciones. Por ejemplo, (González-Almagro, Luengo, *et al.*, 2021) proponen un enfoque mono-objetivo basado en evolución diferencial para resolver este problema. También, realizar una validación usando pruebas Bayesianas (Carrasco *et al.*, 2020; Benavoli *et al.*, 2017) es parte del análisis a explorar como trabajo futuro.

Referencias

Affenzeller, M., Wagner, S., Winkler, S., y Beham, A. (2009). *Genetic algorithms and genetic programming: modern concepts and practical applications*. CRC Press.

Arias, V., Salazar, J., Garicano, C., Contreras, J., Chacón, G., Chacín-González, M., ... Bermúdez-Pirela, V. (2019). Una introducción a las aplicaciones de la inteligencia artificial en medicina: Aspectos históricos. *Revista Latinoamericana de Hipertensión*, *14*(5), 590–600.

Balducci, F., Impedovo, D., y Pirlo, G. (2018). Machine learning applications on agricultural datasets for smart farm enhancement. *Machines*, *6*(3), 38 pág.

Ball, N. M., y Brunner, R. J. (2010). Data mining and machine learning in astronomy. *International Journal of Modern Physics D*, *19*(07), 1049–1106.

Basu, S., Banerjee, A., y Mooney, R. J. (2004). Active semi-supervision for pairwise constrained clustering. , 333–344.

Benavoli, A., Corani, G., Demšar, J., y Zaffalon, M. (2017). Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. *Journal of Machine Learning Research*, *18*(77), 1-36. Descargado de http://jmlr.org/papers/v18/16-305.html

Bradley, P. S., Bennett, K. P., y Demiriz, A. (2000). Constrained k-means clustering. *Microsoft Research, Redmond*, *20*(0), 0.

Carrasco, J., García, S., Rueda, M., Das, S., y Herrera, F. (2020). Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. *Swarm and Evolutionary Computation*, *54*, 100665. doi: 10.1016/j.swevo.2020.100665

Charrad, M., Ghazzali, N., Boiteau, V., y Niknafs, A. (2014). NbClust: an R package for determining the relevant number of clusters in a data set. *Journal of statistical software*, *61*, 1–36.

Chen, S.-C., Zhao, T., Gordon, G. J., y Murphy, R. F. (2006). A novel graphical model approach to segmenting cell images. , 1–8.

Coello Coello, C. A., Lamont, G. B., y Veldhuizen, D. A. V. (2007). *Evolutionary algorithms for solving multi-objective problems* (2nd ed.). Boston MA: Springer US. doi: 10.1007/978-0-387-36797-2

Cohn, D., Caruana, R., y McCallum, A. (2003). Semi-supervised clustering with user feedback. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, *4*(1), 17–32.

Darwin's, C. (1859). On the origin of species. *published on*, *24*.

Das, I., y Dennis, J. E. (1998). Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM journal on optimization*, *8*(3), 631–657.

Davidson, I., y Basu, S. (2007). A survey of clustering with instance level constraints. *ACM Transactions on Knowledge Discovery from data*, *1*(1-41), 2–42.

Davidson, I., y Ravi, S. (2005a). Agglomerative hierarchical clustering with constraints: Theoretical and empirical results. , 59–70.

Davidson, I., y Ravi, S. (2005b). Clustering with constraints: Feasibility issues and the k-means algorithm. , 138–149.

Davidson, I., y Ravi, S. (2007). The complexity of non-hierarchical clustering with instance and cluster level constraints. *Data mining and knowledge discovery*, *14*(1), 25–61.

de Siqueira, V. S., Borges, M. M., Furtado, R. G., Dourado, C. N., y da Costa, R. M. (2021). Artificial intelligence applied to support medical decisions for the automatic analysis of echocardiogram images: A systematic review. *Artificial Intelligence in Medicine*, *120*, 102165. doi: 10.1016/j.artmed.2021.102165

Gaffney, S. J., Robertson, A. W., Smyth, P., Camargo, S. J., y Ghil, M. (2007). Probabilistic clustering of extratropical cyclones using regression mixture models. *Climate dynamics*, *29*(4), 423–440.

González Almagro, G., Rosales-Pérez, A., Luengo, J., Cano, J.-R., y García, S. (2020). Improving constrained clustering via decomposition-based multiobjective optimization with memetic elitism. , 333–341.

González Almagro, G. (2018). Clustering con restricciones. un marco unificado. *Universidad de Granada*.

González-Almagro, G., Luengo, J., Cano, J.-R., y García, S. (2021). Enhancing instance-level constrained clustering through differential evolution. *Applied Soft Computing*, *108*, 107435. doi: 10.1016/j.asoc.2021.107435

González-Almagro, G., Rosales-Pérez, A., Luengo, J., Cano, J.-R., y García, S. (2021). ME-MEOA/DCC: multiobjective constrained clustering through decomposition-based memetic elitism. *Swarm and Evolutionary Computation*, *66*, 100939. doi: 10.1016/j.swevo.2021.100939

Handl, J., Knowles, J., y Kell, D. B. (2005). Computational cluster validation in post-genomic data analysis. *Bioinformatics*, *21*(15), 3201–3212.

Hartigan, J. A., y Wong, M. A. (1979). Algorithm as 136: A K-Means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*(1), 100–108.

Hubert, L., y Arabie, P. (1985). Comparing partitions. *Journal of classification*, *2*(1), 193–218.

Jain, A. K., y Moreau, J. (1987). Bootstrap technique in cluster analysis. *Pattern Recognition*, *20*(5), 547–568.

Jiang, K., Kulis, B., y Jordan, M. (2012). Small-variance asymptotics for exponential family dirichlet process mixture models. *Advances in Neural Information Processing Systems*, *25*, 3158–3166.

José-García, A., y Gómez-Flores, W. (2016). Automatic clustering using nature-inspired metaheuristics: A survey. *Applied Soft Computing*, *41*, 192-213. doi: 10 .1016/j.asoc.2015.12.001

Kan, A. (2017). Machine learning applications in cell image analysis. *Immunology and cell biology*, *95*(6), 525–530.

Khashabi, D., Wieting, J., Liu, J. Y., y Liang, F. (2015). Clustering with side information: From a probabilistic model to a deterministic algorithm. *arXiv preprint arXiv:1508.06235*.

Milligan, G. W., y Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, *50*(2), 159-179. doi: 10.1007/BF02294245

Min, S., Lee, B., y Yoon, S. (2017). Deep learning in bioinformatics. *Briefings in bioinformatics*, *18*(5), 851–869.

Nasraoui, O., y N'Cir, C.-E. B. (2019). Clustering methods for big data analytics. *Techniques, Toolboxes and Applications*, *1*, 91–113.

Pelleg, D., y Baras, D. (2007). K-means with large and noisy constraint sets. , 674–682.

Pétrowski, A., y Ben-Hamida, S. (2017). *Evolutionary algorithms*. John Wiley & Sons.

Romano, S., Bailey, J., Nguyen, V., y Verspoor, K. (2014). Standardized mutual information for clustering comparisons: one step further in adjustment for chance. , 1143–1151.

Romano, S., Vinh, N. X., Bailey, J., y Verspoor, K. (2016). Adjusting for chance clustering comparison measures. *The Journal of Machine Learning Research*, *17*(1), 4635–4666.

Rosenberg, A., y Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. , 410–420.

Seret, A., Verbraken, T., y Baesens, B. (2014). A new knowledge-based constrained clustering approach: Theory and application in direct marketing. *Applied Soft Computing*, *24*, 316–327.

Sun, J., Zhao, W., Xue, J., Shen, Z., y Shen, Y. (2010). Clustering with feature order preferences. *Intelligent Data Analysis*, *14*(4), 479–495.

Takahama, T., y Sakai, S. (2006). Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites. , 1–8.

Tang, B., Pan, Z., Yin, K., y Khateeb, A. (2019). Recent advances of deep learning in bioinformatics and computational biology. *Frontiers in genetics*, *10*, 214.

Tizón Galisteo, D. (2017). Big data clustering. *Universidad Nacional de Educación a Distancia (España). Escuela Técnica*.

Wagstaff, K., y Cardie, C. (2000). Clustering with instance-level constraints. *AAAI/IAAI*, *1097*, 577–584.

Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S., y cols. (2001). Constrained k-means clustering with background knowledge. , *1*, 577–584.

Yan, R., Zhang, J., Yang, J., y Hauptmann, A. G. (2006). A discriminative learning framework with pairwise constraints for video object classification. *IEEE transactions on pattern analysis and machine intelligence*, *28*(4), 578–593.

Zhang, Q. (2021). *Metaheuristic optimization, machine learning, and AI virtual workshop: "Introduction to Decomposition Based Multi-Objective Evolutionary Algorithms".* City University of Hong Kong. Descargado Fecha de consulta: 01/07/2021, de https://vimeo.com/522808115

Zhang, Q., y Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, *11*(6), 712–731.

Zhong, S., y Ghosh, J. (2003). A unified framework for model-based clustering. *The Journal of Machine Learning Research*, *4*, 1001–1037.

Zhou, A., Zhang, Q., y Zhang, G. (2012). A multiobjective evolutionary algorithm based on decomposition and probability model. En *2012 ieee congress on evolutionary computation* (pp. 1–8).

Apéndice A

Mecanismo MOECC

El algoritmo 5, describe de manera general a MOECC.

Durante la ejecución del algoritmo cada una de las generaciones se conforma por los siguientes elementos:

* Una población de N individuos que son solución actual de cada subproblema.
* La evaluación de las funciones objetivo en cada individuo de la población actual, $F(p^i)$, $i = 1, \ldots, N$.
* La proporción de restricciones violadas por cada uno de los individuos de la población actual, $ctr(p^i), i = 1, \ldots, N$.
* El conjunto de soluciones no dominadas EP encontradas hasta el momento, así como sus correspondientes individuos $EP\_individuals$ y el porcentaje de restricciones que violan.

A continuación se describe cada etapa del algoritmo:

Inicialización:

Los vecinos más cercanos de cada instancia en X se obtienen haciendo uso de la distancia euclídea entre pares de instancias.

A partir de la matriz de restricciones, se generan dos grupos por parejas de instancias, uno correspondiente a las restricciones *must-link* y otro a las *cannot-link*.

El cálculo de la compacidad y conectividad, máxima y mínima, se realizan de acuerdo a la descripción en Sección 3.4. Estos valores son de utilidad para el cálculo de normalización de las evaluaciones. EP se inicializa con los valores máximos de cada función; $EP\_individuals$ con un vector de ceros de tamaño n y $EP\_constrains$ con el valor 1. Estos tres elementos guardan los valores correspondientes a las soluciones no dominadas durante la ejecución del algoritmo. La variable vh guardará el valor del hiper-volumen por cada generación y se inicializa como vacío. $patience\_flag = 0$ será el contador de las generaciones consecutivas en las cuales el cambio en el hiper-volumen no supera el umbral $deta\_early\_stopping$.

Con el propósito de generar vectores de pesos bien distribuidos, se hace uso del método *Normal boundary intersection* (Das y Dennis, 1998). Además, para la generación de $\beta(i)$, que contiene los índices de los vecinos más cercanos del vector de pesos λ_i, se emplea la distancia euclídea.

Se genera la población inicial y para a cada uno de sus individuos se obtiene: su evaluación, la proporción de las restricciones que viola y se actualizan el conjunto de Pareto, la frontera de Pareto y el conjunto de restricciones violadas de este grupo.

Actualización de las generaciones:

Este proceso se actualización corresponde a una generación y se repite hasta que alguna de las condiciones de parada definidas en Sección 3.5 se cumpla.

Por cada uno de los individuos p^i de la población $i = 1, \ldots, N$: se obtienen p^a y p^b, donde a y b se seleccionan de manera aleatoria del conjunto de vecinos más cercanos de su vector de pesos λ^i; a partir de estos individuos se genera un nuevo descendiente p^{new}, aplicando el operador de cruza basada en grupos Subsección 3.2.1; posteriormente, a p^{new} se le aplica una mutación basada en vecindario Subsección 3.2.2, siempre y cuando se cumpla con las condiciones de este operador, además, se obtiene la evaluación y proporción de restricciones que viola.

Posteriormente, del grupo de vecinos más cercanos de su correspondiente vector de pesos $lambda^i$, cada vecino es reemplazado por p^{new} si este último es mejor. Para determinarlo se hace uso de la regla de factibilidad por $\epsilon-$restringido Sección 3.4. Si p^{new} es mejor que alguno de los vecinos, estos son reemplazados por p^{new}. Por último, se actualizan el conjunto de Pareto, la frontera de Pareto y el conjunto de se restricciones violadas, tomando en consideración los datos de p^{new}.

Al término de la evaluación de todos los individuos de la población, se calcula el hiper-volumen de la frontera de Pareto actual y se actualizan las variables involucradas para el criterio de paro. Sino se cumple alguno de los criterios de paro, comienza una nueva generación.

Selección de la solución final:

Una vez que ocurre alguno de los criterios de paro, a partir de las evaluaciones óptimas en la frontera de Pareto EP, se obtiene la solución final, siguiendo el método descrito en Sección 3.6.

Algoritmo 5 MOECC

Función MOECC(X, N, $constraint\_mat$, $max\_evaluations$, $n\_neighbors$, $neighborhood\_size$, $crossover$, $mutation$, $crossover\_rate$, $mutation\_rate$, $delta\_early\_stopping$, $patience\_early\_stopping$.)

Generar los vecinos más cercanos de cada instancia en X.
Calcular min_min = [Mínima compacidad de X, Mínima conectividad en X].
Calcular max_max = [Máxima compacidad de X, Máxima conectividad en X].
$EP = max\_max$.
$EP\_individuals = \vec{0}$.
$\_EP\_constraints = 1$.
$hv = \emptyset$.
$patience\_flag = 0$.
Inicializar los vectores de pesos λ_i, $i = 1, \ldots, N$.
Obtener $\beta(i)$, los índices de los vecinos más cercanos de λ_i, $i = 1, \ldots, N$.
Inicializar la población.
Para cada individuo p^i de la población inicial **hacer:**
  Obtener su evaluación, $F(p^i)$.
  Calcular la proporción de restricciones que viola, $ctr(p^i)$.
  Actualizar EP, $EP\_individuals$ y $\_EP\_constraints$.
Fin Para
$evals$=N.
Repetir:
  Para $i = 1$ hasta N **hacer:**
    Selección aleatoria de p^a y p^b con base en $\beta(i)$.
    Generar al individuo p^{new} al aplicar el operador de cruza sobre p^a y p^b.
    Mutar p^{new} al aplicar mutación por vecinos más cercanos.
    Obtener la evaluación de p^{new}, $F(p^{new})$.
    Calcular la proporción de restricciones que viola p^{new}, $ctr(p^{new})$.
    $evals = evals + 1$.
    Para cada $j \in \beta(i)$ **hacer:**
      Evaluar si p^{new} es mejor que p^j mediante la regla de
        factibilidad por $\epsilon-$restringido.
      Si p^{new} es mejor **entonces:**
        Reemplazar $p^j = p^{new}$, $F(p^j) = F(p^{new})$, $ctr(p^j) = ctr(p^{new})$.
      Fin Si
    Fin Para
    Actualizar EP, $EP\_individuals$ y $\_EP\_constraints$, al considerar
      la evaluación de p^{new}.
  Fin Para
  Calcular el hipervolumen de EP y actualizar $patience\_flag$.
Hasta que: ($patience\_flag \geq patience\_early\_stopping$) ó
      ($evals = max\_evaluations$)
Selección de la solución final, p^{best}; su evaluación, $F(p^{best})$ y
  la proporción de restricciones que viola, $\_EP\_constraints(p^{best})$.
Regresa $p^{best}, F(p^{best})$ y $\_EP\_constraints(p^{best})$.
Fin Función

Apéndice B

Algoritmo *Constrained Vector Quantization Error (CVQE)*

A continuación de describe brevemente las propiedades que se consideran de mayor relevancia en el método CVQE y sus adaptaciones, para poder obtener el algoritmo LCVQE (2.5.2).

Este algoritmo es propuesto en (Pelleg y Baras, 2007) y modifica la función objetivo del algoritmo K-Medias tradicional al incluir una penalización cuando se viola alguna restricción. Su propósito es mejorar el agrupamiento resultante cuando se tiene un conjunto de datos y restricciones de grandes dimensiones.

El algoritmo K-Medias se caracteriza porque en cada iteración, sus centros se van recalculando mediante un promedio de todas las instancias asignadas al grupo asociado al centro v_i, de la siguiente manera:

$$v_i = \frac{1}{\mid c_i \mid} \sum_{x_i \in c_i} x_i,$$

Una vez actualizados los centros, se reasignan las instancias de forma que cada una esté asociada al grupo más cercano. De esta regla de actualización se deriva una función de minimizar que se conoce como *Vector Quantization Error (VQE)*:

$$VQE = \frac{1}{2} \sum_{j=1}^{K} \sum_{x_i \in c_i} (v_j - x_i)^2.$$

El método CVQE incorpora un término de penalización de acuerdo a las restricciones incumplidas. Antes de mostrar su regla de actualización de los centros, considere la siguiente notación: sea M la función que regresa el índice del grupo que contiene a la instancia $x \in X$, tal que, $M = \{j \mid x \in c_j\}$. Sean $g(i) = M(x_1(i))$ y $g'(i) = M(x_2(i))$. Además, se define como $h(i)$ a la función que regresa el centro más cercano al centro v_i. Finalmente, sea $vl(i)$ la variable que indica si la restricción i-ésima ha sido violada, de tal manera que, $vl(i) = 1$ si y sólo si $g(i) \neq g'(i)$ para $i = 1, \ldots, \mid ML \mid$. De forma análoga, $vl(i) = 1$ si y sólo si $g(i) = g'(i)$, para $i = \mid ML \mid +1, \ldots, \mid R \mid$.

La regla de actualización de los centros bajo el método CVQE es la siguiente:

$$v_j = \frac{1}{N_j}\left[\sum_{x_i \in c_j} x_i + \sum_{l=1, g(l)=j}^{|ML|} vl(l) v_{g'(l)} + \sum_{l=|ML|+1, g(l)=j}^{|R|} vl(l) v_{h(g'(l))}\right],$$

donde N_j se define como:

$$N_j = \mid c_j \mid + \sum_{l=1, g(l)=j}^{|R|} vl(l).$$

La manera en que trabaja esta actualización es que, para las restricciones ML que se incumplen, uno de los dos centros afectados se mueve hacia el otro. Para el caso de las restricciones CL incumplidas, se mueve una de las instancias hacia el siguiente centro más cercano a su grupo.

Una vez actualizados los centros, cada instancia se reasigna después de cada iteración para minimizar la función de error $CVQE = \sum_{j=1}^{K} CVQE_j$, donde $CVQE_j$ se define de la siguiente manera:

$$CVQE_j = \frac{1}{2}\sum_{x_i \in c_j} T_{j,1} + \frac{1}{2}\sum_{l=1, g(l)=j}^{|ML|} T_{j,2} + \frac{1}{2}\sum_{l=|ML|+1, g(l)=j}^{|R|} T_{j,3},$$

con $T_{j,1}$, $T_{j,2}$, $T_{j,3}$ definidos como:

$$\begin{aligned} T_{j,1} &= (v_j - x_i)^2, \\ T_{j,2} &= \left[(v_j - v_{g'(l)})^2 \cdot vl(l)\right], \\ T_{j,3} &= \left[(v_j - v_{h(g'(l))})^2 \cdot vl(l)\right]. \end{aligned}$$

Como características a considerar en este método se tiene que un grupo c_i puede contener instancias para las que el centro v_i no es necesariamente el más cercano a ella. Además, el orden en que se especifican las instancias implicadas en las restricciones sí importa. Computacionalmente hablando, este método es costoso para grandes conjuntos de datos o restricciones, ya que determinar la asignación que minimiza la función de error requiere $\mathcal{O}(k^2)$ cálculos para cada restricción. Finalmente, el término de penalización de restricciones sólo depende de la distancia entre los centros implicados en la restricción.

Apéndice C

Métricas de evaluación

Permiten medir la calidad en agrupamientos y principalmente se tienen dos categorías, las métricas de validación interna y las métricas de validación externa.

Considere las siguientes variables:

$n =$ Número de observaciones,
$p =$ Número de variables,
$k =$ Número de grupos,
$X = \{x_{ij}\},\ i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, p,$ la matriz de datos con n instancias y p variables,
$\bar{x} =$ Centros de la matriz de datos X,
$C = \{c_1, \ldots c_k\}$, una partición con k grupos,
$n_k =$ Número de instancias en el grupo c_k,
$\mu_k =$ Centros del grupo c_k,
$x_i =$ Vector de observaciones p-dimensional de la i-ésima instancia en el grupo c_k,

$$W_k = \sum_{k=1}^{k} \sum_{i \in c_k} (x_i - \mu_k)(x_i - \mu_k)^T,$$

C.1. Métricas de validación interna

Las métricas de validación interna se caracterizan porque miden el agrupamiento únicamente basadas en la información de los datos. En este apartado se describe la métrica de validación interna Hartigan, la cual se usa como función auxiliar para encontrar el mejor tamaño de grupos en los algoritmos de agrupamiento. Lo que se intenta conseguir son grupos que tengan instancias muy similares entre sí, y que a su vez sean lo más diferentes con respecto al resto de las instancias.

C.1.1. Índice Hartigan

El índice Hartigan se calcula mediante la siguiente ecuación (Charrad *et al.*, 2014):

$$\text{Hartigan} = \left(\frac{trace(W_k)}{trace(W_{k+1})} - 1 \right) (n - k - 1), \tag{C.1}$$

Donde $k \in \{1, \ldots, n-1\}$. La máxima diferencia entre niveles jerárquicos se toma como el número correcto de grupos en el conjunto de datos, es decir, el mejor número de grupos será aquel k en el cual se produce un mayor salto entre el índice Hartigan para el número de grupos anterior y el índice Hartigan para el número de grupos actual (Tizón Galisteo, 2017).

C.2. Métricas de validación externa

Estas métricas miden la calidad en los agrupamientos y deben su nombre a que requieren de información externa para la validación, es decir, información proveniente de algún método de agrupamiento. A continuación se describen cinco métricas.

C.2.1. ARI *(Adjusted Rand Index)*

Este índice es un ajuste al índice *Rand*, que es un medida de similaridad entre las dos particiones y mide la proporción de pares de instancias correctamente clasificadas. Hubert y Arabie (1985) define el índice *Rand* del modo siguiente:

$$RI = \frac{a+b}{\binom{n}{2}},$$

donde a es la cardinalidad de pares de instancias que están en el mismo grupo en ambas particiones; b la cardinalidad de pares de instancias que están en distintos grupos en ambas particiones y n el número de instancias del conjunto de datos.

Rand toma valores en [0,1]. Una de su desventajas es que no garantiza que en agrupamientos con etiquetas generadas de manera aleatoria se obtenga un valor cercano a cero. ARI hace una corrección sobre *Rand* normalizándola del siguiente modo:

$$ARI = \frac{RI - \mathbf{E}[RI]}{Max(RI) - \mathbf{E}[RI]}.$$

ARI entonces mide la correspondencia entre etiquetas de clase de dos agrupamientos. Su rango de valores está comprendido entre [-1,1]. Un valor muy cercano a 1 es indicador de que los grupos son idénticos, mientras que un valor cercano a -1 indica poca concordancia en las etiquetas. Romano *et al.* (2016) propone usar ARI cuando los agrupamientos de referencia contienen un número alto de grupos.

C.2.2. AMI *(Adjusted Mutual Information)*

La información mutua (MI) mide el valor de información que comparten dos variables aleatorias, es decir, mide la reducción en la incertidumbre o entropía de una de ellas dado el conocimiento de la otra. Sean C_1 y C_2 dos agrupamientos. De acuerdo a Romano *et al.* (2014), MI se puede calcular del siguiente modo:

$$MI(C_1, C_2) = H(C_1) + H(C_2) - H(C_1, C_2) = \sum_{i=1}^{l}\sum_{j=1}^{k} \frac{n_{ij}}{n} log\left(\frac{n_{ij}n}{n_i n_j}\right),$$

donde $H(C_1)$ y $H(C_2)$ son las entropías correspondiente a cada agrupamiento; $H(C_1, C_2)$ la entropía conjunta; n_{ij} el número de instancias compartidas por el grupo i y j; l y k el número de grupos de la partición C_1 y C_2, respectivamente.

AMI normaliza a MI de la siguiente forma:

$$AMI = \frac{MI - \mathbf{E}[MI]}{\sqrt{H(C_1)\dot{H}(C_2)} - \mathbf{E}[MI]}.$$

Su rango de valores esta comprendido entre [-1,1]. Si los etiquetados en dos particiones son independientes, AMI toma un valor cercano -1. Si las asignaciones de las etiquetas son aleatorias, AMI toma una puntuación cercana a 0. Si ambas particiones son idénticas, AMI toma valores cercanos a 1. Romano *et al.* (2016) propone usar AMI cuando los agrupamientos de referencia son desequilibrados y además contienen un número bajo de grupos.

C.2.3. Homegeneidad

Esta métrica se basa en una medida de entropía condicional normalizada. Condiciona el etiquetado del agrupamiento a evaluar, dado el conocimiento de las etiquetas de clase reales del mismo conjunto de datos. De acuerdo a Rosenberg y Hirschberg (2007) se calcula mediante la siguiente fórmula:

$$H = 1 - \frac{H(C \mid K)}{H(C)},$$

donde,

$$H(C \mid K) = -\sum_{c,k} \frac{n_k^c}{n} log\left(\frac{n_k^c}{n_k}\right).$$

donde, n_k^c representa el número de instancias etiquetadas con c dentro del grupo k. La homogeneidad toma valores en el intervalo [0,1]. Cuando todas las instancias en el grupos K tienen la misma etiqueta de clase c, la homogeneidad es 1, por lo que la homogeneidad se basa en la premisa de que cada grupo contiene solo instancias de una sola clase.

C.2.4. Medida de completitud

Esta métrica se basa en una medida de entropía condicional normalizada. Condiciona el conocimiento de las etiquetas de clase reales, dado el etiquetado del agrupamiento a evaluar del mismo conjunto de datos. De acuerdo a Rosenberg y Hirschberg (2007) se calcula mediante la siguiente fórmula:

$$C = 1 - \frac{H(K \mid C)}{H(K)},$$

donde,

$$H(K \mid C) = -\sum_{c,k} \frac{n_k^c}{n} log\left(\frac{n_k^c}{n^c}\right).$$

donde, n_k^c representa el número de instancias etiquetadas con c dentro del grupo k y n^c el número de instancias etiquetadas con c. La completitud toma valores en el intervalo [0,1]. Cuando todas las instancias con etiqueta real c son asignadas en el mismo grupo k, la completitud vale 1. La completitud se basa en la premisa de que todas las instancias con misma etiqueta deben pertenecer al mismo grupo.

Completitud no implica homogeneidad, sin embargo, un agrupamiento con valores altos en ambas medidas es deseable para considerar un agrupamiento de buena calidad.

C.2.5. Medida V

La medida V es la media armónica entre la homogeneidad y la completitud. De acuerdo con Rosenberg y Hirschberg (2007), la medida V se calcula del modo siguiente:

$$V = 2\frac{H \cdot C}{H + C}.$$

Su rango de valores es [0,1] y toma el valor de 1 cuando la homogeneidad y la completitud son 1, mientras que toma el valor de 0 a medida que alguna de las dos medidas no sea satisfecha.