CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
DEL INSTITUTO POLITÉCNICO NACIONAL
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
SECCIÓN DE COMPUTACIÓN

Algoritmo de clustering basado en entropía para descubrir grupos en atributos de tipo mixto

Tesis que presenta

Hernández Valadez Edna

Para obtener el grado de

Maestro en Ciencias

En la especialidad de

Ingeniería Eléctrica

Opción Computación

Director: **Dra. Xiaoou Li Zhang**
Co-director: **Dr. Luis E. Rocha Mier**

México, D.F., Agosto de 2006

Agradecimientos

Quiero agradecer a mis padres, hermanos y seres queridos, por su apoyo constante y por sus incontables enseñanzas y valores.

A ti enrique, por tu cariño, apoyo y consejos, gracias por acompañarme en momentos difíciles y ser la motivación para seguir adelante.

A mis asesores la Dra. Xiaoou Li y el Dr. Luis Enrique Rocha, por sus valiosas ideas y comentarios aportados al trabajo de tesis, por su amabilidad y sobre todo disponibilidad para atenderme a discutir temas de la tesis.

Al Dr. Joshua Huang de la Universidad de Hong Kong por enviarnos amablemente el código fuente del algoritmo *k-prototype* y al Dr. Zengyou He del Instituto Harbin de Tecnología en China por enviarnos información útil para realizar las evaluaciones de error y la medición de calidad en los resultados del proceso de clustering.

A todos mis maestros de la Sección de Computación, que me brindaron sus conocimientos y experiencia y sentaron las bases para poder desarrollar correctamente éste trabajo de tesis y a las secretarias de la sección, principalmente a Sofí, porque sin su ayuda no podría titularme.

A mis amigos de siempre: Mario, Mireya y Noel, por éstar ahí, por ayudarme en múltiples situaciones y ser parte de mi formación.

A mis demás compañeros de la sección, por su amistad y compañerismo.

Al CINVESTAV por permitirme cursar los estudios de maestría facilitándome el uso de sus instalaciones.

Al CONACyT por apoyarme con la beca económica durante toda mi estancia en el programa de maestría.

Índice general

Índice de tablas

Índice de figuras

Resumen

La mayoría de los algoritmos de clustering se basan en analizar datasets que contienen ya sea atributos de tipo numérico o categórico. Recientemente, el problema del análisis de clustering en datasets con tipos de datos mixtos ha comenzado a tomar gran interés, ya que en aplicaciones de la vida real los datasets con atributos de tipo mixto son muy comunes.

En la literatura, los primeros algoritmos de clustering se diseñaron para trabajar en datasets que contenían exclusivamente datos de tipo numérico o categórico. Al utilizar algún dataset con datos mixtos, se tenía la problemática de convertir variables categóricas a numéricas o viceversa, lo cual puede representar pérdidas de información en algunas características de los datos originales.

En este trabajo de tesis, proponemos un algoritmo de clustering denominado *ACEM*, el cual es capaz de manejar datasets con tipos de datos mixtos. El algoritmo propuesto pre-clasifica los datos categóricos puros del dataset y realiza una evaluación de entropía de los clústers utilizando el conjunto de datos mixtos para verificar la pertenencia de los datos a los clústers. En caso necesario, cambia los datos al clúster con más características en común (menor valor de entropía). Con la presentación de esta tesis, proponemos un algoritmo de clustering para datos mixtos que extienda las características de un algoritmo de clustering de datos categóricos, introduciendo nociones de entropía para medir la heterogeneidad de los clústers.

Para medir el desempeño del algoritmo propuesto, se realizaron experimentos de comparación con otros algoritmos de clustering utilizando datasets de la vida real con tipos de datos categórico y mixto obtenidos de la UCI Machine Learning Repository [38]. En general, los resultados experimentales demuestran que nuestro algoritmo presenta un comportamiento estable y un buen desempeño en la medición del error tanto para datasets de tipo categórico como para los datasets de tipo mixto evaluados en este trabajo.

Abstract

Most clustering algorithms are focused on datasets with only numeric or categorical attributes. Recently, the problem of clustering mixed data has drawn interest due to the fact that many real life applications have mixed data.

The first clustering algorithms reported in the literature were designed to work on datasets that exclusively contained either numerical or categorical data types. If a dataset contains mixed data, these algorithms have the drawback that they need to transform variables from numeric to categorical or from categorical to numeric, which involves loss of information regarding of the original data characteristics.

In this thesis work, we propose a clustering algorithm called *ACEM* that is able to deal with mixed data. This algorithm makes a pre-clustering on the pure categorical data. Then upon including all mixed data, it evaluates the clusters using an entropy-based criterion in order to verify the cluster membership of the data. The data is changed to the cluster with lowest entropy measure. As a result, we obtain a clustering algorithm for mixed data whose core idea is to extend a categorical clustering algorithm by introducing an entropy criterion to measure the cluster heterogeneity.

We perform comparisons with respect to other clustering algorithms using real life datasets with categorical and mixed data, which were obtained from the UCI Machine Learning Repository [38]. In general, the experimental results show that our algorithm has an stable behavior and a better performance in the clustering accuracy measure error for the categorical and mixed datasets reviewed in this tesis work.

Capítulo 1.

Introducción

En los últimos años se ha visto un gran crecimiento en la generación de información, debido principalmente a la automatización de procesos y a los avances en las capacidades de almacenamiento de información.

Cada día, la gente almacena gran cantidad de información representada como datos para posteriormente realizar un análisis y administración de estos. El principal objetivo de interactuar con estos datos es clasificarlos en pequeños grupos que describan sus características principales, basándose en la similitud o diferencia entre ellos. Es imposible analizar directamente dicha cantidad de datos, por lo que comúnmente se recurre a la utilización de técnicas de clustering que ayudan a particionar un conjunto de datos en pequeños grupos que permiten un análisis eficiente de la información. El análisis de grandes volúmenes de datos no sólo puede brindar información adicional, sino también conocimiento nuevo.

1.1. Antecedentes

La minería de datos es considerada uno de los puntos más importantes de los sistemas de bases de datos y uno de los desarrollos más prometedores interdisciplinariamente en la industria de la información. La minería de datos representa la posibilidad de buscar información dentro de un conjunto de datos con la finalidad de extraer información nueva y útil que se encuentra oculta en grandes volúmenes de datos [43].

El clustering es una de las principales tareas en el proceso de minería de datos para descubrir grupos e identificar distribuciones y características interesantes en los datos. Es una disciplina científica muy joven que actualmente se encuentra bajo un vigoroso desarrollo. Existen un gran número de trabajos de investigación presentados en diversas conferencias y revistas sobre diferentes aspectos y técnicas de la minería de datos. El análisis de clustering en minería de datos ha desempeñado un rol muy importante

en una amplia variedad de áreas tales como: reconocimiento de patrones, análisis de datos espaciales, procesamiento de imágenes, cómputo y multimedia, análisis médico, economía, bioinformática y biometría principalmente [19].

El proceso de agrupar un conjunto de objetos físicos o abstractos dentro de clases con objetos similares se denomina *clustering*. El clustering consiste en agrupar una colección dada de datos no etiquetados en un conjunto de grupos de tal manera que los objetos que pertenecen a un grupo sean homogéneos entre sí [43] , buscando además que la heterogeneidad entre los distintos grupos sea lo más elevada posible (ver ejemplo en la figura 1.1).

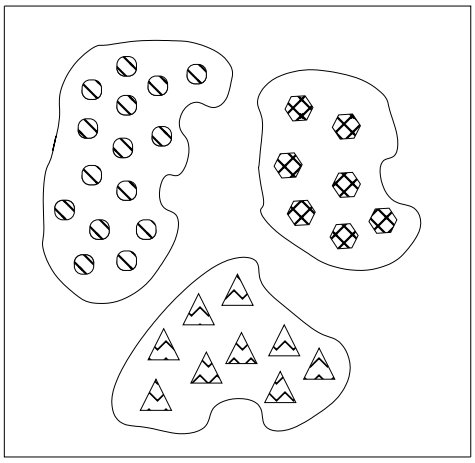

Figura 1.1.: Ejemplo de clustering.

Expresado en términos de variabilidad hablaríamos de minimizar la variabilidad dentro de los grupos para al mismo tiempo maximizar la variabilidad entre los distintos grupos. En el proceso de clustering, no hay clases predefinidas ni registros muestra que permitan conocer las relaciones existentes entre los datos, esto se puede ver como un proceso no supervisado. En este sentido, los clústers o grupos se van creando de acuerdo a las características de los datos, no a una asignación de clases ya predefinidas, por lo que el clustering es también conocido como *clasificación no supervisada* [18].

El proceso de clustering en minería de datos involucra una serie de complicaciones al trabajar grandes datasets con atributos de diferentes tipos. En los últimos años, investigadores han estudiado distintos métodos para un análisis de clustering efectivo y eficiente en datasets con estas características, generando así una gran variedad

de algoritmos de clustering que sean capaces de resolver problemas de escalabilidad, efectividad para descubrir clústers de formas complejas, dimensionalidad y capacidad para agrupar clústers en datasets con distintos tipos de atributos tales como: numérico, categórico y mixto.

Con la presentación de esta tesis se busca proveer de un nuevo algoritmo de clustering para datos mixtos que extienda las características de un algoritmo de clustering de datos categóricos, introduciendo nociones de entropía para evaluar la heterogeneidad de los clústers.

1.2. Trabajo Relacionado

En la literatura, existen un gran número de algoritmos de clustering y la selección de cada uno de ellos depende principalmente del tipo de dato y el propósito particular de la aplicación. De acuerdo a los tipos de atributos contenidos en las base de datos, los algoritmos diseñados exclusivamente para tipos de dato numérico o categórico, tienen además la problemática de tener que convertir variables categóricas a numéricas o numéricas a categóricas (según sea el caso) para poder implementar su metodología, lo cual puede representar pérdidas de información en algunas características de los datos originales. Los algoritmos de clustering pueden ser clasificados en 3 categorías: numéricos, categóricos y mixtos. Actualmente, la mayoría de los algoritmos de clustering propuestos, han sido diseñados con la consideración de que todos los atributos sean de tipo numérico o categórico (ver [9], [16], [17] y [47]). Por lo tanto, es importante contar con algoritmos de clustering que tengan la capacidad de manipular datasets con atributos de tipo mixto.

La mayor parte de los primeros algoritmos de clustering se centran en datasets con atributos numéricos cuyas propiedades inherentemente geométricas pueden ser explotadas naturalmente para definir distancias entre los puntos, tales como *DBSCAN* [9], *CURE* [16] y *CHAMALEON* [32].

En años recientes, se han propuesto algunos algoritmos que manejan atributos de tipo categórico, algunos de ellos son: *Squeezer* [47], *K-Histograms* [21], *K-ANMI* [22],*ROCK* [17], *CACTUS* [13], *K-modes* [28]. Estos algoritmos solo fueron diseñados para atributos categóricos y su posible extensión a datos mixtos aún no ha sido publicada.

Los algoritmos diseñados exclusivamente para tipos de dato numérico o categórico, tienen además la problemática de tener que convertir variables categóricas a numéricas

o numéricas a categóricas (según sea el caso) para poder implementar su metodología, lo cual puede representar pérdidas de información en algunas características de los datos originales.

En lo que respecta a algoritmos de clustering para atributos de tipo mixto, se han hecho varios esfuerzos por parte de algunos autores. En [27] el autor presenta el algoritmo *k-prototypes*, el cual es una extensión del algoritmo *K-means* para atributos de tipo mixto. El algoritmo involucra una suma de pesos: la distancia euclidiana para los atributos numéricos (d_n) y la distancia medida para los atributos categóricos multiplicada por un factor gamma ($\gamma * d_c$), en donde el factor γ se debe determinar de manera a priori.

En [6] se introduce un algoritmo que se basa en la estructura del algoritmo *BIRTH* [48] utilizando una medida de distancias. La distancia medida se deriva de un modelo probabilístico en donde la distancia entre dos clústers es equivalente al decrecimiento de la función de probabilidad logarítmica como resultado de la mezcla. El algoritmo *BIRTH* tiene la desventaja de no trabajar bien para clústers con geometrías no esféricas y además sus resultados se ven afectados por el orden de entrada de los parámetros, por lo tanto el algoritmo en [6] tiene los mismos problemas.

En el algoritmo propuesto en [36], se describe el algoritmo O-Cluster, el cual maneja eficientemente bases de datos muy grandes con atributos de tipo numérico y categórico. El algoritmo utiliza un esquema de particionamiento de ejes paralelos para construir una jerarquía e identificar las regiones hiper-rectangulares de densidad uni-modal dentro del espacio de entrada. Sin embargo para esta implementación es necesario realizar un análisis previo de los datos.

En [35], se propone el algoritmo *SBAC*, el cual adopta una medida de semejanza que le da mayor peso a las características poco comunes que concuerdan con evaluaciones similares. Emplea un algoritmo aglomerativo para construir un dendograma y una técnica heurística para extraer una partición de los datos. Sin embargo, la complejidad del *SBAC* es cuadrática de acuerdo al número de datos en el dataset, debido a lo cual es casi imposible manejar bases de datos muy grandes.

En [23] se presenta un algoritmo que se basa en ensambles. Este algoritmo divide el dataset en datos categóricos puros y datos numéricos puros. Utiliza 2 algoritmos de clustering bien establecidos, diseñados para particionar datos de tipo numérico y categórico respectivamente y realiza el clustering de cada una de las divisiones del dataset. De los resultados de clustering obtenidos por cada uno de los algoritmos, se combinan los resultados en un dataset de tipo categórico sobre el cual se efectuará un último clus-

tering de los datos. Utilizando el mismo algoritmo de clustering categórico, se ejecuta el clustering sobre los resultados combinados y se obtiene de esta manera el resultado final de los clústers. Sin embargo, este trabajo aún se encuentra en revisión y no se pueden realizar comparaciones con respecto a él.

Como se puede observar, no existen muchos algoritmos de clustering que sean capaces de manipular atributos de tipo mixto, por lo tanto, en esta tesis proponemos un nuevo algoritmo de clustering para datos mixtos.

1.3. Motivación

La naturaleza creciente de la información requiere de herramientas especializadas que resulten adecuadas para el correcto y efectivo análisis de la información. El análisis manual de datos es caro, consume muchos recursos en especialistas, dinero y tiempo, ya que se tiene la necesidad de formular hipótesis, probar, ajustar y tomar en consideración un número cada vez mayor de parámetros para conocer las relaciones existentes entre los datos.

En la mayoría de las técnicas de análisis de datos, se debe conocer y especificar previamente los objetivos que se pretende alcanzar, para llevar acabo la toma de decisiones. Sin embargo, existe una gran cantidad de información generada, que resulta casi imposible definir un objetivo previo y por lo tanto, no se tiene una definición de características previa.

El clustering es una de las tareas del aprendizaje no supervisado en minería de datos en la que no se requiere una clasificación predefinida de los datos, para particionarlos y obtener conocimiento de los mismos. Las técnicas de clustering proveen una variedad de algoritmos con características deseables para el descubrimiento de conocimiento contenido en los datos. La mayoría de estos algoritmos de clustering, han sido diseñados para manejar ya sea todos los atributos de tipo numérico o categórico. Sin embargo, el problema de analizar datasets con tipos de datos mixtos ha comenzado a tomar gran interés ya que en aplicaciones de la vida real los datasets con atributos de tipo mixto son muy comunes.

Además, los algoritmos diseñados exclusivamente para tipos de dato numérico o categórico, pueden representar pérdidas de información en algunas características de los datos originales al convertir variables categóricas a numéricas o numéricas a categóricas (según sea el caso) para poder implementar su metodología.

Por estas razones, en esta tesis implementamos una adaptación alternativa de un algoritmo de clustering de datos categóricos, para que pueda manejar datasets con tipos de datos mixtos.

1.4. Objetivos

Los principales objetivos de la tesis son:

- Reconocer la problemática del análisis de grandes volúmenes de datos y de los beneficios de su uso sistemático para la obtención de modelos y patrones descriptivos.
- Realizar un análisis de los datos con la finalidad de establecer relaciones y características que demuestren la existencia de grupos en los datos.
- Implementar una adaptación alternativa de un algoritmo de clustering de datos categóricos para que pueda manejar datasets con tipos de datos mixtos. De tal manera, se busca proponer un algoritmo de clustering que utilice nociones de entropía para clasificar atributos de tipo mixto.

El algoritmo propuesto, denominado *ACEM* (***A****daptación alternativa de un algoritmo de clustering* ***C****ategórico utilizando nociones de* ***E****ntropía para agrupar datos de tipo* ***M****ixto*), pre-clasifica los datos categóricos puros del dataset y realiza una evaluación de entropía de los clústers utilizando el total de los datos mixtos para verificar la pertenencia de los datos a los clústers (en función del menor valor de entropía evaluado).

1.5. Organización

El resto de esta tesis está organizada de la siguiente manera:

En el capítulo 2 se mencionan algunas consideraciones para la definición, tratamiento, medición y evaluación de los datos. Se describen algunas estrategias de manipulación de datos y características importantes que se deben conocer antes de implementar una técnica de clustering en minería de datos.

En el capítulo 3 se proporcionan los conceptos básicos acerca de la minería de datos y el clustering, enfocándonos en las características principales de las técnicas y metodologías comúnmente utilizadas en estas áreas.

En el capítulo 4 se describe el algoritmo de clustering propuesto, incluyendo el algoritmo usado como inspiración, así como la estructura y la descripción básica del algoritmo propuesto. Se presenta además un análisis de la complejidad del algoritmo y una descripción de los algoritmos empleados para la comparación de resultados.

En el capítulo 5 se muestran los experimentos y comparaciones efectuadas del algoritmo propuesto, utilizando datasets con tipos de dato categórico y mixto. Se evalúa el comportamiento y desempeño del algoritmo propuesto respecto a algoritmos de clustering publicados previamente.

En el capítulo 6 se dan las conclusiones de este trabajo, señalando el desempeño obtenido por el algoritmo propuesto en datasets de tipo categórico y mixto. También se indican algunas de las posibles rutas de trabajo futuro que podrían derivarse de esta tesis.

Capítulo 2.

Datos y Mediciones

El objetivo principal de las técnicas de clustering en minería de datos es descubrir relaciones existentes en los datos. Estas relaciones se descubren por medio del análisis exhaustivo de los datos. Sin embargo, para conocer el tipo de análisis de datos es importante conocer el significado y la representación de los *datos*.

En las siguientes secciones se presenta una descripción detallada del concepto de *dato*, su representación simbólica, los procedimientos y técnicas de medición y algunas probemáticas generadas al manejar varios tipos de datos en el proceso de clustering.

2.1. Representación de datos

"Los *datos* son una colección de entidades mapeadas en un dominio de interés. Su representación simbólica se basa en las relaciones existentes entre un conjunto de *atributos* que describen a un conjunto de *objetos*" [20].

Los *atributos* representan las propiedades y características básicas de los *objetos*. Son también conocidos como: variables, campos o características (ej., color de ojos, temperatura, edad, etc.). Un atributo puede ser mapeado a distintos valores (ej. la *Altura* puede ser medida en metros o pies). Varios atributos pueden ser mapeados a un mismo tipo de valor (ej. el *ID* y la *Edad* pueden ser representados con variables numéricas enteras, pero cada uno tiene propiedades diferentes).

Los *objetos* son aquellos vectores que se describen en función del conjunto de *atributos*. Un *objeto* es también conocido como punto, muestra, entidad o instancia.

En la figura 2.1 muestra la representación básica de un conjunto de datos.

A t r i b u t o s

O b j e t o s	ID	Casa	Estado Civil	Ingreso	Crédito
	1	Si	Soltero	125K	**No**
	2	No	Casado	100K	**No**
	3	No	Soltero	70K	**No**
	4	Si	Casado	120K	**Si**
	5	No	Divorciado	85K	**No**
	6	No	Casado	50K	**No**
	7	Si	Divorciado	220K	**Si**
	8	No	Soltero	35K	**No**
	9	No	Casado	75K	**No**
	10	No	Soltero	90K	**Si**

Figura 2.1.: Representación de datos.

2.1.1. Tipos de atributos

Los atributos pueden tener diversas representaciones, las cuales dependen principalmente de las propiedades de los datos [43]. Generalmente, los tipos de atributos se clasifican en:

1. **Nominal - Cualitativo categórico.** Los valores de un atributo nominal son variables diferentes que proveen la suficiente información para distinguir un objeto de otro.

2. **Ordinal - Cualitativo categórico.** Los valores de un atributo ordinal proveen suficiente información para ordenar los objetos.

3. **Intervalo - Cuantitativo numérico.** Los atributos de intervalo pertenecen a un rango específico de valores y las diferencias entre valores son fundamentales.

4. **Radio - Cuantitativo numérico.** Para los atributos de radio las variables se mapean a diferentes tipo de valores.

Además de las características presentadas en la clasificación anterior, los valores de los atributos tienen algunas propiedades que los hacen más específicos en su contenido [43]. Estas propiedades son:

- *Distinción.* Permite realizar operaciones de igualdad y diferencia ($=$,$\neq$).
- *Orden.* Permite realizar operaciones de ordenamiento ($>$,$<$).

- *Adición.* Permite realizar operaciones adición y diferencia $(+,-)$.
- *Multiplicación.* Permite realizar operaciones multiplicativas y divisorias en los datos $(*,/)$.

En la tabla 2.1 se representan los tipos de atributos con sus propiedades y algunos ejemplos.

Tabla 2.1.: Tipos de atributos.

Atributo	**Propiedad**	**Operaciones**	**Ejemplo**
Nominal	$(=,\neq)$	moda, entropía correlación.	sexo, color de ojos empleo.
Ordinal	$(=,\neq,>,<)$	mediana, porcentajes.	grados, dureza de metales.
Intervalo	$(=,\neq,>,$ $<,+,-)$	media, desviación estándar.	temperatura Celsius, calendario.
Radio	$(=,\neq,>,<$ $+,-,*,/)$	media geométrica, media armónica.	edad, masa, distancia corriente eléctrica.

La representación simbólica de los datos (atributos y objetos) es almacenada comúnmente en un dataset. Dicho dataset será la base para el análisis de datos y la implementación de actividades específicas de la minería de datos.

2.1.2. Tipos de datasets

En los últimos años, investigadores han estudiado distintos métodos para un efectivo y eficiente análisis de clustering en datasets con distintos tipos de datos.

En función del tipo de dato contenido en el dataset, se han presentado 3 categorías generales:

1. *Numéricos.*
2. *Categóricos.*
3. *Mixtos.*

• Los datasets con datos de tipo *numérico* pueden ser analizados en función de las características inherentemente geométricas de los datos. Comúnmente se utilizan medidas geométricas (ej. funciones de distancias geométricas).

• Los datasets con datos de tipo *categórico* se analizan de acuerdo a las características cualitativas de los datos. Se utilizan medidas de similitud y análisis de frecuencia para evaluar la estructura representativa de los datos.

• Los datasets con datos *mixtos*, son una combinación de los dos datasets anteriores en donde se presentan datos de tipo *numérico* y tipo *categórico*.

El análisis de datasets con tipos de datos mixtos ha comenzado a tomar gran interés ya que en aplicaciones de la vida real los datasets con atributos de tipo mixto son muy comunes. En la literatura, los primeros algoritmos de clustering se diseñaron para trabajar en datasets que contenían exclusivamente datos de numérico o categórico. Al utilizar algún dataset con datos mixtos, se tenía la problemática de convertir variables categóricas a numéricas o viceversa.

2.1.3. Conversión datos categóricos a numéricos

Existen varias maneras de convertir variables categóricas a numéricas. Por ejemplo, en [39], se convierten múltiples atributos categóricos en atributos binarios, utilizando una representación de $0's$ y $1's$ para definir presencia o ausencia de atributos. Estas variables binarias pueden ser tratadas como numéricas en algún algoritmo para datos numéricos.

La mayoría de los métodos de conversión de variables categóricas a numéricas tienen la problemática de no incluir las características particulares de los datos. Además, si se maneja una gran cantidad de atributos numéricos o binarios es inevitable el incremento en el costo computacional.

2.1.4. Conversión datos numéricos a categóricos

Se han diseñado algunos métodos que convierten variables numéricas a categóricas. Por ejemplo, en los mapas auto-organizativos (*self-organizing maps*) se pueden producir estados discretos en las variables numéricas que pueden ser manipulados como de tipo categórico.

En el algoritmo *k-modes* [28], se categorizan atributos numéricos utilizando un método descrito en [2]. Se realizan tres modificaciones al algoritmo *k-means* referentes a la distancia y la media utilizadas.

Estos métodos, al igual que los utilizados en la conversión de datos categóricos a numéricos, tienen la problemática de perder significativamente algunas propiedades de los datos.

Hasta este punto, se observa que las diversas representaciones de los datos hacen necesario aplicar procedimientos de medición para el análisis de datos y para su adecuada implementación en técnicas de clustering en minería de datos.

2.2. Calidad de datos

La calidad de los datos es una de las características deseables en cualquier proceso de análisis de datos. Es importante considerar la presencia de ruido, outliers, valores faltantes, inconsistencia, datos duplicados y algunas otras imperfecciones que pueden describir el conjunto de datos.

Las aplicaciones que obtienen sus datos de mediciones y están asociadas a ruido se pueden considerar como *outliers*. Alternativamente, los *outliers* representan objetos que tienen un comportamiento anormal. En general, la mayoría de las técnicas de clustering no distinguen entre el ruido o las anormalidades dentro de los clústers. La manera más preferible de manipular los *outliers* es particionar el conjunto de datos y conservar el conjunto de outliers separados para tratarlos en forma diferente con la finalidad de que no interfieran con el proceso de clustering.

Estadísticamente se puede definir a los datos outliers como puntos que no tienen una distribución de probabilidad. En minería de datos, un punto puede ser declarado como outlier si sus puntos vecinos no contienen una fracción significativa del dataset completo.

Existen múltiples maneras eficientes para manipular los outliers. Es importante tomarlos en cuenta desde una etapa de pre-procesamiento. Algunos algoritmos implementan ciertas metodologías para trabajar con datos outliers, por ejemplo: *CURE* [16], *DBSCAN* [9] y otros más. El algoritmo CURE utiliza la concentración de los clústers representativos para eliminar el efecto de los outliers. Por otro lado, el algoritmo DBSCAN utiliza conceptos de datos núcleo (internos), frontera (alcanzables) y outliers (no alcanzables) para manipularlos en forma independiente.

Agregado a la necesidad de eliminar el efecto negativo de los outliers en el proceso de clustering, existen varias razones de gran interés para la detección temprana de outliers. Estas razones se basan principalmente en que actualmente algunas aplicaciones definen a los outliers como elementos negativos. Se pueden presentar en diagnósticos médicos, redes de seguridad y operaciones financieras como anomalías, inmunidades computacionales y fraudes respectivamente. Por lo tanto, es muy importante considerar la manipulación de outliers en los diversos campos de aplicación.

2.3. Mediciones

Es natural que nos preguntemos qué tipo de características debemos de considerar para determinar la cercanía de un conjunto de puntos o para medir la distancia o similitud entre un par de objetos.

Las mediciones pueden ser categorizadas de diversas formas. Algunas de ellas dependen de la naturaleza de los datos a ser medidos mientras que otras dependen de la aplicación para la cual fueron diseñadas [20]. La mayoría de las mediciones son clasificadas en términos de transformaciones que presentan las características empíricas entre los datos.

La evaluación de la similitud/proximidad entre dos objetos que pertenecen a un mismo espacio de variables es fundamental para la definición de clústers. Debido a la cantidad de tipos de datos y a la variedad en sus representaciones simbólicas, la selección del tipo de medición debe ser realizada cuidadosamente.

2.3.1. Medidas de distancia y similitud

La mayor parte de las técnicas de minería de datos (ej. métodos de clasificación del vecino más cercano y análisis de clustering) se basan en la medición de similitud entre objetos.

Existen básicamente dos maneras de obtener las funciones de similitud. Se pueden obtener directamente de los datos o indirectamente por medio de los vectores o características que describen a cada objeto [20].

En este contexto, se introducen dos términos importantes: *distancia* y *similitud*. Estos términos se utilizan para referir las mediciones derivadas de las características que

describen a los objetos.

Típicamente, las distancias son utilizadas como mediciones para variables continuas, mientras que las medidas de similitud son representativas para variables cualitativas.

• Una ***función de distancia*** D en un conjunto de datos X debe satisfacer las siguientes condiciones[45].

1. Simetría. $D(x_i, x_j) = D(x_j, x_i)$.
2. Positiva. $D(x_i, x_j) \geq 0$ para todo x_i y x_j.
3. Desigualdad del triángulo. $D(x_i, x_j) \leq D(x_i, x_j) + D(x_k, x_j)$ para todo x_i, x_j y x_k.
4. Reflexión. $D(x_i, x_j) = 0$ si $x_i = x_j$.

Las medidas de distancia son una especialización de las medidas de proximidad. Son métricas utilizadas para cuantificar la similitud o cercanía entre datos. La métrica para variables continuas más popular es la *distancia euclidiana* [1]. La *distancia euclidiana* es usada para evaluar la proximidad de objetos en dos o tres dimensiones. En el apéndice A.1 se presentan algunas de las mediciones más utilizadas para variables continuas.

• Una ***función de similitud*** S en un conjunto de datos X debe satisfacer las siguientes condiciones[45].

1. Simetría. $S(x_i, x_j) = S(x_j, x_i)$.
2. Positiva. $0 \leq S(x_i, x_j) \leq 1$ para todo x_i y x_j.
   y además satisface:
3. $S(x_i, x_j)S(x_j, x_k) \leq [S(x_i, x_j) + S(x_k, x_j)]S(x_i, x_k)$ para todo x_i, x_j y x_k.
4. $S(x_i, x_j) = 1$ si $x_i = x_j$.

Existe también un número considerable de mediciones de similitud en lo que respecta a atributos cualitativos (binarios, nominales, ordinales), ver referencias [8], [10] y [15]. Las funciones de similitud pueden construirse axiomáticamente basándose en información de los datos con consideraciones teóricas.

[1] La distancia euclidiana entre dos puntos está definida como: $D_{ij} = \left(\sum_{l=1}^{d} \mid x_{il} - x_{jl} \mid^{1/2}\right)^2$

La mayoría de los atributos cualitativos se mapean a variables binarias para utilizar las estrategias de medición. En datos de tipo binario, los coeficientes de las funciones de similitud se pueden representar como la presencia o ausencia de una variable (1 y 0 respectivamente). La tabla 2.2 muestra ésta representación por medio de una matriz de asociación de 2x2.

Tabla 2.2.: Mediciones para variables cuantitativas.

	i=1	i=0
i=1	$n_{1,1}$	$n_{1,0}$
i=0	$n_{0,1}$	$n_{0,0}$

Para medir la similitud en datos binarios, las dos medidas de similitud más comunes se ilustran en las ecuaciones 2.1 y 2.2.

$$S_{ij} = \frac{n_{1,1} + n_{0,0}}{n_{1,1} + n_{0,0} + w(n_{1,0} + n_{0,1})} \tag{2.1}$$

$w = 1,$ coeficiente de mapeo simple.

$w = 2,$ medida de Rogers y Tanimoto.

$w = 1/2,$ medida de Gower y Legendre.

$$S_{ij} = \frac{n_{1,1}}{n_{1,1} + w(n_{1,0} + n_{0,1})} \tag{2.2}$$

$w = 1,$ coeficiente de Jaccard.

$w = 2,$ medida de Sokard y Sneath.

$w = 1/2,$ medida de Gower y Legendre.

Estas medidas evalúan el mapeo directamente entre dos objetos. Se centran en la co-ocurrencia de las variables ignorando los efectos de la co-ausencia.

2.4. Comentarios

Es de gran interés, conocer las características de los datos para seleccionar las técnicas y metodologías adecuadas para el tratamiento de datos en aplicaciones específicas. Las propiedades de los datos, proporcionan información útil para su análisis y descripción,

siendo un elemento fundamental en el proceso de análisis de clustering.

Actualmente, la mayoría de los datasets son susceptibles al ruido, datos faltantes o inconsistentes. Existen muchas técnicas para la manipulación y el procesamiento de los datos. Algunas de ellas consisten en la obtención consistente de datos, eliminación de ruido, aplicación de transformaciones y normalizaciones de los datos y reducción en el tamaño de los datos previniendo redundancia e inconsistencia en los datos.

Utilizar alguna técnica de tratamiento de datos ayuda substancialmente a mejorar los resultados generados en las técnicas de clustering de minería de datos.

Capítulo 3.

Minería de Datos y Clustering

La minería de datos es una de las técnicas incluida en el proceso KDD (Knowledge Discovery in Data bases, por sus siglas en inglés), en la que el clustering de datos juega un papel muy importante. En este capítulo, se presenta una introducción a los conceptos básicos de minería de datos y el análisis de clustering.

3.1. Descubrimiento de Conocimiento en Bases de Datos

El procesar automáticamente grandes cantidades de datos para encontrar conocimiento útil para un usuario y satisfacerle sus metas, es el objetivo principal del área de Descubrimiento de Conocimiento en Bases de Datos o KDD. "El **KDD** es un proceso no trivial de identificación válida, novedosa, potencialmente útil y entendible de patrones comprensibles que se encuentran ocultos en los datos"[43]. KDD es un proceso multidisciplinario que encierra: conocimiento, aprendizaje, bases de datos, estadística, sistemas expertos y representación gráfica, entre otros.

Las fases principales que integran el proceso de KDD son (ver figura 3.1):

Selección. Selección de los datos relevantes para el análisis (son obtenidos de la base de datos).

Preprocesamiento. Limpieza de datos, estudio de la calidad de los datos y determinación de las operaciones de minería que se pueden realizar.

Transformación. Conversión de datos en un modelo analítico, donde los datos se transforman o consolidan en formas apropiadas para la minería.

Minería de Datos. Tratamiento automatizado de los datos seleccionados con una combinación apropiada de algoritmos para extraer conocimiento de los datos.

Interpretación y evaluación. Identificación de patrones representativos del conocimiento.

Conocimiento. Aplicación del conocimiento descubierto.

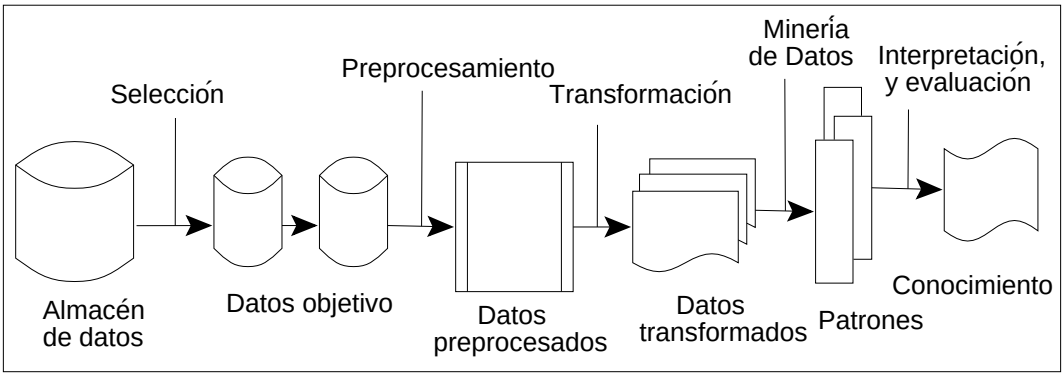

Figura 3.1.: Proceso KDD.

Comúnmente se suele denominar de la misma forma al proceso KDD con minería de datos a pesar que esta última es sólo una fase del proceso KDD.

3.2. Minería de Datos

La minería de datos es considerada uno de los puntos más importantes de los sistemas de bases de datos, y uno de los desarrollos más prometedores interdisciplinariamente en la industria de la información.

3.2.1. Concepto de Minería de Datos

"La minería de datos es un mecanismo de explotación, consistente en la búsqueda de información valiosa en grandes volúmenes de datos, con el fin de descubrir patrones, relaciones, reglas, asociaciones o incluso excepciones útiles para la toma de decisiones" [43].

Las técnicas de minería de datos son el resultado de un largo proceso de investigación y desarrollo de productos. Esta evolución comenzó cuando los datos de negocios fueron almacenados por primera vez en computadoras, y continuó con mejoras en el acceso a los datos, y más recientemente con tecnologías generadas para permitir a los usuarios navegar a través de los datos en tiempo real. Minería de datos toma este proceso de

evolución más allá del acceso y navegación retrospectiva de los datos, hacia la entrega de información prospectiva y proactiva.

La minería de datos representa la posibilidad de buscar información dentro de un conjunto de datos con la finalidad de extraer información nueva y útil que se encuentra oculta en grandes volúmenes de datos. Involucra un conjunto de técnicas de múltiples disciplinas tales como: tecnología de bases de datos, estadística, aprendizaje, reconocimiento de patrones, redes neuronales, visualización de datos, obtención de información, procesamiento de imágenes y de señales, y análisis de datos (ver figura 3.2).

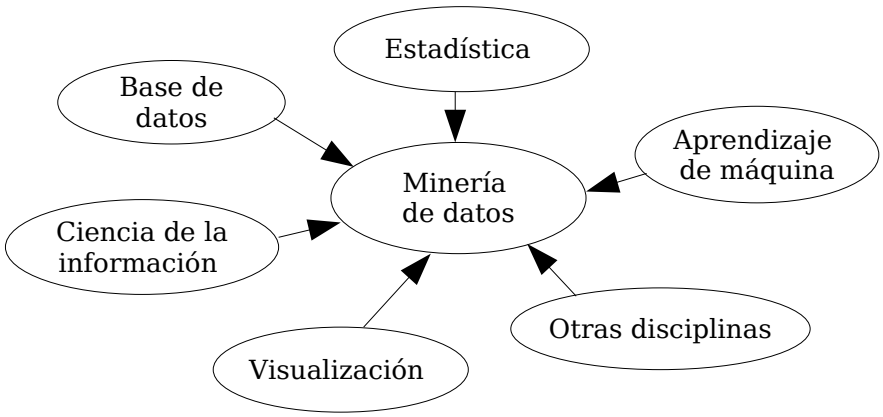

Figura 3.2.: Minería de datos como una confluencia de múltiples disciplinas.

3.2.2. Componentes básicos

La arquitectura básica de un sistema de minería clásico contiene la mayoría de los siguientes componentes [20] (ver figura 3.3):

1. Bases de datos, datawarehouse o algún otro repositorio de información como por ejemplo planillas de cálculo.

2. Servidor de bases de datos o de Datawarehouse que es el responsable de capturar los datos relevantes basándose en los requerimientos de minería de datos del usuario.

3. Base de conocimientos que guiará la búsqueda o que evaluará el interés de los patrones resultantes. Dicho conocimiento puede incluir jerarquías de conceptos usadas para organizar valores de distintos atributos en diferentes niveles de abstracción.

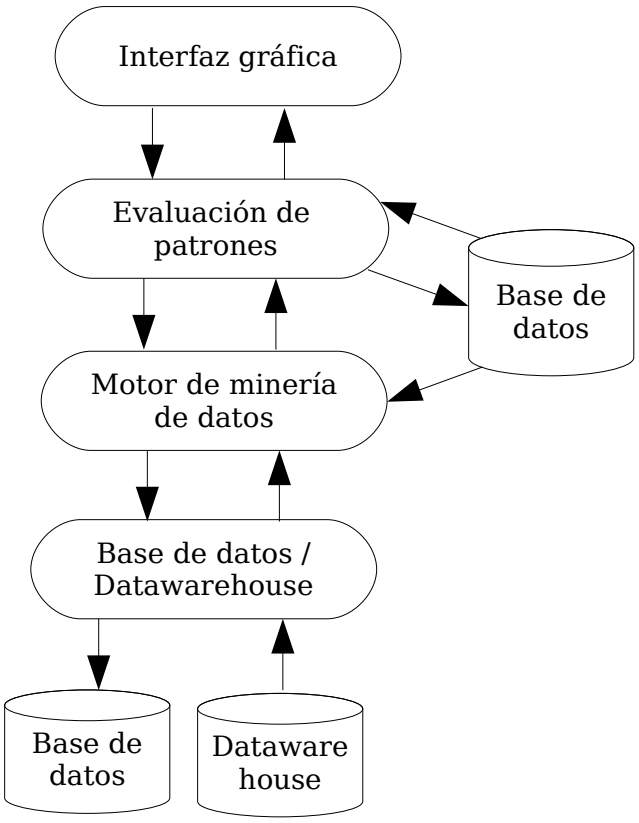

Figura 3.3.: Arquitectura típica de un sistema de minería de datos.

4. Motor de minería de datos, el cual es esencial para el sistema de minería y consiste de un conjunto de módulos funcionales que llevan a cabo distintas tareas tales como caracterización, asociación, clasificación, análisis de evolución y desviación.

5. Módulo de evaluación de patrones , que generalmente utiliza medidas de interés e interactúa con los módulos de minería de datos para enfocar la búsqueda hacia patrones interesantes, así como también para filtrar o descartar patrones ya reconocidos. Alternativamente, el módulo de evaluación de patrones puede ser in-

tegrado al módulo de minería, dependiendo de la implementación del método de minería utilizado. Para una minería de datos eficiente, es altamente recomendado el incluir la evaluación de patrones de interés tanto como sea posible dentro del proceso para dirigir la búsqueda sólo hacia los patrones de interés.

6. Interfaz gráfica de usuario; este módulo comunica a los usuarios y al sistema de minería de datos, permitiendo al usuario interactuar con el sistema especificando la consulta de datamining, proveyendo información que ayude a enfocar la búsqueda y realizar exploración de datos basándose en resultados de minerías intermedias. Esta componente le permite además al usuario visualizar datos, bases de datos, esquemas de datawarehouse o estructuras de datos, evaluar y visualizar de distintas maneras los patrones minados.

3.2.3. Técnicas de Minería de Datos

Las técnicas de minería de datos pueden dividirse en 2 categorías principales: predictivas (aprendizaje supervisado) y descriptivas (aprendizaje no supervisados) [37].En la figura 3.4 se muestra la taxonomía general de las categorías de minería de datos.

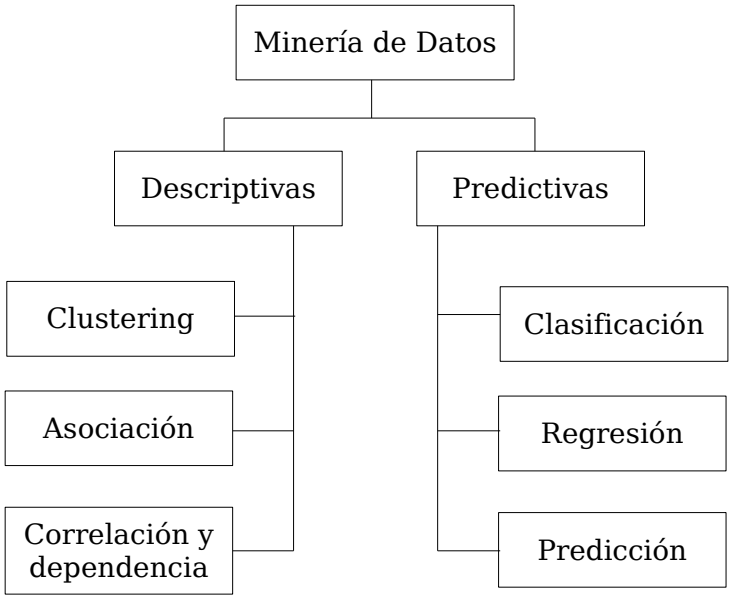

Figura 3.4.: Taxonomía de las técnicas de minería de datos.

Según la funcionalidad, las categorías *predictivas o supervisadas* (por ejemplo clasificación, regresión) predicen el valor de un atributo (etiqueta) de un conjunto de datos a partir de datos previamente conocidos.

Cuando una aplicación no es lo suficientemente madura, no tiene el potencial necesario para una solución predictiva. En ese caso hay que recurrir a las categorías *descriptivas o no supervisados* (por ejemplo: clustering, asociación) que descubren patrones y tendencias en los datos. El descubrimiento de esa información sirve para llevar a cabo acciones y obtener un beneficio o conocimiento de ellas.

3.2.3.1. Técnicas descriptivas

En las tareas predictivas, cada observación incluye un valor de la clase a la que corresponde. El objetivo de estas tareas es predecir el valor de un atributo particular basado en los valores de otros atributos. El atributo a ser predecido se conoce como variable dependiente u objetivo (target), mientras que los atributos utilizados para realizar la predicción se llaman variables independientes o de exploración. Algunas de las tareas descriptivas son:

Clustering: El agrupamiento/segmentación es la detección de grupos de individuos. Se diferencia de la clasificación en que en este caso, no se conocen ni las clases ni su número (aprendizaje no supervisado), con lo que el objetivo es determinar grupos o racimos (clústers) diferenciados del resto. Ejemplo, agrupar los pacientes de un hospital a partir de su historial clínico.

Asociaciones: Una asociación entre dos atributos ocurre cuando la frecuencia de que se den dos valores determinados de cada uno conjuntamente, es relativamente alta. Ejemplo, en un supermercado se analiza si los pañales y la leche del bebé se compran conjuntamente.

Dependencias: Una dependencia funcional (aproximada o absoluta) es un patrón en el que se establece que uno o más atributos determinan el valor de otro. Ejemplo: que un paciente haya sido ingresado en maternidad determina su sexo.

3.2.3.2. Tareas predictivas

En las tareas descriptivas, el conjunto de observaciones no tienen clases asociadas. El objetivo es derivar características (correlaciones, clústers, trayectorias, anomalías) que describan las relaciones entre los datos. Estas tareas son comúnmente de exploración natural y frecuentemente requieren de técnicas de postprocesamiento para explicar los resultados. Algunas de las tareas predictivas son:

Clasificación: Una clasificación se puede ver como el esclarecimiento de una dependencia, en la que el atributo dependiente puede tomar un valor entre varias clases, ya conocidas. Ejemplo: se sabe (por un estudio de dependencias) que los atributos edad, número de miopías y astigmatismo han determinado los pacientes para los que su operación de cirugía ocular ha sido satisfactoria. Se pueden determinar las reglas exactas que clasifican un caso como positivo o negativo a partir de esos atributos.

Regresión: El objetivo es predecir los valores de una variable continua a partir de la evolución sobre otra variable continua, generalmente el tiempo. Ejemplo, se intenta predecir el número de clientes o pacientes, los ingresos, llamadas, ganancias, costos, etc. a partir de los resultados de semanas, meses o años anteriores.

Predicción: Encuentra una clasificación de valores faltantes o sin conocimiento previo. Se refiere tanto a la predicción de valores en los datos como a la predicción de clases utilizando la identificación de distribuciones en los datos disponibles. Ejemplo, predecir qué gobernador será elegido de acuerdo a las actuales encuestas electorales.

De estas tareas de minería de datos, aquellas que son comúnmente utilizadas son: la clasificación, el clustering y la asociación. A continuación se presenta una definición más formal de cada uno de ellos [43].

- **Clasificación** Es el proceso de encontrar un conjunto de modelos (o funciones), los cuales describen y distinguen las clases definidas de los datos, con la finalidad de predecir clases de objetos cuyas clasificaciones no se han definido [19]. La clasificación requiere de un aprendizaje supervisado, es decir, se deben especificar los objetivos (clases) a los que se pretende llegar. El modelo se deriva principalmente del análisis de un conjunto de datos de entrenamiento previamente clasificado.

- **Clustering:** Es una de las tareas del aprendizaje no supervisado en la que no se requiere una clasificación predefinida. El objetivo es particionar los datos obteniendo el conocimiento de acuerdo a las características de los mismos. En general, las clases de los datos no se presentan en el conjunto de datos y los objetos son agrupados basándose en el principio de maximización de similitud dentro de los clústers y minimización de similitud entre clústers diferentes [19].

- **Asociación:** Se basa en el descubrimiento de reglas de asociación demostrando condiciones en los valores de los atributos que ocurren simultáneamente de forma frecuente en un determinado conjunto de datos. La regla de asociación $X \Rightarrow Y$

es interpretada como: *los registros de la base de datos que satisfacen la condición X también satisfacen la condición Y* [19].

En esta tesis, nos concentraremos en analizar las técnicas clustering en minería de datos para el descubrimiento de grupos escondidos en los datos, ya que no todos los conjuntos de datos requieren de una clasificación predefinida para obtener conocimiento de ellos.

3.3. Clustering

El clustering consiste en agrupar una colección dada de patrones no etiquetados en un conjunto de grupos. En este sentido, las etiquetas están asociadas con los grupos, pero las categorías se obtienen únicamente de las propiedades de los datos.

3.3.1. Concepto de clustering

"El clustering representa la división de datos en grupos de objetos similares llamados clústers" [37]. Clustering es también conocido como *clasificación no supervisada*, en donde no se tienen asignación de grupos a clases ya predefinidas, sino que los grupos se van creando de acuerdo a las características de los datos.

Los grupos o *clústers*, son un conjunto de objetos que comparten características similares y juegan un papel muy importante en la manera en como la gente analiza y describe el mundo que los rodea. De forma natural, el humano se encarga de dividir objetos en grupos (clustering) y asignar objetos particulares a dichos grupos (clasificación).

Clustering es una de las técnicas más útiles para descubrir conocimiento oculto en un conjunto de datos. En la actualidad el análisis de clustering en minería de datos ha jugado un rol muy importante en una amplia variedad de áreas tales como: reconocimiento de patrones, análisis de datos espaciales, procesamiento de imágenes, cómputo y multimedia, análisis médico, economía, bioinformática y biometría principalmente [19]. Esto ha hecho posible que el análisis de clustering se considere como una de las mejores técnicas para obtener conocimiento y realizar exploraciones en los datos.

3.3.2. Análisis de clustering

Un problema de análisis de clustering, parte de un conjunto de casos u objetos cada uno de los cuales está caracterizado por varias variables (ver tabla 3.1).

A partir de dicha información se trata de obtener grupos de objetos, de tal manera que los objetos que pertenecen a un grupo sean muy homogéneos entre sí y, por otra parte, la heterogeneidad entre los distintos grupos sea muy elevada. Expresado en términos de variabilidad hablaríamos de minimizar la variabilidad dentro de los grupos para al mismo tiempo maximizar la variabilidad entre los distintos grupos.

Tabla 3.1.: Casos de partida para el análisis de clustering

$C_1 \ldots C_i \ldots C_k$
$O_1\ {x_1}^1 \ldots {x_i}^1 \ldots {x_k}^1$
.....
$O_j\ {x_1}^j \ldots {x_i}^j \ldots {x_k}^j$
.....
$O_n\ {x_1}^N \ldots {x_i}^N \ldots {x_k}^N$

Denotando por $O = O_1, \ldots, O_N$ al conjunto de N objetos, se trata de dividir O en k grupos o clústers, $C_1, \ldots, C_k$ de tal forma que: $\bigcup_{j=1}^{k} x_j = O$.

A partir del planteamiento de un problema de clustering, las actividades del análisis de clustering típicamente involucran los siguientes pasos [31]:

1. *Representación de patrones*. Se refiere al establecimiento del número de clases, número de patrones, y el número, tipo y tamaño de las características disponibles para el algoritmo de clustering.

2. *Definición de proximidad*. La proximidad de los patrones es usualmente medida por una función distancia entre un par de datos.

3. *Clustering*. La etapa de agrupamiento puede desarrollarse en un gran número de formas. Se pueden utilizar agrupamientos de clústers jerárquicos, particionales y otros más abarcan métodos probabilísticos o de teoría de grafos.

4. *Abstracción de datos*. Es el proceso de extraer una representación simple y compacta del conjunto de datos.

5. *Verificación de resultados*. Consiste en validar el análisis de clustering realizado evaluando los resultados obtenidos.

3.3.3. Características de los algoritmos de clustering

Las características deseables de la mayoría de los algoritmos de clustering son las siguientes:

- *Escalabilidad*. La mayoría de los algoritmos de clustering trabajan de manera apropiada con un número pequeño de observaciones (hasta 200 aproximadamente), mientras que se necesita una gran escalabilidad para realizar agrupamiento de datos en bases con millones de observaciones.

- *Habilidad para trabajar con distintos tipos de atributos*. Muchos algoritmos se han diseñado para trabajar sólo con datos numéricos, mientras que en una gran cantidad de ocasiones, es necesario trabajar con atributos asociados a tipos numéricos, binarios, discretos y alfanuméricos.

- *Descubrimiento de clústers con formas arbitrarias*. La mayoría de los algoritmos de clustering se basan en la distancia euclidiana, lo que tiende a encontrar clústers todos con forma (circular) y densidad similares. Es importante diseñar algoritmos que puedan establecer clústers de formas arbitrarias.

- *Requerimientos mínimos en el conocimiento del dominio para determinar los parámetros de entrada*. La herramienta no debería solicitarle al usuario que introduzca la cantidad de clases que quiere considerar, ya que dichos parámetros en muchas ocasiones no son fáciles de determinar, y esto haría que sea difícil controlar la calidad del algoritmo.

- *Habilidad para tratar con datos ruidosos*. La mayoría de las BD contienen datos con comportamiento extraño, datos faltantes, desconocidos o erróneos. Algunos algoritmos de clustering son sensibles a tales datos y pueden derivarlos a clústers de baja calidad.

- *Insensibilidad al orden de las observaciones de entrada*. Algunos algoritmos son sensibles al orden en que se consideran las observaciones. Por ejemplo, para un mismo conjunto de datos, dependiendo del orden en que se analicen, los clústers devueltos pueden ser diferentes. Es importante entonces que el algoritmo sea insensible al orden de los datos, y que el conjunto de clústers devuelto sea siempre el mismo.

- *Alta dimensionalidad*. Una BD o DW (DataWarehouse) puede contener varias dimensiones o atributos, por lo que es bueno que un algoritmo de clustering pueda trabajar de manera eficiente y correcta no sólo en repositorios con pocos

atributos, sino también en repositorios con un alto espacio dimensional, o gran cantidad de atributos.

- *Clustering basado en restricciones*. Es un gran desafío el agrupar los datos teniendo en cuenta no sólo el comportamiento, sino también que satisfagan ciertas restricciones.

- *Interpretación y uso*. Los usuarios esperan que los resultados del clustering sean comprensibles, fáciles de interpretar y de utilizar.

Con estas características, se busca diseñar algoritmos más flexibles que sean capaces de manipular una gran variedad de requerimientos de acuerdo a las necesidades de los usuarios.

3.4. Técnicas de Clustering

Los algoritmos de agrupación de clustering varían entre sí por las reglas heurísticas que utilizan y el tipo de aplicación para el cual fueron diseñados. La mayoría de ellos se basa en el empleo sistemático de distancias entre vectores (objetos a agrupar) así como entre clústers o grupos que se van formando a lo largo del proceso de clustering. Las características básicas por las que los algoritmos de clustering pueden ser clasificados son en función de:

1. El tipo de dato que manejan (numérico, categórico y/o mixto).

2. El criterio utilizado para medir la similitud entre los puntos.

3. Los conceptos y técnicas de clustering empleadas (ej. lógica difusa, estadísticas).

En la literatura existen una gran cantidad de técnicas de clustering que varían de acuerdo a la arquitectura que utilizan [31]. Una clasificación general divide los algoritmos en: clustering particional, clustering jerárquico, clustering basado en densidad y clustering basado en grid.

Para cada una de las categorías presentadas en la figura 3.5, existen una variedad de sub-clasificaciones que presentan algoritmos con diferentes técnicas para encontrar clústers en los datos. Algunas de ellas son:

- *Técnicas estadísticas*, basadas en la utilización de medidas de similitud y análisis estadístico para agrupar los datos.

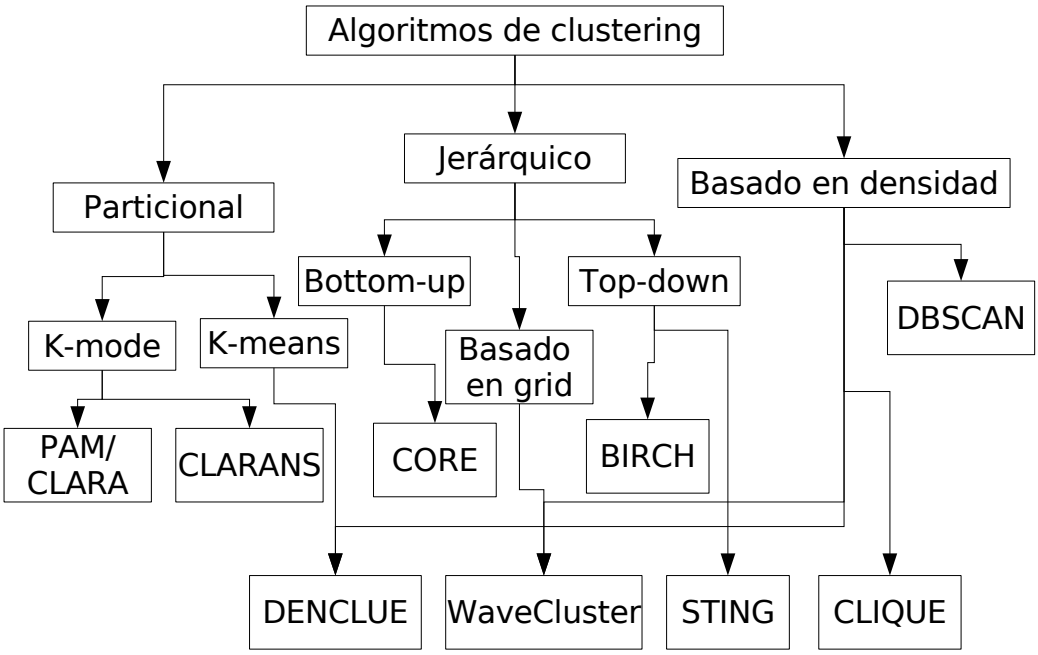

Figura 3.5.: Algoritmos básicos de clustering.

- *Técnicas conceptuales*, basadas en la clasificación de características cualitativas de los datos.
- *Técnicas excluyentes*, basadas en el agrupamiento de datos sin traslape, es decir un dato única y exclusivamente puede pertenecer a una sola clase. La mayoría de los algoritmos de clustering se basan en esta técnica.
- *Técnicas con traslapes*, basadas en técnicas de lógica difusa que consideran grados de pertenencia en los datos. Los objetos pueden pertenecer a más de una clase.

En las siguientes subsecciones se describirán las técnicas de clustering más representativas en minería de datos.

3.4.1. Clustering jerárquico

Un método jerárquico crea una descomposición jerárquica de un conjunto de datos, formando un dendograma (árbol) que divide recursivamente el conjunto de datos en conjuntos cada vez más pequeños [31] . La figura 3.6 muestra la representación gráfica

de un dendograma.

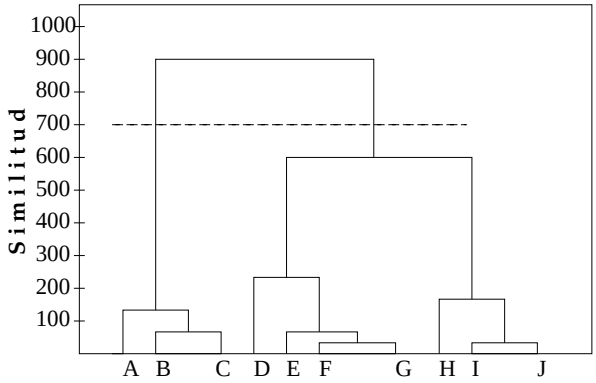

Figura 3.6.: Dendograma de un conjunto de datos.

El árbol puede ser creado de dos formas: de abajo hacia arriba (bottom-up) o de arriba hacia abajo (top-down). En el caso *bottom-up*, también llamado aglomerativo, se comienza con cada objeto formando un grupo por separado. Los objetos o grupos se combinan sucesivamente según determinadas medidas, hasta que todos los grupos se hayan unido en uno solo, o hasta que se cumpla alguna condición de terminación.

En el caso *top-down*, también llamado divisivo, se comienza con todos los objetos en el mismo cluster, y a medida que se va iterando, se dividen los grupos en subconjuntos más pequeños según determinadas medidas, hasta que cada objeto se encuentre en un clúster individual o hasta que se cumplan las condiciones de terminación.

Algunos algoritmos de clustering que pertenecen a esta clasificación son: CURE (Clustering Using Representatives) [16], CHAMALEON [32], BIRCH (Balanced Iterative Reducing and Clustering using Hierarchical) [48] y ROCK (RObust Clustering algorithm using linKs) [17].

3.4.2. Clustering particional

Un algoritmo de clustering particional obtiene una partición simple de los datos en vez de la obtención de la estructura del clúster tal como se produce con los dendogramas de la técnica jerárquica [31]. En la figura 3.7 se muestra un ejemplo de clustering particional.

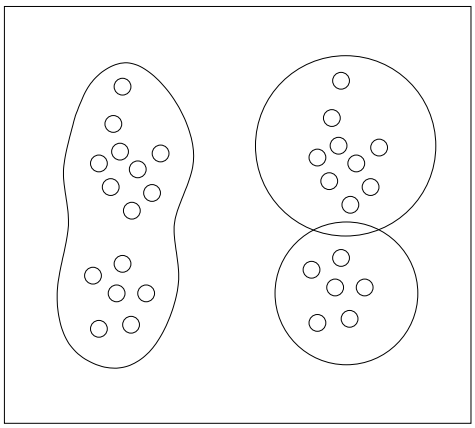

Figura 3.7.: Clustering particional.

El clustering particional organiza los objetos dentro de k clústers de tal forma que sea minimizada la desviación total de cada objeto desde el centro de su clúster o desde una distribución de clústers. La desviación de un punto puede ser evaluada en forma diferente según el algoritmo, y es llamada generalmente función de similitud.

Los métodos particionales tienen ventajas en aplicaciones que involucran gran cantidad de datos para los cuales la construcción de un dendograma resultaría complicado. El problema que se presenta al utilizar algoritmos particionales es la decisión del número deseado de clústers de salida. Las técnicas particionales usualmente producen clústers que optimizan el criterio de función definido local o globalmente. En la práctica, el algoritmo se ejecuta múltiples veces con diferentes estados de inicio y la mejor configuración que se obtenga es la que se utiliza como el clustering de salida.

Algunos algoritmos de clustering que pertenecen a esta clasificación son: CLARA (Clustering Large Applications) [33], CLARANS (Clustering Large Applications based on Randomized Search) [12], K-prototypes [36], K-mode [28], K-Means [29].

3.4.3. Clustering basado en densidad

Los algoritmos basados en densidad obtienen clústers basados en regiones densas de objetos en el espacio de datos que están separados por regiones de baja densidad (estos elementos aislados representan ruido). En la figura 3.8 se muestra un ejemplo de clustering basado en densidad.

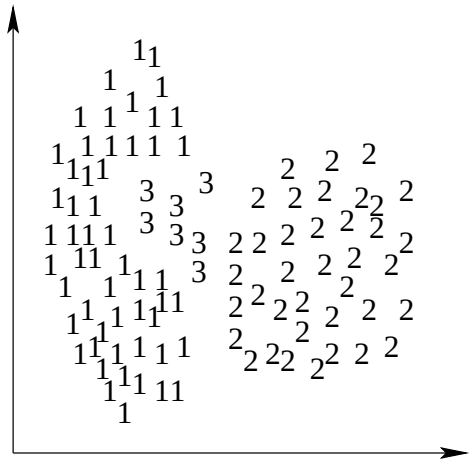

Figura 3.8.: Clustering basado en densidad.

Este tipo de métodos es muy útil para filtrar ruido y encontrar clústers de diversas formas. La mayoría de los métodos de particionamiento, realizan el proceso de clustering con base en la distancia entre dos objetos [31]. Estos métodos pueden encontrar sólo clústers esféricos y se les dificulta hallar clústers de formas diversas.

Algunos algoritmos de clustering que pertenecen a esta clasificación son: DBSCAN (Density-Based Spatial Clustering of Aplications with Noise) [9], OPTICS (Ordering Points To Identify the Clustering Structure) [3] y DENCLUE (DENsity-based CLUstEring) [25].

3.4.4. Clustering basado en grid

Recientemente un número de algoritmos de clustering han sido presentados para datos espaciales, éstos son conocidos como algoritmos basados en grid [31]. Estos algoritmos cuantifican el espacio en un número finito de celdas y aplican operaciones sobre dicho espacio. La mayor ventaja de este método es su veloz procesamiento del tiempo, el cual generalmente es independiente de la cantidad de objetos a procesar. En la figura 3.9 se muestra un ejemplo de clustering basado en grid.

Algunos algoritmos de clustering que pertenecen a esta clasificación son: STING (Statistical Information Grid-based method) [44], CLIQUE [1] y Waveclúster [42].

Figura 3.9.: Clustering basado en grid.

3.4.5. Clustering difuso

En los algoritmos de clustering tradicionales, cada patrón pertenece única y exclusivamente a un solo cluster. El clustering fuzzy asocia cada patrón con cada clúster utilizando funciones de pertenencia [46]. La salida de cada algoritmo es un agrupamiento, no una partición excluyente. La figura 3.10 muestra un ejemplo de clustering difuso generando dos clústers F_1 y F_2 cuyos datos tienen un nivel de pertenencia en los 2 clústers.

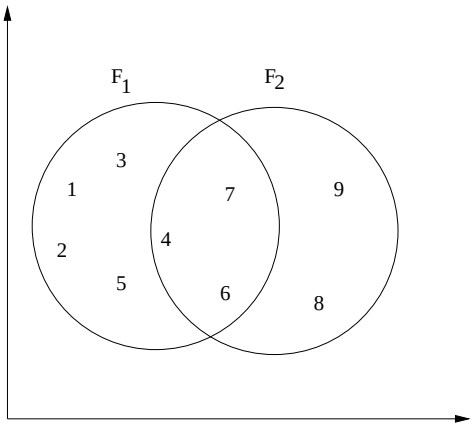

Figura 3.10.: Clustering difuso.

El algoritmo de clustering difuso se describe de la siguiente manera [4]:

1. Selecciona una partición difusa inicial de n en k clústers por medio de una matriz de pertenencia U $n \times k$. Un elemento u_{ij} de esta matriz representa el grado de pertenencia del objeto al clúster ($u_{ij} \epsilon [0, 1]$).

2. Se utiliza U para encontrar el valor de la función de criterio difuso. Se reasignan los datos a los clústers para reducir el valor de la función de criterio y se re-evalúa U.

3. Se repite el paso 2 hasta que los valores de U no cambien significativamente.

3.4.6. Clustering de redes neuronales artificiales

Las redes neuronales artificiales (ANN) [24], fueron motivadas por las redes neuronales biológicas. Las ANN tienen una extensa utilización tanto en técnicas de clasificación como de clustering en minería de datos. Dentro del proceso de clustering las ANN's presentan las siguientes características:

- Procesan vectores numéricos y por lo tanto requieren patrones para poder representarse utilizando únicamente características cuantitativas [31].
- Son inherentemente paralelas y utilizan arquitecturas de procesamiento distribuido.
- Pueden aprender por medio de la interconexión y adaptabilidad de los pesos. Más específicamente, pueden actuar como patrones normalizados y selectores si se seleccionan adecuadamente los pesos.

La arquitectura de estas redes es simple: todas tienen una sola capa. En la figura 3.11 se muestran las conexiones de los pesos de entrada en un nodo de la red.

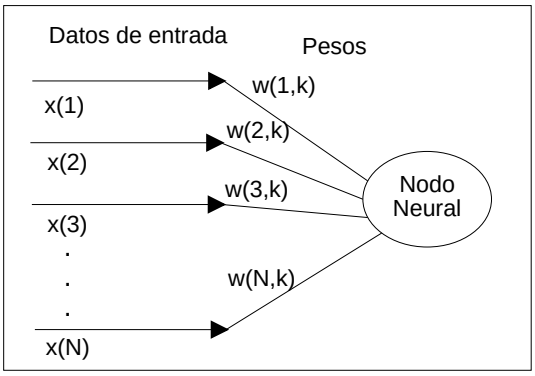

Figura 3.11.: Conexión de los datos de entrada a un nodo neuronal.

Los patrones se introducen en la entrada y son asociadas a los nodos de salida. Los pesos entre los nodos de entrada son iterativamente modificados hasta que se satisfaga

el criterio de terminación. El aprendizaje competitivo tiene su semejanza en las redes neuronales biológicas.

Las ANN más utilizadas en clustering son los vectores de cuantización de aprendizaje de Kohonen (learning vector quantization -LVQ), los mapas de auto-organización de Kohonen (self-organizing map -SOM) [34] y modelos adaptativos de resonancia (adaptive resonance theory - ART) [5].

3.4.7. Clustering de algoritmos evolutivos

Los algoritmos evolutivos, motivados por la evolución natural, utilizan operadores genéticos y una población de soluciones para obtener el óptimo global en la partición de los datos [31]. Los operadores genéticos más utilizados son: selección, cruza y mutación. La figura 3.12 muestra la operación de cruza del proceso evolutivo entre los padres $P1$ y $P2$ generando los hijos $H1$ y $H2$.

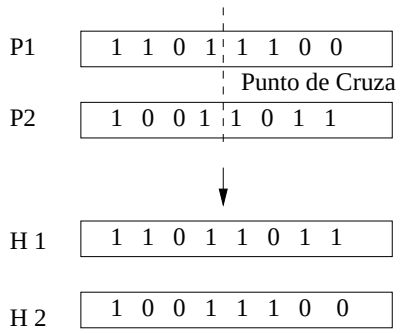

Figura 3.12.: Operación de cruza de un modelo evolutivo.

Las soluciones candidatas para los problemas de clustering se presentan codificadas por medio de cromosomas. Una o más entradas de cromosomas se transforman en uno o más cromosomas de salida. La evaluación de la función de aptitud de un cromosoma determina la probabilidad de un cromosoma a sobrevivir en la siguiente generación.
Los algoritmos de clustering para cómputo evolutivo se describen de la siguiente manera:

1. Seleccionan individuos aleatorios de una población de soluciones. Cada solución corresponde a las k particiones de los datos asociada a una función de aptitud.

2. Se utilizan operadores genéticos de selección, cruza y mutación para generar la siguiente población de soluciones. Se evalúan los nuevos valores de aptitud para estas soluciones.

3. Se repite el paso 2 hasta satisfacer alguna condición de terminación.

Las mejores y más utilizadas técnicas de cómputo evolutivo son: los Algoritmos Genéticos (por sus siglas en inglés: GA's, [26], [14]), las Estrategias Evolutivas (por sus siglas en inglés: ES, [40]) y Programación Evolutiva (por sus siglas en inglés: EP, [11]).

3.4.8. Clustering basado en entropía

Las técnicas de clustering tienen la problemática de diseñar algoritmos que puedan medir la calidad en el particionamiento de los clústers en función de la similitud y diferencia entre los datos. Para el clustering con tipos de datos numéricos, es natural el uso de medidas basadas en distancias geométricas. Sin embargo, si los vectores contienen variables de tipo categórico, los métodos geométricos son inapropiados y se deben buscar otras alternativas.

Recientemente, se han utilizado las medidas de entropía para establecer la similitud entre un conjunto de datos.

"La *entropía* es una medida de la incertidumbre de una variable aleatoria "[35]. Denotando a X como una variable aleatoria discreta, X es el conjunto de posibles valores de X y $p(x)$ es la función de probabilidad de la variable aleatoria X. La entropía $E(X)$ está definida por la ecuación 3.1.

$$E(X) = -\sum_{x \epsilon X} p(x) \log p(x) \tag{3.1}$$

La entropía de un vector multidimensional $\vec{x}$=$X_1, \cdots, X_n$ se puede evaluar como se muestra en la ecuación 3.2.

$$E(\vec{x}) = -\sum_{x_1 \epsilon S(X_1)} \sum_{x_n \epsilon S(X_n)} \cdots p(x_1, \cdots, x_n) \log p(x_1, \cdots, x_n) \tag{3.2}$$

Asumiendo una independencia en los atributos del vector de datos, entonces la ecuación 3.2 se transforma en la ecuación 3.3. Es decir, la unión de las probabilidades de los atributos con valores combinados se convierte en el producto de las probabilidades de cada atributo y por lo tanto, la entropía global se puede calcular como la suma de las entropías de los atributos.

$$E(\vec{x}) = -\sum_{x_1 \epsilon S(X_1)} \sum_{x_n \epsilon S(X_n)} \cdots p(x_1, \cdots, x_n) \log p(x_1, \cdots, x_n)$$

$$= E(X_1) + E(X_2) + \cdots + E(X_n) \tag{3.3}$$

La *entropía* también se define como una medida para evaluar la cantidad de "desorden"que puede haber en un sistema [41]. Por ejemplo, un cuarto que tiene regados calcetines por todos lados presenta mayor valor de entropía que uno en donde los calcetines se encuentren acomodados con su par correspondiente en un lugar específico.

Basándonos en el criterio de entropía de un conjunto de datos de clustering, la formulación del problema de entropía en clustering se presenta definiendo a D como el conjunto de n puntos $d_1, d_2, \cdots, d_n$ con a lo más m posibles valores, donde para cada i, $1 \leq i \leq n$, d_i es un vector de r atributos. Con ésto, se busca particionar el conjunto de n puntos en K clústers dados.

La *entropía* estimada E de una partición de K clústers se puede establecer como la ecuación 3.4:

$$E(C) = \frac{1}{n} \sum_{k=1}^{K} n_k E'(C_k)$$

$$= -\frac{1}{n} \sum_{k=1}^{K} \sum_{j=1}^{r} \sum_{t=0}^{m} n_k \frac{N_{j,k,t}}{n_k} \log \frac{N_{j,k,t}}{n_k} \tag{3.4}$$

donde C representa una partición de D datos en $C_1, C_2, \cdots, C_K$ clases. De aquí se espera que los puntos pertenecientes a un mismo grupo o clase sean similares y los que pertenezcan a grupos diferentes sean lo más distintos posible [35].

En la tabla 3.2 se muestra un ejemplo de la medición de la entropía en clustering. Se tienen 3 vectores v_1,v_2 y v_3 con todas sus posibles combinaciones de tal forma que se formen dos clústers con ellos. Cada agrupamiento tiene la evaluación de entropía por cada clúster E' y la entropía esperada E del total de clústers por agrupamiento. Se puede observar que el agrupamiento del clustering 1 es el que tiene menor evaluación de la entropía esperada E. Esta es obviamente la manera correcta de agrupar los vectores utilizando 2 clústers.

Algunas proposiciones que se tienen al utilizar conceptos de entropía son:

Proposición 1. $E(X) \geq E(C) = \frac{1}{n} \sum_{k=1}^{K} n_k E'(C_k)$.
Prueba Se tiene:

Tabla 3.2.: Medición de entropía en clustering.

No. de clúster	***Clustering 1***		***Clustering 2***		***Clustering 3***	
	vectores	***E'***	***vectores***	***E'***	***vectores***	***E'***
Clúster 0	$\{red, heavy\}$ $\{red, medium\}$	1.0	$\{red, heavy\}$ $\{blue, light\}$	2.0	$\{red, heavy\}$	0
Clúster 1	$\{blue, light\}$	0	$\{red, medium\}$	0	$\{red, medium\}$ $\{blue, light\}$	2.0
$E =$		0.66		1.33		1.33

$$\sum_{k=1}^{K}\sum_{j=1}^{r}\sum_{t=0}^{m} N_{j,k,m}\log\frac{N_{j,k,m}}{n_k}$$

$$=\sum_{j=1}^{r}\sum_{t=0}^{m}\sum_{k=1}^{K} N_{j,k,t}\log\frac{N_{j,k,t}}{n_k}$$

$$\geq\sum_{j=1}^{r}\sum_{t=0}^{m}\left(\sum_{k=1}^{K} N_{j,k,t}\right)\log\frac{\sum_{k=1}^{K} N_{j,k,t}}{\sum_{k=1}^{K} n_k}$$

$$=\sum_{j=1}^{r}\sum_{t=0}^{m} N_{j,t}\log\frac{N_{j,t}}{n}$$

Notar que:

$$E'(X)=-\sum_{j=1}^{r}\sum_{t=0}^{m} n_k\frac{N_{j,t}}{n}\log\frac{N_{j,t}}{n}$$

$$E(C)=-\frac{1}{n}\sum_{k=1}^{K}\sum_{j=1}^{r}\sum_{t=0}^{m} N_{j,k,t}\log\frac{N_{j,k,t}}{n_k}$$

De aquí se tiene $E'(X)\geq E(C)$.

La proposición 1 muestra que cualquier proceso de clustering decrementa el valor de la entropía. El objetivo del clustering es encontrar una partición C tal que el decremento en la entropía se maximice. En otras palabras, se busca minimizar $E(C)$.

Proposición 2. *E(C) se maximiza cuando todos los datos están en el mismo cluster.*

Prueba Si todos los puntos pertenecen a un cluster, entonces $E(C) = E'(X)$. Por lo tanto de acuerdo a la Proposición 1, $E(C)$ se maximiza.

Lo anterior muestra que un sub-espacio con clústers tiene típicamente menor entropía que un sub-espacio sin clústers. Por lo tanto, utilizar el criterio de entropía en los procesos de clustering para evaluar la variabilidad de los datos dentro de los clústers representa una metodología eficiente para datasets que contengan tipos de dato mixtos.

3.5. Comentarios

Existe gran cantidad de información representada como datos, en donde no se define (en la mayoría de los casos) el objetivo previo que se pretende alcanzar. El clustering es una de las tareas del aprendizaje no supervisado en minería de datos en la que no se requiere una clasificación predefinida de los datos, para particionarlos y obtener conocimiento de los mismos.

De la variedad de técnicas de clustering anteriores, se puede observar que existe una gran cantidad de algoritmos que nos ayudan a descubrir propiedades particulares de los datos. Cada uno de ellos está enfocado a tareas específicas y por lo tanto su utilización en la mayoría de los casos, depende del tipo de dato que presentan y la aplicación hacia la cual está enfocada.

Por lo tanto, el desarrollo de la tesis se basa en la implementación de un algoritmo de clustering jerárquico con la capacidad de manipular tipos de datos mixtos (numéricos y categóricos).

Capítulo 4.

Algoritmo Propuesto ACEM

El algoritmo propuesto *ACEM* (**A**daptación alternativa de un algoritmo **C**ategórico utilizando **E**ntropía para manejar datos **M**ixtos) se basa en el clustering de datasets con atributos de tipo mixto (es decir, numérico y categórico). *ACEM* está inspirado en el algoritmo de clustering para datos categóricos *ROCK* [17], el cual emplea ligas y no distancias para medir la similitud y proximidad entre un par de clústers. En las siguientes secciones se describe la estructura básica del algoritmo *ACEM* y se muestran algunas características del algoritmo original sobre el cual se inspiró nuestro algoritmo, resaltando las diferencias principales con respecto al algoritmo *ACEM*.

4.1. Algoritmo de inspiración

Usualmente los algoritmos de clustering emplean una distancia entre clústers basada en métricas, por ejemplo: distancia euclidiana, minkowski, mahalanobis, etc. Sin embargo, estas medidas no son apropiadas para atributos de tipo booleano y categórico, debido a las posibles pérdidas de información.

El algoritmo *ROCK* (RObust Clustering using LinKs) emplea ligas y no distancias para medir la similitud y proximidad entre el par de clústers a ser mezclados [17]. Las características básicas del algoritmo *ROCK* son:

1. Pertenece a la clase de algoritmos jerárquicos aglomerativos que trabajan con datos de tipo categórico
2. Utiliza una muestra aleatoria del conjunto de total de datos para trabajar.
3. Maneja atributos con valores faltantes.
4. Utiliza una función para medir la calidad de la mezcla de los clústers.
5. Utiliza ligas para medir el número de vecinos en común entre 2 puntos.

6. Mezcla los clústers hasta que se obtengan los k clústers especificados por el usuario.

7. Descarta del proceso de clustering los datos con pocos o ningún vecino (outliers).

8. Encuentra clústers de formas arbitrarias.

4.1.1. Algoritmo ROCK

ROCK es un algoritmo diferente, diseñado con la estabilidad y robustez suficiente para trabajar en aplicaciones con datasets de tipo de dato categórico. En la tabla 4.1 se resumen algunas características del algoritmo de clustering jerárquico *ROCK*.

Tabla 4.1.: Características *ROCK*

Nombre	**Tipo de dato**	**Complejidad**	**Geometría**	**Maneja Ruido**
ROCK	Categórico.	$O(n^2 + nm_m m_a + n^2 \log n)$, $O(n^2, nm_m m_a)$	Formas aleatorias.	Sí

*Aquí m_m es el número máximo de vecinos por punto y m_a es el número promedio de vecinos por punto.

En la vida real es muy común que se presente el problema de analizar datasets con tipos de datos mixtos. En los algoritmos 1 y 2 se muestra el pseudo-código del algoritmo de clustering *ROCK*.

El algoritmo *ROCK* evalúa las distancias entre objetos utilizando el coeficiente Jaccard [30]. Utiliza el parámetro θ para determinar quienes son los vecinos en cada uno de los objetos. Dado un punto p, un punto q es vecino de p si el coeficiente de Jaccard $sim(p, q)$ excede el valor de θ.

Se generan los valores de la matriz de ligas ($links$), la cual consiste en la evaluación de $links(p, q)$ como el número de vecinos comunes entre los puntos p y q. El algoritmo comienza con el procedimiento de clustering de los datos utilizando una técnica aglomerativa en la cual en un inicio todos los datos pertenecen a un clúster diferente y sucesivamente se van agrupando. El procedimiento de mezcla se basa en la selección de

Algoritmo 1: Algoritmo principal ROCK

```
1  Entrada: S, conjunto de datos
2  Entrada: k, numero de clústers
3  Salida: clústers con los datos
4  begin
5      link := compute links(S)
6      foreach  s ∈ S do
7          q[s]:= buildLocalHeap(link,s)
8      end
9      Q := buildGlobalHeap(S, q)
10     while  size(Q) > k do
11         u := extractMax(Q)
12         v := max(q[u])
13         delete(Q, v)
14         w := merge(u, v)
15         foreach  x ∈ q[u] ∪ q[v] do
16             link[x,w] := link[x, u] + link[x, v]
17             delete(q[x], u)
18             delete(q[x], v)
19             insert(q[x], w
20             g(x,w))
21             insert(q[w], x, g(x,w))
22             update(Q, x, q[x])
23         end
24         insert(Q,w, q[w])
25         deallocate(q[u])
26         deallocate(q[v])
27     end
28 end
```

Algoritmo 2: Función que evalúa las ligas

```
1  Entrada: S // conjunto de datos
2  begin
3      nbrlist[i] //evalua para cada i en S link[i,j] //inicializa a cero para todo i, j  for
       i=1 to n do
4          N := nbrlist[i]
5          for  j= 1 to |N| - 1 do
6              for  l= j + 1 to |N| do
7                  link[N[j],N[l]] := link[N[j],N[l]]+ 1
8              end
9          end
10     end
11 end
```

los mejores clústers u y v a ser mezclados, los cuales se ordenan en forma decreciente de acuerdo a la mejor medida de efectividad de la función $g(j, max(q[j]))$. De aquí, los clústers j en Q (la pila global de los clústers) y el máximo elemento en $q[j]$ (la pila local de vecinos de j) son los mejores clústers a ser mezclados. La secuencia $while - loop$ se repite hasta que la pila global Q contenga los k clústers definidos previamente por el usuario.

En la literatura aún no se ha publicado una posible extensión del algoritmo *ROCK* para manejar datos mixtos. Por lo tanto, de todo lo anterior surgió la idea de implementar una extensión alternativa de un algoritmo categórico (en este caso se pensó en *ROCK*) para manejar tipos de datos mixtos agregando además nociones de entropía para evaluar la pertenencia de los datos a los clústers.

4.2. Modelo inicial alternativo

Uno de los primeros modelos propuestos para evaluar la pertenencia de los datos mixtos en el proceso de clustering, se basó en la utilización de medidas geométricas. Se planteó utilizar la *distancia euclidiana* como medida para evaluar la distancia de los datos a cada centro de los clústers. La *distancia euclidiana* es una de las métricas más utilizadas para medir propiedades continuas en los datos.

En nuestra primera propuesta se planteó dividir la estructura general en 2 fases:

- En la primera fase, se utilizaría la implementación alternativa del algoritmo ca-

tegórico de inspiración, para generar clústers iniciales con los datos categóricos puros del dataset.

- En la segunda fases se utilizaría la *distancia euclidiana* como base para medir la distancia de los datos y evaluar su pertenencia a los clústers. Es decir, utilizando los resultados de la etapa anterior, se localizan los puntos representativos de cada clúster y se evaúa la distancia euclidiana en los datos para verificar su cercanía (pertenencia) a los clústers correctos. Al utilizar una métrica de distancia para datos numéricos, fue necesario implementar una metodología de conversión y normalización de datos categóricos a datos numéricos.

En este modelo inicial, se obtuvieron buenos resultados en algunos experimentos realizados (se presentarán en el capítulo de comparación y evaluación de resultados). Sin embargo, la conversión de variables puede representar pérdidas de información en el proceso de clustering y no es adecuado para todos los datasets con datos mixtos.

De esta manera, surgió la idea de implementar una nueva metodología de evaluación de datos mixtos utilizando nociones básicas de entropía en clustering. La evaluación de entropía representa la cantidad de desorden acumulado en los datos. Por lo tanto, su utilización en el proceso de verificación de pertenencia de los datos en los clústers, asegura el agrupamiento de datos similares en un mismo clúster.

4.3. Estructura del algoritmo propuesto

La estructura básica del algoritmo *ACEM* utilizando entropía está divida en 2 fases principales: a) *fase 1*, pre-clasifica los datos categóricos puros del dataset y b) *fase 2*, evalúa el valor de entropía de cada clúster verificando la pertenencia de los datos (ver figura 4.1).

4.3.1. Fase 1 - Datos categóricos

Toma los datos categóricos puros del dataset y los pre-clasifica en clústers utilizando la implementación de un algoritmo de clustering para datos con atributos categóricos. El algoritmo utilizado en esta fase, es una adaptación alternativa de un algoritmo de datos categóricos, inspirada en el algoritmo *ROCK*. Se utiliza la misma estructura del algoritmo de inspiración pero se modifican algunas características que hacen de *ACEM* un algoritmo diferente. Dichas características son:

- Se evalúan internamente las operaciones para obtener los vecinos más cercanos tomando como base únicamente el conjunto de datos de entrada.

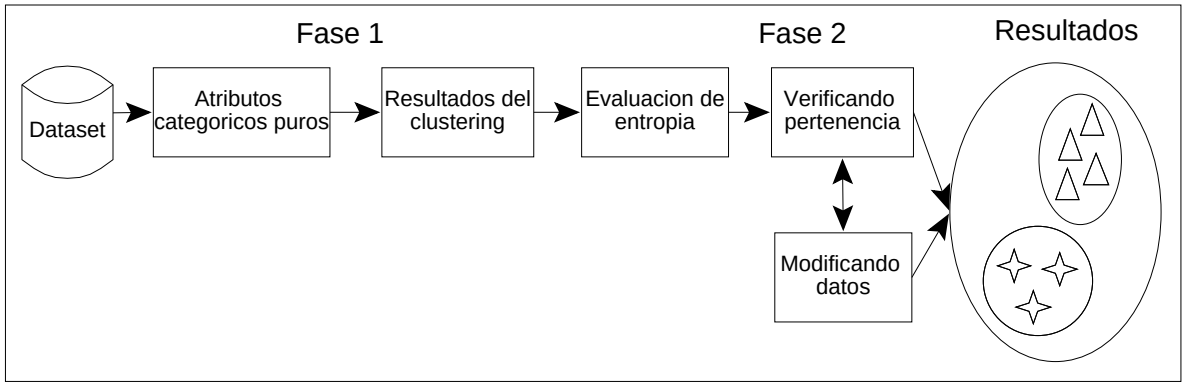

Figura 4.1.: Estructura del algoritmo *ACEM*.

- Se permite al usuario excluir variables que a decisión propia no interesen para el clustering de los datos.

- El algoritmo puede encontrar una clasificación diferente a los k clústers especificados inicialmente. Es decir, la definición de k representa el número mínimo de clústers deseados.

- El algoritmo no saca definitivamente los datos del proceso de clustering. Los datos se asignan a los clústers que compartan más características en común con ellos.

4.3.2. Fase 2 - Datos mixtos

En esta etapa, se evalúan los resultados de la pre-clasificación realizada en la *fase 1*, midiendo el valor de entropía de cada clúster. En esta fase del proceso de clustering, se incluyen tanto los datos numéricos como los datos categóricos para verificar su pertenencia al clúster asignado. Al medir la entropía del conjunto de datos mixtos se busca asegurar que los datos permanezcan o sean cambiados al clúster con más características en común. Es decir, los datos cuya variación de entropía dentro del clúster sea grande son cambiados al clúster con menor variación de entropía.

En nuestro algoritmo se propuso utilizar la entropía (originada en teoría de la información) como una medida para evaluar la pertenencia de los datos a un clúster. Este método es motivado por el hecho de que en un sub-espacio con clústers se tiene típicamente menor entropía que en un sub-espacio sin clústers.

4.4. Algoritmo ACEM

El algoritmo *ACEM* toma como entrada los siguientes parámetros: 1) el conjunto de datos D conteniendo los n puntos a ser agrupados, 2) el número de clústers k y 3) el valor definido para θ en un rango [0:1]. El algoritmo toma en un inicio los datos categóricos puros del dataset y los clasifica en clústers iniciales evaluando las ligas entre los datos y midiendo la similitud entre 2 transacciones con el *Coeficiente de Jaccard* y el parámetro θ ($sim(T_1, T_2) \geq \theta$).

Inicialmente, cada punto se toma como un clúster diferente. Se construye una pila local $q[i]$ para cada clúster y una pila global Q que contiene todos los clústers ordenados en forma decreciente en función a la medida de efectividad. En cada iteración, el último clúster j en Q y el último clúster en $q[j]$ son el mejor par de clústers que serán mezclados.

Del proceso anterior, se obtiene una pre-clasificación de los datos utilizando únicamente los atributos de tipo categórico y se utiliza el conjunto completo de datos (numéricos y categóricos) para realizar una medición de entropía de los clústers y verificar la pertenencia de los datos a un clúster. El algoritmo termina cuando todos los datos sean asignados a los clústers con mayores características en común. En el algoritmo 3 se describe en pseudo-código nuestro algoritmo propuesto.

El algoritmo *ACEM* utiliza métodos específicos para realizar la evaluación de las ligas entre vecinos, medir la efectividad de los mejores clústers a ser mezclados y evaluar los valores de entropía de cada clúster para establecer la pertenencia de los datos. En las siguientes subsecciones se describirán dichas metodologías.

4.4.1. Evaluación de vecinos

Los vecinos son aquellos objetos que son considerablemente similares entre sí. La función de similitud $sim(p_i, p_j)$ se encarga de obtener el par de puntos p_i y p_j más cercanos, es decir, aquellos con más características en común. Esta función sim se encuentra entre valores de 0 y 1 con la finalidad de evaluar si dos objetos son similares o no. Al definir el parámetro de entrada θ entre valores de 0 y 1, se dice que los puntos p_i y p_j son vecinos si cumplen con la ecuación 4.1.

$$sim(p_i, p_j) \geq \theta \tag{4.1}$$

El parámetro θ se utiliza como variable de control para restringir qué tan cerca deben de estar un par de puntos para ser considerados como vecinos. Por lo tanto, valores

Algoritmo 3: Algoritmo *ACEM*

```
1  Entrada: D,k,θ
2  Salida: Q // Pila global con los clústers
3  begin
4      atributo ∈ D // tipo de dato
5      if atributo == categorico then
6          link := evalua-vecinos(D)
7          foreach d ∈ D do
8              q[d] := pila-local(link,d)
9              if d.vecinos ≤ 1 then
10                 faltantes:=faltantes+1
11             end
12         end
13         Q := pila-global(D,q)
14         while size(Q) > k do
15             u := extrae-max(Q)
16             if u.pila == NULL then
17                 if pilas-vacias(Q) !=0 then
18                     u:= extrae-sig(Q)
19                     else
20                         break
21                     end
22                 end
23             end
24             v := max(q[u])
25             delete(Q, v)
26             w := mezcla(u, v)
27             update-links(u,v)
28         end
29         insert(Q,w, q[w])
30     end
31     E:= obtiene-entropia(Q)
32     for j=1 to faltantes do
33         cluster-menor-entropia := menor-entropia(E)
34         insert(d.faltante,cluster-menor-entropia)
35         E:= update-entropia(Q)
36     end
37     // evaluando pertenencia
38     foreach d ∈ D do
39         cluster-menor-entropia := menor-entropia(E,d)
40         if d.clúster != cluster-menor-entropia then
41             cambia-dato(cluster-menor-entropia)
42         end
43     end
44 end
```

cercanos a 1 en θ corresponden a objetos con altas probabilidades de ser considerados como vecinos.

El parámetro θ se define inicialmente por el usuario. Si su valor es definido en 1 se permite obtener únicamente vecinos idénticos al objeto. Si su valor es definido en 0 se permite obtener como vecino a cualquier objeto.

La ecuación 4.2 define la función de similitud para las transacciones T_1 y T_2 basada en el coeficiente de Jaccard [30].

$$sim(T1, T2) = \frac{\mid T_1 \cap T_2 \mid}{\mid T_1 \cup T_2 \mid} \tag{4.2}$$

Donde $\mid T_i \mid$ es el número de objetos en T_i. La mayor cantidad de variables que tengan en común las transacciones T_1 y T_2 en $\mid T_1 \cap T_2 \mid$ representa el grado de similitud entre ellos. Al dividir este valor entre $\mid T_1 \cup T_2 \mid$ se obtiene el factor de escalamiento que asegura que θ se encuentre en valores entre 0 y 1.

El algoritmo puede manipular atributos con valores faltantes. Si falta algún valor de un atributo, simplemente se ignora dicho valor y se consideran aquellos atributos que permanezcan definidos en ambos objetos de evaluación de similitud.

4.4.2. Evaluación de ligas

El clustering de datos basándose únicamente en la cercanía y similitud entre ellos no es suficientemente robusto para distinguir dos clústers diferentes, porque es posible que puntos en diferentes clústers sean vecinos. Incluso, si los puntos p_i y p_j son vecinos y pertenecen a clústers diferentes, es casi imposible que tengan un número grande de vecinos en común.

La evaluación de ligas $link(p_i p_j)$ se define como el número de vecinos en común entre los puntos p_i y p_j. Si el valor de $link(p_i p_j)$ es grande, es muy probable que p_i y p_j pertenezcan al mismo clúster.

Una manera de ver el problema de obtener las ligas entre un par de puntos, es considerar una matriz adyacente A de $n \times n$ en donde los datos de entrada son 0 o 1 dependiendo de si los puntos i y j son o no vecinos respectivamente. El número de ligas entre un par de puntos i y j puede ser obtenido multiplicando la fila i con la columna j (lo cual es: $\sum_{l=1}^{n} A[i,l] * A[l,j]$). Por lo tanto, el problema de evaluar el número de ligas de un par de puntos es simplemente realizar la multiplicación de la matriz adyacente

A por sí misma, es decir, $A \times A$. El tiempo de complejidad del algoritmo para evaluar el cuadrado de la matriz es $O(n^3)$. Sin embargo, el problema de calcular el cuadrado de una matriz ha sido bien estudiado y algoritmos bien conocidos como el algoritmo Strassen se ejecutan en un tiempo $O(n^{2,81})$. Se espera que, en promedio, el número de vecinos por cada punto sea pequeño comparado con el número de puntos de entrada n.

Para cada punto, después de evaluar una lista de vecinos, el algoritmo considera todos los pares de sus propios vecinos. Para cada par, el punto contribuye a una liga. Si el proceso se repite para cada punto y el número de ligas se incrementa para cada par de vecinos, al final se obtendrá el número de ligas de todos los pares de puntos.

4.4.3. Medida de efectividad

Los mejores puntos de clustering son aquellos que resultan en maximizar la función de efectividad. Como se busca encontrar un clúster que maximice la función de efectividad, se utiliza la función de efectividad con la finalidad de determinar el mejor par de clústers que se pueden mezclar en cada paso del algoritmo. Para el par de clústers C_i,C_j la función $link[C_i, C_j]$ almacena el número de ligas entre los clústers C_i y C_j, lo cual se define como: $\sum link(p_q, p_r)$. Por lo tanto, la medida de efectividad $g(C_i, C_j)$ al mezclar los clústers C_i y C_j se muestra en la ecuación 4.3.

$$g(C_i, C_j) = \frac{link[C_i, C_j]}{(n_i + n_j)^{1+2f(\theta)} - {n_i}^{1+2f(\theta)} - {n_j}^{1+2f(\theta)}} \tag{4.3}$$

El par de clústers para los cuales se maximiza la medida de efectividad (descrita previamente), son los mejores pares de clústers que pueden ser mezclados en cualquier paso del algoritmo. Por lo tanto, se puede ver que cualquier par de clústers con un gran número de ligas (links) representan buenos candidatos a ser mezclados por el algoritmo. Sin embargo, al utilizar únicamente el número de ligas entre el par de clúster como un indicador de buenas mezclas puede no ser completamente apropiado. Este método propuesto puede trabajar bien para clústers bien separados, pero en caso de que el clúster presente ruido (outliers) o puntos que sean vecinos, un clúster muy grande podría devorar otros clústers más pequeños y se podrían mezclar puntos de diferentes clústers en uno solo. Esto sucede debido a que la mayoría de los clústers de tamaño grande tienen un gran número de ligas respecto a los demás clústers.

Para remediar este problema, se divide el número de ligas entre los clústers entre el número esperado de ligas entre ellos. De aquí, si cada punto en el clúster C_i tiene un total de $n_i^{f(\theta)}$ vecinos, entonces el número esperado de ligas que incluyen únicamente

puntos es $n_i^{1+2f(\theta)}$. La función $f(\theta)$ es una función que depende de los datos y del tipo de clúster que se va a utilizar, por lo que no es fácil determinar el valor adecuado para dicha función. Sin embargo, en la literatura se sabe que si los clústers están considerablemente bien distribuidos, se puede definir a la función $f(\theta)$ como: $f(\theta) = 1+2*\frac{1-\theta}{1+\theta}$.

Para clústers muy grandes, se puede asumir que los puntos fuera del clúster contribuyen mínimamente al número de ligas entre el par de puntos en el cluster. Por lo tanto, el número esperado de ligas entre puntos dentro del clúster es aproximadamente $n_i^{1+2f(\theta)}$. Como resultado, se tiene que si 2 clústers considerablemente grandes de tamaños n_i y n_j se mezclan, el número de ligas entre pares de puntos en el clúster mezclado es $(n_i + n_j)^{1+2f(\theta)}$ mientras que el número de ligas entre los clústers antes de mezclarse eran $n_i^{1+2f(\theta)}$ y $n_j^{1+2f(\theta)}$, respectivamente. Por lo tanto, el número esperado de ligas entre el par de puntos dentro clústers diferentes está dada como $(n_i + n_j)^{1+2f(\theta)} - (n_i)^{1+2f(\theta)} - (n_j)^{1+2f(\theta)}$.

4.4.4. Evaluación de entropía

La entropía se utiliza para evaluar la similitud entre un conjunto de datos. Permite demostrar que un sub-espacio de clústers tiene típicamente menor entropía que un sub-espacio sin clústers. Por lo tanto, retomando el criterio de entropía en clustering definido en la ecuación 3.4 del capítulo 3, se evalúa la variabilidad de entropía de los datos contenidos en los clústers. El clustering basado en nociones de entropía nos permite medir la cantidad de desorden (variación) contenida en los clústers.

El proceso de clustering de los datos categóricos puros (fase 1 del algoritmo *ACEM*), genera clústers en donde los datos incluidos en un mismo cúster son similares y los datos pertenecientes a otros cústers son diferentes. Por lo tanto, al utilizar datasets con la mayor parte de atributos de tipo categórico, se espera que la variación de entropía en los clústers iniciales sea casi cero, ya que los datos fueron agrupados utilizando una implementación alternativa de un algoritmo de clústering de tipo categórico. Es decir, como los datos contenidos en un mismo clúster presentan cierta similitud, se tiene poca o nula variación en los datos. Por lo tanto, la evaluación de entropía se minimiza y su valor se puede observar cercano a cero.

Al evaluar la entropía inicial en cada uno de los clústers, se incluyen en el proceso de evaluación el conjunto de datos mixtos (numéricos y categóricos) del dataset. De aquí, a partir de los resultados generados en la evaluación de entropía inicial, se verifica que los datos fueron correctamente asignados al conjunto de clústers. Si se utilizan

datasets mixtos con mayor cantidad de datos categóricos que numéricos, los resultados del proceso de clustering no presentarán muchas modificaciones en la evaluación de pertenencia y asignación de los datos a los clústers correctos.

El proceso de evaluación de entropía (pertenencia y modificación de datos) en los clústers se describe en el algoritmo 4.

Algoritmo 4: Proceso de evaluación de entropía.

```
1 begin
2     Evaluar la entropía inicial en cada clúster con datos mixtos.
3     Tomar un dato de un clúster.
4     Agregar el dato a todos los clústers existentes.
5     Evaluar la entropía actual en todo el conjunto de datos.
6     Si la variación de la entropía inicial y la entropía actual es cero, el dato no se
      modifica de clúster.
7     Si la variación de la entropía inicial y la entropía actual es diferente de cero,
      el dato cambia al clúster con menor variación de entropía.
8     Se repite el paso 3 hasta evaluar todos los datos.
9 end
```

Como resultado, se obtienen los k clústers conteniendo los datos con la menor evaluación de entropía, lo cual minimiza la variabilidad dentro de los clústers pero maximiza la variabilidad de los datos fuera de ellos.

4.4.5. Complejidad

La complejidad del algoritmo *ACEM* se presenta como una suma de complejidades de los procedimientos de evaluación de ligas, mezcla/actualización consecutiva de clústers y medición de entropía.

Podemos hacer una similitud de la evaluación de complejidad presentada en el algoritmo de inspiración *ROCK* [17], ya que *ACEM* se basa en su estructura como algoritmo de inspiración. El tiempo de complejidad del algoritmo *ROCK* es $O(n^2+nm_mm_a+n^2logn)$.

El algoritmo *ACEM* en su primera fase (*fase 1*), utiliza la misma estructura que la presentada en la evaluación de ligas del algoritmo *ROCK*, por lo tanto su complejidad se define como: $O(n^2 + nm_mm_a)$. De aquí, la evaluación de los vecinos se puede dar con: $min(n^2, nm_mm_a)$, donde m_m es el número máximo de vecinos por punto y m_a es

el número promedio de vecinos por punto.

El procedimiento de mezcla es básicamente el mismo que el presentado por el algoritmo *ROCK*. Sin embargo, la implementación que utilizamos en el algoritmo *ACEM* difiere de la de *ROCK*. Por lo tanto la complejidad del procedimiento también difiere. En *ROCK* se muestra una complejidad de $O(n^2 logn)$ utilizando árboles para este procedimiento.

En nuestra implementación utilizamos listas ligadas. La complejidad de las pilas locales es $O(n)$ (se pueden construir en un tiempo lineal) y la complejidad de las pilas globales también es $O(n)$ (tiene a lo más n clústers iniciales). El procedimiento de actualización de datos en las n pilas como consecuencia de la mezcla de los clústers tiene una complejidad de $O(n^2)$. Como el proceso de mezcla dentro del *while-loop* se lleva a cabo en un tiempo de $O(n)$, entonces la complejidad total del procedimiento es $O(n^3)$.

Es importante mencionar que el algoritmo *ROCK* presenta su análisis de complejidad de forma teórica, es decir, no considera los tiempos de actualización de árboles (utilizados en su implementación) debido al procedimiento de mezcla. Por lo tanto la complejidad es $O(nlogn)$. Por lo tanto, la complejidad total de *ROCK* quedaría como $O(n^2 + nm_m m_a + (n^2 logn)(nlogn)) = O(n^2 + nm_m m_a + (n^3 (logn)^2)$, lo cual es muy semejante a la que obtiene el algoritmo *ACEM* en la *fase 1* ($O(n^3)$).

En el procedimiento de evaluación de entropía y verificación de pertenencia de los datos a los clústers, se tiene un complejidad general de $O(n^2)$. La medición de entropía está dada en un tiempo $O(n)$. La verificación del conjunto de datos requiere de a lo más n actualizaciones en la evaluación de entropía, teniendo una complejidad de $O(n^2)$. Por lo tanto la complejidad de *ACEM* está dada como $O(n^2 + nm_m m_a + n^3 + n^2)$, lo cual engloba la complejidad total en un tiempo $O(n^3)$.

4.5. Algoritmos de comparación

Seleccionamos dos algoritmos de clustering para la comparación de resultados respecto a nuestro algoritmo propuesto *ACEM*. En el capítulo 5 se muestran todas las evaluaciones y comparación de resultados efectuadas para medir el desempeño de nuestro algoritmo. El primer algoritmo de comparación denominado *GAClust* es un algoritmo de clustering de tipo categórico y se utiliza en la *fase 1* de los experimentos realizados en *ACEM*. El segundo algoritmo de comparación es *k-prototypes*, el cual es un algoritmo de clustering de tipo mixto utilizado en la *fase 2* de comparación del algoritmo *ACEM*.

En las siguientes subsecciones se describirán ampliamente cada uno de ellos.

4.5.1. Algoritmo de clustering para datos categóricos

El algoritmo de comparación utilizado en la *fase 1* de agrupamiento de datos categóricos es el algoritmo *GAClust*[1] [7]. Este es un algoritmo genético en donde las particiones están representadas como cromosomas. Se basa en evaluar la cercanía entre dos particiones utilizando una generalización de la entropía clásica condicional.

Los objetos que se agrupan son representados por vectores en una tabla y el objetivo del algoritmo genético es construir una partición del conjunto de filas cuyas clases son definidas como clústers de las filas. El algoritmo *GAClust* requiere que las particiones satisfagan las siguientes dos condiciones:

1. Deben representar el conjunto de particiones generadas por los atributos de la tabla.

2. El número de clases de la partición no debe exceder el límite máximo predefinido.

La primera condición asegura que los resultados de la partición representen el agrupamiento (clustering) de las filas, mientras que el segundo, representa una restricción en el número de clases del clustering.

El objetivo del algoritmo genético es modificar el conjunto de cromosomas de la población actual utilizando operadores genéticos de cruza y mutación, de tal forma que la nueva población tenga cromosomas que incrementen su cercanía con la partición promedio. En el algoritmo 5 se muestra el pseudocódigo del algoritmo *GAClust*.
El algoritmo utiliza el operador de selección de *la ruleta* y el operador de cruza de *un punto*. La estrategía de la ruleta establece que los cromosomas con valores grandes en su función de aptitud tienen más oportunidad de ser seleccionados.

La cruza de un punto consiste en tomar dos cromosomas padres, seleccionar un punto aleatorio de cruza l entre 1 y N y generar dos cromosomas hijos por medio de combinación de las posiciones 1 a l y $l + 1$ a N de cada uno de los padres. La figura 4.2 muestra la operación de cruza de un punto entre los padres $P1$ y $P2$ generando los hijos $H1$ y $H2$.

[1]Los códigos fuentes del algoritmo están disponibles en: http://www.cs.umb.edu/~dana/GAClust/index.html. Recurrir a este sitio para más detalles acerca de otros parámetros.

Algoritmo 5: Algoritmo *GAClust*

```
1  begin
2      Inicializa la población del algoritmo genético
3      while true do
4          Evalua la aptitud de los cromosomas en la población
5          if no hay una mejora en la mejor aptitud para N_nmax iteraciones then
6              Salida de la partición de Kbest
7              exit;
8          end
9          Copia los mejores cromosomas a la nueva población
10         Selecciona probabilísticamente 2 cromosomas de cruza
11         Aplica el operador de cruza a los cromosomas seleccionados
12         Copia los hijos a la nueva población
13         Selecciona con probabilidad uniforme los cromosomas de mutación
14         Aplica el operador de mutación a los cromosomas seleccionados
15         Copia los cromosomas modificados a la nueva población
16         reemplaza la vieja población por la actual
17     end
18 end
```

4.5.2. Algoritmo de clustering para datos mixtos

El algoritmo de clustering utilizado para la comparación de resultados en la fase de datos mixtos es el algoritmo *k-prototypes* [27]. Este algoritmo se basa en el paradigma del algoritmo *k-means* [29], preservando su eficiencia pero eliminando la limitación de trabajar con únicamente datos de tipo numérico.

El algoritmo agrupa objetos con atributos de tipo numérico y categórico en una forma muy similar a la manera como lo hace el propio algoritmo *k-means*. Es un método que actualiza dinámicamente los k prototipos de los datos, con la finalidad de maximizar la similitud dentro de los clústers. La medida de similitud se deriva de ambos atributos: numéricos y categóricos. Si se introducen datasets con datos exclusivamente numéricos, el algoritmo funciona de forma idéntica al algoritmo *k-means*. El algoritmo *k-prototypes* se muestra en el algoritmo 6.

En el algoritmo *k-prototypes* se utilizan tres procedimientos principales que describen su metodología: selección inicial de prototipos, asignación de datos a los clústers y re-localización.

- El primer proceso simplemente selecciona aleatoriamente k objetos como prototipos

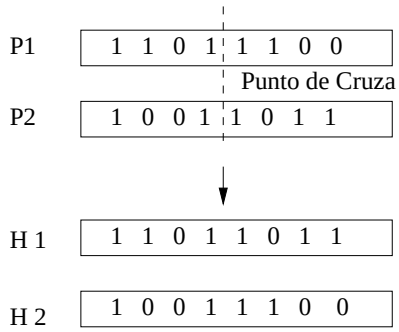

Figura 4.2.: Operación de cruza de un modelo evolutivo.

iniciales para clústers.

• El segundo proceso comienza asignando cada objeto restante a un clúster, actualizando el clúster prototipo en cada asignación. Se manejan dos tipos de arreglo para almacenar las partes numéricas y categóricas de los clústers prototipos. La función de costo se muestra en la ecuación 4.4. Consiste en una función de proximidad para atributos numéricos representada por medio de la evaluación de la distancia euclidiana y una función similitud para atributos categóricos es el número de mapeos existentes entre objetos y clústers prototipo. La ecuación 4.4 representa la suma total del costo de los atributos numéricos (${E_l}^r$) y el costo de los atributos categóricos (${E_l}^c$).

$$E_l = {E_l}^r + \gamma {E_l}^c$$
$$= \sum_{j=1}^{m_r} \left({x_{ij}}^r - {q_{lj}}^2\right)^2 + \gamma \sum_{j=1}^{m_c} \sigma({x_{ij}}^c, {q_{lj}}^c) \tag{4.4}$$

donde $\sigma(p, q) = 0$ para $p = q$ y $\sigma(p, q) = 1$ para $p \neq q$. Las variables ${x_{ij}}^r$ y ${q_{ij}}^r$ representan los valores de los atributos numéricos y ${x_{ij}}^r$ y ${q_{ij}}^r$ representan los valores de los atributos categóricos para el objeto i y el clúster prototipo l. Las variables m_r y m_c son el número total de atributos numérico y categórico. γ es una constante de control para atributos de tipo categóricos, su selección depende de la distribución de los atributos numéricos.

• El proceso de *re-localización* encuentra el clúster cuyo prototipo es el objeto más cercano en el proceso de actualización. Este proceso es similar al proceso de asignación inicial excepto que después de asignar un objeto, los prototipos de los clústers anteriores y actuales se deben actualizar.

Algoritmo 6: Algoritmo *k-prototypes*

1 **Entrada:** X // conjunto de datos
2 **Entrada:** γ // peso en datos categóricos
3 **Salida:** k // clústers
4 **begin**
5 Selecciona k prototipos iniciales del conjunto de datos X, uno para cada clúster.
6 Asigna cada objeto en X al clúster cuya evaluación de similitud sea la más cercana.
7 Evalúa la función de costo $E_l = {E_l}^r + \gamma {E_l}^c$ como la suma de medidas de similitud de los atributos numéricos y categóricos.
8 Al asignar todos los datos a los clústers, se re-evalúa la similitud de los datos respecto al dato prototipo del clúster. Si el dato tiene un dato prototipo más cercano respecto al que fue asignado previamente, entonces el dato es re-localizado en el otro clúster y se actualizan ambos clústers.
9 Se repite el punto 7 hasta que los datos no cambien de localización durante el ciclo completo de clustering.
10 **end**

El costo computacional del algoritmo es $O((t+1)kn)$, donde n es el número de objetos, k es el número de clústers y t es el número de iteraciones en el proceso de *re-localización*. Usualmente $k << n$ y t no excede a 100. Por lo tanto, este algoritmo se considera adecuado para trabajar con datasets grandes.

Capítulo 5.

Evaluación de Resultados

En este capítulo se realizó un estudio para evaluar el comportamiento del algoritmo propuesto *ACEM*. La ejecución de nuestro algoritmo se realizó tomando datasets reales obtenidos del UCI Machine Learning Repository [38] y se compararon resultados con respecto a otros algoritmos de clustering ya propuestos. Los experimentos realizados para medir el desempeño de nuestro algoritmo se basan en los experimentos realizados en la mayoría de los artículos de clustering de los doctores Zengyou He, Xiaofei Xu y Shengchun Deng, todos ellos del Instituto Harbin de China. Estos artículos son parte importante del estado del arte de minería de datos y clustering (ver referencias [21], [22], [23] y [47]).

5.1. Medida de exactitud del clúster

La medida de exactitud de los clústers resultantes se evalúa de la siguiente manera. Suponiendo que el número final de clústers es k, la medida de exactitud r está dada por la ecuación 5.1.

$$r = \frac{\sum_{i=1}^{k} a_i}{n} \tag{5.1}$$

donde n es el número de instancias del dataset, a_i es el número de instancias que aparecen clasificadas correctamente en el clúster i y en su correspondiente clase, la cual es aquella que tenga el número máximo. En otras palabras, a_i es el número de instancias con las etiquetas de la clase que dominan en el clúster i. Por lo tanto, el error esta definido en la ecuación 5.2

$$e = 1 - r \tag{5.2}$$

En la literatura se observa que una medida de error en un rango $\leq 3{,}0$, representa un porcentaje aceptable de error.

5.2. Comparación de ACEM respecto al modelo inicial alternativo

El modelo inicial alternativo (descrito en el capítulo 4), fue el primer modelo propuesto para evaluar la pertenencia de los datos mixtos en el proceso de clustering, utilizando como medida geométrica a la *distancia euclidiana*. En las tablas 5.1 y 5.2 se muestran las comparaciones de las mediciones de error y el valor del parámetro θ del algoritmo propuesto *ACEM* utilizando *entropía* y el modelo inicial utilizando la *distancia euclidiana* para datasets de tipo categórico y mixto respectivamente [1]. La comparación se efectuó únicamente utilizando el valor de θ que minimiza la evaluación de error en el algoritmo *ACEM*.

Tabla 5.1.: Comparación de eficiencia entre el algoritmo *ACEM* (utilizando entropía) y el modelo inicial alternativo (utilizando la distancia euclidiana) en los datasets categóricos.

Dataset	**Modelo**	θ	**Error**
Breast Cancer	Distancia euclidiana	0.53	0.08053
	Entropía	0.6	0.07613
Votes	Distancia euclidiana	0.5	0.16055
	Entropía	0.5	0.15632
Zoo	Distancia euclidiana	0.5	0.13861
	Entropía	0.5	0.12871
Soybean	Distancia euclidiana	0.7	0.0
	Entropía	0.7	0.0

Se puede observar que la diferencia de error entre el algoritmo *ACEM* y el modelo inicial alternativo es muy poca e incluso, en algunos casos, es la misma; sin embargo, en la mayoría de los datasets utilizados se muestra una evaluación de error menor para el modelo que utiliza la entropía como medida para evaluar la pertenencia de los datos

[1] Una descripción de las características de todos los datasets utilizados como caso de estudio se encuentra en el apéndice C

Tabla 5.2.: Comparación de eficiencia entre el algoritmo *ACEM* (utilizando entropía) y el modelo inicial alternativo (utilizando la distancia euclidiana) en los datasets mixtos.

Dataset	**Modelo**	θ	**Error**
Bands	Distancia euclidiana	0.5	0.35464
	Entropía	0.5	0.34227
Cleve	Distancia euclidiana	0.5	0.13531
	Entropía	0.5	0.11551
Flag	Distancia euclidiana	0.5	0.51546
	Entropía	0.5	0.45876
Post-operative	Distancia euclidiana	0.5	0.28736
	Entropía	0.5	0.28736
Bridges	Distancia euclidiana	0.7	0.50476
	Entropía	0.7	0.47619

a los clústers. Por lo tanto, el algoritmo *ACEM* utilizando nociones de entropía, es el modelo más conveniente para el trabajo de tesis.

5.3. Comportamiento de variabilidad

Hemos realizado una evaluación del comportamiento de los algoritmos *ACEM* y *K-prototypes* en función del orden en los datos de entrada. El objetivo es verificar que los resultados obtenidos en nuestro algoritmo de clustering son independientes del orden en que se presenten los parámetros de entrada.

Se evaluaron 10 casos de estudio, midiendo el error de clustering en el dataset con datos de tipo mixto *Cleveland Heart Disease* y modificando tanto el orden de las instancias como el orden en los atributos de entrada.

5.3.1. Variabilidad en las instancias

En la figura 5.1 y la tabla 5.3 se muestra el comportamiento de variabilidad de los dos algoritmos comparados (*ACEM* y *K-prototypes*) en función del error de clustering obtenido al modificar únicamente el orden en las instancias del dataset de entrada.

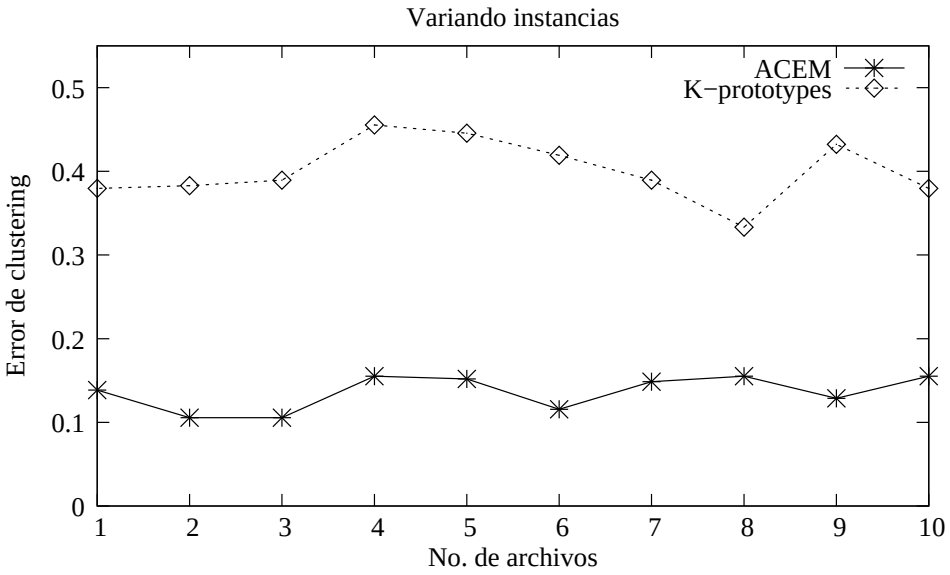

Figura 5.1.: Comportamiento de variabilidad modificando el orden en las instancias de entrada.

De aquí, se puede observar que los resultados obtenidos tienen un error de clustering más o menos constante, con una desviación estándar de 0.02063 para el algoritmo *ACEM* y 0.0385 para el algoritmo *K-prototypes*. El algoritmo *ACEM* tiene una menor desviación estándar en los datos y por lo tanto, no tiene una dependencia directa con el orden de las instancias del dataset.

5.3.2. Variabilidad en los atributos

En la figura 5.2 y la tabla 5.4 se muestra el comportamiento de variabilidad de los dos algoritmos de comparación (*ACEM* y *K-prototypes*) en función del error de clustering obtenido al modificar únicamente el orden en los atributos del dataset de entrada.

Al igual que en la sección anterior, se observan resultados de evaluación de error que varían en un rango muy pequeño. El algoritmo *ACEM* tiene un mejor comportamiento

Tabla 5.3.: Error promedio de los casos de estudio modificando el orden en las instancias de entrada.

Archivo	**Error- ACEM**	**Error- K-prototypes**
1	0.13861	0.37954
2	0.10561	0.38284
3	0.10561	0.38944
4	0.15512	0.45545
5	0.15182	0.44554
6	0.11551	0.41914
7	0.14851	0.38944
8	0.15512	0.33333
9	0.12871	0.43234
10	0.15512	0.37954
promedio	0.13597	0.40066
desviación estándar	0.02063	0.0385

Tabla 5.4.: Error promedio de los casos de estudio modificando el orden en los atributos de entrada.

Archivo	**Error- ACEM**	**Error- K-prototypes**
1	0.14521	0.41254
2	0.11551	0.42244
3	0.13201	0.4224
4	0.13861	0.40264
5	0.11551	0.40594
6	0.11551	0.38944
7	0.13861	0.38614
8	0.14521	0.39274
9	0.14521	0.42244
10	0.14521	0.45545
promedio	0.13597	0.411217
desviación estándar	0.01322	0.02076

tanto en el error promedio como en la desviación estándar obtenidas, respecto al algo-

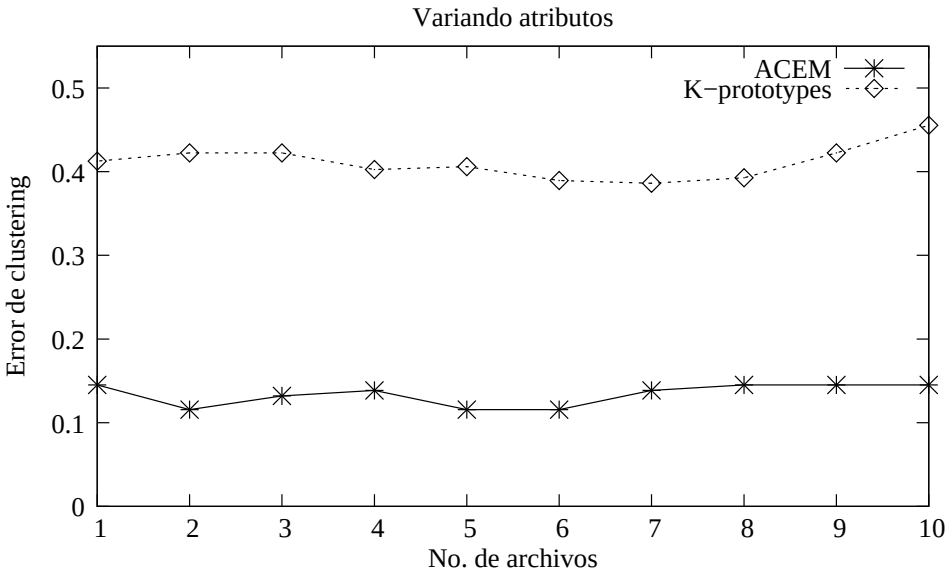

Figura 5.2.: Comportamiento de variabilidad modificando el orden en los atributos de entrada.

ritmo *K-prototypes*. Es decir, la desviación estándar obtenida al modificar el orden de los parámetros de entrada es tan pequeña en el algoritmo *ACEM*, que no se afectan significativamente los resultados de clustering del algoritmo.

5.4. Parámetros de entrada

Debido a que el algoritmo *ACEM* puede manipular datasets de tipo categórico y mixto, realizamos una evaluación de su comportamiento utilizando datasets con dichos tipos de dato.

Para la evaluación del algoritmo *ACEM*, también utilizamos algoritmos de clustering diseñados para atributos de tipo categórico (en la *fase 1*) y mixto (en la *fase 2*) como algoritmos de comparación. Los algoritmos utilizados son: *GAClust*[2] para datos categóricos y *K-prototypes*[3] para datos mixtos. Estos algoritmos se describieron previamente en el capítulo 4.

[2]Es un algoritmo genético y diferentes ejecuciones pueden variar los resultados de salida, por lo tanto en el proceso de evaluación se considera una sola ejecución de éste algoritmo.

[3]El peso del parámetro gamma puede ser calculado internamente por el algoritmo si no se define por el usuario.

Para realizar una comparación justa de *ACEM*[4] con los algoritmos de clustering utilizados para la comparación, se introdujeron los mismos parámetros de entrada en cada una de las pruebas, variando únicamente el número de clústers. La tabla 5.5 muestra los parámetros de entrada utilizados en cada uno de los algoritmos de clustering de comparación.

Tabla 5.5.: Parámetros de entrada de los algoritmos de comparación.

Algoritmo	**Parámetros Fijos**	**Parámetros Variables**
ACEM	Dataset [S] Theta [θ]=0.5	Número de clústers [k]
GAClust	Dataset [S] Tamaño de la población =50 Demás parámetros en su valor por omisión.	Número de clústers [k]
K-protopytes	Dataset [S] Peso gamma [γ] (opcional)	Número de clústers [k]

5.5. Experimentos Fase 1

En la *fase 1* del algoritmo *ACEM* (destinada a clasificar únicamente datos categóricos), utilizamos datasets con datos categóricos citados en la mayoría de los artículos de clustering para datos con atributos categóricos (ver [17], [47], [13], [28]), los cuales son:

1. **Breast Cancer**: Este dataset tiene 699 instancias con 9 atributos categóricos. Las instancias son clasificadas en 2 diagnósticos etiquetados como: "benign" (tumor benigno) y "malignant" (tumor maligno). Este dataset contiene un total de 458

[4]Se asignaron valores de $\theta = 0{,}5$ para todos los experimentos, aunque en la práctica se observo que para algunos datasets se puede mejorar el valor de θ.

benign (65.5 %) y 241 malignant (34.5 %). Todos los atributos tienen un dominio entre 1 y 10. Para las pruebas se mapearon los números de [1 - 10] a las letras [A -J]. Además se eliminaron las instancias con valores faltantes teniendo así un total de 683 registros con 444 benign y 239 malignant.

2. **Congressional Votes**: Este dataset tiene 435 instancias con 16 atributos categóricos. Las instancias son clasificadas en 2 clases etiquetadas como: "republicans" y "democrats". Todos los atributos son booleanos con valores de Yes (y) y No (n), conteniendo un total de 168 republicanos y 267 demócratas.

3. **Soybean** : Este dataset tiene 47 instancias con 35 atributos categóricos. Las instancias son clasificadas en 4 enfermedades etiquetadas como: "diaporthe stem canker" (D1), "charcoal rot" (D2), "rhizoctonia root rot" (D3) y "phytophthora" (D4). Todas excepto la clase D4, la cual 17 instancias, tienen 10 instancias.

4. **Zoo** : Este dataset tiene 101 instancias con 17 atributos categóricos. Las instancias son clasificadas en 7 grupos de animales etiquetados como: 1,2, ... , 7. El número de animales por cada grupo es: 1(41), 2(20), 3(5), 4(13), 5(4), 6(8) y 7(10).

Ahora, para comparar los resultados obtenidos de la *fase 1* del algoritmo *ACEM* para datos categóricos, se tomó como punto de comparación el algoritmo *GAClust* descrito en [7]. Se realizaron 8 experimentos de comparación evaluando el dataset representativo *Congressional Votes* y variando únicamente el número de clústers k para cada algoritmo, midiendo el error promedio obtenido para cada uno.

5.5.1. Resultados del dataset Breast Cancer

En la tabla 5.6 se muestran los resultados del clustering con 2 clústers de entrada en el dataset *Breast Cancer* para los algoritmos *ACEM* y *GAClust* y el error promedio obtenido de las 8 pruebas realizadas.

Los 2 algoritmos de comparación contienen clústers con un número mayor de datos *benignos* en un clúster y un número mayor de datos *malignos* en el otro. Sin embargo, en el clúster de datos *malignos* obtenido por el algoritmo *ACEM* se tienen 97.1 % datos *malignos* y en el clúster de datos *benignos* se tienen 85.57 % datos *benignos*. En el algoritmo *GAClust* se tienen 66.66 % datos *malignos* en el clúster de datos *malignos* y 91.64 % datos *benignos* en el clúster de datos *benignos*. Por lo tanto, el algoritmo *ACEM* mejora la calidad de los clústers en un porcentaje mayor.

ACEM			
Cluster	Benigno	Maligno	Instancias
1	5	165	170
2	439	74	513
total	444	239	683
Error ($k = 2$) = 0.11567			
Error Promedio = 0.11969			

GAClust			
Cluster	Benigno	Maligno	Instancias
1	340	31	371
2	104	208	312
total	444	239	683
Error ($k = 2$) = 0.197657			
Error Promedio = 0.13268			

Tabla 5.6.: Resultados de clustering con $k = 2$ para el dataset *Breast Cancer*.

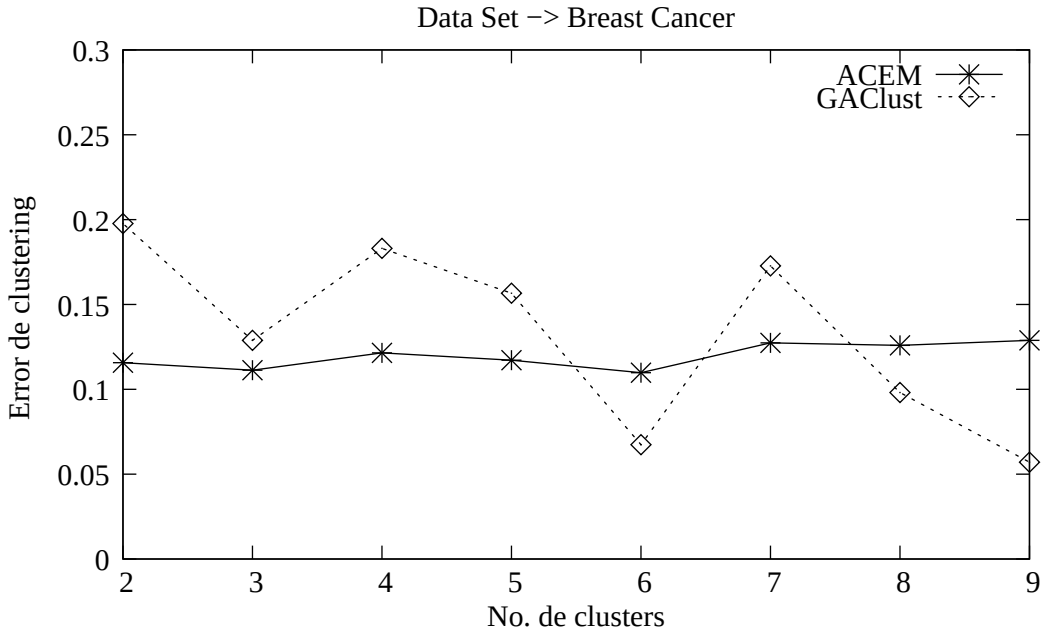

Figura 5.3.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Breast Cancer*.

En los resultados de la figura 5.3 se observa una evaluación de error menor en 5 de los 8 casos de prueba del algoritmo *ACEM* respecto al algoritmo *GAClust*. Por tanto, el algoritmo *ACEM* muestra un comportamiento más estable.

5.5.2. Resultados del dataset Congressional Votes

La evaluación del dataset *Congressional Votes* consiste en variar el número total de clústers de entrada (2-9) y comparar el error obtenido de los resultados de clustering para los algoritmos *ACEM* y *GAClust*.

ACEM			
Cluster	Republicanos	Demócratas	Instancias
1	3	201	204
2	165	66	231
total	168	267	435
Error ($k = 2$) = 0.15862			
Error Promedio = 0.11379			

GAClust			
Cluster	Republicanos	Demócratas	Instancias
1	156	74	230
2	12	193	205
total	168	267	435
Error ($k = 2$) = 0.197701			
Error Promedio = 0.13534			

Tabla 5.7.: Resultados de clustering con $k = 2$ para el dataset *Congressional Votes*.

En la tabla 5.7 se muestran los resultados con 2 clústers de entrada. Se observa que los 2 algoritmos de comparación contienen clústers con un número mayor de datos *republicanos* en un clúster y un número mayor de datos *demócratas* en el otro. Sin embargo, en el clúster de datos *republicanos* obtenido por el algoritmo *ACEM* se tienen 71.42 % de datos correctos en el clúster de *republicanos* y 98.52 % de datos correctos en el clúster de *demócratas*. En el algoritmo *GAClust* se tienen 94.14 % datos *demócratas* en el clúster de datos *demócratas* y 67.82 % datos *republicanos* en el clúster de datos *republicanos*. De aquí, se ve que el algoritmo *ACEM* mejora la calidad de los clústers y por lo tanto el error de clustering obtenido es menor.

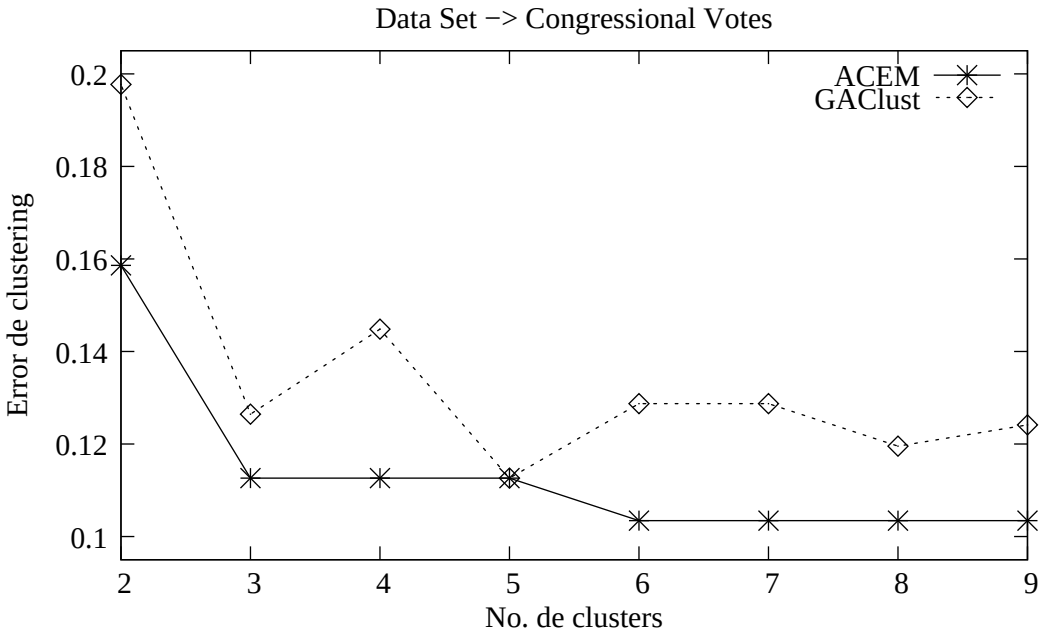

Figura 5.4.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Congressional Votes*.

La figura 5.4 muestra los resultados de todas las evaluaciones de error de los 2 algoritmos de comparación en el dataset *Congressional votes*. La evaluación de error del algoritmo *ACEM* es menor en 7 de 8 casos de prueba y en el último caso de prueba el error es igual respecto al obtenido por el algoritmo *GAClust*. El desempeño del algoritmo *ACEM* muestra una ventaja competitiva sobre el algoritmo de comparación.

5.5.3. Resultados del dataset Soybean

En la tabla 5.8 y la figura 5.5 se muestran los resultados del proceso de clustering con 4 clústers de entrada y las evaluaciones del error promedio variando el número de clústers de entrada en el dataset *Soybean* respectivamente.

En los resultados del proceso de clustering con 4 clústers de entrada, se observa que los algoritmos de comparación contienen clústers con un número mayor de datos representativos para cada una de las clases D1-D4. En esta evaluación, el algoritmo *ACEM* no mejora considerablemente la calidad de los clusters ya que para el clúster D1 se tiene un 100 % de datos de la clase D1, para el clúster D2 se tiene un 100 % de datos de la clase D2 pero para el clúster D3 se tiene un 46 % de datos de la clase D3 y

ACEM					
Cluster	D1	D2	D3	D4	Instancias
1	8	0	0	0	8
2	0	10	0	0	10
3	2	0	6	5	13
4	0	0	4	12	16
total	10	10	10	17	47
Error ($k = 4$) = 0.23404					
Error Promedio = 0.23404					

GAClust					
Cluster	D1	D2	D3	D4	Instancias
1	10	0	0	0	10
2	0	10	0	0	10
3	0	0	10	8	18
4	0	0	0	9	9
total	10	10	10	17	47
Error ($k = 4$) = 0.17021					
Error Promedio = 0.32181					

Tabla 5.8.: Resultados de clustering con $k = 4$ para el dataset *Soybean*.

para el clúster D4 se tiene un 75 % de datos de la clase D4. Mientras que en el algoritmo *GAClust* se tiene un 100 % de datos de la clase D1, un 100 % de datos de la clase D2, un 55 % de datos de la clase D3 y un 100 % de datos de la clase D4. Aunque en este caso no se mejora considerablemente la calidad de los clústers para el algoritmo *ACEM*, el promedio de error de todas las evaluaciones presenta un comportamiento constante en todos los casos, generando así un mejor desempeño en el proceso general de clustering.

5.5.4. Resultados del dataset Zoo

En la tabla 5.9 y la figura 5.6 se muestran los resultados del proceso de clustering con 7 clústers para el dataset *Zoo* y el error promedio obtenido de todas las evaluaciones.

En los resultados del proceso de clustering con 7 clústers de entrada, se observa un desempeño muy semejamente del algoritmo *ACEM* y el algoritmo *GAClust*.

En el algoritmo *ACEM* se tiene un 95 % de datos correctamente asignados a la clase 1, un 100 % de datos correctos en la clase 2, un 66.6 % de datos correctos en la clase 3,

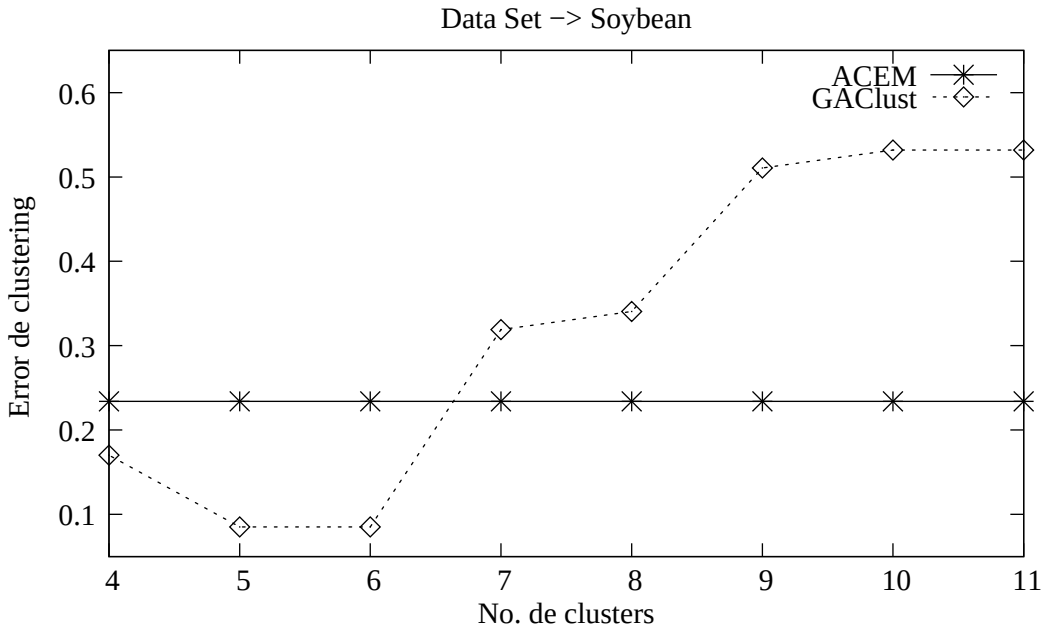

Figura 5.5.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Soybean*.

un 61.53 % de datos correctos en la clase 4, un 62.5 % de datos correctos en la clase 5, un 100 % de datos correctos en la clase 6 y un 100 % de datos correctos en la clase 7. En el algoritmo *GAClust* se tiene un 56 % de datos correctamente asignados a la clase 1, un 100 % de datos correctos en la clase 2, un 92.3 % de datos correctos en la clase 3, un 100 % de datos correctos en la clase 4, un 100 % de datos correctos en la clase 5, un 66.66 % de datos correctos en la clase 6 y un 30.76 % de datos correctos en la clase 7.

En este análisis se observa que para un número de clústers igual a 7 (clases), el error obtenido en *ACEM* es menor mejorando la calidad de los clústers, pero en la evaluación promedio, el algoritmo *GAClust* tiene una diferencia de 0.00125 de menor error. La distribución de datos en el proceso de clustering del algoritmo *ACEM* se puede mejorar considerablemente si se aumenta el valor en la constante θ, la cual para este caso de estudio se tomó como $\theta = 0{,}5$.

En la gráfica de evaluación de error del algoritmo *ACEM* en el dataset *Zoo* se obseva un comportamiento más estable en todas las evaluaciones.

ACEM								
Cluster	1	2	3	4	5	6	7	Instancias
1	23	0	1	0	0	0	0	24
2	0	20	0	0	0	0	0	20
3	0	0	1	0	1	8	2	12
4	0	0	2	8	3	0	0	13
5	2	0	1	5	0	0	0	8
6	16	0	0	0	0	0	0	16
7	0	0	0	0	0	0	8	8
Total	41	20	5	13	4	8	10	101
Error ($k = 7$) = 0.12871								
Error Promedio = 0.12005								

GAClust								
Cluster	1	2	3	4	5	6	7	Instancias
1	0	0	0	0	0	7	9	16
2	18	0	0	0	0	0	0	18
3	0	0	1	12	0	0	0	13
4	0	16	0	1	0	0	0	16
5	19	0	0	0	0	0	0	19
6	4	0	0	0	2	0	0	6
7	0	4	4	1	2	1	1	13
total	41	20	5	13	4	8	10	101
Error ($k = 7$) = 0.18811								
Error Promedio = 0.11880								

Tabla 5.9.: Resultados de clustering con $k = 7$ para el dataset *Zoo*.

5.6. Experimentos Fase 2

Para el análisis de los resultados de la *fase 2*, se ejecuta el algoritmo *ACEM* varias veces (al igual que en los experimentos realizados en la *fase 1*) cambiando el número de clústers deseados y midiendo los porcentajes de error obtenidos en cada uno de ellos a partir de la fórmula para medir la exactitud del clúster descrita en las ecuaciones 5.1 y 5.2.

Se realizaron las comparaciones del algoritmo *ACEM* con respecto al algoritmo de datos mixtos *k-prototype*, basándose principalmente en la evaluación de datasets con datos mixtos. Es necesario eliminar las instancias de los atributos numéricos vacías (en

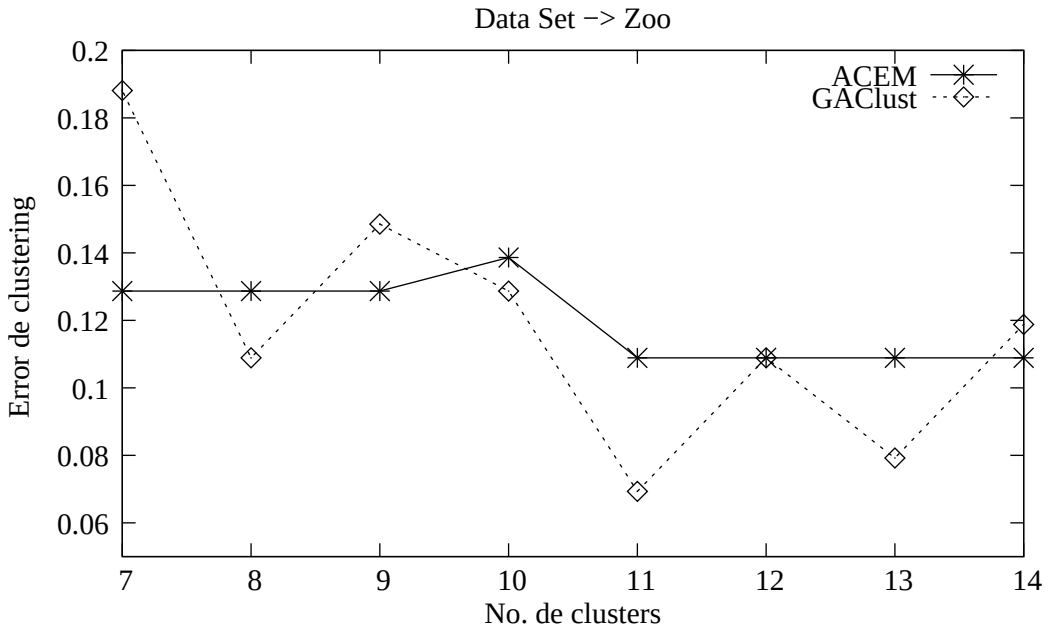

Figura 5.6.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Zoo*.

caso de existir) ya que el algoritmo *k-prototype* no maneja atributos vacíos en atributos numéricos. Los datasets de tipo mixto utilizados en esta fase de las pruebas son:

- **Bridges**: Tiene 108 instancias con 11 atributos de los cuales 4 son atributos numéricos y 7 son categóricos. No hay un atributo definido como clase, es un dominio de diseño en donde 5 propiedades necesitan ser predecidas basándose en 7 propiedades de especificación. Nosotros utilizamos el atributo "river" como clase (contiene 3 grupos). Se eliminaron 3 instancias que tienen valores faltantes en los atributos numéricos ya que el algoritmo de comparación *k-prototypes* requiere valores en los datos numéricos. Por lo tanto se utilizan 105 instancias clasificadas en 3 grupos etiquetados como A,M y O. De aquí, existen 49 instancias del grupo A, 41 instancias del grupo M y 15 instancias del grupo O.

- **Cilinder bands**: Tiene 512 instancias con 40 atributos de los cuales 20 son atributos numéricos y 20 son categóricos. Las instancias son clasificadas en 2 clases etiquetadas como: "No band" y "Band". De aquí se tienen 312 instancias no band y 200 band. Este dataset contiene 302 instancias con valores faltantes. Por lo tanto, para las pruebas hemos utilizado una versión un poco diferente tomando 485 instancias con 18 atributos numéricos y 18 categóricos. Se normalizaron los

valores faltantes, sustituyéndolos con los valores promedio de cada atributo. En total se tienen 293 instancias no band y 142 band.

- **Cleveland clinic heart disease**: Tiene 303 instancias de las cuales 6 son atributos numéricos y 8 son categóricos. Las instancias son clasificadas en 2 clases etiquetadas como: sanos "buff" y enfermos del corazón "sick". Este tiene 5 valores vacíos en atributos numéricos que son puestos a 0.

- **Flags**: Tiene 194 instancias, con 10 atributos numéricos y 19 categóricos. El dataset puede tener distintos atributos de clasificación (clase). Sin embargo, decidimos utilizar las instancias clasificadas en 4 clases etiquetadas como cuadrantes de la zona geográfica de la bandera correspondiente como son : 1=NE, 2=SE, 3=SW y 4=NW. De aquí se tienen 91 instancias que pertenecen a la zona NE, 29 a la zona SE, 16 a SW y 58 a la zona NW.

- **Post-operative**: Tiene 90 instancias, con 1 atributo numérico y 7 atributos categóricos. La distribución de las clases está dada como: I (2), S (24) y A (64). El dataset tiene 3 valores faltantes, por lo que utilizamos 87 instancias agrupados en 3 clases con 24 instancias en S, 1 en I y 62 en A.

5.6.1. Resultados del dataset Bridges

En el dataset *Bridges* se varió el número total de clústers de entrada (3-10) y se comparó el error obtenido de los resultados de clustering para los algoritmos *ACEM* y *k-prototypes* (ver figura 5.7).

En la tabla 5.10 se muestran los resultados con 3 clústers de entrada. La evaluación del error es igual para los 2 algoritmos de comparación, ya que la distribución de los datos fue muy similar en ambos casos. En el algoritmo *ACEM* los datos que predominan en los clústers tienen porcentajes de 43.28 %, 41.17 % y 76.19 % para cada una de las clases. En el algoritmo *k-prototypes* se tienen porcentajes de 58.06 %, 42.10 % y 50 % de los datos representativos para cada una de las 3 clases respectivamente. Por lo tanto, los 2 algoritmos presentan un comportamiento semejante al evaluar la calidad interna de los clústers.

En general, en la gráfica de evaluación de error se observa poca diferencia de error en la mayoría de las pruebas del dataset *Bridges*. Sin embargo, el algoritmo *ACEM* presenta un error promedio menor al algoritmo de comparación *k-prototypes*. El rango de error es $\geq 0{,}4$, lo cual implica que los resultados no son completamente satisfactorios para considerarlos competitivos. Sin embargo, estos resultados se pueden mejorar

ACEM				
Cluster	A	M	O	Instancias
1	27	29	11	67
2	6	7	4	17
3	16	5	0	21
total	49	41	15	105
Error ($k = 3$) = 0.50476				
Error Promedio = 0.44643				

k-prototype				
Cluster	A	M	O	Instancias
1	18	9	4	31
2	13	16	9	38
3	18	16	2	36
total	49	41	15	105
Error ($k = 3$) = 0.50476				
Error Promedio = 0.48989				

Tabla 5.10.: Resultados de clustering con $k = 3$ para el dataset *Bridges*.

considerablemente hasta obtener un error de cero si se aumenta el valor en la constante θ, la cual para este caso de estudio se tomó como $\theta = 0{,}5$.

5.6.2. Resultados del dataset Cylinder Bands

En la tabla 5.11 y la figura 5.8 se muestran los resultados del proceso de clustering con 2 clústers de entrada y las evaluaciones del error promedio variando el número de clústers de entrada en el dataset *Soybean*, respectivamente.

En los resultados de clustering con 2 clústers de entrada, se observa que el algoritmo *ACEM* contiene un número mayor de datos *band* en un clúster y un número mayor de datos *noband* en otro. Mientras que el algoritmo *K-prototypes* tiene la mayor cantidad de datos *noband* en los 2 clústers. Por lo tanto, el algoritmo *ACEM* mejora la calidad de los clústers y minimiza el error. Aunque la diferencia de error entre los algoritmos de clustering al utilizar 2 clústers de entrada es muy pequeña, el algoritmo *ACEM* tiene una evaluación de error promedio menor en los casos de prueba.

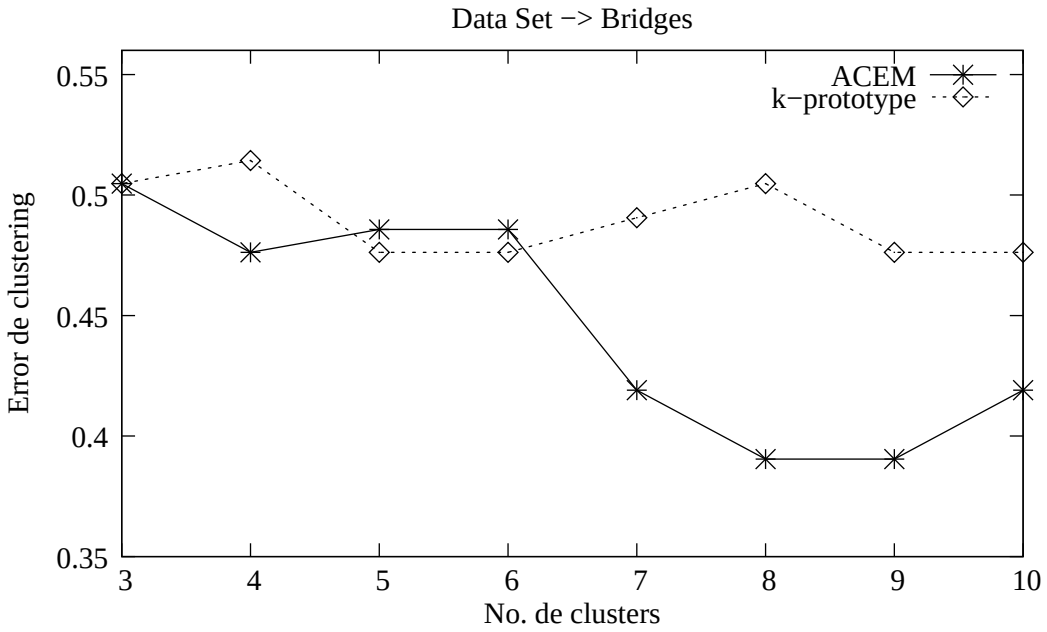

Figura 5.7.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Bridges*.

5.6.3. Resultados del dataset Cleve

Al utilizar el dataset de *Cleve* se obtuvo un muy buen desempeño ya que en los resultados, el porcentaje de error obtenido para todas las pruebas es mucho menor a los resultados del algoritmo *k-prototype* (ver figura 5.9 y tabla 5.12).

En la tabla 5.12 se muestran clústers que contienen un número mayor de datos *buff* en un clúster y un número mayor de datos *sick* en otro. En el clúster de datos *buff* obtenido por el algoritmo *ACEM* se tiene un 5.47 % de datos *sick* y un 94.53 % de datos *buff*; en el clúster de datos *sick* se tienen 82.80 % de *sick* y un 17.19 % de *buff*. En el algoritmo *k-prototypes* se tiene un 41.36 % de *sick* y 58.63 % de *buff* en el clúster de datos *buff* y un 43.38 % de *buff* y 56.62 % de *sick* en el clúster de datos *sick*. Por lo tanto, el algoritmo *ACEM* mejora la calidad de los clústers en un porcentaje mayor y los resultados muestran un desempeño estable y considerablemente bueno del algoritmo *ACEM* sobre el algoritmo de comparación en este dataset.

ACEM			
Cluster	Band	NoBand	Instancias
1	44	37	81
2	129	275	404
total	173	312	485
Error ($k = 2$) = 0.34227			
Error Promedio = 0.32913			

k-prototype			
Cluster	Band	NoBand	Instancias
1	57	108	165
2	116	204	320
total	173	312	485
Error ($k = 2$) = 0.3567			
Error Promedio = 0.35567			

Tabla 5.11.: Resultados de clustering con $k = 2$ para el dataset *Bands*.

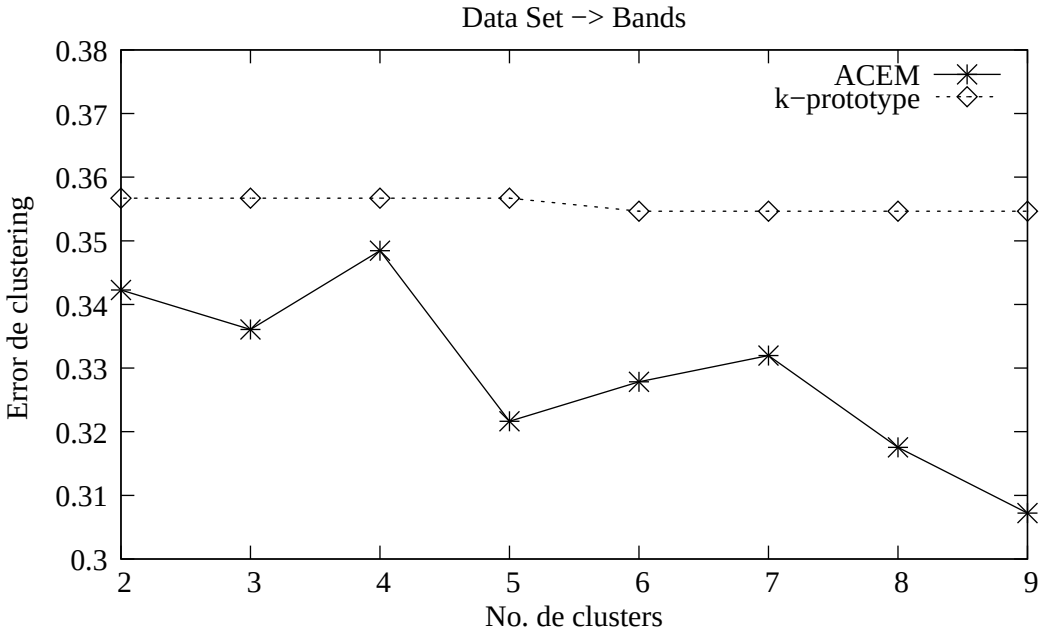

Figura 5.8.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Bands*.

ACEM			
Cluster	buff	sick	Instancias
1	138	8	146
2	27	130	157
total	165	138	303
Error ($k = 2$) = 0.11551			
Error Promedio = 0.136191			

k-prototype			
Cluster	buff	sick	Instancias
1	129	91	220
2	36	47	83
total	165	138	303
Error ($k = 2$) = 0.41914			
Error Promedio = 0.38325			

Tabla 5.12.: Resultados de clustering con $k = 2$ para el dataset *Cleve*.

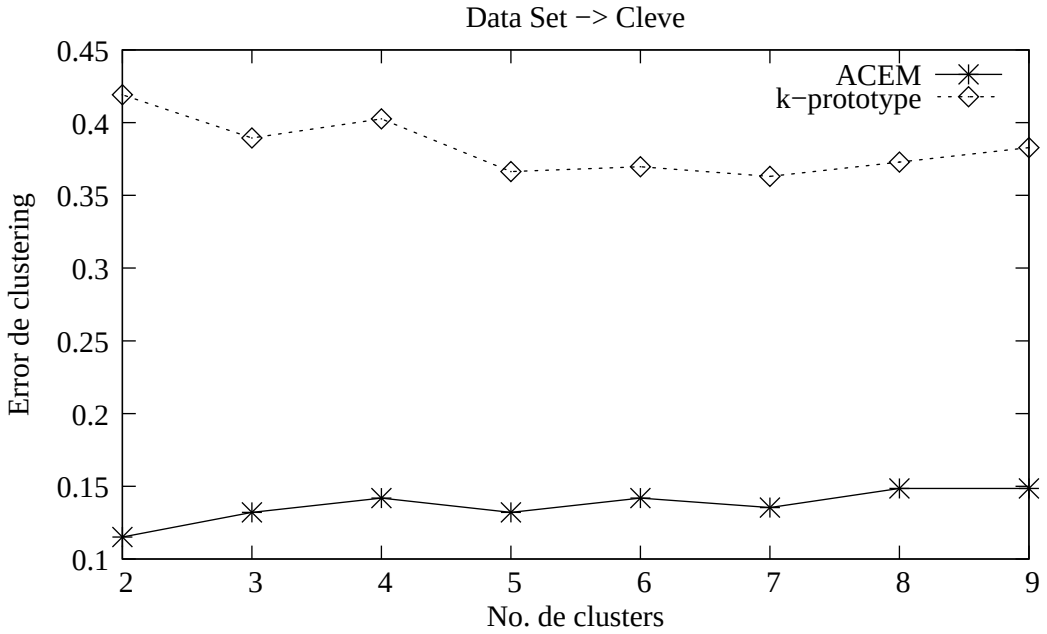

Figura 5.9.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Cleve*.

5.6.4. Resultados del dataset Flag

En la tabla 5.13 y la figura 5.10 se muestran los resultados del proceso de clustering con 4 clústers de entrada para el dataset *Flag* y el error promedio promedio obtenido de las evaluaciones de los algoritmos *ACEM* y *k-prototype*, variando el número de clústers en cada prueba.

ACEM					
Cluster	NE	SE	SW	NW	Instancias
1	21	12	0	13	46
2	22	2	4	5	33
3	4	6	4	18	32
4	44	9	8	22	83
total	91	29	16	58	194
Error ($k = 4$) = 0.45876					
Error Promedio = 0.45876					

k-prototype					
Cluster	NE	SE	SW	NW	Instancias
1	23	8	3	20	54
2	48	16	7	22	93
3	9	3	5	12	29
4	11	2	1	4	18
total	91	29	16	58	194
Error ($k = 4$) = 0.51546					
Error Promedio = 0.498065					

Tabla 5.13.: Resultados de clustering con $k = 4$ para el dataset *Flag*.

En los resultados del proceso de clustering con 4 clústers de entrada, se observa que los algoritmos de comparación contienen clústers con un número mayor de datos representativos para cada una de las clases (NE, SE, SW, NW). En el algoritmo *ACEM* se tiene un 45.65 % de datos representativos de la clase NE, 66.66 % de datos representativos de la clase SE, 56.25 % de datos representativos de la clase SW y 53.01 % de datos representativos de la clase NW. Mientras que en el algoritmo *k-prototype* se tiene un 42.59 % de datos de la clase NE, un 51.61 % de datos de la clase SE, un 41.37 % de datos de la clase SW y un 61.11 % de datos de la clase NW. De aquí, se muestra que el algoritmo *ACEM* presenta una mejor calidad en el clustering de los datos. Sin embargo, como el error evaluado en el dataset *Flag* está en un rango $\geq 0{,}4$, el porcentaje de error

obtenido no se considera competitivo.

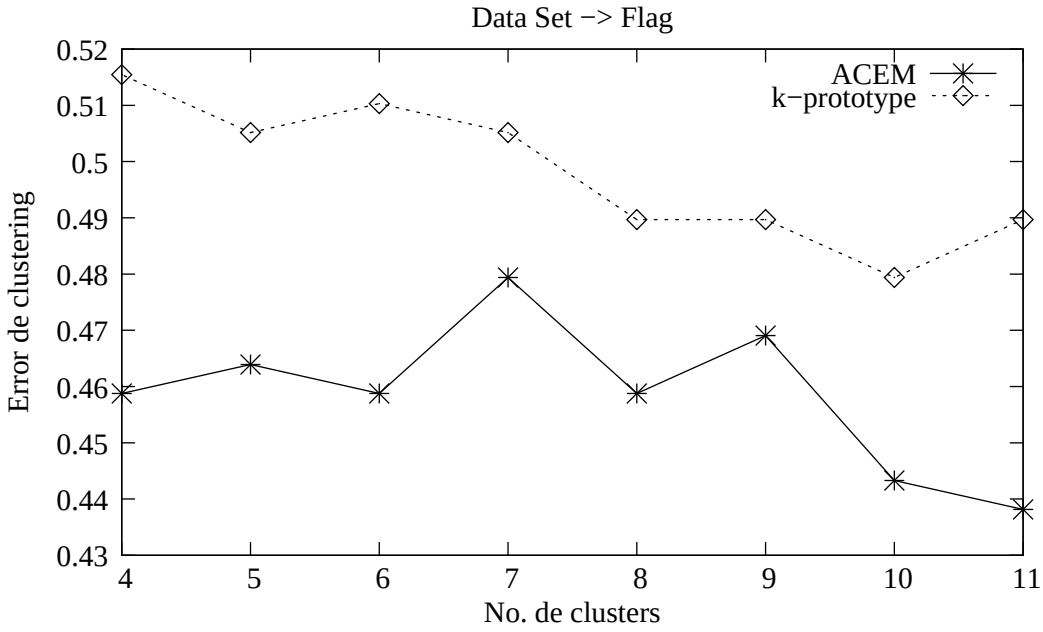

Figura 5.10.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Flag*.

5.6.5. Resultados del dataset Post-operative

En este dataset se observa una evaluación de error muy pequeña, entre los resultados de los algoritmos de comparación *ACEM* y *k-prototypes* (ver figura 5.11 y tabla 5.14).

En los resultados con 3 clústers de entrada se tiene una evaluación de error menor para el algoritmo *ACEM*. Los algoritmos de comparación generaron clústers con datos mayoritarios que predominan en cada una de las clases (A, M, O). En el algoritmo *ACEM* los datos que predominan en los clústers tienen porcentajes de 76 %, 70 % y 69 % para cada una de las clases. En el algoritmo *k-prototypes* se tienen porcentajes de 63 %, 70 % y 85 % de los datos representativos para cada una de las 3 clases respectivamente. Sin embargo, en el algoritmo *k-prototypes* los datos mayoritarios representativos pertenecen al mismo atributo (A), lo cual indica que la distribución de los clústers no es adecuada. Por lo tanto, aunque la diferencia de error es mínima para los resultados de los dos algoritmos de comparación, en general se observa un desempeño más eficiente en el algoritmo *ACEM*.

ACEM				
Cluster	A	M	O	Instancias
1	19	6	0	25
2	7	3	0	10
3	36	15	1	52
total	62	24	1	87
Error ($k = 3$) = 0.28736				
Error Promedio = 0.28592				

k-prototype				
Cluster	A	M	O	Instancias
1	12	7	0	19
2	38	15	1	54
3	12	2	0	14
total	62	24	1	87
Error ($k = 3$) = 0.29885				
Error Promedio = 0.29167				

Tabla 5.14.: Resultados de clustering con $k = 3$ para el dataset *Post-operative*.

5.7. Evaluación del parámetro θ

El algoritmo *ACEM*, tal y como se describió en el capítulo 4, utiliza el parámetro θ para restringir qué tan cerca deben de estar un par de puntos para ser considerados como vecinos. Por lo tanto, es importante la asignación de dicho parámetro, ya que ésto repercute en la evaluación de los resultados y el error promedio. En la literatura se establece que un valor de $\theta \geq 0{,}5$ presenta buenos resultados el clustering de los datos [17].

Para la evaluación de θ en cada uno de los datasets, se introdujeron valores para θ entre [0.3:0.8] y se graficaron los errores que definen la medida de exactitud de los clústers en los casos más representativos. Los valores de θ que nos interesan son aquellos en donde se maximice el valor de θ y se minimice el error obtenido. En algunos casos se observa una minimización de error en valores de $theta \leq 4$, sin embargo estos no pueden ser considerados como resultados competitivos ya que en este caso, los puntos vecinos podrían no ser muy semejantes entre sí afectando la calidad de los clústers.

En las siguientes subsecciones se muestran las gráficas de evaluación del mejor valor para θ en cada uno de los datasets utilizados en el proceso de pruebas. Se puede observar que el valor máximo de θ no es el mismo para todos los casos de estudio.

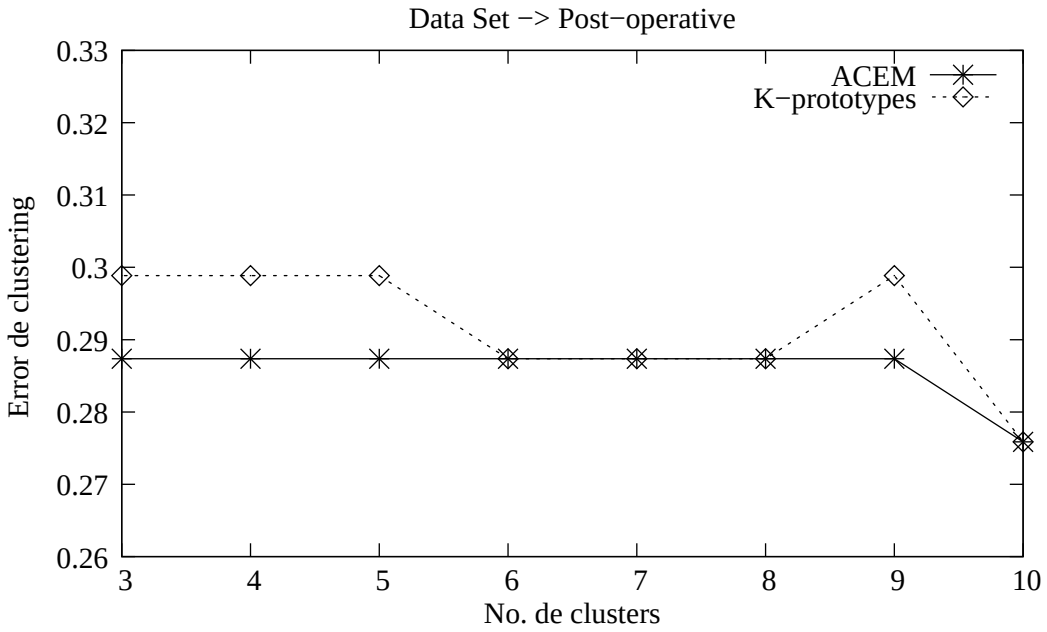

Figura 5.11.: Gráfica de la evaluación del error de clustering vs. número de clusters en el dataset *Post-operative*.

5.7.1. Valores de θ en datasets de tipo categórico

En las figuras 5.12, 5.13, 5.14 y 5.15 y en la tabla 5.15 se representan las evaluaciones de error al variar el valor de θ y los resultados que maximizan el valor de θ y minimizan el error obtenido del clustering en los datasets de tipo categórico respectivamente.

Tabla 5.15.: Datasets categóricos que maximizan θ y minimizan el error del clustering.

Dataset	**No. instancias**	**Clases**	θ	**Error**
Breast Cancer	683	2	0.6	0.07613
Votes	435	2	0.5	0.15632
Zoo	101	7	0.5	0.12871
Soybean	47	4	0.7	0.0

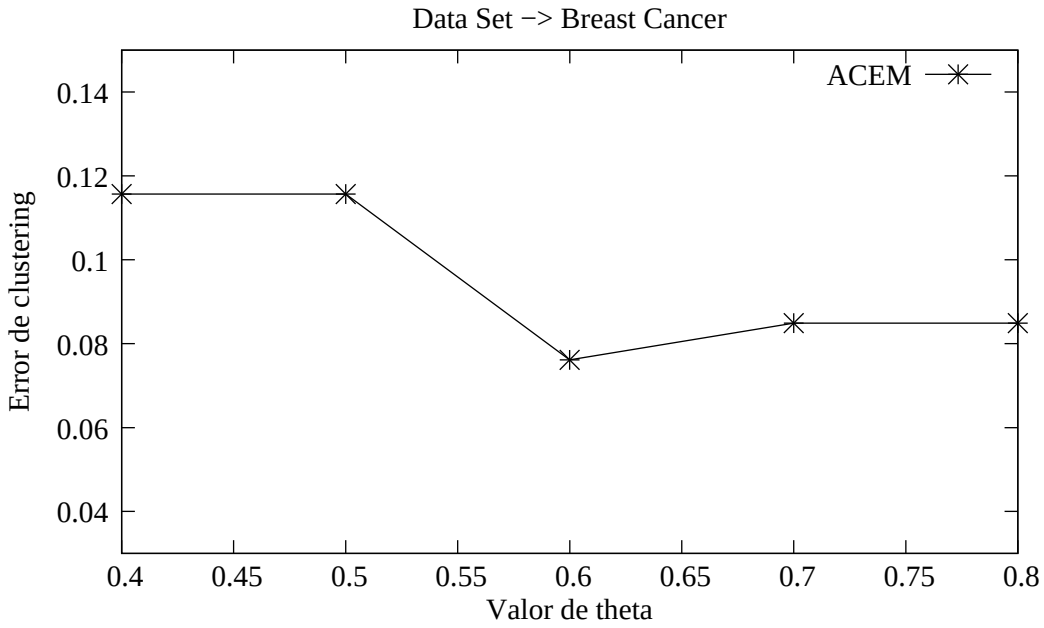

Figura 5.12.: Evaluación del mejor valor de θ en el dataset *Breast Cancer*.

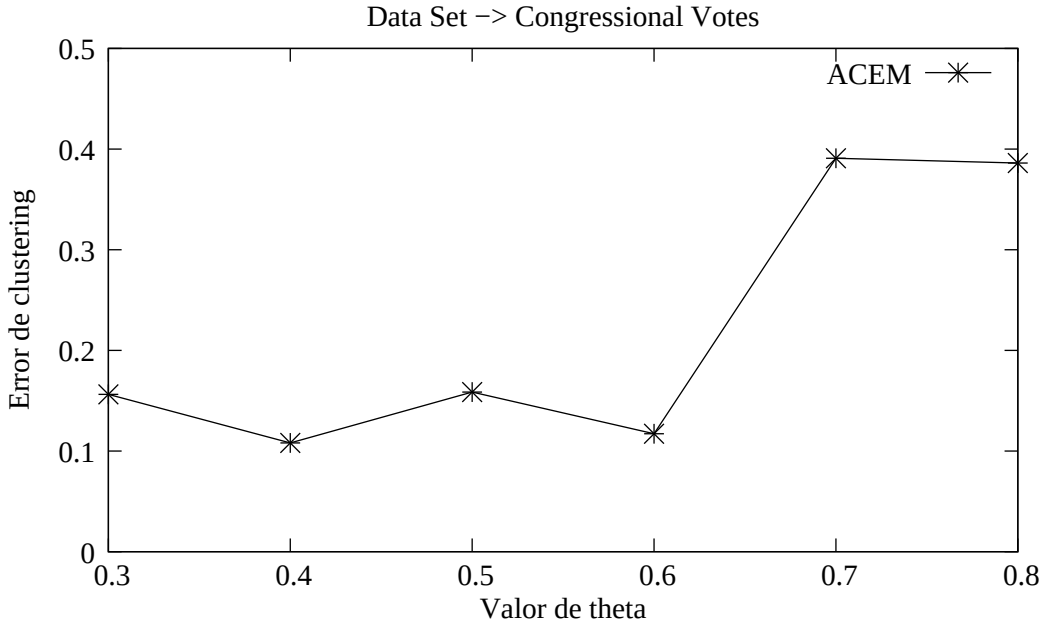

Figura 5.13.: Evaluación del mejor valor de θ en el dataset *Congressional Votes*.

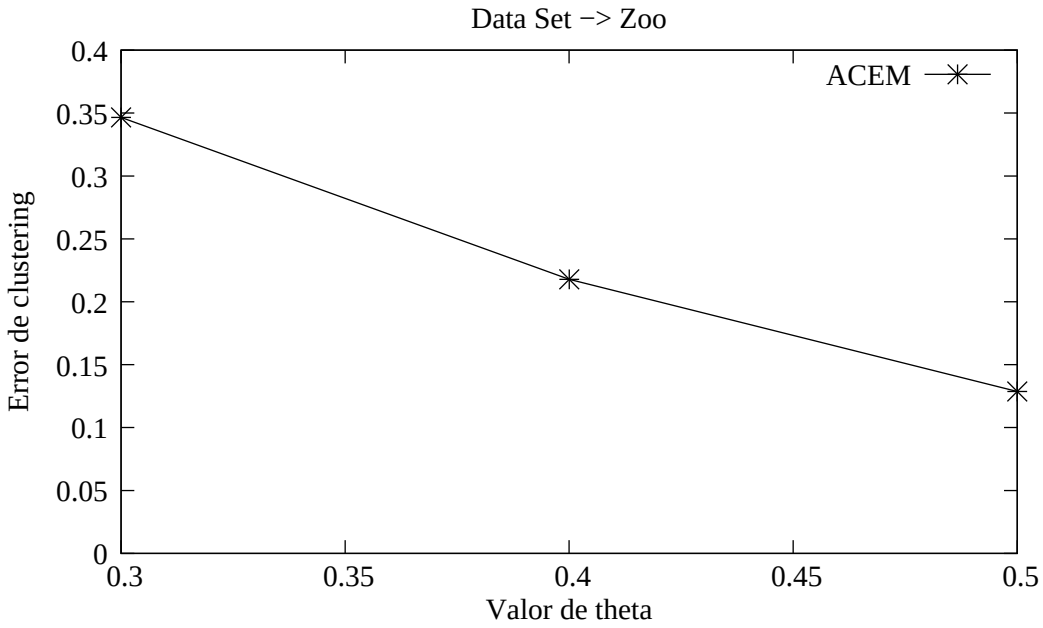

Figura 5.14.: Evaluación del mejor valor de θ en el dataset *Zoo*.

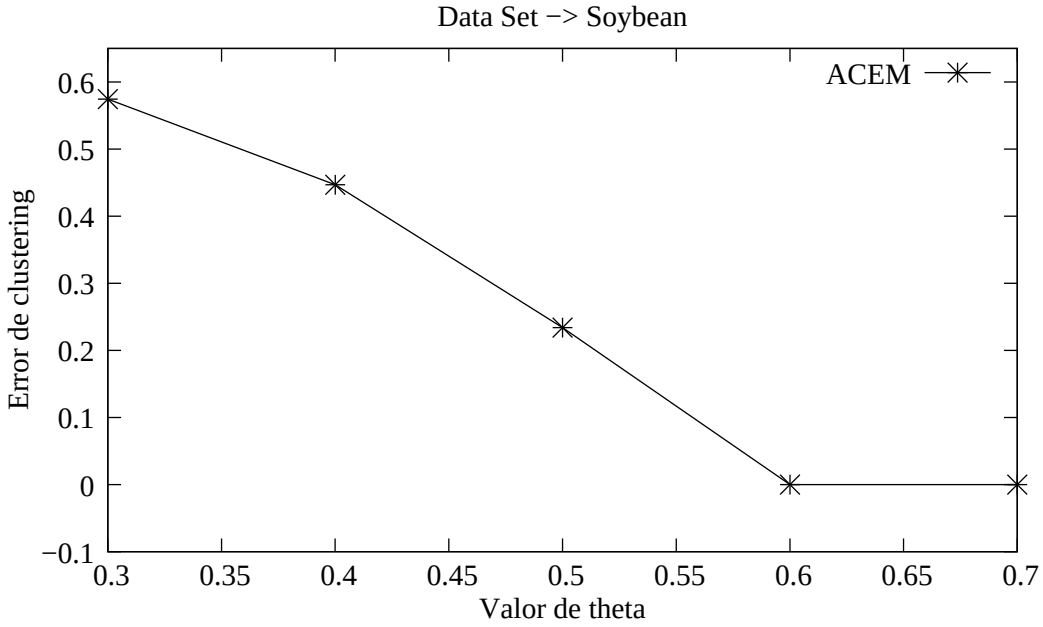

Figura 5.15.: Evaluación del mejor valor de θ en el dataset *Soybean*.

5.7.2. Valores de θ en datasets de tipo mixtos

En las figuras 5.16, 5.17, 5.18, 5.19, 5.20 y en la tabla 5.16 se representan las evaluaciones de error al variar el valor de θ y los resultados que maximizan el valor de θ y minimizan el error obtenido del clustering en los datasets de tipo mixto respectivamente.

Tabla 5.16.: Datasets mixtos que maximizan θ y minimizan el error del clustering.

Dataset	**No. instancias**	**Clases**	θ	**Error**
Bands	485	2	0.5	0.34227
Cleve	303	2	0.5	0.11551
Flag	194	4	0.5	0.45876
Post-operative	683	2	0.6	0.07613
Bridges	105	3	0.7	0.47619

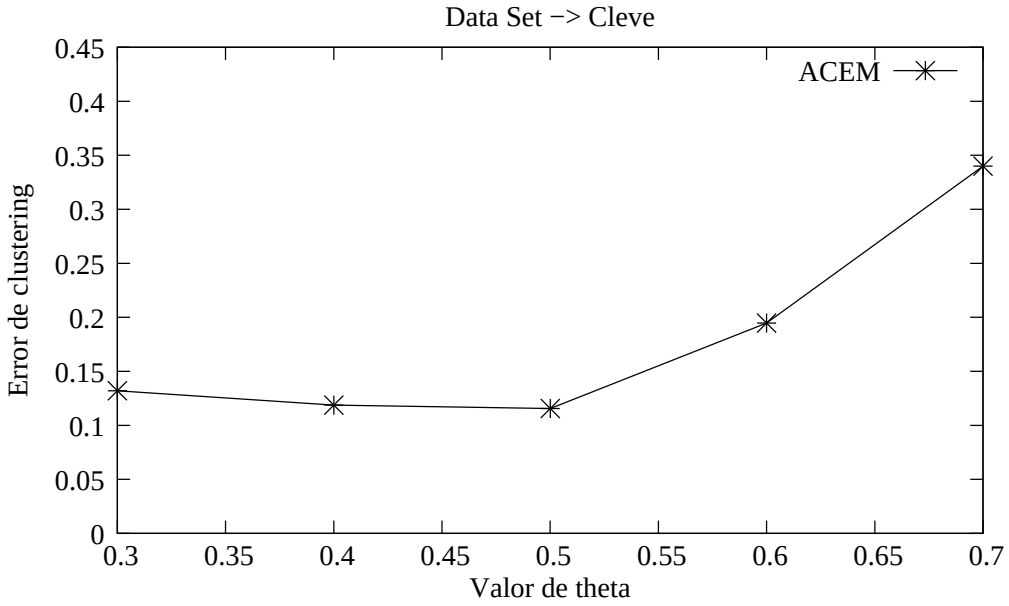

Figura 5.16.: Evaluación del mejor valor de θ en el dataset *Cleve*.

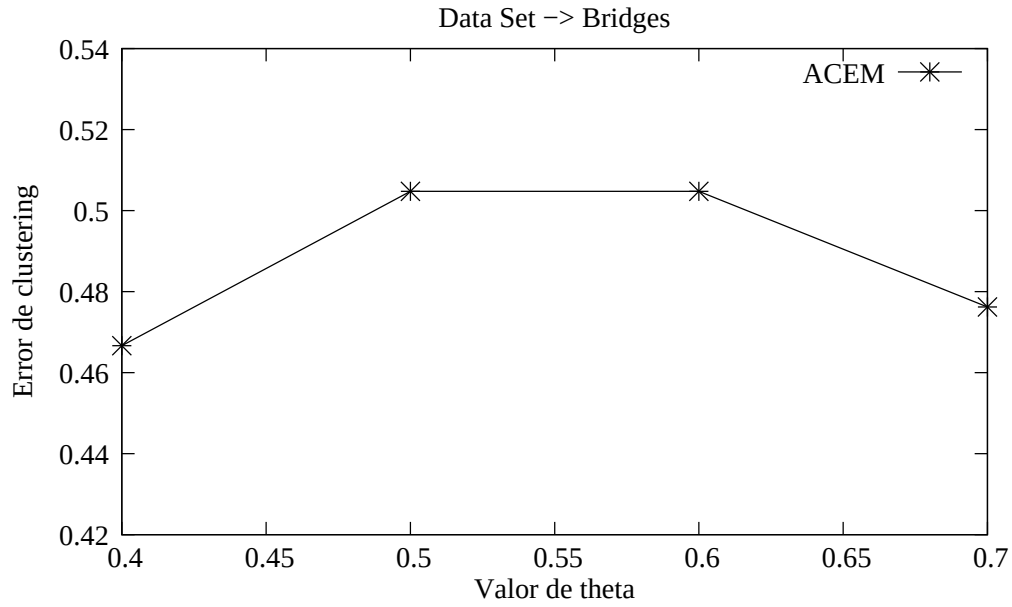

Figura 5.17.: Evaluación del mejor valor de θ en el dataset *Bridges*.

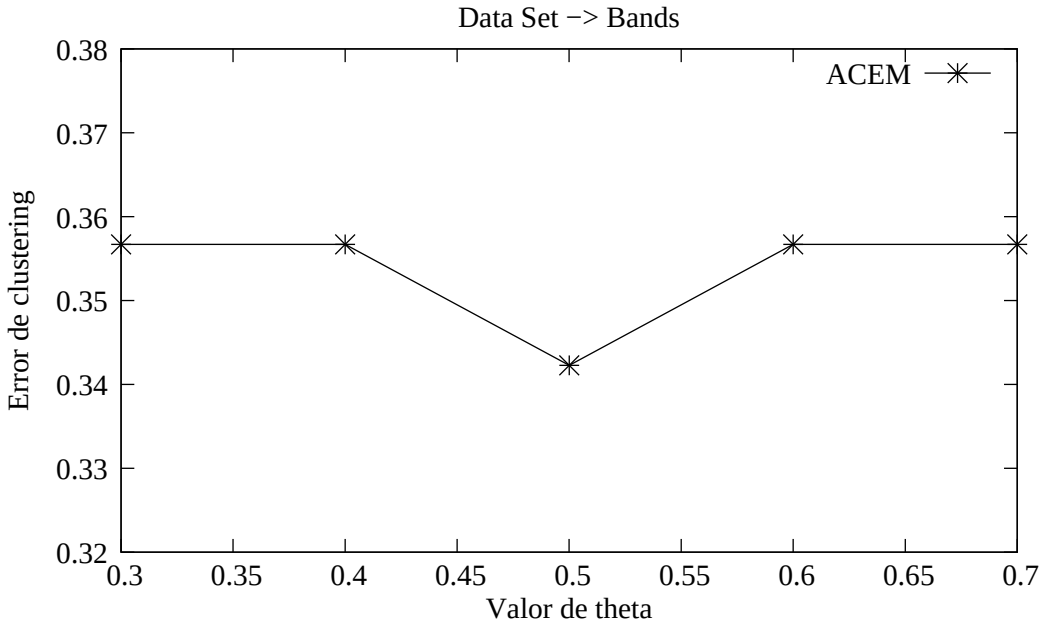

Figura 5.18.: Evaluación del mejor valor de θ en el dataset *Bands*.

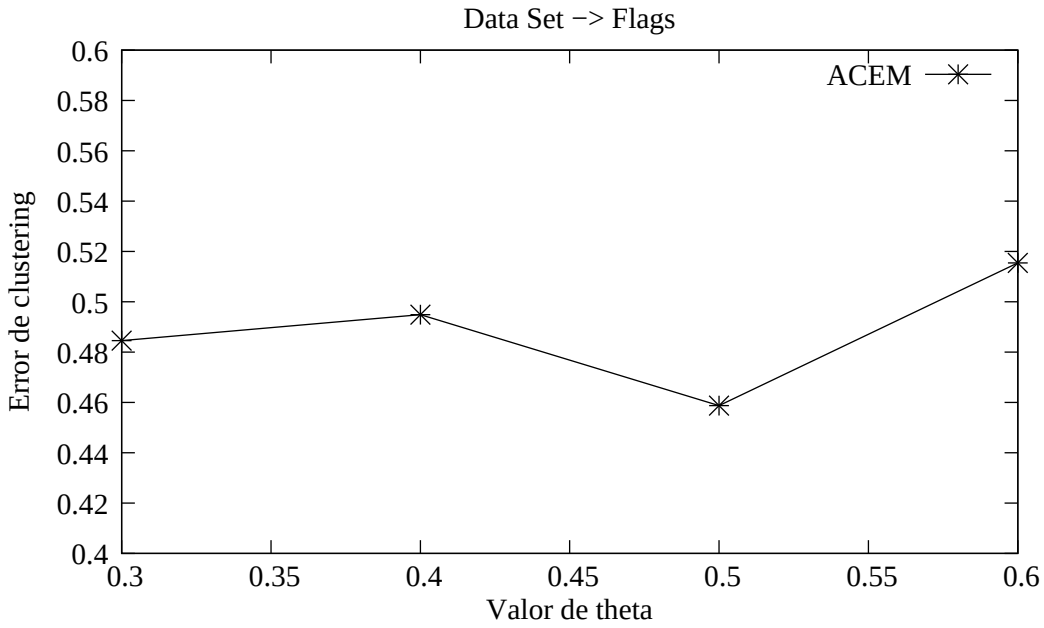

Figura 5.19.: Evaluación del mejor valor de θ en el dataset *Flags*.

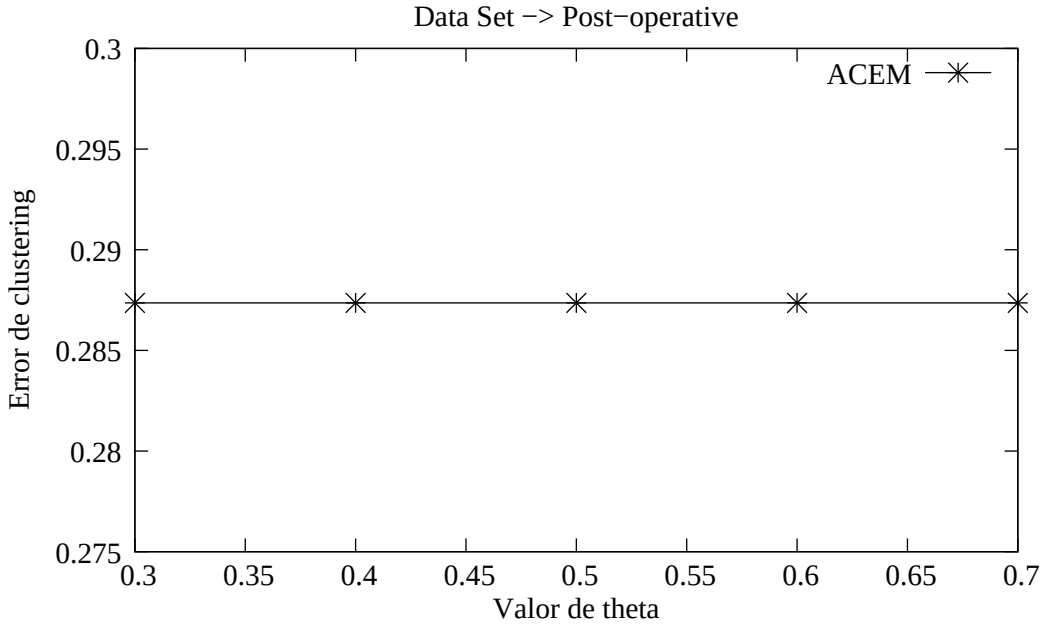

Figura 5.20.: Evaluación del mejor valor de θ en el dataset *Post-operative*.

5.8. Análisis de experimentos

Los resultados experimentales demuestran que el algoritmo propuesto *ACEM* presenta buen desempeño tanto para datasets de tipo categórico como para los datasets de tipo mixto evaluados en éste trabajo de tesis.

En los experimentos realizados en la *fase 1* del algoritmo, diseñada para datasets con atributos de tipo categórico, se tienen evaluaciones de error menores en la mayoría de los datasets analizados respecto al algoritmo de comparación *GAClust*. Al medir el error en el número exacto de clústers por dataset, mejoramos la calidad en 3 de los 4 datasets evaluados (*Breast cancer*, *Congressinal votes* y *Zoo*). En la medición del error promedio en 8 casos de prueba, igualmente mejoramos el error en 3 de los 4 datasets (*Breast cancer*, *Congressinal votes* y *Soybean*). En general, el algoritmo *ACEM* presenta un comportamiento muy estable en todas las mediciones del error. Por lo tanto, ésto implica que el algoritmo *ACEM* tiene buen desempeño en datasets de tipo categórico.

En los experimentos realizados en la *fase 2*, diseñada para datasets con atributos de tipo mixto, se pudo observar que se disminuye en gran medida las evaluaciones de error tanto con el número exacto de clústers como en la evaluación promedio de error en todos los datasets analizados. En dos de los cinco datasets analizados (*Bridges* y *Flag*), nuestro algoritmo obtiene un menor porcentaje de error que el algoritmo de comparación *GAClust*. Sin embargo, este porcentaje se encuentra entre un rango $\geq 0{,}4$, pudiéndose definir como resultados no satisfactorios para considerarlos competitivos.

En el algoritmo *ACEM*, se puede mejorar considerablemente la calidad en la distribución de datos del proceso de clustering, si se aumenta el valor de la constante θ en los datasets en los que se demostró poder maximizar su valor.

En general, lo anterior establece un criterio de efectividad de nuestro algoritmo *ACEM* respecto a los algoritmos de comparación *GAClust* y *k-prototypes* para los datasets analizados de tipo categórico y mixto respectivamente.

Capítulo 6.

Conclusiones y Trabajo Futuro

Con este trabajo de tesis se pretende proveer de un nuevo algoritmo de clustering llamado *ACEM* que tenga la capacidad de manipular datasets con tipos de datos mixtos. El algoritmo *ACEM* pre-clasifica los datos categóricos puros del dataset, extendiendo las características de un algoritmo de clustering para datos exclusivamente categóricos e introduciendo nociones de entropía para medir la heterogeneidad de los clústers y evaluar la pertenencia de los datos a un clúster específico. Un punto importante de nuestra contribución se centra en la implementación particular del modelo basándonos en una adaptación del algoritmo original *ROCK* sobre el cual se inspiró nuestro trabajo.

Los resultados experimentales demuestran que el algoritmo propuesto presenta buen desempeño tanto para datasets de tipo categórico como para los datasets de tipo mixto evaluados en este trabajo. En general, el algoritmo *ACEM* en la mayoría de las pruebas evaluadas, presenta un comportamiento estable en la medición del error. Por lo tanto, esto implica que los resultados en el proceso de clustering mantienen estable la variabilidad de los datos dentro de los clústers, maximizando su variabilidad fuera de ellos.

ACEM se puede visualizar como un algoritmo de clustering eficiente que puede manipular tipos de dato categórico y mixto obteniendo un desempeño considerablemente bueno.

En cuanto al trabajo futuro, se puede mejorar el mecanismo para el manejo de outliers en los datos, de forma que se encuentren nuevas soluciones para asignar o eliminar estos datos de los clústers de análisis.

Un problema que se presenta en el algoritmo *ACEM* es el incremento en el tiempo de ejecución, afectando la manipulación de datasets muy grandes. Sin embargo, existen algoritmos muy eficientes para seleccionar muestras aleatorias de datos que representen las características generales de los datasets y no sea necesario utilizar todo el datset.

Se puede agregar además, como trabajo futuro, investigar nuevas alternativas de evaluación de pertenencia al clúster tratando de maximizar la heterogeneidad de los datos en clústers diferentes y minimizar la variabilidad de los datos en un mismo clúster.

Apéndice A.

Métricas de Distancia

Tabla A.1.: Mediciones de distancia más comunes para variables numéricas.

Medición	*Expresión*	*Comentarios*
Distancia Minkowski	$D_{ij} = \left(\sum_{l=1}^{d} \mid x_{il} - x_{jl} \mid^{1/2}\right)^{n}$	Métrica. Variables con valores grandes y varianzas tienden a dominar otras variables.
Distancia Euclidiana	$D_{ij} = \left(\sum_{l=1}^{d} \mid x_{il} - x_{jl} \mid^{1/2}\right)^{2}$	Métrica más utilizada. Tienden a formar clústers hiperesféricos. Caso especial Minkowski n=2.
Distancia City-block	$D_{ij} = \sum_{l=1}^{d} \mid x_{il} - x_{jl} \mid$	Tienden a formar clústers hiperrectángulares. Caso especial de distancia Minkowski n=1.
Distancia Sup	$D_{ij} = \sum_{l=1}^{d} \mid x_{il} - x_{jl} \mid$	Caso especial de la métrica Minkowski cuando n $\rightarrow \infty$.
Distancia Mahalanobis	$D_{ij} = (x_i - x_j)^T S^{-1} (x_i - x_j)$ donde S es la matriz de covarianza	Invariante en cualquier transformación lineal. Forman clústers hiperelipsodiales. Con variables no correlacionadas, la métrica es equivalente a la distancia euclidiana.
Correlación de Pearson	$D_{ij} = (1 - r_{ij})/2$, donde $r = \frac{\sum_{l=1}^{d}(x_{il} - \overline{x}_i)(x_{jl} - \overline{x}_j)}{\sqrt{\sum_{l=1}^{d}(x_{il} - \overline{x}_i)^2 \sum_{l=1}^{d}(x_{jl} - \overline{x}_j)^2}}$	No es una métrica. Se deriva de una correlación de coeficientes. No es adecuada para detectar diferencias de magnitudes en 2 variables.
Distancia Punto de Simetría	$D_{ir} = min_{j=1,..,N, j \neq i} \frac{\left\|(x_i - x_r) + (x_j - x_r)\right\|}{\left\|(x_i - x_r)\right\| + \left\|(x_j - x_r)\right\|}$	No es una métrica. Evalúa la distancia entre x_j y un punto de referencia x_r. $D_i r$ se minimiza cuando existe simetría.
Similitud de Coseno	$S_{ij} = cos\alpha = \frac{{x_i}^T x_j}{\left\|x_i\right\| \left\|x_j\right\|}$	Independiente del vector de longitud. Invariante en rotación, pero no en trasformaciones lineales.

Apéndice B.

Algoritmos de clústering

B.1. Algoritmos de clústering jerárquicos

Tabla B.1.: Características principales de los algoritmos de clústering jerárquico.

Nombre	Tipo de dato	Complejidad	Geometría	Ruido	Parámetros de entrada	Resultado	Criterio de clústering
BIRCH	Numérico.	$O(n)$	Formas no complejas.	Sí	Radio de clústers, factor de ramas.	CF=(número de puntos en clúster N, suma lineal de puntos LS, suma cuadrada SS).	Un punto se asigna al clúster más cercano basándose en la distancia métrica seleccionada.
CURE	Numérico.	$O(n^2 \log n)$	Formas aleatorias.	Sí	No. de clústers, No. de puntos dispersos.	Asigna datos a clústers.	Los clústers con el par de puntos dispersos más cercanos son mezclados en cada paso.
ROCK	Categórico.	$O(n^2 + nm_m m_a + n^2 \log n)$, $O(n^2, nm_m m_a)$	Formas aleatorias.	Sí	No. de clústers	Asigna datos a clústers.	Emplea ligas entre los clústers para medir la similitud y proximidad entre el par de clústers a ser mezclados.

*Aquí m_m es el número máximo de vecinos por punto y m_a es el número promedio de vecinos por punto.

B.2. Algoritmos de clústering particional

Tabla B.2.: Características principales de los algoritmos de clústering particional.

Nombre	Tipo de dato	Complejidad	Geometría	Ruido	Parámetros de entrada	Resultado	Criterio de clústering
K-mean	Numérico	$O(n)$	Formas no complejas.	No	Número de clústers	Centros de clústers.	$E_k = \sum_{i=1}^{k} \sum_{k=1}^{n} d^2(x_k, v_i)$
K-mode	Categórico	$O(n)$	Formas no complejas.	No	Número de clústers	Modas de clústers.	$E = \sum_{i=1}^{k} \sum_{l=1}^{n} d(X_l, Q_i)$ $d(X_l, Q_i)$ distancia categórica entre X_l y Q_i.
PAM	Numérico	$O(k(n-k)^2)$	Formas no complejas.	No	Número de clústers	Medias de clústers.	$TC_{ih} = \sum_j C_{ijh}$ $min(TC_{ih})$
CLARA	Numérico	$O(k(40+k)^2 + k(n-k))$	Formas no complejas.	No	Número de clústers	Medias de clústers.	$TC_{ih} = \sum_j C_{ijh}$ $min(TC_{ih})$
CLARANS	Numérico	$O(kn^2)$	Formas no complejas.	No	Número de clústers, máximo número de vecinos.	Medias de clústers.	$TC_{ih} = \sum_j C_{ijh}$ $min(TC_{ih})$
FCM	Numérico	$O(n)$	Formas no complejas.	No	Número de clústers, creencia.	Centros de clústers.	$min_{U,v_1,..,v_k}(J_m(U,V))$
Fuzzy C-means							$J_m(U,V) = \sum_{i=1}^{k} \sum_{j=1}^{n} {U_{ik}}^m d^2(x_j, v_i)$

*Aquí n es el número de puntos en el datasets a consideración. C_{ih} = costo de reemplazar el centro i con h tan lejos como O_j permita.

B.3. Algoritmos de clústering basados en densidad

Tabla B.3.: Características principales de los algoritmos de clústering basado en densidad.

Nombre	Tipo de dato	Complejidad	Geometría	Ruido	Parámetros de entrada	Resultado	Criterio de clústering
DBSCAN	Numérico.	$O(n \log n)$	Formas aleatorias.	Sí	Radio de los clústers y un número mínimo de puntos.	Asigna datos a clústers.	Se mezclan los puntos que sean densamente alcanzables dentro del clúster.
DENCLUE	Numérico.	$O(n \log n)$	Formas aleatorias.	Sí	Radio de los clústers σ y un número mínimo de puntos.	Asigna datos a clústers.	${f_{Gauss}}^D(x^*) = \sum_{x_1 \epsilon cerca(x^*)} e^{\frac{d(x^*,x_1)^2}{2\sigma^2}}$ x^* densidad del punto x si $F_{Gauss} > \xi$.

*Aquí n es el número de puntos en el datasets a consideración.

B.4. Algoritmos de clústering basados en grid

Tabla B.4.: Características principales de los algoritmos de clústering basado en grid.

Nombre	Tipo de dato	Complejidad	Geometría	Ruido	Parámetros de entrada	Resultado	Criterio de clústering
Wave	Datos especiales	$O(n)$	Formas aleatorias.	Sí	Wavelets, número de celdas, número de aplicación.	Asigna datos a clústers.	Descompone el espacio de variables, aplica la transformación wavelet subdata -¿clústers.
STING	Datos especiales	$O(K)$ K es el número de celdas.	Formas aleatorias.	Sí	Número de objetos por celda	Asigna datos a clústers.	Divide el espacio en celdas rectángulares particionadas en otras más pequeñas.

*Aquí n es el número de puntos en el datasets a consideración.

Apéndice C.

Descripción de Datasets

Son datasets reales obtenidos del UCI Machine Learning Repository [38] para la evaluación del desempeño de los algoritmos de clustering.

C.1. Datasets con tipos de datos categórico

C.1.1. Dataset Breast Cancer

Este dataset tiene 699 instancias con 9 atributos categóricos. Las instancias son clasificadas en 2 diagnósticos etiquetados como: "benign" (tumor benigno) y "malignant" (tumor maligno). Conteniendo un total de 458 benign (65.5 %) y 241 malignant (34.5 %). Existen 16 instancias que contienen valores faltantes (missing), se denotan como "?". Todos los atributos tienen un dominio entre 1 y 10. Para las pruebas se mapearon los números de [1 - 10] a las letras [A -J]. Además se eliminaron las instancias con valores faltantes teniendo así un total de 683 registros con 444 benign y 239 malignant. En la tabla C.1 se describen cada uno de los atributos del dataset.

No.	Atributo	Dominio
1.	Clump Thickness	1 - 10
2.	Uniformity of Cell Size	1 - 10
3.	Uniformity of Cell Shape	1 - 10
4.	Marginal Adhesion	1 - 10
5.	Single Epithelial Cell Size	1 - 10
6.	Bare Nuclei	1 - 10
7.	Bland Chromatin	1 - 10
8.	Normal Nucleoli	1 - 10
9.	Mitoses	1 - 10
10.	Class:	(2 - benign, 4 - malignant)

Tabla C.1.: Lista de atributos del dataset *Breast Cancer*.

C.1.2. Dataset Congressional Votes

Este dataset tiene 435 instancias con 16 atributos categóricos. Las instancias son clasificadas en 2 clases etiquetadas como: "republicans" y "democrats". Todos los atributos son booleanos con valores de Yes (y) y No (n). Conteniendo un total de 168 republicanos y 267 demócratas. Existen 17 atributos que contienen valores faltantes denotados como "?". En la tabla C.2 se describen cada uno de los atributos del dataset.

No.	**Atributo**	**Dominio**
1.	Class Name	(democrat, republican)
2.	handicapped-infants	(y,n)
3.	water-project-cost-sharing	(y,n)
4.	adoption-of-the-budget-resolution	(y,n)
5.	physician-fee-freeze	(y,n)
6.	el-salvador-aid	(y,n)
7.	religious-groups-in-schools	(y,n)
8.	anti-satellite-test-ban	(y,n)
9.	aid-to-nicaraguan-contras	(y,n)
10.	mx-missile	(y,n)
11.	immigration	(y,n)
12.	synfuels-corporation-cutback	(y,n)
13.	education-spending	(y,n)
14.	superfund-right-to-sue	(y,n)
15.	crime	(y,n)
16.	duty-free-exports	(y,n)
17.	export-administration-act-south-africa	(y,n)

Tabla C.2.: Lista de atributos del dataset *Congressional Votes*.

C.1.3. Dataset Soybean

Este dataset tiene 47 instancias con 35 atributos categóricos. Las instancias son clasificadas en 4 enfermedades etiquetadas como: "diaporthe stem canker" (D1), "charcoal rot" (D2), "rhizoctonia root rot" (D3) y "phytophthora" (D4).

Todas excepto la clase D4, la cual 17 instancias, tienen 10 instancias. Para las pruebas se mapearon los números de los atributos a las letras correspondientes del abecedario [A..Z]. En la tabla C.3 se describen cada uno de los atributos del dataset.

No.	Atributo	Dominio
1.	date	(0-6) april, may, june,july, august, september, october.
2.	plant-stand	(0-1) normal, lt-normal
3.	precip	(0-2) lt-norm, norm, gt-norm
4.	temp	(0-2) lt-norm, norm, gt-norm
5.	hail	(0-1) yes,no
6.	crop-hist	(0-3) diff-lst-year, same-lst-yr, same-lst-two-yrs, same-lst-sev-yrs
7.	area-damaged	(0-3) scattered, low-areas, upper-areas, whole-field
8.	severity	(0-2) minor, pot-severe, severe
9.	seed-tmt	(0-2) none, fungicide, other
10.	germination	(0-2) 90-100 %, 80-89 %, lt-80 %
11.	plant-growth	(0-1) norm, abnorm
12.	leaves	(0-1) norm, abnorm
13.	leafspots-halo	(0-2) absent, yellow-halos, no-yellow-halos
14.	leafspots-marg	(0-2) w-s-marg, no-w-s-marg, dna
15.	leafspot-size	(0-2) lt-1/8, gt-1/8, dna
16.	leaf-shread	(0-1) absent, present
17.	leaf-malf	(0-1) absent, present
18.	leaf-mild	(0-2) absent,upper-surf,lower-surf
19.	stem	(0-1) norm,abnorm
20.	lodging	(0-1) yes,no
21.	stem-cankers	(0-3) absent, below-soil, above-soil, above-sec-nde
22.	canker-lesion	(0-3) dna, brown, dk-brown-blk, tan
23.	fruiting-bodies	(0-1) absent, present
24.	external decay	(0-2) absent, firm-and-dry, watery
25.	mycelium	(0-1) absent, present
26.	int-discolor	(0-2) none, brown, black
27.	sclerotia	(0-1) absent,present
28.	fruit-pods	(0-3) norm, diseased, few-present, dna
29.	fruit spots	(0-4) absent, colored, brown-w/blk-specks, distort, dna
30.	seed	(0-1) norm, abnorm
31.	mold-growth	(0-1)absent, present
32.	seed-discolor	(0-1) absent, present
33.	seed-size	(0-1) norm, lt-norm
34.	shriveling	(0-1) absent, present
35.	roots	(0-2) norm, rotted, galls-cysts

Tabla C.3.: Lista de atributos del dataset *Soybean*.

C.1.4. Dataset Zoo

Este dataset tiene 101 instancias con 17 atributos categóricos. Las instancias son clasificadas en 7 grupos de animales etiquetados como: 1,2, ... ,7. El número de animales por cada grupo es: 1(41), 2(20), 3(5), 4(13), 5(4), 6(8) y 7(10). Los atributos con valores enteros fueron mapeados a sus correspondientes valores categóricos de letras [A..Z]. En

las tablas C.4 y C.5 se describen las características de cada uno de los atributos del dataset.

No.	**Cantidad**	**Animales**
1	(41)	aardvark, antelope, bear, boar, buffalo, calf, cavy, cheetah, deer, dolphin, elephant, fruitbat, giraffe, girl, goat, gorilla, hamster, hare, lion, leopard, lynx, mink, mole, mongoose, opossum, oryx, platypus, polecat, pony, porpoise, puma, pussycat, raccoon, reindeer, seal, sealion, squirrel, vampire, vole, wallaby,wolf.
2	(20)	chicken, crow, dove, duck, flamingo, gull, hawk, kiwi, lark, ostrich, parakeet, penguin, pheasant, rhea, skimmer, skua, sparrow, swan, vulture, wren.
3	(5)	pitviper, seasnake, slowworm, tortoise, tuatara.
4	(13)	bass, carp, catfish, chub, dogfish, haddock, herring, pike, piranha, seahorse, sole, stingray, tuna.
5	(4)	frog, frog, newt, toad
6	(8)	flea, gnat, honeybee, housefly, ladybird, moth, termite, wasp.
7	(10)	clam, crab, crayfish, lobster, octopus, scorpion, seawasp, slug, starfish, worm.

Tabla C.4.: Grupos de animales en cada clase del dataset *Zoo*.

No.	**Atributo**	**Dominio**
1.	animal name	valor único
2.	hair	(y,n)
3.	feathers	(y,n)
4.	eggs	(y,n)
5.	milk	(y,n)
6.	airborne	(y,n)
7.	aquatic	(y,n)
8.	predator	(y,n)
9.	toothed	(y,n)
10.	backbone	(y,n)
11.	breathes	(y,n)
12.	venomous	(y,n)
13.	fins	(y,n)
14.	legs	(0,2,4,5,6,8)
15.	tail	(y,n)
16.	domestic	(y,n)
17.	catsize	(y,n)
18.	class	(1-7)

Tabla C.5.: Lista de atributos del dataset *Zoo*.

C.2. Datasets con tipos de datos mixto

C.2.1. Dataset Cilinder bands

Tiene 512 instancias con 40 atributos de los cuales 20 son atributos numéricos y 20 son categóricos. Las instancias son clasificadas en 2 clases etiquetadas como: "No band" y "Band". De aquí se tienen 312 instancias no band y 200 band.

El dataset contiene 302 instancias con valores faltantes por lo tanto para las pruebas hemos utilizado una versión un poco diferente tomando 485 instancias con 18 atributos numéricos y 18 categóricos. Se normalizaron los valores faltantes, sustituyéndolos con los valores promedio de cada atributo. En total se tienen 293 instancias no band y 142 band. En la tabla C.6 se describen cada uno de los atributos del dataset.

C.2.2. Dataset Bridges

Tiene 108 instancias con 11 atributos de los cuales 4 son atributos numéricos y 7 son categóricos. No hay un atributo definido como clase, es un dominio de diseño en donde 5 propiedades necesitan ser predecidas basándose en 7 propiedades de especificación.

Nosotros utilizamos el atributo "River" como clase (contiene 3 grupos). Se eliminaron 3 instancias que tienen valores faltantes en los atributos numéricos ya que el algoritmo de comparación *k-prototypes* requiere valores en los datos numéricos. Por lo tanto se utilizan 105 instancias clasificadas en 3 grupos etiquetados como A,M y O. De aquí, existen 49 instancias del grupo A, 41 instancias del grupo M y 15 instancias del grupo O. En la tabla C.7 se describen cada uno de los atributos del dataset.

No.	Atributo	Tipo	Dominio
1.	timestamp	numérico	valor único
2.	cylinder number	categórico	–
3.	customer	categórico	–
4.	job number	categórico	–
5.	grain screened	categórico	yes, no
6.	ink color	categórico	key, type
7.	proof on ctd ink	categórico	yes, no
8.	blade mfg	categórico	benton, daetwyler, uddeholm
9.	cylinder division	categórico	gallatin, warsaw, mattoon
10.	paper type	categórico	uncoated, coated, super
11.	ink type	categórico	uncoated, coated, cover
12.	direct steam	categórico	yes, no
13.	solvent type	categórico	xylol, lactol, naptha, line
14.	type on cylinder	categórico	yes, no
15.	press type	categórico	wood hoe, motter, albert, motter
16.	press	categórico	821, 802, 813, 824, 815, 816, 827, 828
17.	unit number	categórico	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
18.	cylinder size	categórico	catalog, spiegel, tabloid
19.	paper mill location	categórico	north us, south us, canadian, scandanavian, mid european
20.	plating tank	categórico	1910, 1911
21.	proof cut	numérico	0-100
22.	viscosity	numérico	0-100
23.	caliper	numérico	0-1.0
24.	ink temperature	numérico	5-30
25.	humifity	numérico	5-120
26.	roughness	numérico	0-2
27.	blade pressure	numérico	10-75
28.	varnish pct	numérico	0-100
29.	press speed	numérico	0-4000
30.	ink pct	numérico	0-100
31.	solvent pct	numérico	0-100
32.	ESA Voltage	numérico	0-16
33.	ESA Amperage	numérico	0-10
34.	wax	numérico	0-4.0
35.	hardener	numérico	0-3.0
36.	roller durometer	numérico	15-120
37.	current density	numérico	20-50
38.	anode space ratio	numérico	70-130
39.	chrome content	numérico	80-120
40.	class	categórico	band, no band

Tabla C.6.: Lista de atributos del dataset *Cilinder bands*.

C.2.3. Dataset Cleveland clinic heart disease

Tiene 303 instancias con 14 atributos de los cuales 6 son atributos numéricos y 8 son categóricos. Las instancias son clasificadas en 2 clases etiquetadas como: sanos “buff” y enfermos del corazón “sick”. Este tiene 5 valores vacíos en atributos numéricos que son puestos a 0. En la tabla C.8 se describen cada uno de los atributos del dataset.

No.	Atributo	Tipo	Dominio
1.	RIVER - class	categórico	A, M, O
2.	LOCATION	numérico	(1 - 52)
3.	ERECTED	numérico	(1818-1986)
4.	PURPOSE	categórico	WALK, AQUEDUCT, RR, HIGHWAY
5.	LENGTH	numérico	804-4558
6.	LANES	numérico	(1-6)
7.	CLEAR-G	categórico	N, G
8.	T-OR-D	categórico	THROUGH, DECK
9.	MATERIAL	categórico	WOOD, IRON, STEEL
10.	SPAN	categórico	SHORT, MEDUIM, LONG
11.	REL-L	categórico	S, S-F, F
12.	TYPE	categórico	WOOD, SUSPEN, SIMPLE-T, ARCH, CANTILEV, CONT-T CANTILEV, CONT-T

Tabla C.7.: Lista de atributos del dataset *Bridges*.

No.	Atributo	Tipo	Dominio
1.	age	numérico	edad en años
2.	sex	categórico	male, female
3.	chest pain	categórico	typical angina, atypical angina non-anginal pain, asymptomatic
4.	resting blood pressure	numérico	presión en mm de Hg
5.	cholestoral	numérico	colesterol en mg/dl
6.	fasting blood sugar	categórico	(true, false)
7.	electrocardiographic	categórico	normal, ST-T wave abnormality , ventricular hypertrophy
8.	thalach	numérico	maximum heart rate
9.	exang	categórico	true, false
10.	oldpeak	numérico	ST depression
11.	slope	categórico	upsloping, flat, downsloping
12.	ca	numérico	(0-3)
13.	thal	categórico	normal, fixed defect, reversable defect
14.	num	categórico	buff, sick

Tabla C.8.: Lista de atributos del dataset *Cleve*.

C.2.4. Dataset Flags

Tiene 194 instancias, con 10 atributos numéricos y 19 categóricos. El dataset puede tener distintos atributos de clasificación (clase), sin embargo decidimos utilizar las instancias clasificadas en 4 clases etiquetadas como cuadrantes de la zona geográfica de la bandera correspondiente como son : 1=NE, 2=SE, 3=SW y 4=NW. De aquí se tienen 91 instancias que pertenecen a la zona NE, 29 a la zona SE, 16 a SW y 58 a la zona NW. En la tabla C.9 se describen cada uno de los atributos del dataset.

No.	Atributo	Tipo	Dominio
1.	name	categórico	Name of the country concerned
2.	landmass	categórico	NAmerica, SAmerica, Europe, Africa, Asia, Oceania
3.	zone -class	numérico	1=NE, 2=SE, 3=SW, 4=NW
4.	area	numérico	in thousands of square km
5.	population	numérico	in round millions
6.	language	categórico	1=English, 2=Spanish, 3=French, 4=German, 5=Slavic, 6=Other Indo-European, 7=Chinese, 8=Arabic, 9=Japanese/Turkish/Finnish/Magyar, 10=Others
7.	religion	categórico	0=Catholic, 1=Other Christian, 2=Muslim, 3=Buddhist, 4=Hindu, 5=Ethnic, 6=Marxist, 7=Others
8.	bars	numérico	Number of vertical bars in the flag
9.	stripes	numérico	Number of horizontal stripes in the flag
10.	colours	numérico	Number of different colours in the flag
11.	red	categórico	(s,n)
12.	green	categórico	(s,n)
13.	blue	categórico	(s,n)
14.	gold	categórico	(s,n)
15.	white	categórico	(s,n)
16.	black	categórico	(s,n)
17.	orange	categórico	(s,n)
18.	mainhue	categórico	predominant colour in the flag
19.	circles	numérico	Number of circles in the flag
20.	crosses	numérico	Number of (upright) crosses
21.	saltires	numérico	Number of diagonal crosses
22.	quarters	numérico	Number of quartered sections
23.	sunstars	numérico	Number of sun or star symbols
24.	crescent	categórico	(s,n)
25.	triangle	categórico	(s,n)
26.	icon	categórico	(s,n)
27.	animate	categórico	(s,n)
28.	text	categórico	(s,n)
29.	topleft	categórico	color
30.	botright	categórico	color

Tabla C.9.: Lista de atributos del dataset *Flags*.

C.2.5. Dataset Post-operative

Tiene 90 instancias, con 1 atributo numérico y 7 atributos categóricos. La distribución de las clases ésta dada como: I (2), S (24) y A (64). El dataset tiene 3 valores faltantes, por lo tanto utilizamos 87 instancias agrupados en 3 clases con 24 instancias en S, 1 en I y 62 en A. En la tabla C.10 se describen cada uno de los atributos del dataset.

No.	**Atributo**	**Tipo**	**Dominio**
1.	L-CORE (internal temperature)	categórico	high, mid, low
2.	L-SURF (surface temperature)	categórico	high, mid, low
3.	L-O2 (oxygen saturation)	categórico	excellent, good, fair, poor
4.	L-BP (blood pressure)	categórico	high, mid, low
5.	SURF-STBL (stability SURF)	categórico	stable, mod-stable, unstable
6.	CORE-STBL (stability CORE)	categórico	stable, mod-stable, unstable
7.	BP-STBL (stability blood pressure)	categórico	stable, mod-stable, unstable
8.	COMFORT	numérico	(0 - 20)
9.	ADM-DECS -class	categórico	I (patient sent to Intensive Care Unit), S (patient prepared to go home), A (patient sent to general hospital floor)

Tabla C.10.: Lista de atributos del dataset *Post-operative*.

Bibliografía

[1] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. pages 94–105, 1998.

[2] M.R Anderberg. *Cluster analysis for applications*. Academic press., 1973.

[3] M. Ankerst, M. Breuning, H.P. Kriegel, and J. Sander. Optics: Ordering points to identify clustering structure. In *Proceedings of the ACM SIGMOD Conference*, pages 49–60, Philadelphia, PA., 1999.

[4] J. C. Bezdek. *Pattern Recognition With Fuzzy Objetive Function Algorithms*. Plenum Press, New York, NY, 1981.

[5] G. Carpenter and S. Grossberg. Art3: Hierarchical search using chemical transmiters in self-orginizing pattern recognition architectures. *Neural Networks.*, 3:129–152, 1990.

[6] Tom Chiu, DongPing Fang, John Chen, Yao Wang, and Christopher Jeris. A robust and scalable clustering algorithm for mixed type attributes in large database environment. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 263–268, New York, NY, USA, 2001. ACM Press.

[7] Dana Cristofor and Dan Simovici. Finding median partitions using information-theoretical-based genetic algorithms, 2002.

[8] Richard C. Dubes. Cluster analysis and related issues. *Handbook of pattern recognition & computer vision*, pages 3–32, 1993.

[9] M. Esther, H.P Krieguel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases. In *KDD '96: Conf. Knowledge Discovery Data Mining*, pages 226–231, 1996.

[10] B. Everitt. *Cluster Analysis*. Edward Arnold, third edition, 1993.

[11] L. J. Fogel, A. J. Ownes, and M. J. Walsh. *Artificial Intelligence Through Simulated Evolution*. John Wiley and Sons Inc, NY, 1965.

[12] R. G and J. Han. Very large data bases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 144–155, Berkeley, CA., 1994.

[13] V. Ganti, J. Gehrke, and R. Ramakrishnan. Cactus- clustering categorical data using summaries. In *International Conference on Knowledge Discovery and Data Mining*, pages 73–83, New York, NY, USA, 1999.

[14] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Co., Reading, Massachusetts, 1989.

[15] J. Gower. A general coefficient of similarity and some of its properties. *Biometrics*, 27:857–872, 1971.

[16] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Cure: an efficient clustering algorithm for large databases. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 73–84, New York, NY, USA, 1998. ACM Press.

[17] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5):345–366, 2000.

[18] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3):107–145, 2001.

[19] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc, 2000.

[20] D. J. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. Addison Wesley, 2000.

[21] Zengyou He, Xiaofei Xu, and Shengchun Deng. K-histograms: An efficient clustering algorithm for categorical dataset, 2003.

[22] Zengyou He, Xiaofei Xu, and Shengchun Deng. K-anmi: A mutual information based clustering algorithm for categorical data, 2004.

[23] Zengyou He, Xiaofei Xu, and Shengchun Deng. Clustering mixed numeric and categorical data: A cluster ensemble approach, 2005.

[24] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the theory of Neural computation*. Addison-Wesley Longman Publishing Co, Inc., MA, Santa Fe Institute Studies in the Sciences of Complexity lecture notes., 1991.

[25] Alexander Hinneburg and Daniel A. Keim. An efficient approach to clustering in large multimedia databases with noise. In *Knowledge Discovery and Data Mining*, pages 58–65, 1998.

[26] J. H. Holland. *Adaption in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[27] Z. Huang. Clustering large data sets with mixed numeric and categorical values. In *First Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1998.

[28] Zhexue Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *Research Issues on Data Mining and Knowledge Discovery*, pages 0–, 1997.

[29] Zhexue Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2(3):283–304, 1998.

[30] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice Hall, Englewood Cliffs, New Jersey, 1988.

[31] A. K. Jain, M.N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[32] George Karypis, Eui-Hong (Sam) Han, and Vipin Kumar NEWS. Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, 1999.

[33] L. Kauffman and P. Rousseuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley and Sons, New York, NY., 1990.

[34] T. Kohonen. *Self-orgniztion and assosiative Memory*. Springer information sciences series, Springer-Verlag, NY, third edition, 1989.

[35] Tao li, Sheng Ma, and Mitsunori Ogihara. Entropy-based criterion in categorical clustering. In *Proceedings of the 21 International Conference on Machine Learning*, pages 213–228, 2004.

[36] B. L. Milenova and M. M. Campos. Clustering large databases with numeric and nominal values using orthogonal projections. In *Proc. of the ACM SIGMOD Conf. Management of Data*, pages 0–, 2002.

[37] S. Mitra and T. Acharya. *Data Mining: Multimedia, Soft Computing and Bioinformatics*. Wiley Inter-Science, 2003.

[38] D.J. Newman, C.L. Blake S. Hettich, and C.J. Merz. UCI repository of machine learning databases, 1998. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[39] H. Ralambondrainy. A conceptual version of the k-means algorithm. *Pattern Recogn. Lett.*, 16(11):1147–1157, 1995.

[40] H. P. Schwefel. *Numerical Optimization of Computer Models.* John Wiley and Sons Inc, NY, 1981.

[41] C. E. Shannon. A mathemtical theory of communications. *Bell System Technical Journal*, pages 379–423, 1948.

[42] Gholamhosein Sheikholeslami, Surojit Chatterjee, and Aidong Zhang. WaveCluster: A multi-resolution clustering approach for very large spatial databases. In *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pages 428–439, 24–27 1998.

[43] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining.* Addison Wesley, 2006.

[44] W. wang, J. Yang, and R. Muntz. Sting: a statistical information grid approach to spatialdata mining. In *Proceedings of the 23rd Conference on VLDB*, pages 186–195, Athens, Greece., 1997.

[45] R. Xu. Survey of clustering algorithms. Technical report, Department of Electrical and Computer Engineering University of Missuri-Rolla, University of Missuri-Rolla, Rolla, MO 65409 USA, 2003.

[46] L. A. Zadeh. Fuzzy sets. *Inf. Control*, 8:338–353, 1965.

[47] He Zengyou, Xu Xiaofei, and Deng Shengchun. Squeezer: an efficient algorithm for clustering categorical data. *J. Comput. Sci. Technol.*, 17(5):611–624, 2002.

[48] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, pages 103–114, New York, NY, USA, 1996. ACM Press.