



TRABAJO FIN DE GRADO  
INGENIERÍA INFORMÁTICA

# Clustering Con Restricciones

---

Un Marco Unificado

**Autor**

Germán González Almagro

**Directores**

Salvador García López  
Julián Jesús Luengo Martín



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE  
TELECOMUNICACIÓN

—  
Granada, mes de 201



# **Clustering Con Restricciones**

---

**Un Marco Unificado.**

**Autor**

Germán González Almagro

**Directores**

Salvador García López  
Julián Jesús Luengo Martín



# **Clustering Con Restricciones: Un Marco Unificado**

Germán González Almagro

**Palabras clave:** palabra\_clave1, palabra\_clave2, palabra\_clave3, .....

## **Resumen**

Poner aquí el resumen.



**Project Title: Project Subtitle**

First name, Family name (student)

**Keywords:** Keyword1, Keyword2, Keyword3, ....

**Abstract**

Write here the abstract in English.



---

Yo, **Germán González Almagro**, alumno de la titulación en ingeniería informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 76593910T, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Germán González Almagro

Granada a X de mes de 201 .



---

**D. Salvador García López**, Profesor del Departamento Ciencias de la Computación en Inteligencia Artificial de la Universidad de Granada.

**D. Julián Jesús Luengo Navas**, Profesor del Departamento Ciencias de la Computación en Inteligencia Artificial de la Universidad de Granada.

**Informan:**

Que el presente trabajo, titulado *Clustering Con Restricciones, Un Marco Unificado*, ha sido realizado bajo su supervisión por **Germán González Almagro**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a X de mes de 201 .

**Los directores:**

**Salvador García López      Julián Jesús Luengo Navas**



## AGRADECIMIENTOS

---

Poner aquí agradecimientos...



## ÍNDICE GENERAL

---

<b>1</b>	<b>INTRODUCCIÓN</b>	<b>1</b>
1.1	Motivación Personal . . . . .	2
1.2	Objetivos . . . . .	2
<b>2</b>	<b>BREVE INTRODUCCIÓN AL CLUSTERING</b>	<b>3</b>
2.1	Motivación para la clasificación . . . . .	3
2.2	Métodos numéricos para el Clustering . . . . .	3
2.2.1	¿Qué es un cluster? . . . . .	5
2.3	Aplicaciones del clustering . . . . .	7
2.3.1	Aplicaciones en marketing . . . . .	7
2.3.2	Aplicaciones en astronomía . . . . .	8
2.3.3	Aplicaciones en psiquiatría . . . . .	8
2.3.4	Aplicaciones en meteorología y climatología . .	9
2.3.5	Aplicaciones en arqueología . . . . .	9
2.3.6	Aplicaciones en bioinformática y genética . . .	10
2.4	Resumen . . . . .	11
<b>3</b>	<b>CLUSTERING CON RESTRICCIONES</b>	<b>13</b>
3.1	Motivación para el clustering con restricciones . . . . .	13
3.2	Definición de las restricciones . . . . .	14
3.3	Uso de las restricciones . . . . .	14
3.3.1	Métodos basados en restricciones . . . . .	15
3.3.2	Métodos basados en distancia . . . . .	16
3.4	Aplicaciones del clustering con restricciones . . . .	17
3.4.1	Aplicaciones en análisis de imágenes . . . . .	17
3.4.2	Aplicaciones en análisis de vídeos . . . . .	19
3.4.3	Aplicaciones en genética . . . . .	19
3.4.4	Aplicaciones en análisis de textos . . . . .	20
3.4.5	Aplicaciones en datos web . . . . .	20
3.4.6	Aplicaciones en datos de audio . . . . .	21
3.4.7	Aplicaciones en datos de GPS . . . . .	21
3.5	Beneficios del uso de restricciones . . . . .	22
3.6	Problemas del uso de restricciones . . . . .	23
3.6.1	El problema de la factibilidad . . . . .	23
3.6.2	El problema de la utilidad de conjuntos de res- tricciones . . . . .	24
3.6.3	Soluciones al problema de la factibilidad . . . .	25
3.6.4	Soluciones al problema de la utilidad de con- juntos de restricciones . . . . .	26
3.7	Resumen . . . . .	27
<b>4</b>	<b>ALGORITMOS DE CLUSTERING CON RESTRICCIONES</b>	<b>29</b>
4.1	Formalización del problema . . . . .	29
4.2	COP-K-means (Constrained K-means) . . . . .	29
4.3	CEKM (Constrained Evidential K-means) . . . . .	31

4.3.1	Funciones de creencia . . . . .	31
4.3.2	FKM (fuzzy K-means) y sus variantes . . . . .	32
4.3.3	El algoritmo EKM (Evidential K-means) . . . . .	33
4.3.4	Incorporación de restricciones a EKM . . . . .	35
4.3.5	Función objetivo de CEKM . . . . .	36
4.3.6	Proceso de optimización de CEKM . . . . .	36
4.4	Linear Constrained Vector Quantization Error (LCVQE)	38
4.4.1	El algoritmo CVQE . . . . .	38
4.4.2	El algoritmo LCVQE . . . . .	41
4.5	Relational Dirichlet Process - Means (RDP - Means) . .	44
4.5.1	El modelo Bayesiano no paramétrico . . . . .	44
4.5.2	Reparametrización de la familia exponencial pa- ra RDP-means . . . . .	48
4.5.3	Escalando las distribuciones para RDP-means .	49
4.5.4	Asintóticos de TVClust . . . . .	50
4.5.5	Muestreo de los parámetros de los clusters . . . .	50
4.5.6	Función objetivo de RDP-means . . . . .	51
4.5.7	Efectos de los cambios sobre $\xi_1$ y $\xi_2$ . . . . .	51
4.6	El algoritmo RDP-means . . . . .	51
5	IMPLEMENTACIÓN	53
5.1	Entorno de desarrollo . . . . .	53
5.2	Bibliotecas empleadas en el desarrollo . . . . .	53
5.3	Funcionalidades desarrolladas . . . . .	54
5.3.1	Función para generar restricciones . . . . .	54
5.3.2	Función para aplicar COP-K-Medias . . . . .	55
5.3.3	Función para aplicar CEKM . . . . .	56
5.3.4	Función para aplicar LCVQE . . . . .	56
5.3.5	Función para aplicar RDPM . . . . .	57
5.3.6	Función para aplicar TVClust . . . . .	58
6	EXPERIMENTACIÓN	59
A	EL ALGORITMO K-MEDIAS	61
	BIBLIOGRAFÍA	63

## ÍNDICE DE FIGURAS

---

Figura 2.1	Clusters con cohesión interna y/o aislamiento externo . . . . .	6
Figura 2.2	Distribución uniforme de puntos . . . . .	6
Figura 2.3	Distribución uniforme de puntos clasificados .	6
Figura 3.1	Restricciones de tipo delta y epsilon. . . . .	15
Figura 3.2	Restricciones sobre un conjunto de datos. . . .	16
Figura 3.3	Clustering que satisface todas las restricciones.	16
Figura 3.4	Restricciones sobre un conjunto de datos. . . .	17

Figura 3.5	Clustering basado en métrica aprendida en base a las restricciones. . . . .	17
Figura 3.6	Caras de la base de datos de CMU. . . . .	18
Figura 3.7	Restricciones de tipo Cannot-Link (CL) entre caras de la misma persona. . . . .	18
Figura 3.8	Método de clustering empleado en el sistema de navegación del robot Aibo. . . . .	18
Figura 3.9	Diferentes tipos de restricciones en datos de video. . . . .	19
Figura 3.10	Clustering de genes basado en microarrays. . .	20
Figura 3.11	Uso de información GPS. . . . .	21
Figura 3.12	Clusters encontrados en datos GPS sin uso de restricciones. . . . .	22
Figura 3.13	Ejemplo de conjunto de restricciones informativo. . . . .	26
Figura 3.14	Ejemplo de conjunto de restricciones contradictorio. . . . .	27
Figura 3.15	Representación de la medida de coherencia. . .	27
Figura 4.1	Ejemplo de CVQE. . . . .	40
Figura 4.2	Ejemplo de LCVQE. . . . .	40

## ÍNDICE DE TABLAS

---

Tabla 3.1	Complejidad del problema del clustering en función del tipo de restricciones. . . . .	24
Tabla 4.1	Ejemplo de partición de creencia . . . . .	34
Tabla 4.2	Valores de CVQE para el ejemplo . . . . .	40
Tabla 4.3	Valores de LCVQE para el ejemplo . . . . .	42

## ÍNDICE DE ALGORITMOS

---

Algoritmo 1	COP-K-medias . . . . .	30
Algoritmo 2	Constrained Evidential K-means (CEKM) . . .	37
Algoritmo 3	Linear Constrained Vector Quantization Error (LCVQE) . . . . .	43
Algoritmo 4	Relational Dirichlet Process - Means (RDPM) .	52
Algoritmo 5	K-medias . . . . .	61

## ACRONYMS

---

ML	Must-Link
CL	Cannot-Link
KM	K-medias
URL	Uniform Resource Locator
GPS	Global Positioning System
CEKM	Constrained Evidential K-means
FKM	Fuzzy K-means
EKM	Evidential K-means
NC	Noise Clustering
VQE	Vector Quantization Error
CVQE	Constrained Vector Quantization Error
LCVQE	Linear Constrained Vector Quantization Error
RDPM	Relational Dirichlet Process - Means
DPM	Dirichlet Process - Means
TVClust	Two Views Clustering

## INTRODUCCIÓN

---

*An intelligent being cannot treat every object it sees as unique entity unlike anything else in the universe. It has to put objects in categories so that it may apply its hard-won knowledge about similar objects encountered in the past, to the object at hand.*

**Steven Pinker, How the Mind Works, 1997**

Una de las habilidades más básicas y primitivas de la que están dotados los seres humanos es la de agrupar objetos similares para producir una clasificación que les resulte útil. Habilidad que ya nuestros más antiguos ancestros debieron poseer, por ejemplo, debieron ser capaces de darse cuenta de qué objetos eran comestibles, cuales eran venenosos y cuales intentarían matarles.

La capacidad de clasificación, en su sentido mas amplio, es necesaria para el desarrollo del lenguaje, que esta formado por palabras que nos ayudan a reconocer diferentes tipos de eventos, acciones y entidades. En esencia, cada sustantivo es una etiqueta que empleamos para agrupar un colectivo de seres u objetos con características similares, de manera que podemos hacer referencia a todos ellos empleando la palabra que los une.

De igual forma que la clasificación es una habilidad básica para las personas en su vida cotidiana, es también esencial en la mayoría de las ramas de la ciencia. En biología, por ejemplo, la clasificación de los diferentes tipos de organismos ha sido objeto de estudio desde el comienzo de su existencia. Aristóteles construyó un elaborado sistema de clasificación animal que dividía a todas las criaturas en dos grupos, aquellos con sangre roja y aquellos que carecían de ella. Más tarde propuso una subdivisión que los clasificaba según la forma en la que nuevos individuos venían al mundo, ya sea vivos, mediante huevos, crisálidas, etc.

Siguiendo a Aristóteles, Teoprastos escribió el primer documento que recopilaba las directrices para la clasificación de las plantas. Los trabajos resultantes fueron tan amplios y detallados que sentaron las bases para la investigación en biología durante los siguientes siglos. Este trabajo no fue sustituido hasta 1737, cuando Carlos Linneo publicó su trabajo *Genera Plantarum*, del que el siguiente fragmento ha sido extraído:

*All the real knowledge which we possess, depends on methods by which we distinguish the similar from the dissimilar. The greater the number of natural distinctions this method comprehends the clearer becomes our idea of things. The more nume-*

*rous the objects which employ our attention the more difficult it becomes to form such a method and the more necessary. For we must not join in the same genus the horse and the swine, though both species had been one hoof'd nor separate in different genera the goat, the reindeer and the elk, tho' they differ in the form of their horns. We ought therefore by attentive and diligent observation to determine the limits of the genera, since they cannot be determined a priori. This is the great work, the important labour, for should the genera be confused, all would be confusion.*

Carlos Linneo, *Genera Plantarum*, 1737

La clasificación de los animales y las plantas ha sido de gran importancia en campos como la biología y la zoología. Particularmente, esta clasificación sentó las bases para el desarrollo de la teoría de la evolución de Darwin. Pero también ha sido de gran relevancia en áreas de conocimiento como la química y la física, con la clasificación de los elementos en la tabla periódica, propuesta por Mendeleev en la década de 1860; o en astronomía, con la clasificación de estrellas en enanas o gigantes empleando las directrices de Hertzsprung–Russell.

#### MOTIVACIÓN PERSONAL

Durante mi formación he escuchado de multitud de profesores que incorporar conocimiento humano a una máquina es una de las tareas más complejas a las que se ha enfrentado la humanidad.

Algo que realmente me resultó interesante fue que Deep Blue, la primera máquina en ganar al campeón del mundo de ajedrez, Gary Kasparov en aquel momento (1996), no ganó por un avance significativo en el algoritmo que ejecutaba la máquina, sino por avances en el hardware, que permitían que la máquina analizase más movimientos por unidad de tiempo, es decir, la máquina no empleaba conocimiento del que no disponía anteriormente, simplemente “pensaba” más rápido.

Por ello, he querido profundizar en el campo de la incorporación de conocimiento a las máquinas algo más de lo que he podido hacerlo durante estos años. El clustering con restricciones puede verse como un ejemplo de ello, al fin y al cabo no es más que guiar el proceso de toma de decisiones de una máquina incorporando conocimiento extraído de las personas.

#### OBJETIVOS

# 2

## BREVE INTRODUCCIÓN AL CLUSTERING

---

En primer lugar, será necesario realizar una introducción, siquiera sea breve, al clustering y sus aplicaciones, que facilite la comprensión de las siguientes secciones. A tal fin se seguirá el estudio realizado por Brian S. Everitt et al. (2011) [1].

### MOTIVACIÓN PARA LA CLASIFICACIÓN

La clasificación puede ser entendida como una forma de simplificar la información contenida en grandes conjuntos de datos, de un modo que sea fácilmente comprensible por las personas. De esta manera, los procesos de extracción de información útil y aplicación de la misma se simplifican. Así, si somos capaces de dividir de forma válida un gran conjunto de datos en subconjuntos o grupos, podremos extraer información común a todos los elementos del subconjunto y proporcionar una descripción precisa que los englobe.

La necesidad de analizar la información de esta manera crece con la aparición y disponibilidad de grandes conjuntos de datos en el ámbito de la ciencia. El análisis de este tipo de información mediante clasificación, o clustering, hoy en día es conocido como *Ciencia de datos*. En el siglo XXI surge un particular interés en la ciencia de datos desde la aparición de la *World Wide Web*, conocida como Internet, donde el objetivo se ha convertido en extraer información relevante de las páginas Web que forman esta vasta red.

Es importante destacar que en la mayoría de las ocasiones no hay un sólo criterio de clasificación para un mismo conjunto de datos, sino que existe una amplia variedad de los mismos. En el caso de las personas, podrían ser clasificadas, por ejemplo, en base a sus ingresos económicos, o según la cantidad de calorías que consumen a lo largo de un periodo de tiempo definido. Así, distintos criterios de clasificación no tienen por qué dar como resultado la misma división en grupos del conjunto a clasificar; de esta manera, diferentes criterios servirán a diferentes propósitos.

### MÉTODOS NUMÉRICOS PARA EL CLUSTERING

Los métodos numéricos para el clustering surgen en ramas de las ciencias naturales, como la biología o la zoología, en un intento de eliminar la subjetividad implícita en el proceso de clasificación que se desencadena al descubrir una nueva especie. El objetivo es pro-

porcionar un método no subjetivo y estable para clasificar y agrupar elementos.

Estos métodos adoptan diversos nombres que varían según el campo en el que se apliquen: taxonomía numérica (*numerical taxonomy*) en biología, *Q* análisis (*Q analysis*) en psicología, o reconocimiento de patrones no supervisado (*unsupervised pattern recognition*) en el campo de la inteligencia artificial. No obstante, hoy en día, análisis de clusters (*Clusters analysis*) o simplemente clustering son los términos más ampliamente aceptados y extendidos para referirse a tareas que involucran el descubrimiento de subgrupos dentro de un conjunto de elementos.

En la mayoría de las aplicaciones del clustering, el objetivo es obtener una partición de los datos, en la que cada instancia u objeto pertenezca a un único cluster, y la unión de todos ellos contenga a todos los objetos individuales. Dicho esto, debe destacarse que en algunas circunstancias son aceptables soluciones en las que existe solapamiento entre clusters, así como el hecho de que puede no existir una partición aceptable de los datos.

La manera más ampliamente extendida de representar la información sobre la que se debe aplicar clustering es una matriz  $X$  de dimensión  $n \times p$ , en la que cada fila corresponde a una instancia u objeto a procesar, y cada columna corresponde a una de las variables que caracterizan dichas instancias u objetos. El término comúnmente aceptado para referirse a cada fila es el de *vector de características*.

$$X = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & \cdots & \cdots & x_{n,p} \end{pmatrix}$$

La entrada  $x_{i,j}$  en  $X$  se corresponde con el valor de la variable  $j$ -esima en la instancia  $i$ .

Las variables en  $X$  pueden ser una mezcla de atributos en un dominio continuo, discreto o categórico. Además, es posible que, en problemas reales, algunas entradas no estén disponibles. Esta mezcla de tipos de variables y los valores perdidos pueden complicar la tarea del clustering. Sin embargo, existen métodos para tratar estos casos, como la inferencia de valores perdidos o las transformaciones de dominio.

En algunas aplicaciones, las filas de la matriz pueden contener medidas repetidas de la misma variable, aunque bajo diferentes condiciones, o en diferentes momentos, incluso en diferentes localizaciones espaciales. Un ejemplo de ello pueden ser las medidas de altura de un grupo de niños en un mismo mes a lo largo de diferentes años. Este tipo de datos posee una estructura que, de nuevo, puede complicar la tarea del clustering.

Algunos métodos de clustering conllevan realizar transformaciones sobre la matriz  $X$  para transformarla en una matriz de  $n \times n$  en la que se almacenan medidas extraídas de la matriz  $X$  que relacionen una instancia con todas las demás, como pueden ser la similitud, distancia o disimilitud.

El clustering es, dicho de manera simple, descubrimiento de grupos en datos, y no debe ser en ningún caso confundido con los métodos de discriminación o asignación, conocidos en el ámbito de la inteligencia artificial como aprendizaje supervisado. En ellos los grupos son conocidos a priori y el objetivo del análisis es obtener una regla de clasificación o clasificador que permita asignar nuevas instancias o individuos a uno de los grupos ya conocidos.

Una vez definida la estructura general de los métodos de clustering, estaría justificado preguntar, ¿qué es un cluster? La siguiente sección (2.2.1) intentará dar respuesta a esta pregunta.

### *¿Qué es un cluster?*

Hasta este momento, los términos cluster, grupo y clase han sido empleados de una manera completamente intuitiva, sin necesidad alguna de definición formal, una prueba más de lo innato de estos conceptos en el ser humano. De hecho, dar una definición formal de cluster resulta una tarea no, sólo complicada, sino en muchas ocasiones poco útil. Bonner, por ejemplo, propuso en 1964 una definición de cluster completamente dependiente de la interpretación del usuario: en lo que a él respecta, un cluster es aquello que el usuario entiende como cluster sin haberle propuesto una definición formal del mismo.

Aunque la definición de Bonner es acertada en una amplio rango de situaciones, autores como Cormack, en 1971, o Gordon en 1999, proponen una definición algo más analítica desde el punto de vista matemático, definiendo un cluster en términos de cohesión interna (homogeneidad), y aislamiento externo (separación). La Figura 2.1 muestra de manera informal las propiedades descritas anteriormente de forma que, a cualquier observador le resultarán aparentes los clusters presentes en ella sin necesidad de una definición formal de cluster. Este hecho puede explicar por qué alcanzar una definición matemáticamente precisa de homogeneidad y separación puede llegar a ser innecesario.

No queda completamente clara la manera en que las personas reconocen diferentes clusters cuando éstos son representados en un plano, pero una de las variables que con certeza influye es la distribución de distancias relativas entre los objetos o puntos.



Figura 2.1: Clusters con cohesión interna y/o aislamiento externo

Por otra parte, como ya mencionamos anteriormente en esta sección, puede darse el caso de que en un conjunto de datos no exista una partición justificada. En la Figura 2.2 se muestra un conjunto de datos para el que la mayoría de observadores llegaría a la conclusión de que no existen grupos diferenciados, simplemente una nube de puntos uniformemente distribuidos. Idealmente, es de esperar que un método de clustering aplicado a este mismo conjunto de datos llegue a la misma conclusión.

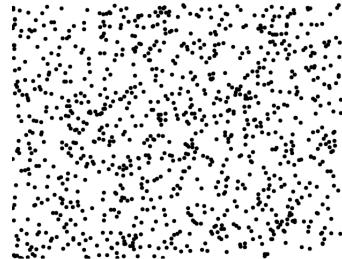


Figura 2.2: Distribución uniforme de puntos

Sin embargo, la mayoría de métodos de aprendizaje no supervisado darán como resultado un particionamiento uniforme como el que se muestra en la Figura 2.3. El número de particiones encontradas dependerá del método aplicado, si bien en cualquier caso obtendremos un particionamiento uniforme.

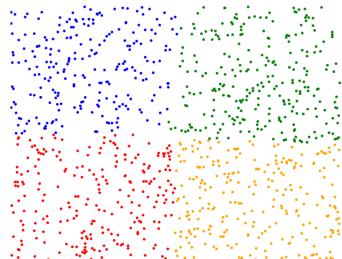


Figura 2.3: Distribución uniforme de puntos clasificados

El proceso de dividir una distribución homogénea de datos en diferentes grupos se conoce como disección, y tal proceso puede ser útil en ciertas circunstancias. Sin embargo, dado que en la mayoría de las ocasiones de aplicación real de métodos de clusters no se conoce a priori la estructura de los datos, existe el riesgo de interpretar todas

las soluciones en términos de existencia de subgrupos, lo que conllevaría la imposición de una estructura ficticia en datos en los que no hay estructura presente.

#### APLICACIONES DEL CLUSTERING

Como ya se ha indicado, el problema general al que intenta dar solución el clustering está presente en muchas ramas de la ciencia: biología, botánica, medicina, psicología, geografía, marketing, procesamiento de imágenes, psiquiatría, arqueología, etc. En esta sección se presentan algunas de las aplicaciones del clustering relacionadas con las citadas disciplinas.

##### *Aplicaciones en marketing*

Dividir los clientes en grupos homogéneos es una de las tareas más frecuentes en marketing. Un director de marketing podría preguntarse cómo agrupar los posibles clientes según los beneficios potenciales del producto que intenta introducir en el mercado. Por otra parte, un analista de marketing podría estar interesado en agrupar las empresas según sus características financieras, para poder analizarlas y predecir sus estrategias de mercado.

Un ejemplo de aplicación del clustering en este ámbito fue publicado por Green et al. (1967). Así, con un gran número de ciudades disponibles para el análisis, debieron restringir los lugares en los que llevar a cabo sus estudios, debido a motivos económicos. Para ello hicieron uso del análisis de clusters, clasificando las ciudades en pequeños grupos basándose en 14 características de las mismas, entre ellas el tamaño e ingresos medios *per capita*. Dado que se esperaba que las ciudades incluidas en un mismo grupo fueran muy similares, escogieron una ciudad de cada uno de ellos para realizar sus estudios.

Otra aplicación del análisis de clusters en el marketing fue descrita por Chakrapani (2004). En este caso, un fabricante de coches cree que comprar un coche deportivo no es una decisión basada sólo en capacidades económicas o edad, sino que es una decisión relacionada con el estilo de vida que llevan aquellos que deciden comprar un coche de estas características, frente a aquellos que no lo hacen. En consecuencia, el fabricante decide realizar un estudio, empleando análisis de clusters, que le permita identificar todas las características relacionadas con las personas que comprarían un coche deportivo, para así enfocar sus campañas de marketing a este sector específicamente.

### *Aplicaciones en astronomía*

Dado un conjunto de datos astronómicos, los investigadores quieren saber, por ejemplo, cuantas clases de estrellas hay presentes en ellos, basándose en algun criterio estadístico. Las preguntas más frecuentes dentro de este ámbito son: ¿Cuantos objetos estadísticamente diferentes están presentes en los datos y a que clase debe ser asignado cada objeto? ¿Aparecen clases de objetos previamente desconocidas?. El análisis de clusters puede ser aplicado para dar respuesta a estas cuestiones, ayudando a detectar objetos estadísticamente anómalos, así como a guiar el proceso de clasificación de los mismos. Algunos ejemplos incluyen el descubrimiento de quasars con alto corrimiento al rojo, quasars de tipo 2 (altamente luminosos, núcleos galácticos activos a menudo oscurecidos por polvo y gas), y enanas marrones.

Un ejemplo específico viene dado por el estudio de Faúndez-Abans et al. (1996), que aplicó técnicas de clustering a datos sobre la composición química de 192 nebulosas planetarias. Se identificaron 6 grupos diferentes que eran similares en muchos aspectos a una clasificación previa de dichos objetos, pero que también mostraban diferencias interesantes que hasta ese momento los investigadores habían pasado por alto.

Un segundo ejemplo lo encontramos en el estudio de Celeux y Govaert (1992), quienes aplicaron clustering basado en distribuciones normales a un conjunto de 2370 estrellas, descriptas por su velocidad relativa al núcleo galáctico y a la rotación galáctica. Usando un modelo de tres clusters, encontraron un cluster de gran tamaño y pequeño volumen, y dos de pequeño tamaño y gran volumen.

### *Aplicaciones en psiquiatría*

Las enfermedades de la mente son a menudo más difíciles de diagnosticar que las enfermedades del cuerpo; es por ello que en el campo de la psiquiatría ha crecido el interés por las técnicas de análisis de clusters que permitan refinar, o incluso redefinir, las técnicas de diagnosis para este tipo de enfermedades. Gran parte de este trabajo involucra pacientes deprimidos, casos en los que el interés reside en distinguir entre dos tipos de depresión, la endógena (congénita), y la neurótica.

Pilowsky et al. (1968), por ejemplo, usando métodos desarrollados por otros autores, aplicaron técnicas de clustering a 200 pacientes en base a sus respuestas a un cuestionario sobre la depresión, junto a información sobre su sexo, edad, estado mental y enfermedad padecida. Éste es un claro ejemplo de variables de diferentes tipos incluidas en el mismo conjunto de datos. Uno de los grupos obtenidos como resultado de este estudio fue identificado como marcador de la depresión endógena.

El análisis de clusters también ha sido empleado para encontrar una clasificación de individuos que intentaron cometer suicidio, que podría sentar las bases para estudios posteriores sobre las causas y tratamientos del problema. Paykey y Rassaby (1978), por ejemplo, estudiaron 236 casos de suicidas fallidos registrados por el servicio de emergencias de una ciudad de los Estados Unidos de América. Del conjunto de posibles variables, 14 fueron seleccionadas como particularmente relevantes para la clasificación y, por tanto, fueron usadas en el análisis. Entre ellas se encontraban: edad, número de intentos de suicidio, gravedad de la depresión, grado de hostilidad, además de una serie de características demográficas. Al conjunto de datos resultante se le aplicaron métodos de clustering; el resultado más significativo obtenido corresponde a una división en tres clusters bien definidos.

#### *Aplicaciones en meteorología y climatología*

Diariamente se recogen enormes cantidades de datos sobre la meteorología mundial. Explorar estos datos mediante técnicas de clustering puede aportar nuevos enfoques para la climatología y el medio ambiente.

Littmann (2000), por ejemplo, aplicó clustering a los datos recogidos sobre los cambios diarios en la presión superficial en la cuenca Mediterránea, y encontró 20 grupos que explicaban la varianza de las lluvias en las regiones centrales del Mediterráneo. Otro ejemplo viene de la mano de Liu y George (2005), quienes usaron el algoritmo K-medias difuso (*Fuzzy K-means*, FKM) con datos espaciotemporales de la climatología de las regiones del sur central de EEUU.

#### *Aplicaciones en arqueología*

La arqueología es otra de las disciplinas en la que resulta útil la aplicación del clustering. La clasificación de los diferentes objetos encontrados en los yacimientos puede ayudar a descubrir su uso, los períodos a los que pertenecen, así como la población que los utilizó. De forma similar, el estudio de materiales fosilizados puede ayudar a revelar cómo vivieron las sociedades prehistóricas.

Un ejemplo temprano de la aplicación de clustering a objetos arqueológicos viene dado por Hodson et al. (1966), que aplicó técnicas de clustering a un grupo de broches que datan de la Edad de Hierro, encontrando una clasificación para los mismos de demostrada relevancia arqueológica. Otro ejemplo de la mano de Hodson (1971) es la aplicación del algoritmo K-medias (KM, apéndice A) para construir una taxonomía de hachas de mano encontradas en las Islas Británicas. Las variables tenidas en cuenta para describir cada hacha incluyen longitud, anchura y valores en una escala que describen cómo de

puntiaguda era la herramienta. El clustering dio como resultado dos grupos de hachas, uno formado por las pequeñas y delgadas, y otro formado por las grandes y gruesas.

Respecto a materiales fosilizados, Sutton y Reinhard (1995) realizaron un estudio sobre 155 coprolitos encontrados en *Antelope House*, un yacimiento prehistórico en el Cañón de Chelly en Arizona. El estudio arrojó como resultado una interpretación de las diferencias entre coprolitos basada en la alimentación.

#### *Aplicaciones en bioinformática y genética*

Tiempos recientes están siendo testigo de un tremendo auge en el interés por la Bioinformática, acompañada por la biología molecular, ciencias de la computación, matemáticas y estadística. Tal aumento ha sido acelerado por la siempre creciente base de datos genómica y proteica, que son por sí mismas resultado de un grandísimo avance en las técnicas de secuenciación del ADN, medidas de expresión de los genes y compresión de las estructuras macromoleculares. La estadística ha resultado relevante en el estudio de la expresión de los genes. Los genes contenidos en el ADN de cada célula proporcionan las plantillas necesarias para la generación de las proteínas implicadas en la mayoría de los procesos estructurales y biomecánicos que tienen lugar en cada uno de nosotros. Sin embargo, aunque la mayoría de las células en los seres humanos contiene todos los complementos genéticos que componen el genoma humano, los genes se expresan de manera selectiva en cada célula dependiendo del tipo de la misma, del tejido y de las condiciones generales tanto dentro como fuera de la célula. La biología molecular ha puesto de manifiesto que la mayoría de los procesos en la vida de una célula están regulados por factores que afectan a la expresión de sus genes.

Como hemos visto, uno de los campos de investigación más activos hoy en día es el que estudia los procesos que regulan la expresión de los genes. Con el fin de almacenar la información relativa a esta área de estudio surgen los microarrays, (Cortesse, 2000). Desde el punto de vista del análisis de datos, una de las características relevantes en este tipo de información es que el número de características de cada instancia ( $p$ ), supera con creces al número de instancias disponibles ( $n$ ); conjuntos de datos como este son calificados como *datos de alta dimensionalidad*.

La mayoría de métodos estadísticos clásicos no pueden ser aplicados a este tipo de conjuntos de datos sin ser modificados de forma sustancial. Sin embargo, el análisis de clusters acepta bien tales conjuntos de datos y puede ser empleado para identificar grupos de genes con patrones de expresión similares, y dar respuesta a preguntas como por qué un gen se ve afectado por cierta enfermedad, o qué genes son responsables de enfermedades genéticas hereditarias.

Un ejemplo de aplicación lo encontramos en el trabajo de Selinski e Ickstadt (2008), quienes usaron clustering sobre polimorfismos de nucleótidos simples para detectar diferencias entre enfermedades a nivel genético.

#### RESUMEN

El clustering consiste en la exploración de conjuntos de datos. Su objetivo es discernir si pueden o no ser resumidos de manera significativa, en términos de un número relativamente pequeño de grupos o clusters de objetos o individuos, que se parecen unos a otros y que se diferencian de los que se encuentran en otros clusters.

Muchas ramas de la ciencia han hecho uso de las técnicas de clustering, de manera exitosa, para avanzar en sus respectivos campos, y procesar grandes cantidades de datos, cuyo análisis sería impensable afrontar con técnicas tradicionales.



# 3

## CLUSTERING CON RESTRICCIONES

---

Realizada la introducción sobre los conceptos básicos del clustering, el objetivo de esta sección es definir los conceptos relacionados con el clustering con restricciones, así como dar ejemplos de su aplicación y destacar los beneficios y problemas que presenta su uso. A tal fin tomamos como referencia principal el trabajo de Ian Davidson y Sugato Basu (2007) [2] entre otros.

### MOTIVACIÓN PARA EL CLUSTERING CON RESTRICCIONES

Tal y como hemos estudiado en secciones anteriores (2), los métodos de clustering no supervisado son útiles para dotar de estructura a datos referentes a un área concreta. Un ejemplo de ello lo encontramos en la clasificación de textos; Cohn et al. (2003) [3] afrontan un problema propuesto por Yahoo!, que consiste en, dada una gran cantidad de documentos de texto, agruparlos según una taxonomía en la que los documentos con temáticas similares se encuentren cercanos. Para ello, los métodos de clustering no supervisado resultan de utilidad, ya que la información sobre el problema de la que se dispone inicialmente es limitada. Sin embargo, Wagstaff et al. (2001) mostraron que aplicando clustering no supervisado a ciertos problemas, como el de agrupar datos de GPS de forma que los clusters definan los carriles de una vía, no se obtienen resultados significativos, pues los clusters obtenidos distan mucho de la forma alargada que se esperaría como resultado. Para atajar el problema, introdujeron en el clustering un nuevo elemento, las restricciones a nivel de instancia, que permitían incluir conocimiento sobre los clusters que guiarían los métodos de clustering para obtener los resultados esperados. Bastaba con indicar que los carriles de la vía por la que circulan los vehículos miden cuatro metros de ancho, y por tanto cualquier vehículo que se encuentre a una distancia mayor de 4 metros de otro, en dirección perpendicular a la del desplazamiento, debe ser ubicado en un cluster diferente.

Nos situamos entonces en un nuevo escenario: es posible incorporar información adicional al proceso de clustering, además de la contenida en el propio conjunto de datos, para guiarlo en la formación de la partición y obtener resultados más precisos. Esto sitúa al clustering con restricciones en el marco del aprendizaje semisupervisado, a diferencia de los métodos de clustering tradicionales que se enmarcan en el área del clustering no supervisado.

### DEFINICIÓN DE LAS RESTRICCIONES

El nuevo tipo de información que incorporamos al clustering viene dado en forma de restricciones a nivel de instancia, esto es, especificar si dos instancias del conjunto de datos deben estar en el mismo cluster o, por el contrario, deben estar en clusters separados.

A las restricciones que indican que dos puntos deben ser situados en el mismo cluster se las denomina *Must-link*, y se notan por  $ML(x, y)$ , donde  $x$  e  $y$  son dos instancias del conjunto de datos. De manera similar, a las restricciones que especifican lo contrario se las denomina *Cannot-link*, y se notan por  $CL(x, y)$ . [4]

Aunque pueden parecer simples, las restricciones definidas de la anterior forma poseen propiedades interesantes. Las Restricciones de tipo Must-link son un ejemplo de relación de equivalencia, y por tanto son simétricas, reflexivas y transitivas, formalizando:

**Observación 3.1** *Las restricciones de tipo ML son transitivas.* Sean  $CC_i$  y  $CC_j$  componentes conexas, conectadas mediante restricciones ML, y sean  $x$  e  $y$  dos instancias en  $CC_i$  y  $CC_j$  respectivamente. Entonces  $ML(x, y) : x \in CC_i, y \in CC_j \rightarrow ML(a, b) \forall a, b : a \in CC_i, b \in CC_j$ . [2]

**Observación 3.2** *Las restricciones de tipo CL pueden ser (encadenadas).* Sean  $CC_i$  y  $CC_j$  componentes conexas, conectadas mediante restricciones ML, y sean  $x$  e  $y$  dos instancias en  $CC_i$  y  $CC_j$  respectivamente. Entonces  $CL(x, y) : x \in CC_i, y \in CC_j \rightarrow CL(a, b) \forall a, b : a \in CC_i, b \in CC_j$ . [2]

Un claro ejemplo de uso de las restricciones lo encontramos en casos de aplicación de clustering en los que existen limitaciones en cuanto a las medidas de distancia, como sucedía en el supuesto de los datos GPS. Así, si queremos que las instancias que forman dos clusters estén separadas por una distancia mayor o igual a  $\delta$ , basta con establecer restricciones de tipo ML entre todas aquellas instancias cuya distancia sea menor que  $\delta$ . De manera similar, si queremos que el diámetro de los clusters sea como mucho  $\epsilon$ , debemos establecer un conjunto de restricciones de tipo CL entre todas aquellas instancias que se encuentren a una distancia mayor que  $\epsilon$ . La Figura 3.1 muestra una representación gráfica de estos dos tipos de restricciones.

### USO DE LAS RESTRICCIONES

Mientras que el aprendizaje completamente supervisado implica conocer la etiqueta asociada a cada instancia, en el aprendizaje semi-supervisado solo se dispone de un subconjunto de instancias etiquetadas. Por otra parte, en gran cantidad de dominios la información disponible se refiere a relaciones entre instancias, y no a la clase concreta a la que pertenecen las mismas. Es más, en montajes de clustering interactivo, un usuario no experto en el dominio del problema

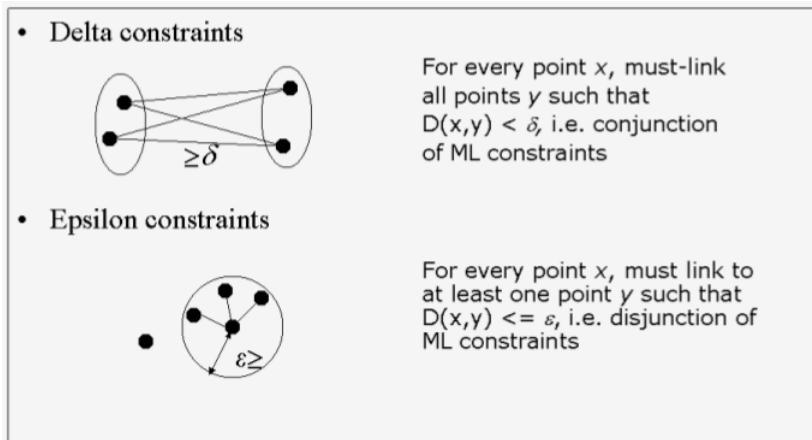


Figura 3.1: Restricciones de tipo *delta* y *epsilon*. [2]

podrá, probablemente, aportar información en forma de restricciones de tipo Must-Link (ML) y Cannot-Link (CL) [3][5], antes que aportar información sobre a qué clase concreta pertenecen ciertas instancias.

Habitualmente, las restricciones se incluyen en los problemas de clustering de dos maneras. Pueden ser empleadas para modificar la regla de asignación de instancias a clusters del método en cuestión, de forma que la solución satisfaga el máximo número de restricciones posible. Alternativamente, cabe la posibilidad de entrenar la función de distancia empleada por el método en base a las restricciones, ya sea antes o durante la aplicación del mismo. En cualquier caso, la fase de inicialización puede tomar en consideración las restricciones, de forma que las instancias asociadas con restricciones Must-Link (ML) serán situadas en el mismo cluster, y aquellas entre las que exista una restricción Cannot-Link (CL), quedarán en clusters diferentes. Basándonos en esta distinción, identificamos dos maneras de aproximar el problema, las basadas en restricciones (*constraint-based*), y las basadas en distancias (*distance-based*).

#### *Métodos basados en restricciones*

En los métodos basados en restricciones, el propio método de clustering es modificado de manera que la información disponible se emplea para sesgar la búsqueda y obtener una partición de los datos apropiada.

Existen dos modelos de métodos basados en restricciones: (1) aquellos que fuerzan el cumplimiento de las restricciones, e intentan encontrar la mejor asignación posible que no infrinja ninguna de ellas [6][7], y (2) los que hacen una interpretación relajada de las restricciones [8][9][10][11], permitiéndose incumplir un número mínimo de ellas para optimizar la función objetivo; de esta manera surge un compromiso entre el número de restricciones incumplidas y el valor de la

función objetivo. Existen diversas técnicas para obtener una partición atendiendo a las restricciones:

- Modificar la función objetivo de manera que incluya una penalización por incumplir restricciones. [12] [10]
- Agrupar instancias con información adicional obtenida de una distribución condicional en un espacio auxiliar. [13]
- Forzar el cumplimiento de todas las restricciones modificando la regla de asignación del método. [6]
- Inicializando los clusters en base a restricciones inferidas del conjunto de instancias etiquetadas.[14]

La Figura 3.2 muestra un conjunto de datos junto a sus restricciones asociadas. La Figura 3.3 propone un posible agrupamiento que satisface todas las restricciones.

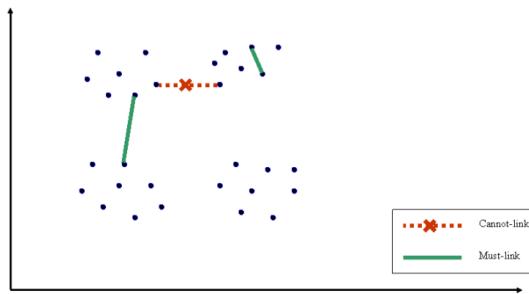


Figura 3.2: Restricciones sobre un conjunto de datos. [2]

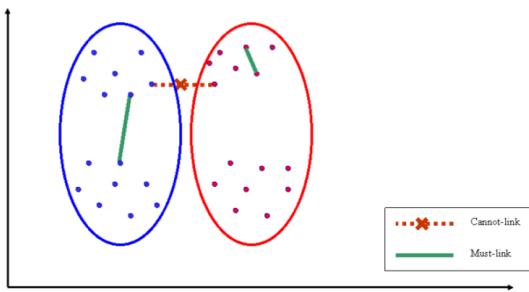


Figura 3.3: Clustering que satisface todas las restricciones. [2]

### *Métodos basados en distancia*

En las aproximaciones basadas en distancias, se emplean métodos de clustering clásicos que hacen uso de una medida de distancia, de forma que dicha medida se modifica para que tenga en consideración las restricciones. En este contexto, satisfacer las restricciones significa que las instancias relacionadas con restricciones Must-Link (ML) se sitúan juntas en el espacio, y las relacionadas mediante Cannot-Link (CL) se encuentran separadas.

La Figura 3.5 muestra un posible agrupamiento basado en una métrica aprendida a partir de las restricciones especificadas en la Figura 3.4. Cabe destacar que en la Figura 3.5 el espacio en el que se encuentran los datos ha sido comprimido en el eje vertical y ensanchado en el eje horizontal para ajustarlo a la métrica de distancia aprendida.

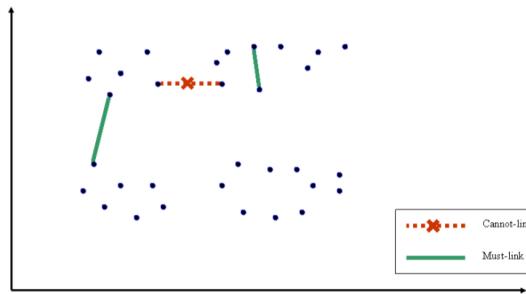


Figura 3.4: Restricciones sobre un conjunto de datos. [2]

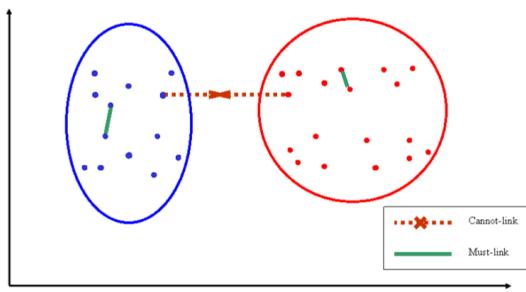


Figura 3.5: Clustering basado en métrica aprendida en base a las restricciones. [2]

#### APLICACIONES DEL CLUSTERING CON RESTRICCIONES

Esta sección muestra algunos casos de aplicación en los que el clustering con restricciones ha resultado ser una herramienta más útil que el clustering no supervisado. Para cada caso analizaremos cómo se obtuvieron las restricciones y cómo éstas mejoran los resultados en el clustering resultante.

##### *Aplicaciones en análisis de imágenes*

La Figura 3.6 muestra un extracto del conjunto de datos de caras de CMU (Carnegie Mellon University), en el que la tarea es agrupar caras en base a diferentes criterios. En este caso, el objetivo es agrupar las caras según su orientación.



Figura 3.6: Caras de la base de datos de CMU. [2]

El método empleado para extraer las restricciones es uno de los más populares en la literatura: establecer el número de clusters de la partición resultado igual al número de clases en la base de datos, y generar las restricciones a partir de un subconjunto de instancias etiquetadas; esto es, si dos instancias tiene diferentes etiquetas, establecer una restricción Cannot-Link (CL) entre ellas, en caso contrario una de tipo Must-Link (ML). De esta forma, entre las imágenes mostradas en la Figura 3.7 se establecen restricciones Cannot-Link (CL), ya que, aunque pertenecen a la misma persona, no presentan la misma orientación.



Figura 3.7: Restricciones de tipo CL entre caras de la misma persona. [2]

En la Figura 3.8 se muestra otro conjunto de datos de imágenes sobre el que se aplican técnicas de clustering con restricciones. En este caso, la tarea es realizar reconocimiento de objetos para incorporar el método al sistema de navegación del robot Aibo [10]. Para ello se emplean restricciones de distancia de tipo  $\delta$  y  $\epsilon$  como las descritas en la Figura 3.1. De esta manera se consiguen clusters bien diferenciados y por tanto útiles para las tareas de búsqueda de caminos que el robot realiza durante la navegación.



Figura 3.8: Método de clustering empleado en el sistema de navegación del robot Aibo. [2][10]

### *Aplicaciones en análisis de vídeos*

Las bases de datos de vídeo son uno de los ejemplos en los que las restricciones pueden ser generadas directamente desde el dominio de datos, especialmente disponiendo de datos espacio-temporales sobre el vídeo [15]. En datos temporalmente sucesivos es posible establecer restricciones de tipo Must-Link (ML) entre grupos de píxeles de fotogramas (*frames*) cercanos en el tiempo. Esto es especialmente útil cuando la tarea es implementar reconocimiento de objetos basado en clustering y segmentación. También es posible añadir restricciones Cannot-Link (CL) a clusters localizados en el mismo fotograma, ya que existe una baja probabilidad de que estén asociados al mismo objeto. De hecho, en el dominio asociado a problemas de análisis de vídeo existen gran variedad de métodos de extracción de restricciones [15], la Figura 3.9 muestra algunos ejemplos.

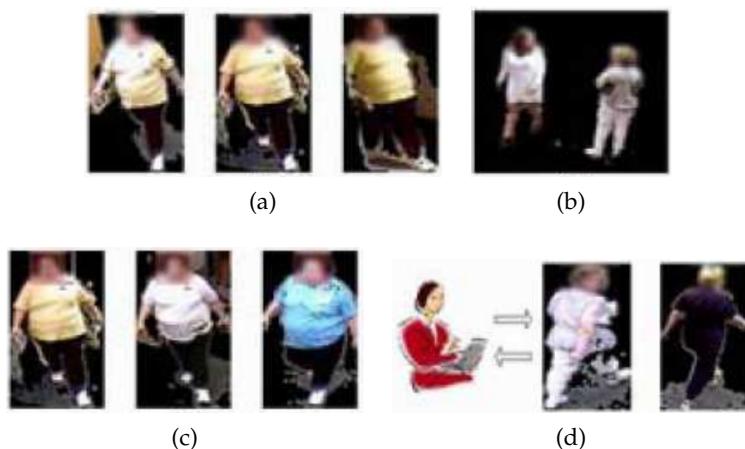


Figura 3.9: Diferentes tipos de restricciones en datos de video. [15] [2]

En la Figura 3.9, la imagen (a) corresponde a restricciones extraídas del seguimiento de una persona durante un periodo de tiempo, la (b) corresponde a restricciones espaciales que asocian dos objetos localizados en el mismo fotograma, la imagen (c) corresponde a restricciones obtenidas mediante reconocimiento facial y la (d) a las proporcionadas por el usuario.

Disponiendo de tantos métodos para extraer restricciones, cabe plantearse: ¿qué sucede si se establecen demasiadas restricciones? ¿Hace esto que el problema esté sobrestringido? En secciones posteriores (3.6) abordaremos estas cuestiones.

### *Aplicaciones en genética*

En clustering de genes basado en microarrays, los genes vienen representados por su perfil de expresión en diferentes experimentos y agrupados empleando diferentes métodos, en este caso métodos de

clustering con restricciones. La Figura 3.10 muestra un ejemplo: se trata de restricciones de tipo Must-Link (ML) que se establecen entre genes en base a los datos de co-ocurrencia almacenados en la base de datos de interacciones de proteínas, que contiene información sobre qué genes (y sus proteínas asociadas) están presentes en los mismos procesos celulares [16]. Esta información puede ser empleada para mejorar los resultados que proporcionan los métodos de clustering. [9]

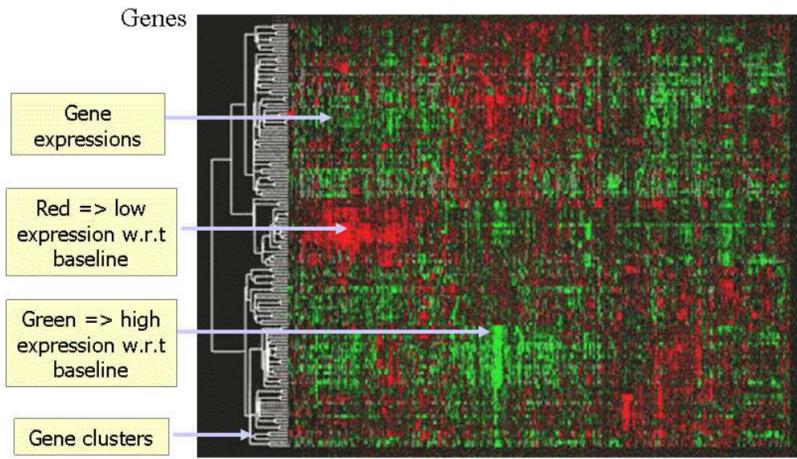


Figura 3.10: Clustering de genes basado en microarrays. [2]

#### *Aplicaciones en análisis de textos*

En tareas de clasificación de contenido, el objetivo es dividir de manera automática grandes cantidades de documentos en grupos o clusters. En este caso es posible extraer las restricciones de múltiples recursos auxiliares. Por ejemplo, si dos documentos se encuentran en el mismo directorio se podría inferir una restricción de tipo Must-Link (ML) entre ellos. De esta manera es posible modificar el clustering resultante para que se adapte a un criterio concreto, como por ejemplo crear una jerarquía de documentos semejante a la forma en la que se encuentran organizados en la estructura de directorios de entrada.

#### *Aplicaciones en datos web*

El clustering con restricciones resulta de gran utilidad en el procesamiento de datos de búsqueda en páginas web. Aquí, el objetivo es agrupar, de manera automática, los resultados de una consulta ambigua en el motor de búsqueda en clusters de URLs, que se refieran al concepto introducido como consulta en diferentes contextos. En este ámbito es posible extraer las restricciones a partir de búsquedas

realizadas anteriormente por los usuarios, de manera que se establece una restricción de tipo Must-Link (ML) entre URLs visitadas en la misma sesión de usuario. Aplicar clustering utilizando estas restricciones puede ayudar a sesgar el resultado de las búsquedas hacia las preferencias del usuario.

#### *Aplicaciones en datos de audio*

En ciertas tareas de análisis de audio, es posible que no se conozca el número de clases de objetos presentes en los datos, si bien las restricciones pueden extraerse directamente del dominio de datos. Esto sucede, por ejemplo, al aplicar clustering para reconocimiento de hablantes en una conversación [17]. En este caso el número de participantes no se conoce a priori, pero es fácil detectar si dos hablantes son diferentes o similares y establecer las restricciones en base a ello.

#### *Aplicaciones en datos de GPS*

Tal y como se indicó en el inicio de este capítulo, el clustering con restricciones sobre datos GPS se utiliza para identificar el carril por el que circula cada vehículo, como se muestra en la Figura 3.11. Cada instancia viene representada por la posición que ocupa en la vía en coordenadas cartesianas bidimensionales ( $x, y$ ), obtenidas en base a los datos GPS. La Figura 3.12 muestra de manera gráfica esta representación de los datos (cabe destacar que múltiples instancias pueden referirse al mismo vehículo en distintos momentos en el tiempo).

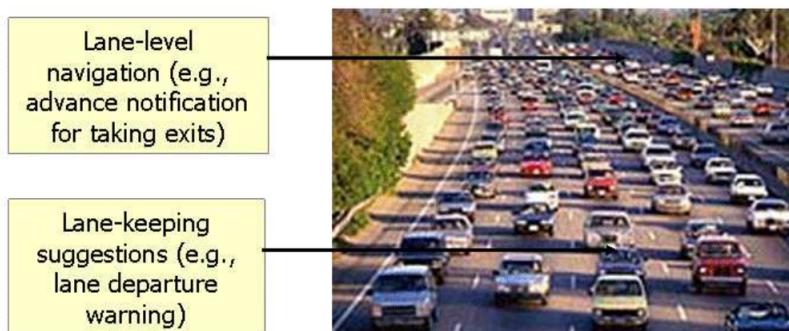


Figura 3.11: Uso de información GPS. [2] [6]

En este dominio, los clusters reales tienen forma alargada en el eje horizontal y se encuentran alineados perpendicularmente a la dirección de desplazamiento. Para lograr que los clusters resultantes tengan esta forma, podemos hacer uso de las restricciones. Estableceremos una restricción de tipo Cannot-Link (CL) entre aquellas instancias separadas más de 4 metros en dirección perpendicular a la del desplazamiento (ya que los carriles tienen una anchura máxima de 4 metros), y Must-Link (ML) entre aquellas instancias que presenten continuidad

en el eje horizontal, puesto que es probable que los vehículos que representan se encuentren en el mismo carril. Este modelo de clustering ha probado ser muy útil en navegación a tiempo real [6], permitiendo notificar al usuario cuando debe cambiar de carril, o cuando no debe abandonarlo.

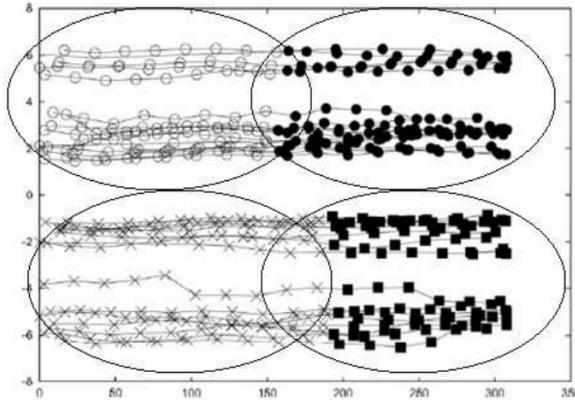


Figura 3.12: Clusters encontrados en datos GPS sin uso de restricciones. [2] [6]

#### BENEFICIOS DEL USO DE RESTRICCIONES

Encontramos dos beneficios principales en el uso de restricciones:

- Incremento de la exactitud en las predicciones de las etiquetas al generar restricciones en base a un subconjunto de instancias etiquetadas.
- Obtención de clusters con geometría adaptable a cada problema.

A continuación se analizan estos dos beneficios:

Dado  $X = \{x_1 \dots x_u\}$ , un gran conjunto de instancias no etiquetadas, y  $L = \{(x_{u+1}, y_{u+1}) \dots (x_{u+l}, y_{u+l})\}$ , un pequeño conjunto de instancias etiquetadas, es común escoger dos elementos de  $L$  (con reemplazamiento) y establecer una restricción ML entre ellos si pertenecen a la misma clase o, en caso contrario, una de tipo CL. Un método apropiado para evaluar los resultados ofrecidos por un método de clustering es medir el nivel de exactitud de éste a la hora de predecir las etiquetas del conjunto  $X$ . Esto normalmente requiere que se especifique el número de clusters deseados igual al número de clases conocidas en  $X$  ( $k = k^*$ ). Para medir la exactitud se emplean métodos como el *Rand index* [18].

El trabajo de Wagstaff y Cardie [4], en el que generaban las restricciones de la manera descrita anteriormente, demostraba que, cuando se realiza un promedio de la exactitud de las predicciones obtenidas con algoritmos de clustering con restricciones, variando estas últimas

entre experimentos, se obtienen resultados hasta un 20 % mejores que con las técnicas clásicas.

**Observación 3.3** *El uso de restricciones, en promedio, incrementa la precisión. El rendimiento de un método al predecir etiquetas aumenta cuando se promedia empleando numerosos conjuntos de restricciones diferentes.* [2]

Esta regla, sin embargo, no es cierta en todos los casos, pues en conjuntos de datos como *Tic-Tac-Toe Endgame*, no se consigue ningún incremento en las predicciones sea cual sea el número de restricciones empleadas. La explicación dada por los autores citados para estas excepciones se basa en que establecer  $k = k^*$  no es apropiado en este caso.

El otro beneficio que reporta el uso de restricciones es la posibilidad de obtener clusters con la geometría deseada, como el ejemplo de aplicar clustering a datos GPS, analizado en la sección 3.4.7 de este trabajo.

#### PROBLEMAS DEL USO DE RESTRICCIONES

Aunque, tal y como hemos comprobado, la incorporación de restricciones a los métodos de clustering reporta beneficios en algunas aplicaciones, existen dos inconvenientes principales que se exponen a continuación, así como posibles soluciones a los mismos.

##### *El problema de la factibilidad*

La introducción de restricciones en el clustering cambia el problema al que éste da solución, que pasa a ser: *Encontrar la mejor partición que satisfaga todas las restricciones*. De esta manera, si aquellas no están bien especificadas o si los métodos de extracción son inadecuados, podemos encontrar que las restricciones se contradicen, lo que deriva en que no existe una asignación de instancias a clusters que las satisfaga todas. Por ejemplo, no existe asignación que satisfaga las restricciones  $ML(x_1, x_2)$  y  $CL(x_1, x_2)$ , independientemente del valor de  $k$ . Lo mismo sucede para  $k = 2$ , y las restricciones  $CL(x_1, x_2)$ ,  $CL(x_2, x_3)$  y  $CL(x_1, x_3)$ . Formalizando, el problema de la factibilidad para problemas de clustering (no jerárquico) con restricciones viene definido por:

**Definición 3.1** *Problema de la factibilidad para clustering con restricciones:* Dado un conjunto de datos  $X$ , un conjunto de restricciones  $R$ , un umbral superior  $K_l$  y un umbral superior  $K_u$  para el número de clusters resultantes, ¿Existe una partición de  $X$  en bloques tal que  $K_l \leq k \leq K_u$  y todas las restricciones en  $R$  se satisfacen? [10] [2]

La complejidad teórica del problema dependerá del tipo de restricciones que se combinen en él. La Tabla 3.1 presenta, de manera resumida, la complejidad esperada en cada caso.

Complejidad del clustering con restricciones	
Restricciones	Complejidad
Restricciones $\delta$	P
Restricciones $\epsilon$	P
ML y $\delta$	P
ML y $\epsilon$	NP-completo
$\epsilon$ y $\delta$	P
CL y otra	NP-completo

Tabla 3.1: Complejidad del problema del clustering en función del tipo de restricciones [2]

Tal y como queda reflejado en la Tabla 3.1, la utilización de restricciones Cannot-Link (CL) eleva el nivel de complejidad del clustering a NP-completo y, por tanto, el problema del clustering con restricciones es intratable. De manera intuitiva puede entenderse fácilmente que, si encontrar una sola partición que satisfaga las restricciones es un problema complejo, más complejo es aún encontrar la mejor.

**Observación 3.4** *Saber que existe una solución factible no nos ayuda a encontrarla. Las implicaciones de este resultado sobre la complejidad del clustering con restricciones implican que, aun en caso de que exista una partición factible, no será fácil de encontrar, hablando en términos de complejidad algorítmica. [2]*

Los autores Wagstaff [19] y Davidson y Ravi [5] muestran que aun especificando el número de clusters de salida igual al de clases verdaderas ( $k = k_*$ ), cosa que garantiza que existe una solución factible, algoritmos simples como la adaptación de K-medias (KM, apéndice A) al clustering con restricciones (COP-K-means [6], sección 4.2), pueden no converger debido al problema de la factibilidad.

#### *El problema de la utilidad de conjuntos de restricciones*

En el clustering con restricciones se asume que éstas son indicaciones que guían al algoritmo para encontrar la partición de los datos deseada. Entonces, está justificado pensar que, de cuanta más información adicional (restricciones) dispongamos, más cercano estará el resultado que obtengamos al que buscamos, tal y como la observación 3.3 afirmaba. Sin embargo, y a pesar de lo dispuesto en dicha

observación, existen casos en los que, aun generando las restricciones sin ruido y en base a las etiquetas verdaderas, existen conjuntos de restricciones que, lejos de mejorar los resultados, los empeoran considerablemente [20]. Esto parece estar en desacuerdo con la observación 3.3, sin embargo, recordemos que en ella se hace referencia al caso medio, y no a casos particulares.

**Observación 3.5** *Conjuntos de restricciones particulares pueden causar efectos adversos. Algunos conjuntos de restricciones generados en base a las etiquetas verdaderas y libres de ruido pueden resultar en una pérdida de precisión a la hora de predecir esas mismas etiquetas. [2]*

#### *Soluciones al problema de la factibilidad*

El problema de la factibilidad puede ser abordado de varias maneras. La más inmediata quizá sea mantener el número de restricciones bajo, en proporción al número de instancias totales, para minimizar la probabilidad de que surjan inconsistencias. Sin embargo, no poder aumentar el número de restricciones si el problema lo requiere no es el escenario ideal. Por ello se debe poner interés en analizar cuando un problema pasa a estar sobrerestringido, ya que, como hemos estudiado en secciones anteriores de este trabajo (3.6.2), incluso generando las restricciones en base a las etiquetas verdaderas, algoritmos como COP-K-medias (sección 4.2) dejan de ser efectivos conforme aumenta el número de restricciones a satisfacer, incluso reiniciando de manera aleatoria el algoritmo varias veces.

El fenómeno de la sobrerestricción de problemas mediante el uso de restricciones Cannot-Link (CL) está íntimamente relacionado con el problema del coloreado de grafos; de hecho, ha sido demostrado que éste es equivalente al problema del clustering con restricciones CL [21]. Así, encontramos que resolver un problema con restricciones CL mediante algoritmos como COP-K-medias es, a efectos prácticos, resolver el problema del coloreado de grafos.

**Observación 3.6** *El problema del clustering con restricciones CL es análogo al problema del coloreado de grafos. [2]*

Este resultado permite trasladar muchas de las propiedades del problema del coloreado de grafos al problema de clustering con restricciones. Por ejemplo, el teorema de Brook establece que el coloreado de grafos es sencillo cuando el número de colores disponibles ( $k$  en nuestro caso), es mayor que el máximo grado del grafo.

**Observación 3.7** *El teorema de Brook es aplicable al problema del clustering con restricciones. Si  $k >$  (Mayor número de restricciones CL sobre una instancia), entonces siempre existirá una partición factible. [2]*

Con esto, y aunque la observación 3.4 indique lo contrario, cuando el problema del clustering cumple la condición expuesta en la observación 3.7, podemos garantizar que siempre se encontrará una solución al problema en tiempo polinómico. Para asegurar la condición de Brook, es posible construir el conjunto de restricciones de manera que ninguna instancia tome partido en más de  $k$  restricciones Cannot-Link (CL). [21]

#### *Soluciones al problema de la utilidad de conjuntos de restricciones*

La solución al problema es simple: identificar aquellos conjuntos de restricciones verdaderamente útiles. Sin embargo, esto involucra aplicar algún tipo de métrica que permita evaluar cuándo un conjunto de restricciones dado cumple esta condición. A tal fin, Davidson, Wagstaff y Basu propusieron dos medidas: **informatividad** y **coherencia**.

La **informatividad** es una medida referida a la cantidad de información presente en el conjunto de restricciones que el algoritmo no puede determinar por sí mismo. Por ejemplo, en la Figura 3.13, un algoritmo como COP-K-medias (sección 4.2) se vería inclinado a agrupar instancias cercanas en el espacio y colocar en clusters separados aquellas que se encuentren lejanas; sin embargo, las restricciones sesgan el espacio de soluciones evitando que esto suceda.

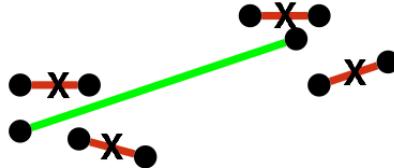


Figura 3.13: Ejemplo de conjunto de restricciones informativo. [2]

La informatividad se estima utilizando el conjunto de restricciones como un conjunto de test, de manera que se mide la habilidad del algoritmo para predecir las restricciones presentes en él. Formalizando, dado un conjunto de restricciones  $R$  y un algoritmo  $A$ , obtenemos la partición  $P_A$  aplicando el algoritmo al conjunto de datos de entrada especificando el conjunto de restricciones vacío. Calculamos entonces la fracción de las restricciones incumplidas por  $P_A$  [2]:

$$I_A(R) = \frac{1}{|R|} \left[ \sum_{r \in R} \text{unsat}(r, P_A) \right] \quad (3.1)$$

Por otra parte, la **coherencia** mide el grado de concordancia dentro del propio conjunto de restricciones respecto a una métrica dada ( $D$ ). Por ejemplo, la Figura 3.14 muestra dos restricciones paralelas y muy cercanas, pero de distinto tipo. Es en casos como este en los que se da una contradicción, ya que las restricciones Must-Link (ML) indican que la distancia entre las instancias involucradas en ellas es pequeña, mientras que las de tipo Cannot-Link (CL) deben indicar lo contrario.



Figura 3.14: Ejemplo de conjunto de restricciones contradictorio. [2]

Entonces, la medida de coherencia viene dada por el grado de solapamiento que presentan las restricciones al interpretarlas como vectores en el espacio y proyectarlas sobre uno de los ejes, tal y como se muestra en la Figura 3.15.

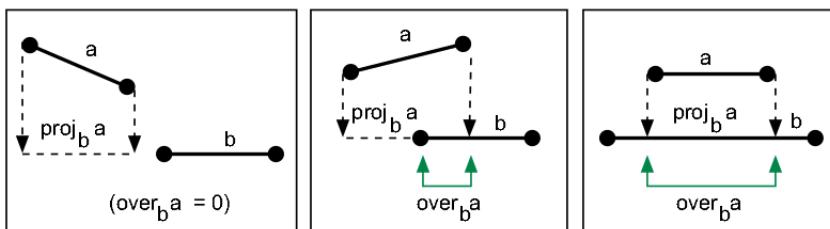


Figura 3.15: Representación de la medida de coherencia. [2]

## RESUMEN

El clustering con restricciones incorpora nueva información al problema del clustering original, las restricciones. Dicha información viene dada en forma de restricciones, ya sean Must-Link (ML), Cannot-Link (CL), o restricciones de distancia, y es utilizada para guiar al método que apliquemos, al conjunto de datos en cuestión, en la búsqueda de la partición resultado.

Los métodos de clustering derivados de este concepto han demostrado ser de gran utilidad en múltiples ámbitos, así como también presenta problemas que pueden ser subsanados estudiando en profundidad las restricciones a emplear para resolver cada problema.



# 4

## ALGORITMOS DE CLUSTERING CON RESTRICCIONES

---

Una vez introducido el problema del clustering con restricciones, pasamos a profundizar en los métodos para su aplicación. La siguiente sección presenta 5 algoritmos de clustering con restricciones, cuyos resultados serán expuestos más tarde, en la sección 6.

### FORMALIZACIÓN DEL PROBLEMA

Definido ya el problema del clustering con restricciones en la sección 3, especificamos la manera de notar sus elementos, de forma que sea sencillo referirse a ellos.

- Notaremos con  $X$  a la matriz de  $n \times p$  que contiene el conjunto de datos de entrada.
- Notaremos con  $x_i$  t.q.  $i \in \{1, \dots, n\}$  a cada instancia de  $X$ , por lo que  $x_i$  es un vector en el espacio  $\mathbb{R}^p$  ( $x_i \in \mathbb{R}^p$ ).
- Notaremos con  $R$  al conjunto de restricciones, tanto las de tipo ML como las CL, es decir  $R = ML \cup CL$ .
- Notaremos con  $K$  el número de clusters de la partición resultante.
- Notaremos con  $C$  el conjunto de clusters, y con  $c_i$  t.q.  $i \in \{1, \dots, K\}$  a cada uno de ellos, por tanto  $C = \{c_1, \dots, c_K\}$ .
- Notaremos con  $V$  la matriz de  $K \times p$  que almacena el conjunto de centroides asociados a los clusters, de manera que  $v_i \in \mathbb{R}^p$  corresponde al cluster  $c_i$ .

Cabe destacar que los elementos y parámetros particulares de cada algoritmo serán definidos en la sección correspondiente al mismo, así como que no todos los elementos expuestos anteriormente son comunes a todos los algoritmos; si bien si que lo son a la mayoría.

### COP-K-MEANS (CONSTRAINED K-MEANS)

El algoritmo K-medias (KM, apéndice A) es uno de los más básicos para aplicar clustering. Así, el algoritmo COP-K-medias (COP-K-means) es la adaptación inmediata de KM al clustering con restricciones. Para realizar un estudio detallado sobre el mismo tomaremos como base el trabajo de Wagstaff et al. (2001) [6].

El cambio más notable que supone COP-K-medias respecto al tradicional K-medias, consiste en modificar la regla de asignación de instancias a clusters de este último, para comprobar que dicha asignación no viola ninguna restricción. De esta manera, en cada iteración se intenta asignar cada instancia  $x_i$  al cluster más cercano  $c_j$ . Ésta asignación solo se llevará a cabo si, como hemos dicho, no se viola ninguna restricción. Si existe una instancia  $x_{ML}$  que debe ser asignada al mismo cluster que  $x_i$ , pero ya ha sido incluida en otro cluster, o existe una instancia  $x_{CL}$  en  $c_j$  que no puede ser agrupada junto a  $x_i$ , entonces  $x_i$  no puede ser asignado a  $c_j$ . El proceso continua hasta encontrar una asignación legal para  $x_i$ , en caso de que no se encuentre se devuelve la partición vacía como resultado. Así el algoritmo da como resultado una partición de  $X$  que cumple necesariamente todas las restricciones especificadas en  $R$ . El Algoritmo 1 corresponde al pseudocódigo asociado a COP-K-medias [6]:

---

**Algoritmo 1:** COP-K-medias

---

**Entrada:** Conjunto de datos  $X$ , conjunto de restricciones  $R$ , número de clusters  $K$ .

**Salida:** Partición  $P$  del conjunto de datos  $X$ .

**función** COP-K-medias( $X, R, K$ ) **begin**

1. Sean  $V = \{v_1, \dots, v_K\}$  los centroides iniciales
2. Asignar cada instancia  $x_i \in X$ , al cluster  $c_j$  asociado al centroide más cercano  $v_j$  tal que  $\text{ViolaRestriccion}(x_i, c_j, R) = \text{falso}$ . Si no existe  $c \in C$  t.q.  $\text{ViolaRestriccion}(x_i, c, R) = \text{falso}$ , **return**  $\emptyset$ .
3. Para cada cluster  $c_i$ , actualizar su centroide  $v_i$  realizando un promedio de todas las instancias  $x_i$  asignadas a él.
4. Iterar entre (1.) y (2.) hasta converger.
5. **return**  $C$

**end**

**Entrada:** Instancia  $x$ , cluster  $c$ , conjunto de restricciones  $R$

**función** ViolaRestriccion( $x, c, R$ ) **begin**

1. Para cada  $(x, x_{ML}) \in ML$ , si  $x_{ML} \notin R$  **return true**.
2. Para cada  $(x, x_{CL}) \in CL$ , si  $x_{CL} \notin R$  **return true**.
3. En otro caso, **return false**.

**end**

---

Existen multitud de criterios de convergencia estandarizados, aunque es común emplear uno adaptado al problema particular que queramos solucionar. El más extendido consiste en calcular la diferencia de la posición de los centroides entre dos iteraciones sucesivas, de forma que cuando esta sea menor que un umbral dado detenemos el proceso de iteración.

### CEKM (CONSTRAINED EVIDENTIAL K-MEANS)

Tal y como indican Violaine el al. (2012) [22], cuyo trabajo es el fundamento de la siguiente sección, para comprender el algoritmo Constrained Evidential K-means (CEKM), primero es necesario realizar una introducción al algoritmo K-medias difuso (*Fuzzy K-means*, FKM). En él, cada instancia puede pertenecer a uno o más clusters, con diferentes grados de pertenencia. La matriz que almacena esta información, es decir, la partición difusa, se nota con  $U$ , y se calcula minimizando la siguiente función:

$$\sum_{j=1}^k u_{ij} \quad t.q. \quad u_{ij} \in [0, 1] \forall i, j \quad (4.1)$$

Donde  $u_{ij}$  representa el grado de pertenencia de la instancia  $i$  al cluster  $j$ , y  $K$  es el número de clusters. Sin embargo, este método puede producir resultados contraintuitivos cuando los datos a los que se aplica son ruidosos o presentan instancias aisladas (*outliers*).

El clustering evidencial (*evidential clustering*) da solución a los problemas que presenta el algoritmo FKM, introduciendo el concepto de partición de creencia (*credal partition*), que extiende los conceptos existentes de particiones fuertes, difusas y probabilísticas. De esta forma, en una partición de creencia, se asigna a cada instancia una masa de creencia (*mass of belief*) no solo para un único cluster, sino para cualquier conjunto de los mismos. El método CEKM combina las ventajas del uso de las restricciones con las del uso de funciones de creencia.

#### *Funciones de creencia*

La teoría de la evidencia de Dempster-Shafer ofrece un marco teórico para trabajar con información parcial y no completamente fiable, tomamos de ella los conceptos relativos a las funciones de creencia.

Consideremos la variable  $c$  que toma valores en el conjunto finito  $C = \{c_1 \dots c_K\}$ . El conocimiento parcial sujeto al valor real que adopta  $c$  puede ser representado mediante una función de masa  $m$ , que es una aplicación de  $C$  al intervalo  $[0, 1]$ :

$$\sum_{A \subseteq C} m(A) = 1 \quad (4.2)$$

A los subconjuntos  $A$  de  $C$  que cumplen que  $m(A) > 0$  se los denomina conjuntos focales (*focal sets*) de  $m$ . El valor del conjunto focal  $m(A)$  se interpreta como la fracción de una unidad de masa de creencia que está asignada a  $A$  y que no puede ser asignada a ningún otro subconjunto de  $A$ . Si el único conjunto focal es  $C$ , nos encontramos en el caso de completa ignorancia sobre los datos. Por el contrario, si la masa de creencia se asigna a un único elemento de  $C$ , estaríamos en el caso de certeza absoluta.

Se dice que una función de masa  $m$  está normalizada si  $m(\emptyset) = 0$ . Sin embargo, bajo la hipótesis de mundo abierto, una función de masa en la que  $m(\emptyset) > 0$  se interpreta como la cantidad de creencia que se le asigna a la hipótesis de que el verdadero valor de  $c$  puede no encontrarse en  $C$ .

Dada una función de masa  $m$ , podemos definir una función de plausibilidad  $pl : 2^C \rightarrow [0, 1]$  y una función de creencia  $bel : 2^C \rightarrow [0, 1]$  de la siguiente manera:

$$pl(A) = \sum_{B \cap A \neq \emptyset} m(B) \quad \forall A \subseteq C \quad (4.3)$$

y

$$bel(A) = \sum_{B \subseteq A, B \neq \emptyset} m(B) \quad \forall A \subseteq C \quad (4.4)$$

De esta manera, las funciones  $pl$  y  $bel$  están relacionadas como sigue:

$$pl(A) = 1 - m(\emptyset) - bel(\bar{A}) \quad (4.5)$$

Donde  $\bar{A}$  representa el complemento de  $A$ . La cantidad  $bel(A)$  se interpreta como el grado de creencia en  $A$ , tomando en consideración la masa de creencia asignada a  $A$  y a los subconjuntos no vacíos de  $A$ . Por el contrario,  $pl(A)$  mide hasta qué punto es erróneo no creer en  $\bar{A}$ .

Con el objetivo de tomar decisiones en base al valor de  $c$ , es posible transformar la función de masa en una distribución de probabilidad pignística, definida, para una función de masa normalizada, como:

$$BetP(c) = \sum_{c \in A} \frac{m(A)}{|A|} \quad \forall c \in C \quad (4.6)$$

### *FKM (fuzzy K-means) y sus variantes*

Cada cluster  $c_j \in C$  con  $j \in \{1, \dots, K\}$  está representado por un vector  $v_j \in \mathbb{R}^p$ , es decir, un centroide. Además, definimos  $V$  como la matriz compuesta por todos los centroides, y  $U = (u_{ij})$  como la partición difusa que contiene los grados de pertenencia de cada instancia de  $X$  a cada cluster. El algoritmo Fuzzy K-means (FKM) calcula las matrices  $U$  y  $V$  de manera que minimiza (sujeto a las ecuaciones 4.1 y 4.2) la siguiente función:

$$J_{FKM}(U, V) = \sum_{i=1}^n \sum_{j=1}^K u_{ij}^\beta d_{ij}^2 \quad (4.7)$$

Donde  $d_{ij}$  representa la distancia Euclídea entre el objeto  $x_i$  y el centroide  $v_j$ , y donde  $\beta > 1$  es el exponente que controla el grado de difusión de la partición. La función objetivo se minimiza mediante un

algoritmo iterativo que optimiza los centroides y los grados de pertenencia de manera alterna. El algoritmo empieza con una asignación inicial sobre la que realiza modificaciones hasta que converge.

Para detectar datos ruidosos u outliers empleamos el algoritmo NC *Noise-Clustering*. Este método consiste en añadir a los  $K$  clusters iniciales uno adicional llamado “cluster ruidoso”, asociado a una distancia fija  $\rho$  respecto a todos los objetos. El parámetro  $\rho$  controla la cantidad de datos que serán considerados como outliers. La pertenencia  $u_{i*}$  de un objeto  $i$  al cluster ruidoso se calcula como:

$$u_{i*} = 1 - \sum_{j=1}^K u_{ij} \quad i = 1, \dots, n \quad (4.8)$$

Por tanto, la función objetivo que minimiza el algoritmo NC no es más que una combinación del cálculo de pertenencia de objetos al cluster ruidoso y la que minimizaba el algoritmo FKM:

$$J_{NC}(U, V) = \sum_{i=1}^n \sum_{j=i}^K u_{ij}^\beta d_{ij}^2 + \sum_{i=1}^K \rho^2 u_{i*}^\beta \quad (4.9)$$

### *El algoritmo EKM (Evidential K-means)*

Es posible obtener una versión credibilística del algoritmo NC reemplazando la matriz asociada a la partición difusa  $U$  con una partición de creencia, que notaremos con  $M$ . En este contexto, el conocimiento parcial asociado a la pertenencia de un objeto a una clase, viene representado por una función de masa aplicada al conjunto  $C$  de posibles clases. Por tanto, la masa de creencia puede ser asignada a cualquier subconjunto  $A$  de  $C$ , y no sólo a elementos únicos de  $C$ . Este esquema hace posible modelar una amplia variedad de circunstancias, que van de la completa ignorancia sobre el conjunto de datos hasta la completa certeza sobre el mismo.

Para ilustrar estas ideas se propone el siguiente ejemplo: consideramos un conjunto de cuatro objetos que deben ser clasificados en dos clases. La Tabla 4.1 contiene la partición de creencia asociada a estos datos. La clase del primer objeto es conocida con certeza, ya que su masa de creencia está asignada a un solo elemento de  $C$ . Por el contrario la clase del segundo objeto es completamente desconocida. La masa de creencia del tercer objeto está repartida entre dos elementos de  $C$ , por tanto tenemos conocimiento probabilístico sobre la clase a la que pertenece. El último objeto representa un outlier, ya que su masa de creencia está asignada al conjunto vacío.

Evidential K-means (EKM) es uno de los algoritmos que opera con una partición de creencia obtenida en base a los datos. Si tomamos  $m_{ij}$  como el grado de creencia de que el objeto  $x_i$  pertenece al subconjunto  $A_j \subseteq C$ , obtener una partición de creencia implica determinar, para cada  $x_i$ , las cantidades  $m_{ij} = m_i(A_j) \quad \forall A_j \neq \emptyset, A_j \subseteq C$ . De esta

Ejemplo de partición de creencia				
$A$	$m_1(A)$	$m_2(A)$	$m_3(A)$	$m_4(A)$
$\emptyset$	0	0	0	1
$\{c_1\}$	1	0	0.3	0
$\{c_2\}$	0	0	0.7	0
$C$	0	1	0	0

Tabla 4.1: Ejemplo de partición de creencia [22]

manera, cuando la distancia  $d_{ij}$  entre  $x_i$  y  $A_j$  es alta (baja),  $m_{ij}$  será un valor bajo (alto).

Tal y como sucedía en Fuzzy K-means (FKM), cada clase  $c_l$  está representada por un centroide  $v_l \in \mathbb{R}^p$ . Entonces, para cada subconjunto  $A_j$  de  $C$  distinto de  $\emptyset$  se calcula su centroide  $\bar{v}_j$  como el baricentro de los centros asociados a las clases presentes en  $A_j$ :

$$\bar{v}_j = \frac{1}{|A_j|} \sum_{l=1}^K S_{lj} v_l \quad (4.10)$$

donde el valor  $S_{lj}$  viene definido por:

$$S_{lj} = \begin{cases} 1 & \text{si } c_l \in A_j \\ 0 & \text{otro caso} \end{cases} \quad (4.11)$$

La distancia  $d_{ij}$  entre la instancia  $x_i$  y el conjunto focal  $A_j$  se calcula como:

$$d_{ij} = ||x_i - \bar{v}_j|| \quad (4.12)$$

Entonces, el algoritmo EKM calcula las matrices  $M$  y  $V$  tratando de minimizar un criterio similar al del algoritmo NC:

$$J_{EKM}(M, V) = \frac{1}{2^c n} \sum_{i=1}^n \sum_{A_j \neq \emptyset} |A_j|^\alpha m_{ij}^\beta d_{ij}^2 + \sum_{i=1}^n \rho^2 m_{i\emptyset}^\beta \quad (4.13)$$

sujeto a las restricciones  $m_{ij} \geq 0 \ \forall i, j$ , y  $m_{i\emptyset} \geq 0 \ \forall i$ , y:

$$\sum_{j/A_j \subseteq C, A_j \neq \emptyset} m_{ij} + m_{i\emptyset} = 1 \quad \forall i = 1, n \quad (4.14)$$

Donde  $m_{i\emptyset}$  denota la cantidad de masa de creencia de la instancia  $x_i$  asignada al conjunto vacío. El parámetro  $\rho$  representa la distancia de cualquier objeto al conjunto vacío, y el parámetro  $\alpha$  se introduce para controlar la penalización por asignar objetos a conjuntos con alta cardinalidad.

Gracias a la restricción expuesta en la ecuación 4.14, podemos obtener equivalencia entre los algoritmos NC y EKM. El algoritmo EKM

asigna una gran masa de creencia al conjunto vacío para un objeto dado cuando éste se encuentra lejos de todos los subconjuntos  $A_j$ .

Como en FKM y NC, la partición de creencia se calcula aplicando un proceso de optimización iterativo, que actualiza las masas y los centroides de forma alterna. La regla de optimización de  $M$  es muy similar a su homóloga en NC, excepto por el número de valores  $m_{ij}$  a calcular, que en este caso es  $2^c$ , en lugar de los  $K + 1$  grados de pertenencia que eran necesarios en NC. De esta manera, la función de masa queda definida como:

$$m_{ij} = \frac{|A_j|^{-\alpha/(\beta-1)} d_{ij}^{-2/(\beta-1)}}{\sum_{A_l \neq \emptyset} |A_l|^{-\alpha/(\beta-1)} d_{ij}^{-2/(\beta-1)} + \rho^{-2/(\beta-1)}} \quad (4.15)$$

y

$$m_{i\emptyset} = 1 - \sum_{A_j \neq \emptyset} m_{ij} \quad i = 1, n \quad (4.16)$$

Por otra parte, la regla de actualización de los centroides resulta un poco más compleja. Requiere resolver un sistema de ecuaciones lineal en cada paso del proceso, en el que cada columna de  $V$  es la solución para un sistema lineal de  $K$  ecuaciones y  $K$  incógnitas. Tomamos  $B$  como la matriz de tamaño  $(K \times p)$  definida por:

$$B_{lq} = \sum_{i=1}^n X_{iq} \sum_{A_j \ni c_l} |A_j|^{\alpha-1} m_{ij}^\beta \quad l = 1, K \quad q = 1, p \quad (4.17)$$

y la matriz  $H$  de tamaño  $(K \times K)$  como:

$$H_{lt} = \sum_i \sum_{A_j \ni \{c_t, c_l\}} |A_j|^{\alpha-2} m_{ij}^\beta \quad t, l = 1, K \quad (4.18)$$

de forma que  $V$  es la solución del sistema de ecuaciones lineal  $H \times V = B$ , que puede ser resuelto mediante técnicas estándar.

### *Incorporación de restricciones a EKM*

Una vez definidos los elementos que conforman el algoritmo EKM, debemos incorporar las restricciones a nivel de instancia al marco de las funciones de creencia, para integrarlas en el cálculo de la partición de creencia.

Siendo  $x_i$  y  $x_j$  dos instancias, podemos calcular la función de masa conjunta en el producto cartesiano  $C \times C = C^2$  conociendo sus funciones de masa particulares  $m_i$  y  $m_j$ :

$$\begin{aligned} m_{i \times j}(A \times B) &= m_i(A)m_j(B) \quad A, B \subseteq C, A \neq \emptyset, B \neq \emptyset \\ m_{i \times j}(\emptyset) &= m_i(\emptyset) + m_j(\emptyset) - m_i(\emptyset)m_j(\emptyset) \end{aligned} \quad (4.19)$$

Conociendo  $m_{i \times j}$  es posible calcular la plausibilidad asociada a que los objetos  $x_i$  y  $x_j$  pertenezcan a la misma clase que, en el espacio  $C^2$ , corresponde al subconjunto  $\theta = \{(c_1, c_1), (c_2, c_2), \dots, (c_K, c_K)\}$ . El caso contrario corresponde al complemento de  $\theta$ , es decir,  $\bar{\theta}$ :

$$\begin{aligned} pl_{i \times j}(\theta) &= \sum_{A \cap B \neq \emptyset} m_i(A)m_j(B) \\ pl_{i \times j}(\bar{\theta}) &= 1 - m_{i \times j}(\emptyset) - \sum_{l=1}^K m_i(\{c_l\})m_j(\{c_l\}) \end{aligned} \quad (4.20)$$

### *Función objetivo de CEKM*

Asumimos ahora que la partición de creencia es desconocida, y que disponemos del conjunto de restricciones. En tal caso será necesario buscar una partición de creencia que considere las similitudes y diferencias obtenidas en base a los datos, así como las restricciones. Para ello debemos buscar que  $pl_{i \times j}(\theta)$  sea tan bajo como sea posible si  $(x_i, x_j) \in CL$ , así como que  $pl_{i \times j}(\bar{\theta})$  lo sea si  $(x_i, x_j) \in ML$ . A tal fin, integramos una penalización en el criterio de optimización de EKM de la siguiente manera:

$$J_{CONST} = \frac{1}{|R|} \left[ \sum_{(x_i, x_j) \in ML} pl_{i \times j}(\bar{\theta}) + \sum_{(x_i, x_j) \in CL} pl_{i \times j}(\theta) \right] \quad (4.21)$$

De esta forma, la función objetivo a minimizar pasa a ser:

$$J_{CEKM}(M, V) = (1 - \xi)J_{EKM}(M, V) + \xi J_{CONST} \quad (4.22)$$

donde el parámetro  $\xi \in [0, 1]$  se utiliza para controlar el compromiso entre las restricciones y el modelo geométrico asociado a la métrica de distancia.

### *Proceso de optimización de CEKM*

De igual forma que en FKM, NC y EKM, el modelo de optimización de CEKM consiste en actualizar  $M$  y  $V$  de forma alterna. Cabe destacar que el término de penalización añadido a CEKM no depende de los centroides de los clusters, y por tanto se pueden aplicar el mismo modelo de actualización que en EKM (ecuaciones 4.17 y 4.18). Generalmente, el problema es mucho más complejo para las masas de creencia. Sin embargo fijando  $\beta = 2$ , la función objetivo 4.22 pasa a ser cuadrática respecto a  $m_{ij}$ . Con esto, y como las restricciones son lineales, se pueden emplear algoritmos de programación cuadrática para resolver el problema de la actualización de las masas de creencia. El proceso de cálculo asociado a CEKM queda resumido en el Algoritmo 2.

---

**Algoritmo 2:** Constrained Evidential K-means (CEKM)
 

---

**Entrada:** Conjunto de datos  $X$ , conjunto de restricciones  $R$ , número de clusters resultantes  $K$

**Salida:** Partición de creencia  $M$ , centroides  $V$

**función** CEKM( $X, R, K$ ) **begin**

1. Sean  $V = \{v_1, \dots, v_K\}$  los centroides iniciales.
2. Actualizar las masas ( $M$ ) resolviendo el problema de programación cuadrática definido por 4.22 sujeto a 4.14
3. Actualizar los centroides ( $V$ ) resolviendo el sistema de ecuaciones lineal definido por 4.17 y 4.18 con  $\beta = 2$
4. Iterar entre (2.) y (3.) hasta que no haya cambios significativos en  $V$ .
5. **return**  $M, V$

**end**

---

En la mayoría de las ocasiones se requiere como resultado de un algoritmo de clustering una partición fuerte del conjunto de datos, y no una partición de creencia. Podemos obtener una partición difusa calculando la probabilidad pignística  $BetP_i(\{c_j\})$  en base a cada función de masa  $m_i$  aplicando la ecuación 4.6, e interpretar este valor como el grado de pertenencia del objeto  $i$  al cluster  $j$ . Una vez obtenida la partición difusa podemos procesarla como precise el problema para obtener la partición fuerte. La manera más común de hacerlo es asignar cada instancia al cluster para el que presente un mayor grado de pertenencia.

Otra manera de extraer información de la partición de creencia es asignar cada objeto al subconjunto de clases que contengan un mayor porcentaje de su masa de creencia. De esta forma obtenemos una partición de creencia fuerte de, como mucho,  $2^K$  clusters. A partir de esta partición es posible discernir que objetos deben ser asignados a un único cluster sin ambigüedad, además de detectar aquellos que se encuentran en la frontera de dos o más clusters, que a menudo suelen ser relevantes.

### LINEAR CONSTRAINED VECTOR QUANTIZATION ERROR (LCVQE)

Tomando como base el trabajo de Pelleg y Dorit (2007) [23], es necesario realizar una introducción al algoritmo Constrained Vector Quantization Error (CVQE) antes de detallar el algoritmo LCVQE, puesto que este último consiste en una modificación sobre el primero para mejorar, principalmente, su orden de complejidad.

#### *El algoritmo CVQE*

El algoritmo CVQE consiste en una generalización del algoritmo K-medias (KM, apéndice A) para incluir las restricciones. En el algoritmo KM los centroides  $v_i$  se actualiza en cada iteración siguiendo la regla:

$$v_i = \frac{1}{|c_i|} \sum_{x_i \in c_i} x_i \quad (4.23)$$

Que no es más que un promedio de todas las instancias asignadas al cluster asociado al centroide  $v_i$ . Tras la actualización, se recalculan las asignaciones de forma que que cada instancia este asociada al cluster más cercano. Esto deriva en una regla que minimiza la función Vector Quantization Error (VQE):

$$VQE = \frac{1}{2} \sum_{j=1}^K \sum_{x_i \in c_j} (v_j - x_i)^2 \quad (4.24)$$

CVQE modifica la función VQE, añadiendo a la misma un término de penalización que considera las restricciones incumplidas. Por simplicidad, notaremos el cardinal de  $ML$  con  $ml$ , así como el de  $CL$  con  $cl$ , y definiremos el conjunto de restricciones  $R$  como una lista de parejas de instancias:

$$\{(x_1(i), x_2(i))\}_{i=1}^{ml+cl} \quad (4.25)$$

Además definimos  $M$  como la función que, dada una instancia, devuelve el índice del cluster que la contiene:  $M = \{j | x \in c_j\}$ , así como las funciones  $g(i) = M(x_1(i))$  y  $g'(i) = M(x_2(i))$ . Por otra parte, definimos  $h(i)$  como la función que devuelve el centroide más cercano al centroide  $v_i$ . Finalmente definimos  $vl(i)$  de manera que indica si la restricción  $i$ -ésima ha sido violada, por tanto, para  $i = 1, \dots, ml$   $vl(i) = 1 \leftrightarrow g(i) \neq g'(i)$ , de forma similar  $i = ml + 1, \dots, ml + cl$   $vl(i) = 1 \leftrightarrow g(i) = g'(i)$ . Definidas estas funciones, la regla de actualización de CVQE es:

$$v_j = \frac{1}{N_j} \left[ \sum_{x_i \in c_j} x_i + \sum_{l=1, g(l)=j}^{ml} vl(l) v_{g'(l)} + \sum_{l=ml+1, g(l)=j}^{ml+cl} vl(l) v_{h(g'(l))} \right] \quad (4.26)$$

donde  $N_j$  viene definido por:

$$N_j = |c_j| + \sum_{l=1, g(l)=j}^{ml+cl} vl(l) \quad (4.27)$$

De manera intuitiva, para las restricciones *ML* incumplidas, uno de los dos centroides afectados se mueve hacia el otro, mientras que para las *CL* incumplidas se mueve una de las instancias hacia el siguiente centroide más cercano a su cluster. De forma similar a KM, cada instancia se reasigna después de cada iteración para minimizar la función de error  $CVQE = \sum_{j=1}^K CVQE_j$ , donde  $CVQE_j$  es:

$$CVQE_j = \frac{1}{2} \sum_{x_i \in c_j} T_{j,1} + \frac{1}{2} \sum_{l=1, g(l)=j}^{ml} T_{j,2} + \frac{1}{2} \sum_{l=ml+1, g(l)=j}^{ml+cl} T_{j,3} \quad (4.28)$$

donde los términos  $T_{j,1}$ ,  $T_{j,2}$  y  $T_{j,3}$  vienen definidos como:

$$\begin{aligned} T_{j,1} &= (v_j - x_i)^2 \\ T_{j,2} &= \left[ (v_j - v_{g'(l)})^2 \cdot vl(l) \right] \\ T_{j,3} &= \left[ (v_j - v_{h(g'(l))})^2 \cdot vl(l) \right] \end{aligned} \quad (4.29)$$

Cabe destacar que, al contrario de lo que sucede en K-medias (KM), en CVQE, un cluster  $c_i$  puede contener instancias para las que el centroide  $v_i$  no es el más cercano a ella.

En cada paso, el algoritmo CVQE asigna una par de instancias implicadas en una restricción de manera que se minimiza la función  $CVQE$ .

A continuación se exponen algunas de las características del algoritmo CVQE que resultan relevantes para la compresión de LCVQE.

En primer lugar, el orden en el que se especifican las instancias implicadas en las restricciones es relevante. Consideremos una restricción formada por las instancias  $(x_1(l), x_2(l))$ , de forma que  $x_1(l) \in c_{g(l)}$ , y  $x_2(l) \in c_{g'(l)}$ . Solo  $c_{g(l)}$  se ve afectado por violar la restricción, mientras que sobre  $c_{g'(l)}$  no se aplica ninguna cambio. Esta regla se cumple tanto para las restricciones Must-Link (ML) como para las Cannot-Link (CL).

El segundo lugar, determinar la asignación que minimiza la función de error requiere  $\mathcal{O}(K^2)$  cálculos para cada restricción. Por tanto, el método puede resultar computacionalmente pesado para grandes conjuntos de restricciones o clusters. Además, no es posible descartar ninguna opción de los cálculos aparte de las triviales. Para exemplificar esto consideramos la Figura 4.1. En ella, el par  $(x, y)$  es una restricción ML, y los centroides existentes son  $\{v_1, \dots, v_6\}$ . Dependiendo de los valores  $R$ ,  $\delta$  y  $\epsilon$  las instancias pueden ser asignadas a los centroides  $(v_1, v_2)$ ,  $(v_3, v_4)$ ,  $(v_5, v_6)$ . Por tanto, las  $K^2$  opciones deben ser consideradas para cada restricción al decidir una asignación.

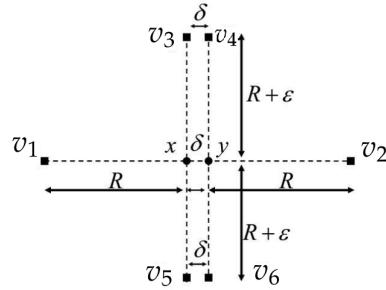


Figura 4.1: Ejemplo de CVQE. [23]

La última observación esta relacionada con el hecho de que la penalización por violar restricciones depende de la distancia entre los centroides implicados en ella, pero no de la distancia entre las instancias. Para mostrar el problema que esto supone tomamos la Figura 4.2, en la que se presentan dos problemas de clustering. Para ambos, los centroides existentes son  $v_1$  y  $v_2$ , mientras que un problema incluye la restricción  $ML(x_1, y)$  y el otro  $ML(x_2, y)$ .

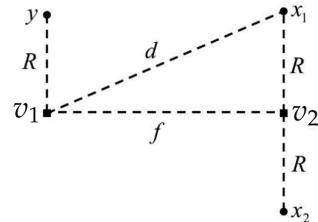


Figura 4.2: Ejemplo de CVQE. [23]

La Tabla 4.2 recoge las asignaciones posibles en cada caso, así como el valor de la función  $CVQE$ . Vemos que, independientemente del valor de  $d$  y  $f$ , ambos problemas tiene la misma solución, mientras que intuitivamente diríamos que violar la restricción  $ML(x_1, y)$  debería acarrear una penalización mayor que violar  $ML(x_2, y)$ . Es más, en ambos casos la acción que aplica  $CVQE$  es la misma, mover  $v_1$  hacia  $v_2$  en la recta que los une.

Valores de $CVQE$			
Restricción	$y \in c_1, x_i \in c_2$	$x_i, y \in c_1$	$x_i, y \in c_2$
$ML(x_1, y)$	$R^2 + R^2 + f^2$	$R^2 + f^2$	$R^2 + d^2$
$ML(x_2, y)$	$R^2 + R^2 + f^2$	$R^2 + f^2$	$R^2 + d^2$

Tabla 4.2: Valores de  $CVQE$  para el ejemplo [22]

### El algoritmo LCVQE

El algoritmo LCVQE minimiza una función similar a la que minimiza CVQE. Con la diferencia de que, para cada asignación, LCVQE considera, como mucho, los dos clusters más naturalmente apropiados para la asignación. Por ello, la complejidad del algoritmo es independiente de  $K$ , al contrario de lo que sucedía en CVQE.

De manera intuitiva, las restricciones ML violadas modifican la actualización de los centroides para desplazar estos hacia la instancia opuesta. Para las restricciones CL violadas, se determina cual es la instancia mas lejana al centroide común, sirviendo esta como guía para mover el segundo centroide mas cercano hacia ella. Con esto tenemos que, en LCVQE, las restricciones son simétricas, es decir, no es relevante el orden en el que se especifiquen las instancias implicadas en ellas.

Para formalizar la regla de actualización es necesario definir nuevas funciones.  $L_j(i)$  devuelve la instancia de entre  $x_1(i)$  y  $x_2(i)$  que más lejos se encuentre del centroide  $v_j$ .  $MM(x)$  devuelve el centroide más cercano a  $x$  distinto de  $v_{M(x)}$ . Con esto la regla de actualización queda definida como:

$$\begin{aligned} v_j &= \frac{1}{N_j} \left[ \sum_{x_i \in c_j} x_i + \frac{1}{2} S_1 + \frac{1}{2} S_2 + S_3 \right] \\ S_1 &= \sum_{l=1, g(l)=j}^{ml} v_l(l) \cdot x_2(l) \\ S_2 &= \sum_{l=1, g'(l)=j}^{ml} v_l(l) \cdot x_1(l) \\ S_3 &= \sum_{l=ml+1, j=MM(L_{M(x_1(l))}(l))}^{ml+cl} v_l(l) \cdot L_{M(x_1(l))}(l) \end{aligned} \quad (4.30)$$

donde  $N_j$  en este caso se calcula como:

$$\begin{aligned} N_j &= |c_j| + \frac{1}{2} \sum_{l=1, g(l)=j}^{ml} v_l(l) + \frac{1}{2} \sum_{l=1, g'(l)=j}^{ml} v_l(l) + \\ &\quad \sum_{l=ml+1, j=MM(L_{M(x_1(l))}(l))}^{ml+cl} v_l(l) \end{aligned} \quad (4.31)$$

Esta regla de actualización minimiza la función de error:

$$\begin{aligned} E_j &= \frac{1}{2} \sum_{x_i \in c_j} T_{j,1} + \frac{1}{2} \sum_{l=1, g(l)=j}^{ml} T_{j,2} + \frac{1}{2} \sum_{l=ml+1, g'(l)=j}^{ml} T_{j,3} + \\ &\quad \frac{1}{2} \sum_{l=ml+1, j=MM(L_{M(x_1(l))}(l))}^{ml+cl} T_{j,4} \end{aligned} \quad (4.32)$$

donde los términos  $T$  vienen definidos como:

$$\begin{aligned} T_{j,1} &= (v_j - x_i)^2 \\ T_{j,2} &= \left[ \frac{1}{2}(v_j - x_2(l))^2 \cdot vl(l) \right] \\ T_{j,3} &= \left[ \frac{1}{2}(v_j - x_1(l))^2 \cdot vl(l) \right] \\ T_{j,4} &= \left[ (v_j - L_{M(x_1(l))}(l))^2 \cdot vl(l) \right] \end{aligned} \quad (4.33)$$

El Algoritmo 3 recoge de manera resumida el proceso que sigue LCVQE para obtener una partición de los datos.

El criterio de convergencia puede estar basado en la diferencia de los centroides entre iteraciones sucesivas, así como en criterios de asignación de instancias a clusters o evaluaciones totales de la función de error.

El algoritmo LCVQE requiere  $\mathcal{O}(p)$  operaciones en cada paso, con  $p$  el número de dimensiones, ya que solo se consideran tres posibles asignaciones, independientemente del valor de  $K$ . Por tanto LCVQE supera en eficiencia a CVQE, que era altamente sensible al número de clusters y de restricciones.

Finalmente, podemos estudiar como se comporta el algoritmo frente al ejemplo de la Figura 4.2, la Tabla 4.3 muestra las asignaciones posibles en cada caso, así como el los valores LCVQE. Asumimos que la asignación en ambos problemas es  $x_i \in c_2$  e  $y \in c_1$ . En el caso de la restricción  $(x_1, y)$  el centroide se actualiza según la regla  $c_1 = (y + \frac{1}{2}x_1)/1,5$ , y en el caso de  $(x_1, y)$  tenemos que  $c_1 = (y + \frac{1}{2}x_2)/1,5$ , resultados intuitivamente mejores que los que obteníamos con LCVQE, ya que el centroide se mueve hacia la distancia media de las instancias en lugar de hacia el otro centroide  $c_2$ .

Valores de LCVQE			
Restricción	$y \in c_1, x_i \in c_2$	$x_i, y \in c_1$	$x_i, y \in c_2$
$ML(x_1, y)$	$(d^2 + d^2)/2$	$d^2$	$d^2$
$ML(x_2, y)$	$(d^2 + d^2)/2$	$d^2$	$d^2$

Tabla 4.3: Valores de LCVQE para el ejemplo [22]

---

**Algoritmo 3:** Linear Constrained Vector Quantization Error (LCVQE)

---

**Entrada:** Conjunto de datos  $X$ , conjunto de restricciones  $R$ , centroides iniciales  $V$ , número de clusters  $K$

**Salida:** Partición  $P$  del conjunto de datos  $X$ , centroides  $V$

**función** LCVQE( $X, R, K$ ) **begin**

1. Inicialización de  $GMLV_j = GCLV_j = \emptyset \forall j \in \{1, \dots, K\}$

2. Inicialización de  $C$  por la regla del centroide mas cercano.

3. Para cada  $\{(x_1(i), x_2(i))\} \in ML$  con  $v_j$  el centroide mas cercano a  $x_1(i)$  y  $v_n$  el centroide mas cercano a  $x_2(i)$  calcular:

$$(a) \frac{1}{2} [(x_1(l) - v_j)^2 + (x_2(l) - v_n)^2] + \frac{1}{4} [(x_1(l) - v_n)^2 + (x_2(l) - v_j)^2]$$

$$(b) \frac{1}{2}(x_1(l) - v_j)^2 + \frac{1}{2}(x_2(l) - v_j)^2$$

$$(c) \frac{1}{2}(x_1(l) - v_n)^2 + \frac{1}{2}(x_2(l) - v_n)^2$$

Si (a) es minimal  $\rightarrow GMLV_j = GMLV_j \cup x_2(l)$  y  $GMLV_n = GMLV_n \cup x_1(l)$ . Asignar  $x_1$  a  $c_j$  y  $x_2$  a  $c_n$ .

Si (b) es minimal  $\rightarrow$  asignar  $x_1$  y  $x_2$  a  $c_j$ .

Si (c) es minimal  $\rightarrow$  asignar  $x_1$  y  $x_2$  a  $c_n$ .

4. Para cada  $\{(x_1(i), x_2(i))\} \in CL$  con  $v_j$  el centroide mas cercano a  $x_1(i)$  y  $v_n$  el centroide mas cercano a  $x_2(i)$ , y siendo  $Max_n(l) = argmax_{x_i(l)} (x_i(l) - v_n)^2$ . Tomamos  $j$  como el centroide mas cercano a  $Max_n(l)$  que no sea  $n$  y calculamos:

$$(a) \frac{1}{2}(x_1(l) - v_j)^2 + \frac{1}{2}(x_2(l) - v_j)^2 + \frac{1}{2}(Max_n(l) - v_{MM(Max_j(l))})^2$$

$$(b) \frac{1}{2} [(x_1(l) - v_j)^2 + (x_2(l) - v_n)^2]$$

Si (a) es minimal  $\rightarrow$

$GCLV_{MM(Max_j(l))} = GCLV_{MM(Max_j(l))} \cup Max_j(l)$  y asignar  $x_1$  y  $x_2$  a  $c_j$ .

Si (b) es minimal  $\rightarrow$  Asignar  $x_1$  a  $c_j$  y  $x_2$  a  $c_n$ .

5. Actualizar cada centroide  $v_j$  en  $V$  como sigue:

$$v_j = \frac{1}{N_j} [Sum(c_j) + \frac{1}{2} Sum(GMLV_l) + Sum(GCLV_j)]$$

6. Iterar entre (2.) y (5.) hasta converger

7. **return**  $C, V$

**end**

---

### RELATIONAL DIRICHLET PROCESS - MEANS (RDP - MEANS)

Tomamos como referencia principal el trabajo de Daniel Khashabi et al. (2015) [24]. En él, los autores incorporan las restricciones desde una perspectiva distinta a las anteriores. Así, modelan el conjunto de instancias y el conjunto de restricciones de manera diferente, es por ello que llaman Two Views Clustering (TVClust) al método base para Relational Dirichlet Process - Means (RDPM).

Para ser mas específicos, TVClust combina una mezcla de Procesos de Dirichlet, aplicados sobre el conjunto de instancias, y una interpretación del conjunto de restricciones que considera las mismas como un grafo aleatorio. Además, los autores derivan este modelo, basándose en el algoritmo Dirichlet Process - Means (DPM) [25] combinado con el modelo de las restricciones, para obtener un algoritmo determinista que incorpora las mismas a DPM, esto es, el algoritmo Relational Dirichlet Process - Means (RDPM).

#### *El modelo Bayesiano no paramétrico*

Introducimos las particularidades relacionadas con el modelo de datos sobre el que aplicaremos clustering. El conjunto de instancias  $X$  viene definido tal y como se especifica al inicio de la sección 4, y definimos el conjunto de restricciones  $R$  como una matriz simétrica de  $n \times n$ . En  $R$  se encuentran almacenadas todas las posibles restricciones que involucran a parejas de instancias. Así, tomando  $x_i$  y  $x_j$  como dos instancias, basta con obtener el valor  $R_{i,j}$  en la matriz para saber la relación que existe entre ellas. Si  $R_{i,j} = 1$ , entonces existe una restricción de tipo Must-Link (ML) entre  $x_i$  y  $x_j$ , si  $R_{i,j} = 0$  la restricción será de tipo Cannot-Link (CL), y no existirá relación entre las instancias en cualquier otro caso.

Consideramos los dos conjuntos de datos de los que disponemos,  $X$  y  $R$ , como dos formas distintas de ver la estructura de clusters subyacente. Cabe destacar que cualquiera de estas dos estructuras es suficiente para llevar a cabo procesos de clustering clásicos, ya sean métodos como K-medias (KM) (sección A), en el caso de  $X$ , o *normalized graph-cut* en el caso de  $R$ .

La aproximación mediante TVClust consiste en agregar información de los dos modelos mediante un enfoque Bayesiano, de manera que sea posible alcanzar un consenso entre ambos, que tenga como resultado la partición de  $X$ . Dada la estructura de clustering latente, que se supone común, los datos de ambas interpretaciones se modelan mediante dos procesos generativos distintos:  $X$  se modela mediante Mezcla de Procesos de Dirichlet, y  $R$  se modela mediante un grafo aleatorio.

Considerar modelos distintos para entender los datos y las restricciones es de especial utilidad cuando ninguno de los dos puede ser

tomado como completamente fiable. Mientras que otros métodos de clustering como COP-K-medias (sección 4.2) confían en la exactitud de las restricciones, TVClust hace una interpretación relajada de las mismas, haciéndolo más robusto ante errores. Podría decirse que, en TVClust,  $R_{i,j} = 1$  equivale a una restricción *may-link*, mientras que  $R_{i,j} = 0$  se asociaría a una *may-not-link*, en contraste con las ya conocidas Must-Link (ML) y Cannot-Link (CL).

### Modelo para las instancias

Como ya hemos visto, utilizamos una Mezcla de Procesos de Dirichlet como modelo de clustering subyacente para el conjunto de datos X. Tomamos  $\theta_i$  como el parámetro del modelo asociado a la instancia  $x_i$ , que es modelado como una muestra independiente e idénticamente distribuida<sup>1</sup> de una distribución aleatoria G, que se obtiene en base a un Proceso de Dirichlet  $DP(\alpha, G_0)$ :

$$\{\theta_0, \dots, \theta_n\} \text{ t.q. } G \stackrel{iid}{\sim} G, \quad G \sim DP(\alpha, G_0) \quad (4.34)$$

Con esto, conociendo la distribución de  $\theta_{\setminus i}$ , definida esta como  $\theta_{\setminus i} = \{\theta_0, \dots, \theta_{i-1}, \theta_{i+1}, \dots, \theta_n\}$  podemos conocer la de  $\theta_i$  aplicando el esquema de Balckwell-MacQueen [26]

$$p(\theta_i | \theta_{\setminus i}) \propto \sum_{k=1}^K n_{-i,k} \delta_{\theta_k^*}(\theta_i) + \alpha G_0(\theta_i) \quad (4.35)$$

donde asumimos que existen K<sup>2</sup> valores únicos entre  $\theta_{\setminus i}$  notados como  $\{\theta_1^*, \dots, \theta_K^*\}$ ,  $\delta_{\theta_k^*}(\cdot)$  es la función delta de Kronecker, y  $n_{-i,k}$  es el número de instancias agrupadas en el cluster  $c_k$  excluyendo la instancia  $i$ . Así, la ecuación 4.35 puede entenderse dentro de los métodos de clustering tradicionales en el sentido de que con una probabilidad positiva,  $\theta_i$  tomará uno de los valores del conjunto  $\{\theta_1^*, \dots, \theta_K^*\}$ , esto es, la instancia asociada se incluirá en uno de los clusters presentes. Para simplificar este concepto podemos emplear la Metáfora del Restaurante Chino, propuesta por Aldous (1983), en la que asignar  $\theta_i$  a un cluster es análogo a sentar un cliente en una mesa del restaurante. El cliente puede sentarse en una mesa ya ocupada u ocupar una que estaba vacía.

Dado  $\theta_i$ , utilizamos una familia de funciones paramétricas  $p(x_i | \theta_i)$  para modelar la instancia  $x_i$ , en concreto tomaremos las familia de las exponenciales:

$$p(x | \theta) = \exp(\langle T(x), \theta \rangle - \psi(\theta) - h(x)) \quad (4.36)$$

donde  $\psi(\theta)$  es la función *log-partition* para la familia exponencial  $\psi(\theta) = \log \int \exp(\langle x, \theta \rangle - h(x)) dx$ , y  $T(x)$  es el vector de estimadores

---

<sup>1</sup> iid es la abreviatura en inglés para “independiente e idénticamente distribuida”

<sup>2</sup> La interpretación de K en esta sección es distinta a la propuesta al inicio de la sección

suficientes de la distribución de probabilidad. En el caso de la familia exponencial la dimensión del vector de estimadores suficientes es igual al número de parámetros estimables. Por simplicidad asumimos que  $x$  es el vector de estimadores suficientes aumentado y obtenemos:

$$p(x|\theta) = \exp(\langle x, \theta \rangle - \psi(\theta) - h(x)) \quad (4.37)$$

Con esta formulación obtenemos:

$$\begin{aligned} \mathbb{E}_p[x] &= \nabla_{\theta}\psi(\theta) \\ Cov_p[x] &= \nabla_{\theta}^2\psi(\theta) \end{aligned} \quad (4.38)$$

Y por conveniencia escogeremos la medida base para  $G_0$  en  $\text{DP}(\alpha, G_0)$  de entre la familia de conjugados, que toma la siguiente forma:

$$dG_0(\theta|\tau, \eta) = \exp(\langle \theta, \tau \rangle - \eta\psi(\theta) - m(\tau, \eta)) \quad (4.39)$$

donde  $\tau$  y  $\eta$  son parámetros de la distribución base. Dada esta definición de probabilidad y conjugado, las posterior distribución sobre  $\theta$  será la familia de exponenciales de la misma forma que la inicial, pero escalando los parámetros a  $\tau + x$  y  $\eta + 1$  (para más detalles consultar [24]).

La familia exponencial contiene multitud de distribuciones empleadas en la práctica. Por ejemplo, las distribuciones Gaussianas se utilizan para modelar valores reales en un espacio  $\mathbb{R}^p$ .

### *Modelo para las restricciones*

Dado  $\theta_{1:n} = \{\theta_1, \dots, \theta_n\}$  podemos resumir las estructura del clustering mediante una matriz  $H$  con dimensiones  $n \times n$ , donde  $H_{i,j} = \delta_{\theta_i}(\theta_j)$ . Cabe destacar que la matriz  $H$  es distinta a la matriz  $R$ , ya que esta última representa las restricciones y puede ser entendida como una muestra aleatoria de la estructura de clustering contenida en  $H$ . Con esto, el objetivo es inferir  $H$  en base a  $R$ .

Modelamos  $R$  mediante el siguiente proceso generativo: con probabilidad  $p$ , una arista existente en el grafo descrito por  $H$  se conserva en  $R$ , y con probabilidad  $q$ , una falsa arista de  $H$  se añade a  $R$ . Por ejemplo, tomando  $(i, j) \in \{1, \dots, n\}$ :

$$\begin{cases} p(R_{i,j} = 0 | H_{i,j} = 1) = p \\ p(R_{i,j} = 1 | H_{i,j} = 0) = 1 - p \\ p(R_{i,j} = 0 | H_{i,j} = 0) = q \\ p(R_{i,j} = 1 | H_{i,j} = 0) = 1 - q \end{cases} \quad (4.40)$$

Dicho de otra forma, los valores  $p$  y  $q$  representan la credibilidad de los valores presentes en la matriz  $R$ , mientras que  $1 - p$  y  $1 - q$  representan la probabilidad de error. El valor concreto que se debe asignar

a  $p$  y  $q$  varía para cada problema; pueden ser obtenidos mediante conocimiento experto o con técnicas de aprendizaje desde datos.

### *Inferencia mediante muestreo de Gibbs*

Es posible derivar un esquema de muestro de Gibbs para el modelo de TVClust, que en esencia es similar al que emplea Dirichlet Process - Means (DPM). La distribución de muestreo que emplearemos viene definida por:

$$p(\theta_i | \theta_{\setminus i}, x_i, R, p, q) \propto \sum_{k=1}^K n_{-i,k} p(x_i | \theta_k^*) \delta_{\theta_k^*}(\theta_i) \left( \frac{p}{1-q} \right)^{f_k^i} \left( \frac{1-p}{q} \right)^{s_k^i} + \alpha p_{G_0}(x_i) p_{G_0}(\theta_i | x_i) \quad (4.41)$$

donde

$$\begin{cases} f_k^i = \#\{j : \theta_j = \theta_k^*, R_{i,j} = 1\} \\ s_k^i = \#\{j : \theta_j = \theta_k^*, R_{i,j} = 0\} \end{cases} \quad (4.42)$$

El proceso de cálculo y simplificación que deriva en estas ecuaciones se encuentra detallado en el trabajo tomado como base para la elaboración de esta sección [24].

Retomando la analogía del restaurante chino, podemos interpretar la distribución de muestreo de  $\theta_i$  de la siguiente manera: si la instancia  $i$  es *amiga* de la instancia  $j$ , entonces  $R_{i,j} = 1$ , por otra parte, si dos instancias son *desconocidas* entonces  $R_{i,j} = 0$ . Con esto, podemos interpretar el valor  $f_i^k$  como el número de amigos de la instancia  $i$  en la mesa  $k$ , y  $s_i^k$  como el número de desconocidos.

Tal y como hemos visto, los valores  $p$  y  $q$  representan la credibilidad de las restricciones, esto es, el nivel de confianza en que son correctas. Para representar un grado de confianza alto generalmente estableceremos que  $p > 1 - q$ . Con esto, y tomando en consideración la ecuación 4.41, la probabilidad de que una persona sea asignada a una mesa no solo aumenta con la popularidad de la misma, sino que también lo hace con el número de amigos de la instancia en la mesa ( $f_i^k$ ) y disminuye con el número de desconocidos ( $s_i^k$ ).

Una mejora aplicable al modelo consiste en modificar la regla de actualización del conjunto de parámetros  $\{\theta_1, \dots, \theta_n\}$ , de manera que, en lugar de actualizar los parámetros particulares de cada instancia de manera secuencial, se actualiza un conjunto de parámetros asociados a los clusters  $\{\theta_1^*, \dots, \theta_K^*\}$ , así como los indicadores de asignación de instancias a clusters (las etiquetas)  $\{z_1, \dots, z_n\}$  donde

$z_i \in \{1, \dots, K\}$  indica el cluster al que esta asignado la instancia  $i$ . Entonces la nueva regla de actualización es:

$$\begin{cases} p(z_i = k) \propto n_{-i,k} p(x_i, \theta_k^*) \left(\frac{p}{1-q}\right)^{f_k^i} \left(\frac{1-p}{q}\right)^{s_k^i} \\ p(z_i = k_{nuevo}) \propto \alpha \int p(x_i, \theta_k^*) dG_0 \end{cases} \quad (4.43)$$

### Reparametrización de la familia exponencial para RDP-means

Para comprender Relational Dirichlet Process - Means (RDPM) es necesario introducir el concepto de divergencia de Bregman, así como su relación con la familia de funciones exponenciales. Comenzamos con la definición formal de la divergencia de Bregman:

**Definición 4.1 Divergencia de Bregman (1967):** Definida una función estrictamente convexa  $\phi : \mathcal{S} \rightarrow \mathbb{R}$ , de forma que el dominio  $\mathcal{S} \subseteq \mathbb{R}^p$  es un conjunto convexo, y  $\phi$  es diferenciable en  $ri(\mathcal{S})$ , siendo este el interior relativo de  $\mathcal{S}$  donde su gradiente  $\nabla\phi$  existe. Dados dos puntos  $x, y \in \mathbb{R}^p$ , la divergencia de Bregman  $D_\phi(x, y) : \mathcal{S} \times ri(\mathcal{S}) \rightarrow [0, +\infty)$  viene definida como: [24]

$$D_\phi(x, y) = \phi(x) - \phi(y) - \langle x - y, \nabla\phi(y) \rangle$$

La divergencia de Bregman es una clase general de medidas de distancia, por ejemplo, tomando  $\phi$  como una función cuadrática, la divergencia de Bregman es equivalente a la distancia Euclidea [27].

Existe una aplicación biyectiva entre la familia exponencial y las divergencias de Bregman, tal y como demostraron Foster y Warmuth (2002) [28]. Dada esta conexión, Banerjee et al. (2005) [27] construyeron un algoritmo en base a K-medias (KM) que utiliza la divergencia de Bregman como medida de distancia, en lugar de la distancia Euclidea.

**Definición 4.2 El Conjugado de Legendre:** Para una función  $\psi(\cdot)$  definida sobre el dominio  $\mathbb{R}^p$ , se define como su conjugado convexo  $\psi^*(\cdot)$  como  $\psi^*(\mu) = \sum_{\theta \in \text{dom}(\psi)} \{\langle \mu, \theta \rangle - \psi(\theta)\}$ . Además, si la función  $\psi(\theta)$  es cerrada y convexa, entonces  $(\psi^*)^* = \psi$ . [24]

Se puede demostrar que la función *log-partition* de la familia exponencial es una función cerrada y convexa. Por tanto, existe una biyección entre el parámetro  $\mu$  del conjugado de Legendre  $\psi^*(\cdot)$ , y el parámetro de la familia exponencial,  $\theta$ , en la función *log-partition* definida para la familia exponencial en la ecuación 4.36. Con esto, podemos reescribir la probabilidad definida por la ecuación 4.37 usando la divergencia de Bregman y el conjugado de Legendre:

$$p(x|\theta) = p(x|\mu) = \exp(-D_{\psi^*}(x, \mu)) f_{\psi^*}(x) \quad (4.44)$$

donde  $f_{\psi^*}(x) = \exp(\psi^*(x) - h(x))$ . Una consecuencia de esta nueva parametrización es que la probabilidad asociada a cualquier instancia  $x$  está relacionada con cómo de lejos está la misma del cluster parametrizado con  $\mu$ , midiendo la distancia según la divergencia de Bregman  $D_{\psi^*}(x, \mu)$ .

De manera similar podemos reescribir la ecuación 4.39 en términos de la divergencia de Bregman y el conjugado de Legendre:

$$p(\theta|\tau, \eta) = p(\mu|\tau, \eta) = \exp\left(-\eta D_{\psi^*}\left(\frac{\tau}{\eta}, \mu\right)\right) g_{\psi^*}(\tau, \eta) \quad (4.45)$$

donde  $g_{\psi^*}(\tau, \eta) = \exp(\eta\psi(\theta) - m(\tau, \eta))$

### *Escalando las distribuciones para RDP-means*

**Lema 4.1 Jiang et al (2012) [25]:** *Dada la familia exponencial definida en la ecuación 4.36, definimos otra distribución de probabilidad con parámetro  $\tilde{\theta}$  y la función log-partition  $\tilde{\psi}(\cdot)$ , donde  $\tilde{\theta} = \gamma\theta$  y  $\tilde{\psi}(\tilde{\theta}) = \gamma\psi(\tilde{\theta}/\gamma)$ , entonces:*

1. *La distribución de probabilidad escalada  $\tilde{p}(\cdot)$  definida con parámetro  $\tilde{\theta}$  y función log-partition  $\tilde{\psi}(\cdot)$ , es una distribución de probabilidad propia que pertenece a la familia exponencial.*
2. *La media y la varianza de la distribución de probabilidad  $\tilde{p}(\cdot)$  son:*

$$\mathbb{E}_{\tilde{p}}(x) = \mathbb{E}_p(x), \quad Cov_{\tilde{p}}(x) = \frac{1}{\gamma} Cov_p(X)$$

3. *El conjugado de Legendre de  $\tilde{\psi}(\cdot)$  es  $\tilde{\psi}^*(\tilde{\theta}) = \gamma\psi^*(\tilde{\theta})$*

El resultado más importante derivado del lema 4.1 es que la covarianza  $Cov_p(x)$  escala con  $1/\gamma$ , que tiende a 0 para valores grandes de  $\gamma$ , mientras que la media  $\mathbb{E}_p(x)$  se mantiene invariante. Por tanto podemos obtener un algoritmo determinista cuando  $\gamma$  tiende a infinito.

Con esto, las ecuaciones 4.44 y 4.45 pueden ser escritas como:

$$\begin{aligned} \tilde{p}(x|\theta, \gamma) &= \tilde{p}(x|\mu, \gamma) = \exp(-\gamma D_{\psi^*}(x, \mu)) f_{\gamma\psi^*}(x) \\ \tilde{p}(\theta|\tau, \eta, \gamma) &= \tilde{p}(\tilde{\mu}|\tau, \eta, \gamma) = \exp\left(-\eta D_{\psi^*}\left(\frac{\tau}{\eta}, \mu\right)\right) g_{\gamma\psi^*}(\tau/\gamma, \eta/\gamma) \end{aligned} \quad (4.46)$$

### Asintóticos de *TVClust*

Usando la distribución escalada definida en la ecuación 4.46 podemos reescribir la regla de actualización de Gibbs descrita en la ecuación 4.43 como sigue:

$$\begin{cases} p(z_i = k) \propto n_{-i,k} \exp(-\gamma D_{\psi^*}(x_i, \mu_k)) \left(\frac{p}{1-q}\right)^{f_k^i} \left(\frac{1-p}{q}\right)^{s_k^i} \\ p(z_i = k_{nuevo}) \propto \alpha \int \tilde{p}(x_i|\theta) \tilde{p}(\theta|\tau, \eta) d\theta \end{cases} \quad (4.47)$$

Siguiendo el proceso de cálculo y simplificación detallado por Daniel Khashabi et al. (2015) [24], podemos transformar la ecuación 4.47 en:

$$\begin{cases} p(z_i = k) \propto n_{-i,k} \exp(-\gamma D_{\psi^*}(x_i, \mu_k)) - \xi_1 f_k^i + \xi_2 s_k^i \\ p(z_i = k_{nuevo}) \propto \nu(x_i; \tau, \eta, \gamma) \exp(-\gamma \lambda) \end{cases} \quad (4.48)$$

Donde  $\xi_1 = \ln\left(\frac{p}{1-q}\right)$  y  $\xi_2 = \ln\left(\frac{q}{1-p}\right)$ , esto es,  $\xi_1$  y  $\xi_2$  representan, respectivamente, la probabilidad de que exista o no exista una restricción.

Cuando  $\gamma$  tiende a infinito, el muestreador de Gibbs degenera en un algoritmo determinista, en el que, en cada iteración, la asignación de  $x_i$  a un cluster se determina comparando los  $K+1$  valores descritos por:

$$\{D_{\psi^*}(x_i, \mu_1) - \xi_1 f_1^i + \xi_2 s_1^i, \dots, D_{\psi^*}(x_i, \mu_K) - \xi_1 f_K^i + \xi_2 s_K^i, \lambda\} \quad (4.49)$$

Si el valor  $k$ -ésimo (con  $k = 1, \dots, K$ ) es el más pequeño, entonces  $x_i$  se asigna al cluster  $k$ -ésimo, por el contrario, si  $\lambda$  es el menor valor, se crea un cluster nuevo con  $x_i$  como única instancia asignada.

### Muestreo de los parámetros de los clusters

Daniel Khashabi et al. (2015) [24] demuestran que la función que muestra los parámetros para los clusters es equivalente a una media de las instancias que estos contiene bajo las condiciones adecuadas, esto es, tomando  $\gamma \rightarrow \infty$  y argumentos iguales para la divergencia de Bregman. De esta manera, la ecuación de actualización de los parámetros para los clusters es:

$$\mu_k = \frac{1}{n_k} \sum_{i:z_i=k} x_i \quad (4.50)$$

### Función objetivo de RDP-means

**Teorema 4.1** El algoritmo de clustering con restricciones RDP-means minimiza de manera iterativa la siguiente función:

$$\min_{\{\mathcal{L}_k\}_{k=1}^K} \sum_{k=1}^K \sum_{i \in \mathcal{L}_k} \left[ D_\phi(x_i, \mu_k) - \xi_1 f_k^i + \xi_2 s_k^i \right] + \lambda K \quad (4.51)$$

donde  $\{\mathcal{L}_1, \dots, \mathcal{L}_k\}$  representan una partición de las  $n$  instancias.

### Efectos de los cambios sobre $\xi_1$ y $\xi_2$

Tomando  $\xi_1, \xi_2 \rightarrow 0$ , el algoritmo RDPM se comporta de manera idéntica a DPM, esto es, no se tiene en consideración la información proporcionada por las restricciones. Por otra parte, si  $\xi_1, \xi_2 \rightarrow \infty$  se pone todo el peso en las restricciones y no se consideran las instancias para el proceso de clustering, en otras palabras, el conjunto de cluster se genera de acuerdo a la matriz  $R$  únicamente. De manera similar, podemos escoger poner más peso en las restricciones *may-link* sobre las *may-not-link* estableciendo  $\xi_1 > \xi_2$  y viceversa.

A pesar del control que  $\xi_1$  y  $\xi_2$  permiten sobre el proceso de clustering, la función objetivo de RDPM (ecuación 4.51) posee un gran número de mínimos locales, y dado que el algoritmo la minimiza con un proceso *greedy*, corremos el riesgo caer en un mínimo local. Los autores Daniel Khashabi et al. (2015) [24] muestran experimentalmente que establecer  $\xi_1 = \xi_2 = \xi$ , con un valor pequeño para  $\xi_0$ , que se incrementará con el paso de las iteraciones en función de un ratio dado, proporciona mejores resultados que un esquema de libre configuración para  $\xi_1$  y  $\xi_2$ .

### EL ALGORITMO RDP-MEANS

Con todo lo estudiado en la sección 4.5, podemos resumir el proceso de cálculo del algoritmo Relational Dirichlet Process - Means (RDPM) el Algoritmo 4:

---

**Algoritmo 4:** Relational Dirichlet Process - Means (RDPM)

---

**Entrada:** Conjunto de datos  $X$ , matriz de restricciones  $R$ , parámetro de la divergencia de Bregman  $\psi^*$ , los parámetros  $\lambda, \xi_0$ , y el ratio de incremento  $\xi_{rate}$

**Salida:** La asignación  $z = [z_1, z_2, \dots, z_n]$

**función** RDPM( $X, R, \psi^*, \lambda, \xi_0, \xi_{rate}$ ) **begin**

```

    1. Inicializar  $\xi \leftarrow \xi_0$  y asignar todas las instancias a un
       mismo cluster
    2. Iterar hasta converger como sigue:
       while no convergencia do
           for  $x_i \in X$  do
               for  $\mu_k \in C$  do
                   Calcular los valores  $f_k^i$  y  $s_k^i$  a partir de  $R$ .
                    $dist(x_i, \mu_k) \leftarrow D_{\psi^*}(x_i, \mu_k) - f_k^i \xi_1 + \xi_2 s_k^i$ 
               end
               //  $d_{min}$  es la distancia mínima e  $i_{min}$  es el
               // índice de la misma
                $[d_{min}, i_{min}] \leftarrow \{dist(x_i, \mu_1), \dots, dist(x_i, \mu_k)\}$ 
               if  $d_{min} < \lambda$  then
                   |  $z_i \leftarrow i_{min}$ 
               else
                   // Crear un nuevo cluster
                    $C \leftarrow \{C \cup \{x_i\}\}$ 
                    $K \leftarrow K + 1$ 
               end
           end
           for  $\mu_k \in C$  do
               if  $|c_k| > 0$  then
                   |  $\mu_k \leftarrow \frac{\sum_{x_i \in c_j} x_i}{|c_j|}$ 
               else
                   | Eliminar el cluster  $c_j$  y propagar los cambios
               end
           end
            $\xi \leftarrow \xi \times \xi_{rate}$ 
       end
       return  $C$ 
   end

```

---

# 5

## IMPLEMENTACIÓN

---

El objetivo de esta sección es dar una visión general de los métodos y herramientas empleadas para la implementación de los algoritmos expuestos en la sección 4, así como servir de guía para su utilización. Para ello tomamos como referencia la implementación de estos métodos en Matlab, resultado de un trabajo colaborativo en el que participan varios de los autores citados en este trabajo (citar github).

### ENTORNO DE DESARROLLO

Para la implementación es esencial disponer de un entorno de desarrollo que permita manejar de manera eficiente los múltiples archivos que contendrán el código. De igual forma debemos disponer de herramientas de depuración de fácil uso. Por ello el entorno recomendado es PyCharm, desarrollado por la empresa JetBrains.

PyCharm ofrece multitud de herramientas que hacen de desarrollo de grandes cantidades de código una tarea asequible, ayudándonos a seguir el estándar de programación de Python y poniendo a nuestra disposición un depurador de código completo.

### BIBLIOTECAS EMPLEADAS EN EL DESARROLLO

(Incluir referencias para todas las bibliotecas)

Para la implementación de las funcionalidades será necesario hacer uso de algunas bibliotecas de Python que nos permitan realizar cálculos y procedimientos básicos de forma sencilla.

Como no podía ser de otra manera, haremos uso de Numpy, una de las bibliotecas más completas y básicas de Python. Numpy nos permite trabajar con matrices de manera eficiente, lo que es esencial para trabajar con grandes cantidades de datos que, a menudo, viene especificados en forma de matriz. Numpy integra operaciones con matrices, como la suma, la resta o la multiplicación y el cálculo de la inversa o del determinante de manera sencilla.

Emplearemos también la biblioteca math, que pone a nuestra disposición multitud de operaciones que no viene definidas en el lenguaje, como pueden ser el cálculo del factorial o el logaritmo.

La biblioteca Scipy es una de las bibliotecas para Python que integra métodos numéricos. Scipy ofrece una API sencilla de usar para el usuario, con funcionalidades como el cálculo de distintas medidas de distancia en un espacio dado, o el de funciones complejas. Un ejemplo de estas últimas pueden ser: el valor absoluto del logaritmo de la

función Gamma para entradas reales (`gammaln`), o el de la derivada logarítmica de la función Gamma (`digamma`), cálculos necesarios para la implementación de algunos métodos.

Otra de las bibliotecas que podemos encontrar de utilidad para el desarrollo puede ser scikit-learn, una de las más ampliamente usadas por la comunidad para el desarrollo de aplicaciones de aprendizaje automático de cualquier tipo. Scikit-learn pone a nuestra disposición métodos clásicos, como K-medias (KM), o Fuzzy K-means (FKM), que a menudo se emplean como métodos de inicialización en funcionalidades más complejas.

Por último, y aunque no sea estrictamente necesaria para la implementación, emplearemos la biblioteca matplotlib para obtener representaciones de los resultados obtenidos con los métodos implementados. Matplotlib permite un alto nivel de personalización en representaciones, haciendo posible incluir toda la información que sea necesaria para su comprensión.

#### FUNCIONALIDADES DESARROLLADAS

Uno de los objetivos de este trabajo es poner a disposición de la comunidad algunos de los métodos de clustering con restricciones que existen en la actualidad. A continuación se ofrece la documentación de los métodos implementados:

##### *Función para generar restricciones*

La función `gen_rand_const` crea un conjunto de restricciones en base a un conjunto de datos. La función que sigue para ello es simple, basta con seleccionar de forma aleatoria dos instancias y establecer una restricción de tipo ML o CL en función de si las instancias pertenecen o no a la misma clase. Dado que el método necesita como argumento las etiquetas (un oráculo) puede no ser de utilidad en un caso real, en el que no conocemos la clase a la que pertenece cada objeto; sin embargo es válido para realizar experimentos. A continuación se muestra el prototipo del método:

---

```
def gen_rand_const(X, y, mat_const, nb_const, noise = 0,
                   prop = False)
```

---

Los parámetros de la función se detallan a continuación:

- **X** → matriz de  $n \times p$  que contiene el conjunto de datos
- **y** → lista de de etiquetas de longitud  $n$  etiquetas asociadas al conjunto de datos.

- **mat\_const** → matriz de  $n \times n$  de restricciones sobre las que se añadirán las nuevas.
- **nb\_const** → número de restricciones a añadir.
- **noise** → porcentaje de ruido a introducir en las restricciones.
- **prop** → booleano que indica si las restricciones deben propagarse, es decir, si la matriz será simétrica respecto a la diagonal principal

La función devuelve una matriz de  $n \times n$  que contiene **nb\_const** nuevas restricciones. Cada elemento de la matriz indica si existe una restricción entre dos instancias y de qué tipo es, esto es, 0 indica que no existe restricción, 1 indica que existe una de tipo ML, y -1 una de tipo CL.

#### *Función para aplicar COP-K-Medias*

La función **COPKM** es la encargada de aplicar el algoritmo COP-k-medias (sección 4.2) a un conjunto de datos dado. A continuación se muestra el prototipo de la función y se detallan sus parámetros:

---

```
def COPKM(X, K, constraints, max_iter = 300, tol = 1e-4,
           init = 'rand')
```

---

- **X** → matriz de  $n \times p$  que contiene el conjunto de datos
- **K** → número de clusters de la partición resultado
- **constraints** → matriz de  $n \times n$  que contiene el conjunto de restricciones. Siendo  $i$  y  $j$  los índices que indican filas y columnas respectivamente, un 1 en la matriz indica una restricción ML entre  $i$  y  $j$ , un -1 indica una de tipo CL, y un 0 indica que no existe restricción.
- **max\_iter** → límite de iteraciones para el algoritmo
- **tol** → factor multiplicativo para el cálculo del umbral del desplazamiento de los centroides por debajo del cual se detiene el proceso de iteración.
- **init** → modelo de inicialización de los centroides: 'rand' para inicialización aleatoria y 'kmpp' para inicialización con K-medias++

La función devuelve la lista de longitud  $n$  que indica la pertenencia de cada instancia a un cluster dado, es decir, la partición del conjunto de datos  $X$ , y los centroides  $V$  asociados a los clusters.

### *Función para aplicar CEKM*

La función `CEKM` aplica el algoritmo CEKM (sección 4.3) a un conjunto de datos dado. A continuación se muestra el prototipo de la función y se detallan sus parámetros:

---

```
def CEKM(X, K, constraints, max_iter = 300, alpha = 1,
rho = 100, bal = 0.5, stop_thr = 1e-3, init = 'rand')
```

---

- **X** → matriz de  $n \times p$  que contiene el conjunto de datos
- **K** → número de clusters de la partición resultado
- **constraints** → matriz de  $n \times n$  que contiene el conjunto de restricciones. Siendo  $i$  y  $j$  los índices que indican filas y columnas respectivamente, un 1 en la matriz indica una restricción ML entre  $i$  y  $j$ , un -1 indica una de tipo CL, y un 0 indica que no existe restricción.
- **max\_iter** → límite de iteraciones para el algoritmo
- **alpha** → exponente que controla la penalización por asignar instancias a conjuntos con alta cardinalidad.
- **rho** → distancia de todos los objetos al conjunto vacío.
- **bal** → compromiso entre la función objetivo  $J_{CEKM}$  y las restricciones:  $J_{CEKM} = (1 - bal) \times J_{CEKM} + bal$
- **stop\_thr** → umbral por debajo del cual la diferencia entre las posiciones de los centroides no es significativa. Por tanto si dicha diferencia es menor se detiene el proceso de iteración.
- **init** → modelo de inicialización de los centroides: 'rand' para inicialización aleatoria y 'fkm' para inicialización con Fuzzy K-means (FKM)

La función devuelve la partición difusa resultado de aplicar la función  $BetP(c)$  (ecuación 4.6) a la partición de creencia, los centroides  $V$ , la matriz que contiene los valores de la función de masa  $M$ , y el valor de la función  $J_{CEKM}$ .

### *Función para aplicar LCVQE*

La función `LCVQE` es la encargada de aplicar el algoritmo LCVQE (sección 4.4) a un conjunto de datos dado. A continuación se muestra el prototipo de la función y se detallan sus parámetros:

---

```
def LCVQE(X, K, constraints, max_iter = 300,
centroids = None)
```

---

- **X** → matriz de  $n \times p$  que contiene el conjunto de datos
- **K** → número de clusters de la partición resultado
- **constraints** → matriz de restricciones de  $|R| \times 3$ , en la que cada fila responde al formato  $\{x_1, x_2, r\}$ , donde  $x_1$  y  $x_2$  son las instancias implicadas en la restricción, y  $r$  indica el tipo de la misma (1 para ML y -1 para CL).
- **centroids** → centroides iniciales
- **max\_iter** → límite de iteraciones para el algoritmo

La función devuelve la lista de longitud  $n$  que indica la pertenencia de cada instancia a un cluster dado, es decir, la partición del conjunto de datos  $X$ , los centroides  $V$  asociados a los clusters, el número de iteraciones realizadas y el valor de la función  $LCVQE$ .

#### *Función para aplicar RDPM*

La función RDPM es la encargada de aplicar el algoritmo RDPM (sección 4.5) a un conjunto de datos dado. A continuación se muestra el prototipo de la función y se detallan sus parámetros:

---

```
def RDPM(X, lamb, constraints, max_iter = 300, xi0 = 0.1,
rate = 1)
```

---

- **X** → matriz de  $n \times p$  que contiene el conjunto de datos
- **lamb** →
- **constraints** → matriz de  $n \times n$  que contiene el conjunto de restricciones. Siendo  $i$  y  $j$  los índices que indican filas y columnas respectivamente, un 1 en la matriz indica una restricción ML entre  $i$  y  $j$ , un -1 indica una de tipo CL, y un 0 indica que no existe restricción.
- **max\_iter** → límite de iteraciones para el algoritmo
- **xi0** →
- **rate** →

La función devuelve la lista de longitud  $n$  que indica la pertenencia de cada instancia a un cluster dado, es decir, la partición del conjunto de datos  $X$ , y el número de clusters presentes en ella.

*Función para aplicar TVClust*

---

```
def TVClust(X, K, constraints, max_iter = 300,
            is_keep_l = 1, alpha0 = 1.2, stop_thr = 0.0005)
```

---

- **X** → matriz de  $n \times p$  que contiene el conjunto de datos
- **K** → número de clusters de la partición resultado
- **constraints** → matriz de  $n \times n$  que contiene el conjunto de restricciones. Siendo  $i$  y  $j$  los índices que indican filas y columnas respectivamente, un 1 en la matriz indica una restricción ML entre  $i$  y  $j$ , un 0 indica una de tipo CL, y un -1 indica que no existe restricción.
- **max\_iter** → límite de iteraciones para el algoritmo
- **is\_keep\_l** →
- **alpha** →
- **stop\_thr** →

# 6

## EXPERIMENTACIÓN

---



# A

## EL ALGORITMO K-MEDIAS

---

El algoritmo K-medias (KM) es el método básico para aplicar clustering, fue ideado por Hugo Steinhaus en 1957, aunque no fue aplicado a un caso real hasta 1967, por James MacQueen [29].

El procedimiento que aplica K-medias es simple: asignar cada instancia al cluster cuyo centroide se encuentre mas cercano a ella, y calcular los centroides en base a las instancias asignadas al cluster asociado a cada uno. De esta manera, siguiendo la notación introducida en la sección 4, la regla de actualización de los centroides es:

$$v_i = \frac{1}{|c_i|} \sum_{x_i \in c_i} x_i \quad (\text{A.1})$$

Así, si cada instancia se asigna al cluster correspondiente al centroide mas cercano, la función de error que minimiza K-medias es:

$$ERR = \frac{1}{2} \sum_{j=1}^k \sum_{x_i \in c_j} (v_j - x_i)^2 \quad (\text{A.2})$$

Formalizando la regla de asignación obtenemos:

$$c^i = \operatorname{argmin}(\|x_i - v_j\|^2) \quad t.q. \quad j \in \{1, \dots, k\} \quad (\text{A.3})$$

donde  $c^i$  representa el cluster seleccionado para la instancia  $x_i$ . Con todo, el proceso que K-medias sigue para obtener una partición de los datos queda resumido en el Algoritmo 5.

---

### Algoritmo 5: K-medias

---

**Entrada:** Conjunto de datos  $X$ , numero de cluster resultantes  $k$ .

**Salida:** Partición  $P$  del conjunto de datos  $X$ , centroides  $V$ .

**función** K-medias( $X, k$ ) **begin**

- 1. Inicialización aleatoria de los centroides  
 $V = \{v_1, \dots, v_k\}$ .
- 2. Asignar cada instancia  $x_i \in X$ , al centroide más cercano  $c_j$  siguiendo la ecuación A.3.
- 3. Para cada cluster  $c_i$ , actualizar su centroide aplicando la ecuación A.1
- 4. Iterar entre (2.) y (3.) hasta converger.
- 5. **return**  $C, V$

**end**

---



## BIBLIOGRAFÍA

---

- [1] Brian Everitt S., Sabine Landau, Morven Leese y Stahl Daniel. *Custer Analysis*. Berlin, Germany: WILEY, 2011.
- [2] Ian Davidson y Sugato Basu. «A survey of clustering with instance level». En: *Constraints* 1 (2007).
- [3] David Cohn, Rich Caruana y Andrew McCallum. «Semi-supervised clustering with user feedback». En: *Constrained Clustering: Advances in Algorithms, Theory, and Applications* 4.1 (2003), págs. 17-32.
- [4] Kiri Wagstaff y Claire Cardie. *Clustering with Instance-level Constraints*. 2000, págs. 1103-1110.
- [5] Ian Davidson y SS Ravi. «Hierarchical clustering with constraints: Theory and practice». En: *Data mining and knowledge discovery* 14.1 (2007), págs. 25-61.
- [6] Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl y col. *Constrained k-means clustering with background knowledge*. 2001, págs. 577-584.
- [7] Ian Davidson y SS Ravi. *Agglomerative hierarchical clustering with constraints: Theoretical and empirical results*. 2005, págs. 59-70.
- [8] Sugato Basu, Arindam Banerjee y Raymond J Mooney. *Active semi-supervision for pairwise constrained clustering*. 2004.
- [9] Eran Segal, Haidong Wang y Daphne Koller. «Discovering molecular pathways from protein interaction and gene expression data». En: *Bioinformatics* 19.suppl\_1 (2003), págs. i264-i272.
- [10] Ian Davidson y SS Ravi. *Clustering with constraints: Feasibility issues and the k-means algorithm*. 2005, págs. 138-149.
- [11] Martin HC Law, Alexander Topchy y Anil K Jain. *Model-based clustering with probabilistic constraints*. 2005, págs. 641-645.
- [12] Ayhan Demiriz, Kristin P Bennett y Mark J Embrechts. «Semi-supervised clustering using genetic algorithms». En: *Artificial neural networks in engineering (ANNIE-99)* (1999), págs. 809-814.
- [13] Janne Sinkkonen y Samuel Kaski. «Semisupervised clustering based on conditional distributions in an auxiliary space». En: (2000).
- [14] Sugato Basu, Arindam Banerjee y Raymond Mooney. *Semi-supervised clustering by seeding*. 2002.
- [15] Rong Yan, Jian Zhang, Jie Yang y Alexander G Hauptmann. «A discriminative learning framework with pairwise constraints for video object classification». En: *IEEE transactions on pattern analysis and machine intelligence* 28.4 (2006), págs. 578-593.

- [16] Ioannis Xenarios, Esteban Fernandez, Lukasz Salwinski, Xiaoqun Joyce Duan, Michael J Thompson, Edward M Marcotte y David Eisenberg. «DIP: the database of interacting proteins: 2001 update». En: *Nucleic acids research* 29.1 (2001), págs. 239-241.
- [17] Aharon Bar-Hillel, Tomer Hertz, Noam Shental y Daphna Weisz-hall. *Learning distance functions using equivalence relations*. 2003, págs. 11-18.
- [18] William M Rand. «Objective criteria for the evaluation of clustering methods». En: *Journal of the American Statistical association* 66.336 (1971), págs. 846-850.
- [19] Kiri Lou Wagstaff y Claire Cardie. «Intelligent clustering with instance-level constraints». En: (2002).
- [20] Ian Davidson, SS Ravi y Kiri Wagstaff. 2006.
- [21] Ian Davidson y SS Ravi. *Identifying and generating easy sets of constraints for clustering*. Vol. 6. 2006, pág. 336341.
- [22] Violaine Antoine, Benjamin Quost, M-H Masson y Thierry Denoeux. «CECM: Constrained evidential C-means algorithm». En: *Computational Statistics & Data Analysis* 56.4 (2012), págs. 894-914.
- [23] Dan Pelleg y Dorit Baras. *K-means with large and noisy constraint sets*. 2007, págs. 674-682.
- [24] Daniel Khashabi, John Wieting, Jeffrey Yufei Liu y Feng Liang. «Clustering With Side Information: From a Probabilistic Model to a Deterministic Algorithm». En: *arXiv preprint arXiv:1508.06235* (2015).
- [25] Ke Jiang, Brian Kulis y Michael I Jordan. «Small-variance asymptotics for exponential family Dirichlet process mixture models». En: (2012), págs. 3158-3166.
- [26] David Blackwell y James B MacQueen. «Ferguson distributions via Pólya urn schemes». En: *The annals of statistics* (1973), págs. 353-355.
- [27] Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon y Joydeep Ghosh. «Clustering with Bregman divergences». En: *Journal of machine learning research* 6.Oct (2005), págs. 1705-1749.
- [28] Jürgen Forster y Manfred K Warmuth. «Relative expected instantaneous loss bounds». En: *Journal of Computer and System Sciences* 64.1 (2002), págs. 76-102.
- [29] James MacQueen y col. *Some methods for classification and analysis of multivariate observations*. Vol. 1. 14. 1967, págs. 281-297.

## DECLARACIÓN

---

Put your declaration here.

*Granada, Junio 2018*

---

Germán González Almagro



## COLOFÓN

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both L<sup>A</sup>T<sub>E</sub>X and LyX:

<https://bitbucket.org/amiede/classicthesis/>

Happy users of `classicthesis` usually send a real postcard to the author, a collection of postcards received so far is featured here:

<http://postcards.miede.de/>

Thank you very much for your feedback and contribution.

*Final Version* as of 23 de mayo de 2018 (`classicthesis` version 4.4).