UNIVERSIDAD NACIONAL AUTONÓMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

"Determinación estadística de códigos numéricos en bases de datos mixtas"

TESIS

QUE PARA OPTAR POR EL GRADO DE:

MAESTRO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

PRESENTA:

RAÚL GALINDO HERNÁNDEZ

DIRECTOR:

DR. ÁNGEL FERNANDO KURI MORALES

POSGRADO DE INGENIERÍA

Ciudad de México, Septiembre de 2019

Resumen

Con la creciente cantidad de información que día a día se envía en todo el mundo y la cantidad de información creada diariamente, el almacenamiento de los datos, ya sean numéricos o no numéricos, se realiza en bases de datos. Cuando es requerido analizar los datos almacenados, tratar con variables numéricas resulta en un panorama más amplio al momento de elegir un algoritmo a aplicar sobre los datos, en comparación, con la cantidad de algoritmos que existen para el tratado de datos no numéricos. Los algoritmos de ML (Machine Learning) intentan obtener mayor información de las BD con el objetivo de aprovechar dicha información y generar un modelo capaz de solucionar problemas futuros. En el presente trabajo se aborda el problema existente en las bases de datos mixtas. Una menor parte de los algoritmos de ML existentes se enfocan en tratar con datos no numéricos (datos categóricos). Siendo un área que ha recibido una menor atención por parte de los investigadores. En este trabajo, se requiere transformar las variables categóricas a variables numéricas con la finalidad de que posteriormente se pueda acceder a la amplia gama de algoritmos que tratan datos solamente numéricos. Debido a que los datos numéricos son mas sencillamente mapeados a espacios métricos, en donde se puede obtener información más *rica* inherente a los datos. En esta tesis se toma un enfoque estadístico para transformar los atributos categóricos de una BDM a atributos numéricos. El teorema de límite central y la aproximación multivariada son las bases de donde parte la solución del problema. Se selecciona un algoritmo que puede ser aplicado a BDM y que posterior a su transformación a una BD completamente numérica, se encuentre dispuesta para la aplicación de algoritmos de inteligencia computacional basados en métricas.

Agradecimientos

Agradezco al Dr. Ángel Fernando Kuri Morales por brindarme la oportunidad de trabajar con él y por guiarme durante todo el proyecto de Tesis.

Al CONACyT por los recursos que me brindó durante toda mi estancia en la maestría que me permitieron dedicarme al posgrado de tiempo completo para terminar en tiempo y forma.

Agradezco a mis padres Maria Olivia Hernández Ayala y Flavio Galindo Cardoso por el apoyo incondicional brindado desde el inicio de mi vida hasta el día de hoy. Sin sus consejos, sus motivaciones, sin ellos, yo no estaría en en lugar que estoy actualmente. Por siempre estaré agradecido por todo el esfuerzo que han hecho por mi.

Mi hermana Yuridia Galindo Hernández y mi hermano Octavio Galindo Hernández, crecí a su lado y jamás olvidaré todos los consejos y enseñanzas que me han guiado durante mi vida. Son el gran ejemplo que sigo como persona y siempre han sido parte fundamental en mi desarrollo.

Mis tíos, Ismael Hernández Ayala e Idania Hernández Ayala por llevarme y aconsejarme como si de su hijo se tratase. Por los cuidados que tuvieron conmigo de niño y que hicieron en buena medida, que tuviera una infancia muy buena.

Mis cuñados, Antonio Pichardo Sánchez y Ángeles León Guzmán por ser las parejas que mis hermanos esperan y por ser siempre bienvenido en sus casas.

Mis sobrinos, Ricardo Galindo León, Emmanuel Pichardo Galindo y Héctor Galindo León, que a pesar de su corta edad significan mucho en mi vida y generan una gran felicidad a toda la familia.

Lista de acrónimos

AA Ascent Algorithm (Algoritmo de ascenso).

BD Base de Datos.

BDM Base de Datos Mixta.

CESAMO Categorical Encoding by Statistical Applied Modeling (Codificación categórica por modelado estadístico aplicado).

CPU Central Processing Unit (Unidad central de procesamiento).

FAA Fast Ascent Algorithm (Algoritmo de ascenso rápido).

GA Genetic Algorithm (Algoritmo genético).

kNN K-nearest neighbor (k-ésimo vecino más cercano).

KS Kolmogorov-Smirnov.

ML Machine Learning (Aprendizaje automatizado).

MLP Multi-layered Perceptron Networks (Redes de perceptrones multicapa).

NN Neural Networks (Redes neuronales).

NP-Difícil Nondeterministic Polynomial time - hard (Tiempo polinomial no determinista - difícil).

SOM Self-Organizing Maps (Mapas auto-organizados de Kohonen).

SVM Support Vector Machines (Máquinas vectoriales de soporte).

Índice general

Índice de figuras

Índice de tablas

I think the brain is essentially a computer and consciousness is like a computer program. It will cease to run when the computer is turned off. Theoretically, it could be re-created on a neural network, but that would be very difficult, as it would require all one's memories.

Stephen Hawking

CAPÍTULO

1

Introducción

1.1 Contexto

Aprendizaje automatizado (Machine Learning - ML) ha sido definido como una aplicación de la inteligencia artificial (campo de las ciencias de la computación) que proporciona la capacidad de aprender automáticamente a partir de la "experiencia". El proceso de aprendizaje comienza con observaciones (datos), para buscar patrones presentes en los datos y tomar mejores decisiones en el futuro en función de los ejemplos que se tienen. El objetivo principal es permitir que las programas informáticos aprendan automáticamente sin intervención o asistencia humana.

Estadística computacional y aprendizaje automatizado comparten características como lo es análisis de datos y generación de modelos predictivos, que son muy útiles para abordar problemas de ML. Una gran parte del campo de estudio de ML se centra en el diseño de soluciones factibles para tales problemas, a través del uso de metodologías matemáticas y estadísticas [1].

En las últimas décadas han surgido diversos algoritmos que buscan atacar los problemas computacionales más complicados (NP-Difícil), entre ellos se encuentran:

1. K-Medias (K-Means) [2].

2. Mapas auto-organizados de Kohonen (Self-organizing maps) [3].

3. C-Medias difusas (Fuzzy C-Means) [4] [5] [6].

4. Redes de perceptrones multicapa (Multi-layered perceptron networks) [7].

5. Máquinas de vectores de soporte (Support vector machines) [8].

Cada uno de los algoritmos mencionados anteriormente comparten la característica de ser aplicables solo para bases de datos completamente numéricas. Es decir, las BD en las que todos sus atributos representan valores numéricos. El análisis de los atributos categóricos (es decir, los atributos cuyo dominio no es numérico) es una tarea difícil pero importante, muchos campos como: estadística o sociología tratan con datos categóricos. Aún conociendo su importancia, la tarea de analizar atributos categóricos en BDM ha recibido una atención relativamente escasa por parte de los investigadores de ML [9].

En años recientes el análisis de la información almacenada en grandes bases de datos se ha convertido en una importante herramienta para las empresas, ya que a través de la preparación, el sondeo y la exploración de los datos, es posible obtener como resultado final información *oculta* y de gran utilidad (comportamientos y futuras tendencias) que brinde soporte para la toma de decisiones y permita hacer proyecciones a futuro para el negocio.

Para realizar el correspondiente análisis de información a la BD, se debe conocer la composición de la misma. Es decir, si solo contiene datos numéricos (números enteros, números con decimales, números positivos o negativos, etc.), o si también contiene datos no numéricos (lugar de nacimiento de un paciente, complexión corporal de un paciente, etc.), ya que de acuerdo a su composición interna será tratada de una forma u otra. Si una base de datos contiene tanto variables númericas como variables categóricos se le denomina base de datos mixta (BDM).

La mayoría de los algoritmos de inteligencia computacional usados en tareas de minería de datos se encuentran basados en métricas, por lo que al momento de aplicar estos algoritmos a un atributo numérico, provee información más *rica*, en comparación con un atributo no numérico. Éstos no

representan un número e implica un problema de mapeo, complicándose la asignación de un valor categórico al dominio de un espacio métrico.

1.2 Problemática

Las bases de datos mixtas contienen datos categóricos y numéricos, por lo cual el problema estriba en que se requiere aplicar algoritmos numéricos de inteligencia computacional basados en métricas, aunque la base de datos no lo sea. La descripción de la problemática que se presenta, se describe en los siguientes puntos.

- ✓ Si alguno de los atributos de la BD no es numérico, los algoritmos previos no son aplicables.
- ✓ Para analizar grandes BD es preciso tratar los datos de tal manera que éstos puedan ser procesados adecuadamente para extraer información útil.
- ✓ Interesa aplicar algoritmos numéricos (basados en métricas) aunque las BD no lo sean.
- ✓ Solamente cuando estas BD se hayan pre-procesado será posible el análisis numérico antes mencionado.

1.3 Objetivos

Objetivo general: Realizar la codificación numérica de los atributos categóricos presentes en una base de datos mixta, preservando los patrones presentes en los datos.

Objetivos específicos:

1. Codificación numérica de las variables categóricas presentes en una BDM.
2. Preservación de patrones presentes en los datos.
3. Aplicación de algoritmos basados en métricas a BD que ha sido codificada numéricamente.
4. Comparación de resultados obtenidos con algoritmos existentes en la literatura.

1.4 Contribuciones

La contribución de esta tesis es introducir un enfoque en el que se busca preservar los patrones inherentes de la BDM y determinar con precisión los códigos numéricos que preservan tales patrones. Se reemplaza una búsqueda exhaustiva sobre toda la BDM de los posibles códigos por muestreo estadístico, con el objetivo de que el tiempo de procesamiento se reduzca considerablemente comparado con otros enfoques. Una vez que las instancias categóricas se reemplazan por los códigos más adecuados, se logra la preservación de los patrones presentes. Por códigos se entiende como a los valores numéricos que reemplazarán a las instancias categóricas de las variables categóricas presentes en la BDM.

Se aplica un algoritmo que reemplace datos categóricos por datos numéricos sin pérdida de patrones, lo cual genera la posibilidad de aplicar a las bases de datos mixtas, algoritmos de inteligencia computacional basados en métricas.

1.5 Restricciones

De acuerdo a resultados de pruebas realizadas con el algoritmo que aplicamos, una de las limitantes a tener en cuenta es que los códigos numéricos encontrados para una determinada BDM (digamos BD0) no son aplicables a otra BD que no sea la original (digamos BD1). Por lo que los códigos encontrados para BD0 no son aplicables para BD1, incluso si ambas son similares en estructura.

Por otra parte, cuando las tuplas de la BDM tienen un orden distinto los códigos numéricos encontrados son diferentes. Es decir, cuando se analiza la BDM ”original” (digamos BD0A) se determinan sus códigos numéricos (digamos CN1), posteriormente se efectúa un ordenamiento diferente a las tuplas de la BDM (digamos BD0B), los códigos encontrados para BD0B (dígamos CN2) son diferentes. Por lo que CN1 $\neq$ CN2, sin embargo, ambos conjuntos de códigos cumplen con el objetivo de preservar los patrones.

1.6 Estructura de la tesis

El resto de este trabajo se estructura en capítulos. En el capítulo 2 se aborda un panorama acerca de las actuales metodologías que atacan la problemática presentada. En el tercer capítulo se encuentra la justificación y el razonamiento que condujo a la solución al problema.

Posteriormente, después de que se ha introducido la temática de ML y del algoritmo que aplicamos (CESAMO - Categorical Encoding by Statistical Applied Modeling) [14] de una manera formal, en el capítulo cuatro se muestra la implementación del algoritmo. Para el capítulo cinco se presentan casos de estudios donde BDM son analizadas, posteriormente de haber realizado preprocesamiento de datos, con el algoritmo propuesto. Finalmente, para el capítulo seis se muestran las conclusiones de la tesis.

Machine consciousness refers to attempts by those who design and analyse informational machines to apply their methods to various ways of understanding consciousness and to examine the possible role of consciousness in informational machines.

Igor Aleksander

CAPÍTULO

2

Estado del arte

2.1 Aprendizaje Supervisado

Las técnicas de ML son un proceso general inductivo, a través del cual se crea automáticamente un clasificador por aprendizaje, a partir de un conjunto de datos. Una de las clasificaciones de ML lo divide en dos vertientes: aprendizaje supervisado y no supervisado.

El aprendizaje supervisado es la búsqueda de algoritmos que razonan desde instancias suministradas externamente para producir hipótesis generales, de las cuales se generan predicciones sobre instancias futuras. Su objetivo es el etiquetado de datos, el cual es, el proceso de asignar para cada uno de los datos una clase determinada. Es decir, se intenta construir un modelo conciso de la distribución de etiquetas de clase en términos de características de predicción. Por ejemplo, el clasificador resultante se utiliza para asignar etiquetas de clase a las instancias de prueba donde se conocen los valores de las características del predictor, pero el valor de la etiqueta de clase es desconocido [11].

Cada instancia en cualquier conjunto de datos utilizado por los algoritmos de aprendizaje automático se representa utilizando el mismo conjunto de características. Las características pueden

ser continuas, categóricas o binarias. Si las instancias se dan con etiquetas conocidas (correspondientes salidas correctas), entonces el aprendizaje se denomina supervisado, en contraste con el aprendizaje no supervisado, donde las instancias no están etiquetadas [1].

	Árboles de Decisión	Redes Neuronales	Naive-Bayes	K-ésimo vecino más cercano (kNN)	Máquinas de vector de soporte (SVM)	Aprendizaje por reglas
Precisión en general	**	***	*	**	****	**
Velocidad de aprendizaje con respecto al número de atributos e instancias	***	*	****	****	*	**
Velocidad de clasificación	****	****	****	*	****	****
Tolerancia a valores faltantes	***	*	****	*	**	**
Tolerancia a atributos irrelevantes	***	*	**	**	****	**
Tolerancia a atributos redundantes	**	**	*	**	***	**
Tolerancia a alta interdependencia de atributos	**	***	*	*	***	**
Analizando atributos discretos/binarios/continuos	****	*** (no discretos)	*** (no continuos)	*** (no directamente discretos)	** (no discretos)	*** (no directamente continuos)
Tolerancia a ruido	**	**	***	*	**	*
Evitando sobre-ajuste	**	*	***	***	**	**
Intentos para aprendizaje incremental	**	***	****	****	**	*
Transparencia de conocimiento/ clasificación	****	*	****	**	*	****
Manejo de parámetros del modelo	***	*	****	***	*	***

Figura 2.1: Comparación de algoritmos de aprendizaje supervisado (1 estrella representa el peor desempeño y 4 estrellas el mejor) [12].

De acuerdo a [12] las máquinas de vector de soporte (SVM) y las redes neuronales son los algoritmos de aprendizaje supervisado que mejor precisión en general obtienen. Al momento de realizar la evaluación del clasificador (mapeo de instancias no etiquetadas a clases) generalmente se basa en la precisión de predicción (el porcentaje de predicción correcta dividido por el número total de predicciones). Para llevar a cabo la medición de la precisión, existen al menos tres técnicas que se utilizan para calcularla. Una de ellas, es dividir el conjunto de entrenamiento utilizando dos tercios para el entrenamiento y el otro tercio para estimar el desempeño. En otra técnica, conocida como validación cruzada, el conjunto de entrenamiento se divide en subconjuntos mutuamente exclusivos e iguales y para cada subconjunto se entrena al clasificador sobre la unión de todos los subconjuntos. El promedio de la tasa de error de cada subconjunto es, por lo tanto, una estimación de la tasa de error del clasificador. Si la evaluación de la tasa de error no es satisfactoria, se debe regresar a una etapa previa del proceso de aprendizaje supervisado.

Un método común para comparar los algoritmos de aprendizaje supervisado es realizar comparaciones estadísticas de las precisiones de los clasificadores entrenados en conjuntos de datos específicos. Si se tiene disponible suficiente suministro de datos, podemos muestrear una serie de conjuntos de entrenamiento de tamaño N, ejecutar los algoritmos de aprendizaje en cada uno de ellos y estimar la diferencia en la precisión de cada par de clasificadores en un conjunto grande de prueba.

2.2 Aprendizaje No Supervisado

El aprendizaje no supervisado estudia la forma en cómo los sistemas pueden aprender a representar patrones de entrada particulares de una manera que refleje la estructura estadística de los patrones de entrada que tienen el conjunto de datos.

En contraste con el aprendizaje supervisado, no hay objetivos de salida explícitos o evaluaciones ambientales asociadas con cada entrada, ya que el aprendizaje no supervisado, aplica sesgos previos sobre qué aspectos de la estructura de entrada deben ser capturados en la salida.

Los métodos de aprendizaje no supervisado tienen que actuar sobre los patrones de entrada, digamos x_i, observados, que frecuentemente se supone que son muestras independientes de una subyacente distribución de probabilidad desconocida, digamos $P_i(x)$, y alguna información a priori explícita o implícita sobre lo que debe tomarse en cuenta.

Algunos métodos de aprendizaje no supervisado buscan descubrir cómo representar las entradas x_i, definiendo la calidad que tienen las buenas características para después buscar esas características en las entradas. Por ejemplo, considere el caso de que la salida $y(x) = w * x$ es una proyección lineal de la entrada en un vector de peso w. El teorema del límite central implica que la mayoría de tales proyecciones lineales tendrán estadísticas gaussianas. Por lo tanto, si se pueden encontrar pesos w tales que la proyección tenga una distribución altamente no gaussiana (por ejemplo, multimodal), entonces es probable que la salida refleje algún aspecto interesante de la entrada.

Esta es la intuición detrás de un método estadístico llamado búsqueda de proyección, en donde, se ha demostrado que puede implementarse utilizando una forma modificada de aprendizaje de

Hebb [10]. Hacer el ajuste para que las diferentes salidas deban representar diferentes aspectos de la entrada se vuelve sorprendentemente complicado.

2.2.1 **Algoritmos de clustering**

En los últimos años, han sido propuestos un gran número de algoritmos de clustering, los cuales se encuentran disponibles en la literatura. En esta sección se describe una clasificación que va de acuerdo al método que tienen los algoritmos de clustering para determinar los grupos [13]:

- ✓ Clustering Particional. Estos algoritmos tratan de descomponer el conjunto de datos en un conjunto de grupos, así también, intentan definir un número de particiones que optimiza una determinada función de criterio, la cual puede hacer énfasis a la estructura local o global de los datos, y su optimización se realiza mediante un proceso iterativo.

- ✓ Clustering Jerárquico. Se trata de un procedimiento sucesivo en el cual pueden ocurrir dos procesos: unir grupos pequeños en unos más grandes, o bien, dividir los grupos más grandes. El resultado provisto es un árbol de grupos llamado dendrograma, en el cual se observa la forma en que están relacionados los grupos. Cuando se realiza un corte al dendrograma en un nivel determinado, se obtiene el clustering de los elementos de los datos en grupos separados.

- ✓ Clustering Basado en Densidad. Intentan agrupar en grupos, objetos vecinos que hay en un conjunto de datos, basado en las condiciones de densidad.

- ✓ Clustering Basado en Celdas. La principal característica de estos algoritmos es que cuantifican el espacio en un número finito de celdas y posteriormente realizan cada una de sus operaciones en dicho espacio.

En términos generales, los algoritmos de clustering están basados en un criterio para evaluar la calidad de una partición dada. Es decir, toman como entrada algunos parámetros (por ejemplo, número y densidad de grupos) e intentan definir la mejor partición de un conjunto de datos de acuerdo a los parámetros dados. Por lo tanto, definen una partición de un conjunto de datos basado

en ciertas suposiciones y no necesariamente la ”mejor” suposición que se ajusta al conjunto de datos.

2.3 Estado del arte

En las últimas décadas los investigadores han puesto escasa atención a las BDM [9], con la intención de minarlas. Recientemente, se han propuesto algunas técnicas para ”codificar” las variables no numéricas. Una de las más populares es la conocida como: One-hot encoding. Ésta propone asignar valores binarios a las instancias de una variable categórica, donde 1 significa existencia y 0 significa ausencia.

Por ejemplo, se tiene un atributo grupo étnico. Sus posibles instancias son: asiático, afroamericano, blanco, hindú, caucásico. Utilizando one-hot encoding quedaría de la siguiente forma.

Grupo Étnico	Bla	Afr	Asi	Hin	Cau
Blanco	1	0	0	0	0
Hindú	0	0	0	1	0
Asiático	0	0	1	0	0
Afroamericano	0	1	0	0	0
Caucásico	0	0	0	0	1

Figura 2.2: Atributo grupo étnico codificado mediante one-hot encoding.

Esto presenta dos grandes problemas. La variable categórica grupo étnico originalmente representaba un atributo en la BDM y ahora es representada por cinco diferentes atributos (blanco, afroamericano, hindú, etc.), que es el número de instancias posibles para la variable categórica. En problemas del mundo real existen variables no numéricas con decenas o centenas de posibles instancias, lo que significaría tener decenas o centenas de atributos agregados a la BDM.

Además, esto solo es para una variable categórica, en dado caso de tener una gran cantidad de este tipo de variables, para cada una de ellas se crearán más atributos. Esto implica que la BD crezca

exponencialmente. Esto conduce a bases de datos más grandes que son más difíciles de almacenar y manejar.

El segundo inconveniente es que a priori se le asigna un valor numérico a cada instancia, sin en realidad conocer si este conjunto de valores representa correctamente la categoría.

Finalmente, con este tipo de esquema, las variables ya no reflejan la esencia de la idea transmitida por una categoría. Una variable correspondiente a la i-ésima instancia de la categoría refleja la forma en que una tupla está afectada al pertenecer al i-ésimo valor categórico, que es correcto. Pero ahora el problema original: ¿Cómo cambia el comportamiento de los individuos de acuerdo con la categoría? se reemplaza por: ¿Cómo cambia el comportamiento de los individuos cuando el valor de la categoría es el i-ésimo? Las dos preguntas no son intercambiables.

En intentos previos [14] se ha adoptado un enfoque diferente para realizar codificación de variables no numéricas:

a) Preservando los patrones incrustados en la base de datos y,

b) Localizando los códigos que preservan dichos patrones.

Estos dos pasos dan como resultado la identificación correcta de un conjunto de códigos numéricos. El algoritmo resultante se llama CENG (Codificación categórica con redes neuronales y algoritmos genéticos) y su versión paralelizada ParCENG [15]. Sin embargo, este enfoque es computacionalmente muy exigente y para aumentar su eficiencia, debe abordarse en conjuntos de múltiples CPU. Aun así, cuando el número de instancias de la categoría es grande, el tiempo de ejecución puede aumentar exponencialmente.

Computers are getting smarter and smarter. Scientists say that soon they will be able to talk to us (and by 'they' I mean computers, I doubt very much that scientists will be able to talk to us).

Dave Barry

CAPÍTULO

3

Justificación

3.1 Importancia del análisis de la información

El análisis de los datos, sin importar la manera en la que son generados, se ha convertido en una tarea de gran relevancia para cualquier empresa u organización, debido a que la información obtenida a través de dicho análisis ha pasado a ser uno de los activos de mayor importancia, tal como menciona Alexandros Labrinidis [16]. Esto se debe básicamente al hecho de que a partir del análisis de los datos es posible describir y predecir comportamientos o hechos.

Debido al crecimiento de información almacenada en BDM y a las exigencias actuales en las organizaciones, resulta relevante analizar las BDM. Un paso de suma importancia es el pre-procesamiento de las mismas. En este paso se intenta adecuar los datos para ser procesados de forma más eficiente, haciendolo sin pérdida de los patrones inherentes a la BDM.

3.2 Codificación preservando patrones de la BDM

Como se indicó en la introducción, la idea básica es aplicar algoritmos de ML diseñados para bases de datos estrictamente numéricas (BDN) a las BDM. Esto, a través de la codificación numérica en los casos de variables categóricas. Esto no representa un nuevo concepto. Sin embargo, las BDM ofrecen un desafío particular cuando se intenta agrupar en grupos porque, en principio, es imposible imponer una métrica a las variables categóricas. No hay forma de asignar códigos numéricos a las variables categóricas en general. Un objetivo alternativo es asignar códigos a todas y cada una de las instancias de cada clase (categoría) que preservarán los patrones presentes para una BDM determinada.

Considerando un conjunto de tuplas d-dimensionales (por ejemplo, U) cuya cardinalidad es t. Suponiendo que hay n funciones desconocidas de $n-1$ variables cada una, denotadas con:

$$f_k(v_1,\cdots,v_{k-1},v_{k+1},\cdots,v_n); k=1,\cdots,n$$

Suponiendo también que hay un método que nos permite aproximar f_k (de las tuplas) con F_k. Denotando las n funciones resultantes de $n-1$ variables independientes con F_i, por lo tanto,

$$F_k \approx f(v_1,\cdots,v_{k-1},v_{k+1},\cdots,v_n); k=1,\cdots,n \tag{3.1}$$

La diferencia entre f_k y F_k se indicará con εk tal que, para el atributo k y las t tuplas en la base de datos

$$\varepsilon k = max[abs(f_{ki}-F_{ki})]; i=1,\cdots,t \tag{3.2}$$

El argumento que se desea mostrar es que, códigos numéricos son los que minimizan εk para todo k. Esto es así porque solo estos códigos conservan las relaciones entre la variable k y las variables restantes $n-1$, y hacen esto para todas las variables en el conjunto. Por lo tanto, conservarán el conjunto completo de relaciones (es decir, patrones) presentes en la base de datos, como en la

siguiente ecuación.

$$\Xi = min[max(\varepsilon k; k = 1, \cdots, n)] \tag{3.3}$$

Se debe tener en cuenta que este es un problema de optimización multi-objetivo. Porque el cumplimiento de la condición k en 3.2 para cualquier valor dado de k puede inducir el incumplimiento de una posible diferente k (esto representa el primer objetivo a cumplir). El segundo objetivo es minimizar el error máximo de 3.2. Usar la expresión min-máx de 3.3 equivale a seleccionar un punto en particular en el frente de Pareto [17].

Para lograr el objetivo propuesto, se debe tener una herramienta que sea capaz de identificar las F_k en 3.1 y los códigos que logren la minimización de 3.3. Esto es posible usando redes neuronales (NN) y algoritmos genéticos (GA). Las consideraciones teóricas (por ejemplo, [18] [19] [20] [21]) aseguran la efectividad del método. Pero, como se indicó, el uso de dicho conjunto implica un número posiblemente exponencial de operaciones de punto flotante. Una alternativa que produce un proceso similar pero más económico es el algoritmo propuesto, descrito a continuación.

3.3 Algoritmo para asignación de códigos numéricos

Se denota el número de tuplas en la BD por t y el número de atributos categóricos por c; el número de atributos numéricos por n; la i-ésima variable categórica por vi; el valor obtenido para la variable i en función de la variable j por $yi(j)$.

Se toman muestras de los códigos que producen yi en función de una relación buscada. Esta relación y el modelo de la población que implica, se seleccionarán con el objetivo de preservar los patrones de comportamiento incrustados en la DB.

Dos temas son de primordial importancia en la metodología propuesta:

a) Cómo definir la función que preservará los patrones.

b) Cómo determinar el número de códigos a muestrear.

Con respecto a (a), usando un modelo matemático que considera relaciones de alto orden, como se

explicará más adelante. Con respecto a (b), se sabe que, independientemente de la distribución de las *yi*, la distribución de las medias de las muestras de *yi* ($y_i(avg)$) se volverán gaussianas.

Una vez que la distribución de $y_i(avg)$ se convierta en gaussiana, se habrá alcanzado una estabilidad estadística, en el sentido de que un mayor muestreo de los *yi* no modificará significativamente la caracterización de la población.

En esencia, por lo tanto, lo que aplicamos es muestrear suficientes códigos para garantizar la estabilidad estadística de los valores calculados a partir de $yi \leftarrow f(vj)$. Si se elige adecuadamente $f(vj)$, los códigos correspondientes a la mejor aproximación serán los que se insertan en la BDM.

Además, el algoritmo propuesto se basa en un muestreo de doble nivel: solo se consideran pares de variables y cada par está, en sí mismo, muestreando el espacio multivariado. Esto evita la necesidad de resolver explícitamente el problema subyacente de la optimización multi-objetivo. El problema de agrupamiento puede ser, entonces, abordado numéricamente.

3.4 Pseudo-código del algoritmo

En este trabajo aplicamos el siguiente algoritmo:

a) Se especifica la base de datos mixta BDM.

b) Se especifica el tamaño de la muestra (*ss*).

c) La BDM se analiza para determinar n, t y $ci(i)$ para $i = 1, ..., c$.

d) Se supone que los datos numéricos se han mapeado entre [0,1). Por lo tanto, cada *ci* estará, igualmente, entre [0,1).

A continuación se presenta el pseudo-código del algoritmo propuesto [14].

Algoritmo 1 Algoritmo para asignación de códigos numéricos.

```
1:  for Para cada i = 1 hasta c do
2:      while Hacer hasta que la distribucion de y_i(avg) sea gaussiana do
3:          Selecciona aleatoriamente la variable j (j diferente de i)
4:          Asignar valores aleatorios a todas las instancias de v_i.
5:          y_i(avg) = 0
6:          for Para k = 1 hasta ss do
7:              y_i = f(v_j)
8:              y_i(avg) = y_i(avg) + y_i
9:          y_i(avg) = y_i(avg)/ss
10:     Selecciona los codigos correspondientes al mejor valor de y_i.
```

El algoritmo inicia seleccionando una variable categórica de la BDM. Se asignan valores numéricos aleatorios consistentes con los valores originales, es decir, si existen dos diferentes instancias en la variable categórica, el algoritmo asigna dos diferentes valores numéricos. Por lo tanto, para una instancia original se asigna un valor numérico y el mismo valor numérico se asignará a cada ejemplo de la instancia original presente en la variable categórica.

En el siguiente paso, los valores aleatorios numéricos propuestos se aproximan con una función (formulada a partir de una variable elegida aleatoriamente en la BDM y que es diferente de la variable categórica seleccionada inicialmente). Es relevante notar que esta variable (vj) puede ser, en sí mismo, categórica. En esos casos, cada instancia categórica de vj se reemplaza por códigos aleatorios para que sea posible calcular $f(vj)$. En este paso se aplica el algoritmo de ascenso rápido (FAA - Fast Ascent Algorithm) [15] que se explicará posteriormente.

El resultado de aproximar las dos funciones formuladas a partir de las variables de la BDM, es un error de aproximación. Este error se suma en la variable $y_i(avg)$ y al finalizar el ciclo se divide entre el tamaño de la muestra (ss). Es importante aclarar que en cada paso de este ciclo, se generan nuevas funciones con el objetivo de encontrar el menor error de aproximación. Una vez encontrado, se realiza la asignación de los códigos numéricos a la variable categórica inicial, ya que estos códigos numéricos corresponden al mejor valor de y_i.

El criterio de parada se cumple hasta que los promedios ($y_i(avg)$) de los errores de aproximación converjan a una distribución gaussiana. En ese momento, se ha cubierto el espacio de posibles

códigos numéricos asignables a la variable categórica. Se conoce que las medias de las muestras (cuando tienden a infinitas medias) eventualmente convergerán a una distribución gaussiana. Por lo que, cuando la convergencia se cumple la distribución se establece y se hace la suposición de que se ha probado un espacio de posibilidades suficientemente grande. Finalmente, en lugar de probar códigos numéricos diferentes infinitas veces se utiliza este criterio de parada.

El proceso descrito previamente se repite para cada una de las variables categóricas (c) de la BDM.

A calculator is a tool for humans to do math more quickly and accurately than they could ever do by hand; similarly, AI computers are tools for us to perform tasks too difficult or expensive for us to do on our own, such as analyzing large data sets or keeping up to date on medical research.

Oren Etzioni

CAPÍTULO

4

Implementación de la propuesta

4.1 Implementación

En este capítulo se detallará la implementación de la arquitectura que ha sido introducida en el capítulo 3, se presentará de manera detallada la forma en la que serán implementados los componentes involucrados en la metodología de desarrollo elegida.

Surgen dos problemas para la asignación de códigos:

1. En la línea número 7 del pseudo-código mostrado previamente (en donde se calcula la función de aproximación), surge la pregunta: ¿Cómo implementar el cálculo del error de aproximación?

2. Y además: ¿Cómo saber cuándo tiene una distribución normal la distribución del ajuste promedio?

4.2 Algoritmo de ascenso

En el pseudo-código mostrado previamente, implica que, dada una tabla de la forma que se muestra en la tabla 4.1, es posible encontrar una expresión algebraica de y como una función de x. Se supone que los valores en la siguiente tabla son instancias de datos experimentales.

x	y
x_1	y_1
x_2	y_2
...	...
x_n	y_n

Tabla 4.1: Tabla con datos numéricos.

Se conocen innumerables algoritmos de aproximación que satisfacen este criterio. Tres de estos multivariados se analizan en [22]. Sin embargo, una característica importante presente en uno de ellos, el llamado algoritmo de ascenso rápido (FAA), es que, para implementarlo, no es necesario cargar en la memoria principal de la CPU todas las tuplas de datos; un hecho que es de mayor importancia cuando (como en el caso de las aplicaciones del mundo real) las bases de datos pueden constar de miles o incluso millones de tuplas. Además, con FAA, la forma del aproximante puede seleccionarse arbitrariamente.

Para responder a la primera pregunta, se tiene lo siguiente: Cualquier función continua puede ser muy cercanamente aproximada, mediante la combinación lineal de una constante y un conjunto de k monomios de a) Grado impar y b) Grado menor o igual que 11.

$$P(l) = c_0 + c_1 S + c_3 S^3 + c_5 S^5 + c_7 S^7 + c_9 S^9 + c_{11} S^{11}$$

se le conoce como combinación lineal de constantes y un conjunto de k monomios.

En [23] se demostró que la función logística $\frac{1}{(1+e^{-x})}$ puede ser aproximada por el polinomio anterior, y que los datos continuos pueden ser aproximados (y sus componentes principales conservados). Los términos de grado superiores a 11 son de poca importancia cuando, como en este caso,

$0 \leq x < 1$. FAA produce coeficientes para la mejor aproximación de L_∞ de y dada la x como en la tabla 4.1. A partir de esto, el algoritmo calcula el error de aproximación *RMS*. El código que logra el error de aproximación promedio (ε_s) más pequeño una vez que se alcanza la normalidad será el conjunto de códigos numéricos seleccionado.

Con este aproximador garantizamos que, con alta precisión, se preservan las relaciones entre pares de variables.

Entonces, se selecciona el algoritmo de ascenso [24] propuesto por Cheney W. en 1966. Este algoritmo permite aproximar una variable como una función de una segunda variable, arrojando como salida un error de aproximación. En nuestro programa calculamos L_2 a partir del polinomio entregado por el algoritmo de ascenso.

4.2.1 **Fundamento teórico**

Este algoritmo tiene su fundamento teórico en lo que se describe a continuación. Se supone que los renglones de una matriz A^i_j satisfacen un requerimiento de no degeneración llamado la ”condición de Haar”: Un conjunto de vectores en espacio n se dice que satisface la condición de Haar si cada conjunto de n de ellos es linealmente independiente. Es decir, cada selección de n vectores de tal conjunto es una base para n espacio.

Se tiene el siguiente teorema. Teorema de intercambio: Sea $A^0,...,A^{n+1}$ un conjunto de vectores en el espacio n satisfaciendo la condición de Haar, si 0 recae en el casco convexo de $A^0,...,A^{n+1}$, entonces hay un índice $j \leq n$ tal que esta condición permanece verdadera cuando A^j es reemplazada por A^{n+1}.

Entonces lo que se busca es un punto donde la función:

$$\Delta(x) = [\max_{1 \leq i \leq m} |r_i(x)| = \max_{1 \leq i \leq m} \langle A^i, x \rangle - b_i] \tag{4.1}$$

alcance su valor mínimo. Esto supone que la condición Haar se satisface por el conjunto de vectores $A^1,...,A^m$.

Es un teorema importante para la implementación del algoritmo de ascenso. Asegura que al realizar el intercambio entre vectores del conjunto interno y del conjunto externo, los coeficientes encontrados para el polinomio minimax, minimizan el error máximo en cualquier intercambio de vectores.

Otro teorema importante es: Cada solución minimax del sistema $\sum_{j=1}^{n} A_j^i x_j = b_i (i = 1, ..., m > n)$ es una solución minimax de un subsistema apropiado que comprende $n+1$ ecuaciones. La idea básica del algoritmo de ascenso es calcular las soluciones minimax de una sucesión de subsistemas, cada uno comprendiendo $n+1$ ecuaciones.

Por el teorema anterior, la solución de uno de estos subsistemas es el punto buscado. Por otro lado, no puede haber más que un número finito de subsistemas y esta observación es la base para la prueba de que el algoritmo es efectivo.

En cada ciclo de cómputo, se tendrá un conjunto de $n+1$ índices $J = \{i_0, ..., i_n\}$ y un vector de signos $\sigma = \{\sigma_0, ..., \sigma_n\}$ tal que,

$$0 \in \{\sigma_0 A^{i_0}, ..., \sigma_n A^{i_n}\}$$

Resolviendo el siguiente sistema de $n+1$ ecuaciones lineales para determinar un vector $y = [y_1, ..., y_n]$ y un número e: $\sigma_j r_{ij}(y) = e$ en donde $j = 0, ..., n$.

Con el objetivo de asegurar que $e > 0$, se aplica el cambio de signos de todos los σ_j sin pérdida de propiedad. Por lo anterior, implica que y es una solución minimax del sistema:

$$\langle A^{i_j}, y \rangle = b_{i_j} (j = 0, ..., n)$$

El teorema de caracterización expresa que: "Dado un punto $z \in R_n$ sea $\sigma_i = sgn\ r_i(z)$ y $M = \{i : |r_i(z)| = \Delta(z)\}$ El punto z minimiza Δ si y solo si el origen de R_n se encuentra en el casco convexo del conjunto $\{\sigma_i A^i : i \in M\}$". Regresando al algoritmo de ascenso, si $e = \Delta(y)$, por el teorema de caracterización y se sabe que es una solución minimax del sistema original de m ecuaciones. En caso contrario, existe al menos un índice α (que no existe en J), tal que $|r_\alpha(y)| > e$. Se podría tomar α tal que $|r_\alpha(y)| = \Delta(y)$, pero esto no es necesario.

Sea $\mu = sgn\, r_\alpha(y)$, usando el teorema del intercambio se reemplaza uno de los vectores $\sigma_0 A^{i_0}, ..., \sigma_n A^{i_n}$ por μA^α de tal forma que el origen permanezca en el casco convexo.

La necesidad de calcular y y e lleva a realizar una suposición acerca de los datos, una forma conveniente es que la matriz:

$$\begin{pmatrix} \sigma_0 & A_1{}^{i_0} & \cdots & A_n{}^{i_0} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_n & A_1{}^{i_n} & ... & A_n{}^{i_n} \end{pmatrix}$$

sea no singular. Los cálculos del algoritmo terminan solo cuando $e = \Delta(y)$, en donde esta ecuación significa que y es una solución. Se sabe que solo un número finito de conjuntos J existen.

Un aspecto que requiere explicación es, ¿Cómo se determina el conjunto inicial J y el vector σ? Para esto, se sabe que J puede ser tomado arbitrariamente y entonces se puede encontrar una solución no trivial a la ecuación:

$$\sum_{j=0}^{n} \theta_j A^{i_j} = 0$$

estableciendo que $\sigma_j = sgn\theta_j$.

Un arreglo que simplifica los cálculos es que se puede re-escribir la ecuación $e = \sigma_j r_{i_j}(y) = \sigma_j[\langle A^{i_j}, y\rangle - b_{i_j}]$ de la siguiente forma: $-\sigma_j e + \langle A^{i_j}, y\rangle = b_{i_j}$ y en notación de matriz es:

$$\begin{pmatrix} \sigma_0 & A_1{}^{i_0} & \cdots & A_n{}^{i_0} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_n & A_1{}^{i_n} & ... & A_n{}^{i_n} \end{pmatrix} \begin{pmatrix} -e \\ y_1 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} b_{i_0} \\ \vdots \\ b_{i_n} \end{pmatrix}$$

Ahora, si se supone que la matriz A_j de la izquierda tiene inversa $C = (C_j{}^i)$ quedaría de la siguiente forma:

$$\begin{pmatrix} -e \\ y_1 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} {C_0}^0 & \cdots & {C_n}^0 & \\ \cdots & \cdots & \cdots & \cdots \\ {C_0}^n & \cdots & {C_n}^n & \end{pmatrix} \begin{pmatrix} b_{i_0} \\ \vdots \\ b_{i_n} \end{pmatrix}$$

Debido a que C es la inversa de A_j:

$$\begin{pmatrix} {C_0}^0 & \cdots & {C_n}^0 \\ \cdots & \cdots & \cdots \\ {C_0}^n & \cdots & {C_n}^n \end{pmatrix} \begin{pmatrix} \sigma_0 & {A_1}^{i_0} & \cdots & {A_n}^{i_0} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_n & {A_1}^{i_n} & ... & {A_n}^{i_n} \end{pmatrix} = \begin{pmatrix} 1 & \cdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \cdots & 1 \end{pmatrix}$$

De acuerdo a lo anterior se puede notar que:

$$\sum_{j=0}^{n} \sigma_j {C_j}^0 = 1, \sum_{j=0}^{n} {C_j}^0 A^{i_j} = 0$$

Por lo tanto, los números $\sigma_j {C_j}^0$ son los coeficientes necesarios para expresar el hecho de que 0 se encuentra en el casco convexo de los puntos $\sigma_j A^{i_j}$. Estos coeficientes entran en los cálculos relacionados al teorema de intercambio. Se puede notar que μA^{α} debe ser expresado como una combinación lineal de $\sigma_0 A^{i_0}, \cdots, \sigma_n A^{i_n}$. Si se asigna $A^{\alpha} = \sum_{j=0}^{n} \lambda_j A^{i_j}$, los coeficientes λ_j pueden ser calculados mediante una ecuación matricial:

$$(\lambda_0, \cdots, \lambda_n) \begin{pmatrix} \sigma_0 & {A_1}^{i_0} & \cdots & {A_n}^{i_0} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_n & {A_1}^{i_n} & ... & {A_n}^{i_n} \end{pmatrix} = (\mu, {A_1}^{\alpha}, \cdots, {A_n}^{\alpha})$$

La solución de la ecuación matricial está dada por:

$$(\lambda_0, \cdots, \lambda_n) = (\mu, {A_1}^{\alpha}, \cdots, {A_n}^{\alpha}) \begin{pmatrix} {C_0}^0 & \cdots & {C_n}^0 \\ \cdots & \cdots & \cdots \\ {C_0}^n & \cdots & {C_n}^n \end{pmatrix}$$

Ya que $\mu A^{\alpha} = \sum_{j=0}^{n}(\mu\sigma_j\lambda_j)(\sigma_j A^{i_j})$, las proporciones a ser calculadas en el teorema de intercambio son $\mu\sigma_j\lambda_j/\sigma_j {C_j}^0 \equiv \mu\lambda_j/{C_j}^0$. El número β es elegido como el índice con la más grande proporción, es decir, el índice que maximiza mayormente la ecuación matricial.

Es importante tener en cuenta que al avanzar de un ciclo completo de cálculos al siguiente, solo una fila de la matriz A_j cambia. El efecto sobre C se puede predecir a partir del siguiente teorema.

Teorema: Sea A una matriz no singular y $C_1, \cdots, C_n$ las columnas de su matriz inversa. Sea $\bar{A}$ la matriz que resulta al reemplazar la β-ésima fila de A por un vector v. Si $\lambda \equiv \langle v, C_\beta \rangle \neq 0$, entonces $\bar{A}$ es no-singular y las columnas de su inversa están dadas por las igualdades $\bar{C}_\beta = \lambda^{-1} C_\beta$ y $\bar{C}_j = C_j - \langle v, C_j \rangle \bar{C}_\beta$ donde $j \neq \beta$.

En el algoritmo de ascenso se reemplaza la fila $(\sigma_\beta, {A_1}^{i_\beta}, \cdots, {A_n}^{i_\beta})$ por una nueva fila $(\mu, {A_1}^{\alpha}, \cdots, {A_n}^{\alpha})$. Por esto, es necesario el número λ. Este es el producto interno de la nueva fila con la β-ésima columna de C, es decir, $\mu C^0 + \sum_{j=1}^{n} {A_j}^{\alpha} {C_\beta}^j$. Y como se vio anteriormente, este es el número λ_β calculado previamente.

4.2.2 **Pseudo-código**

A continuación se muestra el algoritmo de ascenso paso por paso [22].

1. Ingresa los vectores de datos (llámalos D).

2. Ingresa los grados de cada una de las variables del polinomio que se aproxima.

3. Mapea los vectores de datos originales en las potencias de los monomios seleccionados (llámalos P).

4. Estabiliza los vectores de P mediante la alteración aleatoria de los valores originales (llama a los datos resultantes S). Este paso se explica a detalle en el subtema 4.2.2.2.

5. Selecciona un subconjunto de tamaño M de S. Llámalo I. Llama E a los vectores restantes. BOOTSTRAP

6. Obtén los signos minimax (llama A a la matriz incorporando los σ's). Este paso se explica a detalle en el subtema 4.2.2.1.

7. Obtén la inversa de A (llámala B). Este paso se explica a detalle en el subtema 4.2.2.3.

   LOOP

8. Calcula los coeficientes $C = fB$. El máximo error interno ε_θ también es calculado.

9. Calcula el máximo error externo ε_ϕ de C y E. Llama a sus índices I_E.

10. $\varepsilon_\theta \geq \varepsilon_\phi$ Si: Termina el algoritmo; los coeficientes de C son los del polinomio minimax para los vectores de D.

11. Calcula el vector λ de $A^{I_E}B$. Este paso se explica a detalle en el subtema 4.2.2.4.

12. Calcula el vector β que maximiza $\sigma_{I_E}\frac{\lambda_j}{\beta_j}$. Llama a sus índices I_I. Este paso se explica a detalle en el subtema 4.2.2.5.

13. Intercambia los vectores I_E y I_I.

14. Calcula la nueva inversa $\overline{B}$. Haz $B \leftarrow \overline{B}$.

15. Ve al paso 8.

El objetivo del algoritmo de ascenso es expresar el comportamiento de una variable dependiente (y) como función de un conjunto de n variables independientes (v):

$$y = f(v_1, v_2, ..., v_n) \tag{4.2}$$

$$y = f(v)$$

El aproximante es definido con el objetivo de tener la siguiente forma:

$$y = c_1X_1 + c_2X_2 + ... + c_mX_m \tag{4.3}$$

Donde X_i denota una combinación de variables independientes. Es decir, $X_i = f_i(v)$. De acuerdo a la forma en que estas combinaciones son definidas se pueden obtener diferentes aproximantes. El método supone que hay una muestra de tamaño N tal que para cada conjunto de variables independientes v hay un valor conocido de la variable dependiente f. Por convención N representa el número de objetos en la muestra y $M = m + 1$, donde m, es el número de términos deseados del aproximante.

El propósito del algoritmo es encontrar los valores de los coeficientes de la ecuación 4.3 tal que, los valores aproximados minimicen la diferencia entre los valores conocidos de la variable dependiente f en la muestra y los que fueron calculados en la ecuación 4.3 para todos los objetos de la muestra. Un algoritmo genético es usado, para encontrar los coeficientes que minimicen la diferencia mencionada previamente, con el objetivo de que el algoritmo de ascenso converja con mayor rapidez. Este algoritmo es llamado Algoritmo de Ascenso Optimizado (FAA - Fast Ascent Algorithm) y se describe con mayor detalle en la sección 4.2.3.

Se define el error de aproximación como:

$$\varepsilon_{MAX} = max(\varepsilon_1, ..., \varepsilon_m)$$

donde:

$$\varepsilon_i = abs(f_i - y_i)$$

.

Donde, f_i representa el valor de la variable dependiente del objeto i y y_i es el valor que el aproximador produce cuando las X_i son ingresadas.

El algoritmo de ascenso está basado en una metodología iterativa de dos fases. Primero, un subconjunto de la muestra (de tamaño M) es seleccionada (ésta es llamada el conjunto interno) y el mejor aproximador en el sentido minimax (un conjunto de coeficientes) es encontrado. En la segunda parte, el aproximador encontrado en la primera fase es probado para determinar si $y = f(X)$ satisface la norma minimax para los restantes $N - M$ objetos (éste conjunto de cardinalidad $N - M$

es llamado el conjunto externo) de la muestra. Esto significa que, las y_i se calculan para el conjunto externo. Si la condición minimax se cumple para todos los objetos, el algoritmo termina, y los coeficientes son los que se encuentran en el mejor aproximador encontrado. En dado caso de que al menos uno de los objetos en el conjunto externo no cumpla con la condición minimax, entonces un objeto del conjunto interno se intercambia con un objeto del conjunto externo y el proceso se repite.

En cada paso (t) del algoritmo, se calculan dos errores:

a) El error absoluto de aproximación más grande del conjunto interno (denotado como $\varepsilon_\theta(t)$).

b) El error absoluto de aproximación más grande del conjunto externo (denotado como $\varepsilon_\phi(t)$).

El error $\varepsilon_\theta(t)$, se calcula durante la fase uno. Mientras que el error $\varepsilon_\phi(t)$ en la fase dos. La condición de convergencia es:

$$\varepsilon_\theta(t) \geq \varepsilon_\phi(t)$$

Se muestra en [25] que, $\varepsilon_\theta(t+1) > \varepsilon_\theta(t)$ monotónicamente y que $\varepsilon_\phi(t+1) < \varepsilon_\phi(t)$ no monotónicamente. Por lo tanto, en cada paso (t) el error de aproximación del conjunto interno aumenta, mientras que el error del conjunto externo disminuye, resultando en un acercamiento entre sí hasta que el error interno es mayor o igual que el error externo. Por lo cual, la condición de convergencia siempre se cumple.

4.2.2.1 **Polinomio minimax para el conjunto interno**

Se sabe que, $\varepsilon_\theta = max(\varepsilon_1, ..., \varepsilon_M)$, además, $\varepsilon_i = abs(f_i - y_i)$ y que, $y_i = c_1X_{i1} + c_2X_{i2} + ... + c_mX_{im}$. Entonces:

$$\varepsilon_i + c_1X_{i1} + c_2X_{i2} + ... + c_mX_{im} = f_i \leftarrow i = 1, 2, ..., M$$

Haciendo $\varepsilon_i = s_i\varepsilon_\theta$ donde s_i son constantes a ser determinadas, entonces se tiene:

$$\begin{array}{ccccccc} s_1\varepsilon_\theta + & c_1X_{11} + & c_2X_{12} + & \ldots & c_mX_{1m} = & f_1 \\ s_2\varepsilon_\theta + & c_1X_{21} + & c_2X_{22} + & \ldots & c_mX_{2m} = & f_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ s_M\varepsilon_\theta + & c_1X_{M1} + & c_2X_{M2} + & \ldots & c_mX_{Mm} = & f_M \end{array}$$

Aplicando regla de Cramer al sistema anterior, se tiene lo siguiente:

$$\varepsilon_\theta = \frac{\begin{pmatrix} f_1 & X_{11} & \cdots & X_{1m} \\ \cdots & \cdots & \cdots & \cdots \\ f_M & X_{M1} & \cdots & X_{Mm} \end{pmatrix}}{s_1\begin{pmatrix} X_{21} & \cdots & X_{2m} \\ \cdots & \cdots & \cdots \\ X_{M1} & \cdots & X_{Mm} \end{pmatrix} - \ldots + s_M\begin{pmatrix} X_{21} & \cdots & X_{2m} \\ \cdots & \cdots & \cdots \\ X_{m1} & \cdots & X_{mm} \end{pmatrix}}$$

Haciendo que "X_{i*}" sea la i-ésima fila y la primera columna de los determinantes en el denominador del sistema anterior, entonces queda:

$$\varepsilon_\theta = \frac{\Delta}{+s_1|\Delta_{1*}| - s_2|X_{2*}| + s_3|X_{3*}| - \ldots}$$

Donde, $\Delta =$ al determinante del numerador en el sistema anterior. Haciendo que, $\Delta_{i*} = |X_{i*}|$. En notación más compacta:

$$\varepsilon_\theta = \frac{\Delta}{\sum_{i=1}^{M} s_i(-1)^{i-1}\Delta_{i*}}$$

Entonces, para minimizar ε_θ, el denominador de la ecuación previa tiene que ser maximizado. Esto implica que:

a) Todos sus productos deben ser maximizados.

b) Todos sus sumandos deben ser del mismo signo.

Esta observación conduce a un algoritmo que permite resolver la ecuación anterior, minimizando ε_θ, y por lo tanto, encontrar los coeficientes de la ecuación 4.3 en el sentido minimax. Ya que $\varepsilon_i = s_i\varepsilon_\theta$, por lo que, $s_i \leq 1$ y $s_\theta = 1$. Por lo tanto, para que $s_i\Delta_i$ sea maximizado, se tiene que:

1. Tomar el valor más grande de s_i.

2. Asignar los valores de s_i tal que, $s_i = sgn[(-1)^{i-1}\Delta_{i*}]$

La primera condición se cumple si $s_i = 1 \forall i$. Esto es, para que los errores de aproximación absoluta se minimicen, todos deben ser del mismo tamaño absoluto. Para cumplir con la segunda condición, se deben determinar los signos de Δ_{i*}. La mejor forma para hacerlo es recurrir al teorema del cofactor [26]: "Si los cofactores de una columna de un determinante son multiplicados por los elementos de una columna diferente y sumados, el resultado es cero". Haciendo que, $sgn(\Delta_{i*}) = sgn(\sigma_i)$ y ya que, sus valores absolutos son 1, se puede resolver:

$$\begin{pmatrix} \sigma_1 & X_{11} & X_{12} & \cdots & X_{1m} \\ \sigma_2 & X_{21} & X_{22} & \cdots & X_{2m} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \sigma_M & X_{M1} & X_{M2} & \cdots & X_{Mm} \end{pmatrix} \begin{pmatrix} \varepsilon_\theta \\ c_1 \\ c_2 \\ \vdots \\ c_m \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_M \end{pmatrix}$$

Para obtener los signos minimax, denotando la i-ésima fila de A con A^i y su j-ésima columna con A_j, del teorema del cofactor:

$$\sum_{i=0}^{m} K_j A^i = 0$$

Como se observa en la ecuación 4.3, X_i denota una combinación de variables independientes. Una elección común es, alguna combinación lineal de los monomios de las variables independientes. Asumiendo que, nuestra elección ha sido la siguiente:

$$w(x,y) = \sum_{i=0}^{1}\sum_{j=0}^{2} c_{ij}x^i y^j = c_{00} + c_{01}y + c_{02}y^2 + c_{10}x + c_{11}xy + c_{12}xy^2 \tag{4.4}$$

Esto determina que, $X_1 = (1, y_1, {y_1}^2, x_1, x_1 y_1, x_1 {y_1}^2, ..., X_N = (1, y_N, {y_N}^2, x_N, x_N y_N, x_N {y_N}^2)$. Para este caso, $m = 6$, $M = 7$ y $N = 150$. Entonces, se puede tomar cualquier subconjunto de tamaño 7 de XY, y encontrar los coeficientes de la ecuación 4.4 que minimizan el error absoluto del subconjunto seleccionado. Hasta aquí, aún permanece el problema de encontrar los 6 coeficientes deseados para el conjunto completo de tamaño N.

4.2.2.2 **Perturbación y estabilidad**

Se puede ver en el algoritmo que es fácil encontrar datos donde un conjunto de filas o columnas pueden ser linealmente dependientes. En estos casos, el sistema de ecuaciones que será formulado puede volverse numéricamente inestable o simplemente no tener solución. Por esto, los elementos en la matriz de datos X serán reemplazados de acuerdo a lo siguiente:

$$\left.\begin{array}{r} X_i * (1 + \rho_u * \delta_H) si X_i \neq 0 \\ \rho_u * \delta_H si X_i = 0 \end{array}\right\} \quad {X_i}^*$$

donde ρ_u denota una variable aleatoria uniformemente distribuida: $0 \leq \rho_u < 1$ y $\delta_H = O(10^{-6})$. Esto implica, reemplazar vectores linealmente dependientes por vectores linealmente independientes. Los coeficientes de aproximación son cercanamente correctos, por lo que, no se necesitan otros cambios. El error de aproximación relativo es de $O(\delta_H)$. Es decir,

$$\frac{|{F_\tau}^* - F_\tau|}{F_\tau} < \delta_H \sum_i d_i$$

donde ${F_\tau}^*$ es el valor de la función que proxima los vectores alterados en (la convergencia) el paso τ del algoritmo. F_τ es el valor de la función que aproxima los vectores originales no perturbados. δ_H es el tamaño de la constante de perturbación y d_i denota el grado más alto de la i-ésima variable.

4.2.2.3 **Obteniendo la inversa**

Sea:

1. A una matriz no singular,

2. B sea su inversa y $B_1, ..., B_m$ sus columnas,

3. $\overline{A}$ la matriz que se obtiene al reemplazar la $\beta - sima$ fila de A por un vector v.

Si $\lambda_\beta \equiv \langle v, B_\beta \rangle \neq 0$ entonces $\overline{A}$ es no singular y las columnas de su inversa están dadas por $\overline{B}_\beta = \frac{B_\beta}{\lambda_\beta}$ y $\overline{B} = B_j - \langle v, B_j \rangle \overline{B}_\beta$ para $j \neq \beta$.

4.2.2.4 **Obteniendo el vector λ**

Se supone que las filas de A son linealmente independientes:

$$A^j = \sum_{i=1}^{M} -\frac{K_i}{K_j} A^i$$

donde:

$$i \neq j$$

Expresando el vector externo, correspondiente a $\sigma_p hi(A^{I_E})$, como una combinación lineal de los vectores internos:

$$A^{I_E} = \sum_{i=1}^{M} \lambda_i A^i$$

Después se tiene:

$$A^{I_E} - \lambda_j A^j - \sum_{i=1}^{M} \lambda_i A^i = 0$$

Posteriormente:

$$A^{I_E} + (\sum_{i=1}^{M} \frac{\lambda_j K_i}{K_j} - \lambda_i) A^i = 0$$

Se requiere seleccionar j, entonces:

$$\frac{\lambda_j K_i}{K_j} - \lambda_i \geq 0$$

,

por lo que 0 es una combinación lineal no negativa de $A^1,...,A^{I_E}$ sin que aparezca A^j. Esto es equivalente a:

$$\frac{\lambda_j}{K_j} \geq \frac{\lambda_i}{K_i}$$

Ya que:

$$A^{I_E} = \lambda A$$

El vector λ está dado por:

$$\lambda = A^{I_E} B$$

4.2.2.5 **Obteniendo el vector β**

Para obtener el vector β se conoce que, $B = A^{-1}$. En términos del teorema del cofactor B^1 es la fila de los cofactores. Por lo tanto, las proporciones necesarias para determinar el j-ésimo elemento a intercambiar se pueden tomar como las que corresponden al mayor $\frac{\lambda_j}{B_j^1}$. En realidad, se busca el más grande $\sigma_{I_E} \frac{\lambda_j}{B_j^1}$ para que se mantenga el signo del error del vector externo (A^{I_E}).

Este resultado es muy interesante, porque a partir de la segunda instancia de B pueden obtenerse en $O(M^2)$ operaciones. Como el vector λ y las matrices C pueden obtenerse de B, su cálculo eficiente es muy conveniente. De hecho, cada ciclo del algoritmo de ascenso solo necesita $O(M^2)$ operaciones. Este algoritmo es adecuado para resolver sistemas como el presentado en la ecuación 4.3. Es muy conveniente para nuestros propósitos en dos importantes aspectos:

1. Permite aproximar los datos incluso cuando N es grande, sin necesidad de manejar matrices grandes en memoria y,

2. Se puede definir un número arbitrario de términos m^* y usar un algoritmo genético (GA) para seleccionar, del conjunto completo de coeficientes de tamaño $\prod_{i=1}^{n}(d_i+1)$, el mejor aproximador formado por m^* coeficientes.

Además, dado que la forma en que se determina X_i de la ecuación 4.3 es arbitraria, puede aplicarse a cualquier número de variables.

4.2.3 Algoritmo de ascenso optimizado (FAA)

Cualquier función de m_0 variables se puede aproximar con a lo más:

$$T = \sum_{i=1}^{k} [\frac{(2_i - 1 + (\sum_{j=1}^{k} \frac{(2_j - 1 + m_0)!}{(2_j - 1)! m_0!}))!}{(2_i - 1)!([\sum_{j=1}^{k} \frac{(2_j - 1 + m_0)!}{(2_j - 1)! m_0!}])!}]$$

términos de grado k. La expresión anterior produce términos en el orden de 10^{12}, incluso para relativamente pequeños m_0. Esto, no tiene sentido si se está tratando de aproximar cualquier función con un polinomio de muchos términos. Por esta razón, se usa un algoritmo genético (GA) para seleccionar el mejor subconjunto de términos, que se deciden considerar para poder expresar el problema con un número de términos razonable.

4.2.3.1 Polinomios genéticos

La razón básica para elegir una norma minimax es que el método descrito anteriormente no depende del origen de X_i en la ecuación 4.3. Se puede decidir que sean los monomios de un polinomio completo.

$$y = \sum_{i_1=0}^{d_1} \cdots \sum_{i_n=0}^{d_n} c_{i1} \ldots c_{i_n} {v_1}^{i1} \ldots {v_n}^{in}$$

Pero no hace ninguna diferencia para el algoritmo de ascenso si los X_i se obtienen de un conjunto de monomios o si son elementos de los vectores de datos arbitrarios. Esto es importante porque, como se indicó anteriormente, el número de monomios y coeficientes crece exponencialmente. Una forma de evitar el problema de tal explosión de coeficiente es definir a priori el número (digamos μ) de los monomios deseados del aproximador y luego seleccionar apropiadamente cuáles de los p posibles serán los monomios que formarán parte del aproximador resultante.

Hay:

$$\binom{p}{\mu}$$

posibles combinaciones de monomios e incluso para valores modestos de p y μ y la búsqueda

exhaustiva está fuera de discusión. Este problema de optimización se puede abordar utilizando un algoritmo genético (GA), como se indica a continuación.

El genoma es una cadena binaria de tamaño p. Cada bit en ella representa un monomio. Estos monomios están ordenados según la secuencia de las potencias consecutivas de las variables. Si el bit es 1 significa que el monomio correspondiente permanece mientras que si es un 0 significa que tal monomio no debe ser considerado. Lo que se debe asegurar es que, el número de $1's$ sea igual a μ (número deseado de monomios). Suponiendo, por ejemplo, que $y = f(v_1, v_2, v_3)$ y que $d_1 = 1$, $d_2 = d_3 = 2$. En tal caso, las potencias asignadas a las posiciones del genoma son $000, 001, 002, 010, 011, 012, 020, 021, 022, 100, 101, 102, 110, 111, 112, 120, 121, 122$. Para el caso donde $\mu = 6$, el genoma 110000101010000001 corresponde al polinomio:

$$P(v_1, v_2, v_3) = c_{000} + c_{001}v_3 + c_{020}{v_2}^2 + c_{022}v_2{}^2{v_3}^2 + c_{101}v_1v_3 + c_{122}v_1v_2{}^2{v_3}^2$$

La población inicial en el GA consta de un conjunto de cadenas binarias de longitud p en las que solo hay $\mu 1's$. La función de fitness no es, sin embargo, el error minimax. Más bien, el error *RMS*:

$$\varepsilon_{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_i - y_i)^2}$$

se calcula para cada polinomio probado y, al final del proceso, el polinomio que presenta el error más pequeño se selecciona como el mejor aproximador para el conjunto de datos original. Es decir, para cada genoma se calculan los términos correspondientes a los 1. Estos toman el lugar de la X_i en la ecuación 4.3. Después, se aplica el algoritmo de ascenso para obtener los coeficientes correspondientes. A cada combinación de $\mu 1's$ corresponde un conjunto de μ coeficientes minimizando $\varepsilon_{MAX} = max(|f_i - y_i|)\forall i$. Para este conjunto de coeficientes el error ε_{RMS} se calcula. Esta es la función de fitness para el GA. Al final, se conservan los coeficientes que mejor minimizan ε_{RMS} (provenientes del GA), y los que mejor minimizan ε_{MAX} (del algoritmo de ascenso).

4.3 Algoritmo para determinar normalidad de una distribución

Para la segunda pregunta hecha al inicio de este capítulo, se debe conocer el momento en que la distribución del ajuste promedio se vuelve normal. La bondad de ajuste de un modelo estadístico describe qué tan bien se ajusta a un conjunto de observaciones. Las medidas de bondad de ajuste típicamente resumen la discrepancia entre los valores observados y los valores esperados según el modelo en cuestión.

Tales medidas se pueden usar en la prueba de hipótesis estadística, por ejemplo, para probar la normalidad de los residuos, para probar si dos muestras se extraen de distribuciones idénticas (prueba de Kolmogorov-Smirnov), o si las frecuencias de resultado siguen una distribución específica (prueba de chi cuadrado de Pearson). En el análisis de la varianza, uno de los componentes en los que se divide la varianza puede ser una suma de cuadrados, por falta de ajuste.

Una definición importante a tener en cuenta es la hipótesis nula: todas las muestras provienen de la misma distribución.

4.3.1 Pruebas de bondad de ajuste

Lo siguiente es una breve lista de algunas pruebas de bondad de ajuste.

1. Prueba de chi-cuadrada.

Es cualquier prueba de hipótesis estadística en la que, la distribución muestral del estadístico de prueba es una distribución de chi-cuadrado cuando la hipótesis nula es cierta. Las pruebas de chi-cuadrada frecuentemente se construyen a partir de una suma de errores cuadrados. Las estadísticas de prueba que siguen a una distribución de chi-cuadrado surgen de un supuesto de datos independientes distribuidos normalmente, que es válido en muchos casos debido al teorema del límite central. Luego se puede usar una prueba de chi-cuadrado para rechazar la hipótesis de que los datos son independientes [27].

2. Prueba de Kolmogorov - Smirnov.

La prueba de Kolmogorov-Smirnov (prueba K-S o prueba KS) es una prueba no paramétrica de la igualdad de distribuciones de probabilidad unidimensionales y continuas que se puede usar para comparar una muestra con una distribución de probabilidad de referencia (prueba K–S de una muestra), o para comparar dos muestras (prueba K–S de dos muestras) [28].

El estadístico de Kolmogorov-Smirnov cuantifica una distancia entre la función de distribución empírica de la muestra y la función de distribución acumulativa de la distribución de referencia, o entre las funciones de distribución empírica de dos muestras. La distribución nula de esta estadística se calcula bajo la hipótesis nula de que las muestras se extraen de la misma distribución (en el caso de dos muestras) o que la muestra se extrae de la distribución de referencia (en el caso de una muestra). En cada caso, las distribuciones consideradas bajo la hipótesis nula son distribuciones continuas pero por lo demás no están restringidas.

La prueba K – S de dos muestras es uno de los métodos no paramétricos más útiles y generales para comparar dos muestras, ya que es sensible a las diferencias tanto en la ubicación como en la forma de las funciones empíricas de distribución acumulativa de las dos muestras.

La prueba de Kolmogorov-Smirnov se puede modificar para que sirva como prueba de bondad de ajuste. En el caso especial de las pruebas de normalidad de la distribución, las muestras se estandarizan y se comparan con una distribución normal estándar.

La prueba de bondad de ajuste o la prueba de Kolmogorov-Smirnov se construye utilizando los valores críticos de la distribución de Kolmogorov. La hipótesis nula es rechazada en el nivel α si:

$$\sqrt{n}D_n > K_\alpha$$

donde se encuentra $K\alpha$ desde:

$$Pr(K \leq K_\alpha) = 1 - \alpha$$

El poder asintótico de esta prueba es 1.

3. Criterio Cramer-von Mises.

En estadística, el criterio de Cramer-von Mises es utilizado para juzgar la bondad del ajuste de una

función de distribución acumulativa $F*$ en comparación con una función de distribución empírica F_n dada, o para comparar dos distribuciones empíricas. Se define como [29]:

$$\omega^2 = \int_{-\infty}^{\infty} [F_n(x) - F*(x)]^2 dF*(x)$$

En aplicaciones de una muestra, $F*$ es la distribución teórica y F_n es la distribución observada empíricamente. Alternativamente, las dos distribuciones pueden ser estimadas empíricamente. Esto se llama el caso de dos muestras. Si el valor del estadístico T es mayor que los valores tabulados, la hipótesis nula puede rechazarse.

4. Prueba de Anderson-Darling

La prueba de Anderson-Darling es una prueba estadística de cuando una muestra de datos dada se extrae de una distribución de probabilidad dada. En su forma básica, la prueba supone que no hay parámetros que estimar en la distribución que se está probando, en cuyo caso la prueba y su conjunto de valores críticos no tienen distribución [30].

Sin embargo, la prueba se usa más a menudo en contextos donde se está probando una familia de distribuciones, en cuyo caso los parámetros de esa familia deben estimarse y esto se debe tener en cuenta al ajustar el estadístico de prueba o sus valores críticos. Cuando se aplica a las pruebas si una distribución normal describe adecuadamente un conjunto de datos, es una de las herramientas estadísticas más poderosas para detectar la mayoría de las desviaciones de la normalidad. Se rechaza la normalidad si la estadística excede el nivel de significación requerido.

Las pruebas Anderson-Darling de la muestra K están disponibles para probar si varias colecciones de observaciones pueden modelarse como provenientes de una sola población, donde no es necesario especificar la función de distribución.

5. Prueba de Shapiro-Wilk.

Esta prueba utiliza el principio de hipótesis nula para verificar si una muestra x_1,...,x_n proviene de

una población con distribución normal. El estadístico de prueba se llama W [31]:

$$W = \frac{(\sum_{i=1}^{n} a_i x_{(i)})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

donde:

a) $x_{(i)}$ es el i-ésimo estadístico de orden. Es decir, el i-ésimo número más pequeño en la muestra.

b) $\bar{x} = (x_1 + ... + x_n)/n$ es la media muestral.

c) Las constantes a_i están dados por:

$$(a_i, ..., a_n) = \frac{m^T V^{-1}}{(m^T V^{-1} V^{-1} m)^{\frac{1}{2}}}$$

donde:

$$m = (m_1, ..., m_n)^T$$

y $m_1, ..., m_n$ son los valores esperados de las estadísticas de orden de variables aleatorias independientes e idénticamente distribuidas muestreadas de la distribución estándar normal, y V es la matriz de covarianza de esas estadísticas de orden.

El usuario puede rechazar la hipótesis nula si W está por debajo de un umbral predeterminado.

6. Criterio de información de Akaike (AIC).

Es una medida de la calidad relativa de los modelos estadísticos para un conjunto dado de datos. Dado un conjunto de modelos para los datos, AIC estima la calidad de cada modelo, en relación con cada uno de los otros modelos. Por lo tanto, AIC proporciona un medio para la selección del modelo [32].

El AIC se basa en la teoría de la información: ofrece una estimación relativa de la información perdida cuando se usa un modelo determinado para representar el proceso que genera los datos. Al

hacerlo, se ocupa de la compensación entre la bondad de ajuste del modelo y la complejidad del modelo.

AIC no proporciona una prueba de un modelo en el sentido de probar una hipótesis nula; es decir, AIC no puede decir nada sobre la calidad del modelo en un sentido absoluto. Si todos los modelos candidatos se ajustan mal, AIC no dará ninguna advertencia al respecto.

4.3.2 Asegurando normalidad de los datos

En todos los enfoques previos, el énfasis se encuentra en determinar cuando una muestra A no proviene de la misma distribución que la muestra B. En este caso, la muestra A es la distribución de los eventos y B es gaussiana. Lo que se busca es un método que garantice que la normalidad es alcanzada. Es decir, dada una muestra A, está se aproximará a una distribución normal. Esto es muy diferente a ser capaz de determinar que A no se distribuye normalmente.

4.3.2.1 La distribución chi-cuadrada modificada

La idea básica se encuentra en poder responder la pregunta ¿Qué tan probable es calcular un valor experimental de π mayor que ξ (que denotará el 'valor crítico') para un conjunto de datos que se distribuyen normalmente? Se divide el espacio de observación en cuantiles. Suponiendo que Q es el número de cuantiles, O_i es el número de eventos observados en el i-ésimo cuantil, E_i es el número de eventos esperados en el i-ésimo cuantil y Φ es el número mínimo de observaciones requeridas por cuantil. Sea p la que indique la probabilidad de que π exceda de ξ cuando los datos se distribuyen normalmente y hay al menos Φ eventos en todos los cuantiles [33]. Entonces:

$$\pi = \sum_{i=1}^{Q} \frac{(O_i - E_i)^2}{E_i} \wedge [O_i \geq \phi \forall i]$$

Debido a que se requiere que la distribución de los eventos sea normal, se pueden encontrar los intervalos (en desviaciones estándar) que deben ser seleccionadas para asegurar que $E_i = 1/Q$. Esto implica que cuando la distribución alcanca la normalidad $E_i = \sigma O_i/Q \forall i$. Por ejemplo, si $Q = 10$,

los valores izquierdos y derechos de los cuantiles y el área correspondiente de la curva normal serían como se muestra en la siguiente figura.

Cuantil	Izquierdo_*i*	Derecho_*i*	Área_*i*
1	-5.0000	-1.2815	0.1000
2	-1.2815	-0.8416	0.1000
3	-0.8416	-0.5243	0.1000
4	-0.5243	-0.2532	0.1000
5	-0.2532	0.0000	0.1000
6	0.0000	0.2532	0.1000
7	0.2532	0.5243	0.1000
8	0.5243	0.8416	0.1000
9	0.8416	1.2815	0.1000
10	1.2815	5.0000	0.1000

Figura 4.1: Intervalos para $Q = 10$ [31].

π es más pequeño cuanto más cerca están los eventos de los valores normales esperados. Por lo tanto, si $\pi \geq \xi$ se sabe que los datos se distribuyen normalmente con probabilidad $> 1 - p$. Es importante notar que esto es bastante diferente de la prueba de chi-cuadrada donde las estadísticas se pueden usar para rechazar la hipótesis de que los datos son independientes. Un experimento de Monte Carlo [34] puede diseñarse para encontrar el valor de ξ.

Por ejemplo, si se requieren 5 observaciones por cuantil, entonces se necesitan, al menos, 50 observaciones (teniendo 10 cuantiles). Por razones prácticas se exploran hasta 90 observaciones. Se requiere que la probabilidad de encontrar π solo por casualidad sea menor que $1 - p$, en este caso 0.95. Asignando los valores para $\pi_0 = 4.0$ y $\delta_\pi = 0.05$, se obtiene $\pi = 3.2$.

El experimento de Monte Carlo consiste en generar una secuencia de 50, ..., 90 muestras gaussianas y contar el número de observaciones en cada uno de los 10 intervalos. Después, se calculan los valores de π y el número de observaciones por cuantil. Si $\pi < \pi_0$ y $|O_i| > \Phi$ la muestra cumple, en caso contrario, la muestra no cumple lo que significa que no es gaussiana. Se comienza asumiendo un valor de π_0 que es mayor que su valor mínimo verdadero. Es importante notar que los valores más pequeños de π denotarán un mejor ajuste a una distribución gaussiana.

El pseudo-código del algoritmo es el siguiente [31]:

Algoritmo 2 Algoritmo para determinar normalidad de una distribución.

```
1: Q = 10, p = 0.95, ξ = 3.20, isNormal = true ; Inicialización de parámetros
2: while true do
3:     NM = QΦ-1 ; Observaciones mínimas
4:     NM++
5:     E_i = NM/Q ; Observaciones esperadas por cuantil
6:     mean = Media de los promedios de errores de ajuste; Calcula la media
7:     stdDes = Desviación estándar de los promedios de errores de ajuste; Calcula la desviación
       estándar
8:     for i=1 hasta NM do
9:         j = Quantil(x) ; j = Número de cuantil de x
10:        O(j) = O(j) + 1 ; Observaciones en el j-ésimo cuantil
11:    π_TEST = 0
12:    MO = true ; Bandera para observaciones mínimas requeridas
13:    for i=1 hasta Q do
14:        if O(i) < Φ then
15:            MO = false
16:        π_TEST = π_TEST + (O(i) − E_i)^2 / E_i
17:    if MO = false o π_TEST > ξ then
18:        isNormal = false
19:    return isNormal
```

Una vez que tenemos un método para determinar los valores críticos ξ de la distribución π, es posible obtener una tabla para diferentes combinaciones de Q y p. En la siguiente tabla se presentan los valores de algunas ξ calculadas de π.

Cuantiles	p	ξ
10	0.99	1.90
10	0.95	3.20
10	0.90	4.05
12	0.99	2.95
12	0.95	4.55
12	0.90	5.80
14	0.99	4.10
14	0.95	6.55
14	0.90	8.00

Tabla 4.2: Valores críticos de la distribución π [31].

Two ideas lie gleaming on the jeweler's velvet. The first is the calculus, the second, the algorithm. The calculus and the rich body of mathematical analysis to which it gave rise made modern science possible; but it has been the algorithm that has made possible the modern world.

David Berlinski

CAPÍTULO

5

Análisis de Resultados

En diferentes sitios web pueden ser encontradas muchas bases de datos, sin embargo, no todas se encuentran completas o algunas son realmente pequeñas (30 tuplas o menos). Se llevó a cabo la búsqueda de bases de datos que tuvieran datos categóricos y numéricos. Posteriormente, la metodología propuesta en esta tesis fue aplicada a diferentes BDM, de las cuáles se seleccionaron aquellas que arrojaron resultados interesantes y que son ilustrativos para poder explicar el funcionamiento de CESAMO. Los casos de estudio que han sido analizados se presentan en este capítulo.

Es importante mencionar que en cada caso se utilizó validación cruzada para el entrenamiento de las redes neuronales. El 80 % de las tuplas fueron usadas para la fase de entrenamiento y el 20 % restante para la fase de prueba.

5.1 Primer caso: verificando funcionalidad

De <https://www.kaggle.com/ronitf/heart-disease-uci> se extrajo la base de datos mixta llamada "Enfermedad del corazón". Los datos de la BDM fueron registrados por varios años en clínicas médicas de Cleveland, Estados Unidos. En donde cada tupla representa el caso de un paciente y

después de varios estudios, se registró si el paciente presentaba una enfermedad del corazón. De acuerdo a atributos como: presión arterial, colesterol, azúcar en sangre, edad, etcétera, lo que se desea conocer es si el paciente presenta una enfermedad del corazón o no. En total, contiene 303 tuplas y 14 atributos que se describen a continuación.

Atributo	Descripción	Tipo
Edad	Edad en años	Numérico
Sexo	Masculino o Femenino	Categórico
Dolor en Pecho	Tipo de dolor en el pecho	Categórico
Presion Arterial	Presión arterial en reposo (mm/Hg)	Numérico
Colesterol	Colesterol sérico (mg/dl)	Numérico
Azúcar en sangre	Azúcar en la sangre en ayunas	Categórico
Resultados electr.	Resultados electrocardiográficos en reposo	Categórico
Ritmo cardiaco	Máximo ritmo cardiaco alcanzado	Numérico
Angina	Angina inducida por el ejercicio	Categórico
Depresión	Depresión inducida por el ejercicio en relación con el descanso	Numérico
Ejercicio	Pendiente del valor máximo de ejercicio	Categórico
Vasos sanguineos	Número de vasos principales coloreados por fluoroscopia	Categórico
Thal	Puede ser normal, defecto fijo o defecto reversible	Categórico
Objetivo	Se tiene la enfermedad o no	Categórico

Tabla 5.1: Descripción de base de datos mixta Enfermedad del corazón.

En la base de datos no hay tuplas con valores faltantes. A continuación se muestra un pequeño segmento de la BDM. El orden de las columnas es como se observa en la tabla previa, iniciando por Sexo, ya que el atributo Edad no se muestra.

MA	D	145	233	G	J	150	M	2.3	P	S	AB	AG"
MA	C	130	250	F	K	187	M	3.5	P	S	AC	AG"
FA	B	130	204	F	J	172	M	1.4	R	S	AC	AG"
MA	B	120	236	F	K	178	M	0.8	R	S	AC	AG"
FA	A	120	354	F	K	163	N	0.6	R	S	AC	AG"
MA	A	140	192	F	K	148	M	0.4	Q	S	AB	AG"
FA	B	140	294	F	J	153	M	1.3	Q	S	AC	AG"
MA	B	120	263	F	K	173	M	0	R	S	AD	AG"
MA	C	172	199	G	K	162	M	0.5	R	S	AD	AG"
MA	C	150	168	F	K	174	M	1.6	R	S	AC	AG"

Figura 5.1: Pequeño segmento de la BDM Enfermedad del corazón.

En la siguiente figura se muestra la distribución de pacientes que tienen la enfermedad de corazón

y los pacientes que no.

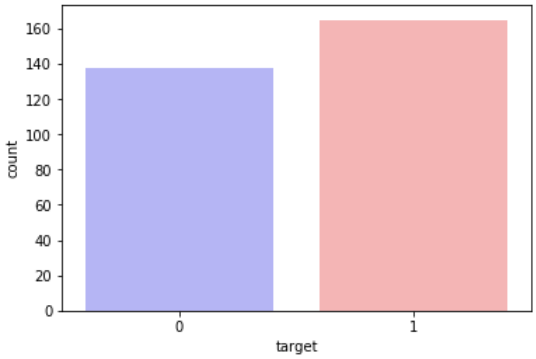

Figura 5.2: Pacientes con enfermedad del corazón (54.46 %) y pacientes que no la presentan (45.54 %).

Se puede observar que existe un mayor número de pacientes con enfermedad del corazón. Graficando la frecuencia por edades de los pacientes enfermos, se obtiene la siguiente figura.

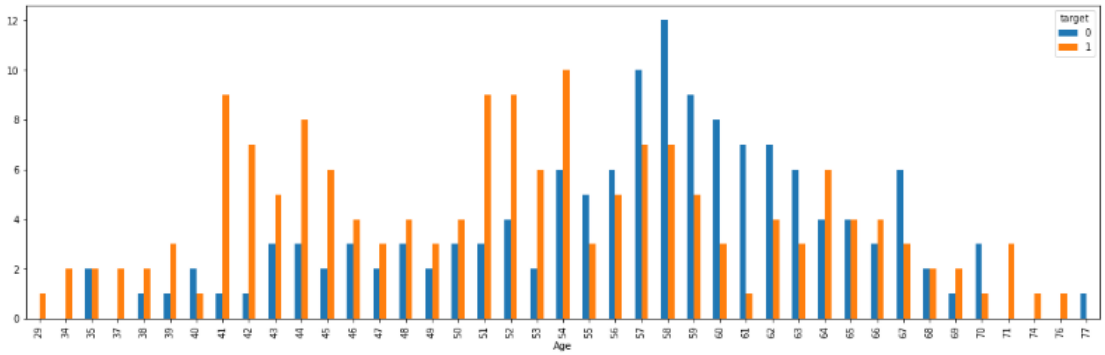

Figura 5.3: Frecuencia por edades de pacientes enfermos.

Posteriormente, CESAMO (Categorical Encoding by Statistical Applied Modeling) se aplicó sobre esta BDM y en la siguiente imagen se ilustra un pequeño segmento de la BD codificada.

Edad	Sexo	Dolor en pecho	Presión art.	Colesterol	Azúcar en sangre	Resultados de elec.	Ritmo cardiaco	Angina	Depresión	Ejercicio	Vasos sanguíneos	Thal	Objetivo
0.81818182	0.63565324	0.655719471	0.725	0.413120567	0.635653244	0.922403096	0.742574257	0.72120342	0.37096774	0.87277505	0.661542242	0.9224031	0.91344496
0.48051948	0.63565324	0.657829293	0.65	0.443262411	0.635774061	0.924161578	0.925742574	0.72120342	0.56451613	0.87277505	0.661542242	0.92416158	0.91344496
0.53246753	0.63577406	0.687251466	0.65	0.361702128	0.635774061	0.922403096	0.851485149	0.72120342	0.22580645	0.85175712	0.661542242	0.92416158	0.91344496
0.57142857	0.63565324	0.665669371	0.55	0.34929078	0.635774061	0.922403096	0.876237624	0.72120342	0	0.85175712	0.617919086	0.92416158	0.87860121
0.72727273	0.63565324	0.687251466	0.6	0.418439716	0.635774061	0.924161578	0.881188119	0.72120342	0.12903226	0.85175712	0.661542242	0.92416158	0.91344496
0.74025974	0.63577406	0.665669371	0.6	0.627659574	0.635774061	0.924161578	0.806930693	0.72087562	0.09677419	0.85175712	0.661542242	0.92416158	0.91344496
0.74025974	0.63565324	0.665669371	0.7	0.340425532	0.635774061	0.924161578	0.732673267	0.72120342	0.06451613	0.85659923	0.661542242	0.9224031	0.91344496
0.76623377	0.63565324	0.657829293	0.63	0.386524823	0.635653244	0.924161578	0.663366337	0.72120342	0.35483871	0.85659923	0.617919086	0.9224031	0.87860121
0.51948052	0.63565324	0.665669371	0.76	0.395390071	0.635774061	0.924161578	0.896039604	0.72120342	0	0.85175712	0.661542242	0.91182773	0.87860121

Figura 5.4: Muestra de la BDM Enfermedad del corazón codificada.

Como se observa en la figura anterior, todos las instancias categóricas fueron reemplazadas por códigos numéricos propuestos por el algoritmo. En la siguiente tabla se muestra la codificación del atributo Dolor en el pecho.

Instancia categórica	Código numérico propuesto
A	0.665669370888723
B	0.687251465682612
C	0.657829292502902
D	0.65571947125351

Tabla 5.2: Codificación del atributo categórico *dolor en el pecho*.

Este mismo proceso fue aplicado para los atributos categóricos restantes. También, los valores de los atributos numéricos existentes en la BDM fueron escalados y se les asignó un valor que se encuentra en el rango entre $(0, 1]$.

5.1.1 Aprendizaje supervisado

El problema original de clasificación es conocer si el paciente se encuentra enfermo del corazón o no (solo dos posibles valores), como una función de las variables restantes. Este problema de clasificación fue abordado por diferentes algoritmos, con los resultados ilustrados en la siguiente tabla.

Una vez que los datos fueron codificados, posteriormente se entrenó una red neuronal. Para determinar su arquitectura se recurrió a lo siguiente [35]:

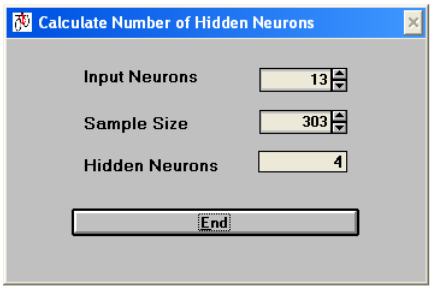

Figura 5.5: Determinación de arquitectura para entrenar la red neuronal.

Algoritmo	Precisión
Árbol de decisión (Decision tree)	0.852459
K-ésimo vecino más cercano (K-Nearest Neighbour)	0.918033
Regresión logística (Logistic regression)	0.934426
SVM Radial (Radial SVM)	0.918033
SVM Lineal (Linear SVM)	0.918033
Bosques aleatorios (Random Forest)	0.8852
Naive-Bayes	0.8689
Potenciación de Gradiente (Gradient Boosting Machine)	0.76
Potenciación de Gradiente (Modelo XGBoost)	0.903225
Gradiente descendente aleatorio (Stochastic Gradient Descent)	0.818181

Tabla 5.3: Resultados reportados con algoritmos tradicionales.

Por lo que en la arquitectura se tienen 13 neuronas para la capa de entrada, 4 neuronas en la capa oculta (con función de activación de tangente hiperbólica) y 1 una neurona para la capa de salida.

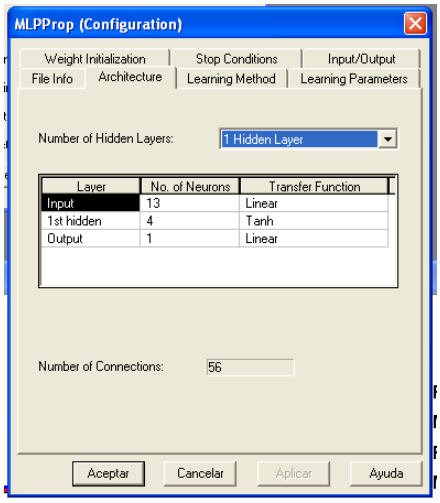

Figura 5.6: Arquitectura determinada para la BDM Enfermedad del corazón.

Se sabe que el error máximo en el entrenamiento de una red neuronal representa la distancia máxima que existe entre el valor que predice la red neuronal y el valor real de la BD. Este error es medido en cada una de las tuplas. De esta forma, el error máximo es el error más grande existente en toda la BD.

Se puede notar que la neurona en la capa de salida corresponde a la variable que determina si

el paciente se encuentra enfermo del corazón o no (dos posibles clases para este problema de clasificación). Suponiendo que el valor 1 significa paciente enfermo del corazón y que el valor 0 significa paciente no enfermo. Si la red neuronal predice un resultado de 0.9 cuando el paciente esta enfermo, significaría un error de 0.1 y sería claro notar que la predicción hecha pertenece a la clase enfermo del corazón. En caso de que la red neuronal arroje un resultado de 0.5, aun cuando el paciente está enfermo, significaría un error de 0.5. La predicción hecha sería equivocada, ya que no es claro saber a qué clase pertenece el resultado. Lo mismo sucede cuando la red neuronal arroja un resultado de 0.5 para el caso de un paciente no enfermo. Para este problema de clasificación, un error máximo de 0.5 permitiría distinguir sin equivocación entre las clases.

Los resultados se observan en la siguiente figura:

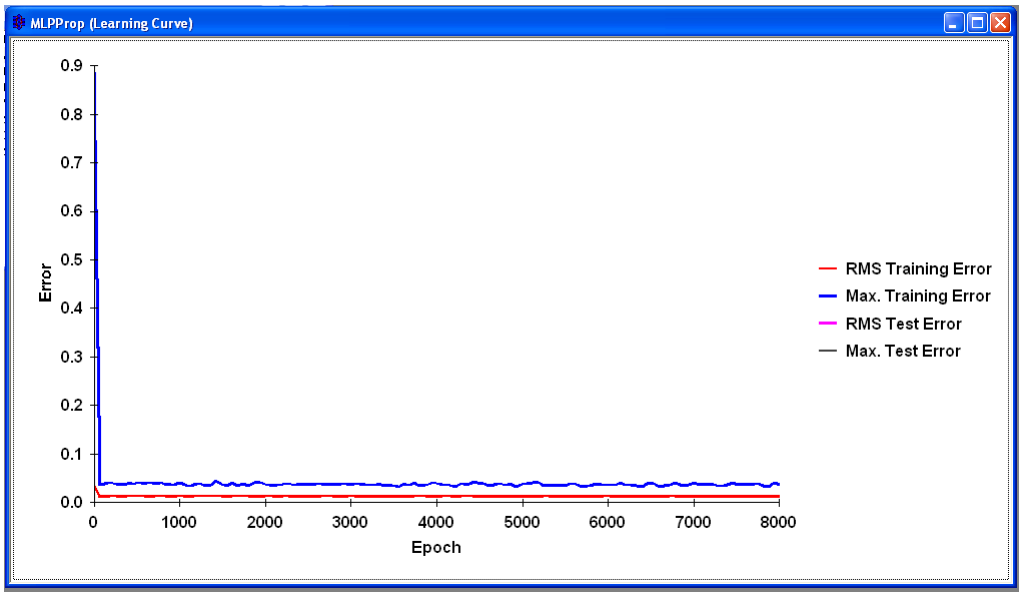

Figura 5.7: Curva de aprendizaje de la red neuronal aplicada sobre la BDM Enfermedad del corazón.

De acuerdo a la gráfica de curva de aprendizaje, el error máximo de entrenamiento y prueba para esta red neuronal es de 0.0375. Debido a que el error máximo de entrenamiento y prueba alcanzado es menor a 0.5, se garantiza una perfecta clasificación. Los resultados completos se muestran en la siguiente tabla.

Comparando la precisión alcanzada por CESAMO en la etapa de prueba (test), se observa que

Entrenamiento de red neuronal	Precisión
Entrenamiento	0.9876
Máximo error de entrenamiento	0.0375
Prueba	0.9874
Máximo error de prueba	0.0361

Tabla 5.4: Resultados del entrenamiento de la red neuronal para la BDM Enfermedad del corazón.

supera a todos los resultados que se mostraron inicialmente con otras técnicas (árbol de decisión, SVN, regresión logística, etc.).

Por último, se realizó una prueba aplicando la técnica one-hot encoding para codificar los atributos categóricos de la BDM. Posteriormente, se analizaron los datos con una red neuronal con la misma arquitectura descrita previamente. Los resultados del entrenamiento se ilustran a continuación.

Entrenamiento de red neuronal	Precisión
Entrenamiento	0.8958
Máximo error de entrenamiento	0.0375
Prueba	0.8458
Máximo error de prueba	0.0361

Tabla 5.5: Resultados del entrenamiento de la red neuronal en la BDM codificada mediante one-hot encoding.

Si se observa la curva de aprendizaje, se nota que la diferencia es muy grande. El máximo error de entrenamiento y prueba es de 0.57. Al obtener este resultado, no se puede garantizar que exista clasificación perfecta para la BDM.

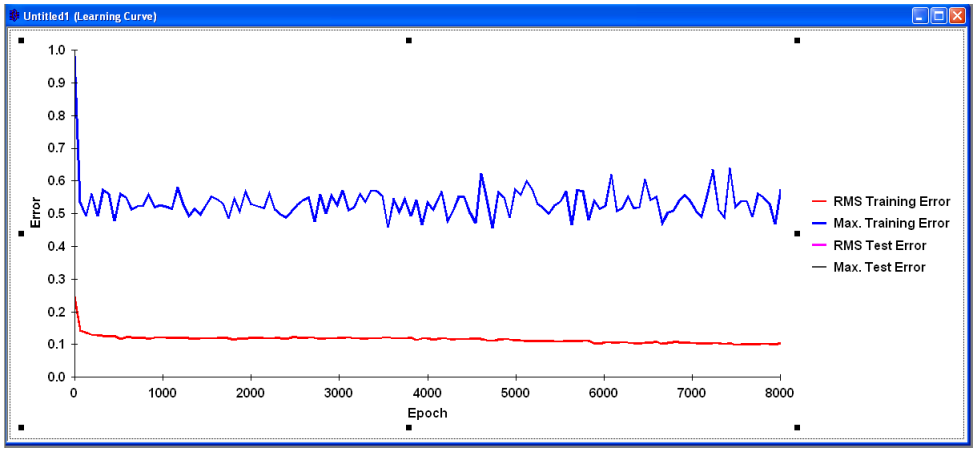

Figura 5.8: Curva de aprendizaje de la red neuronal en BDM codificada mediante one-hot encoding.

Los resultados de cada uno de los algoritmos aplicados sobre la BDM se muestran a continuación. Se observa que CESAMO obtuvo la mayor precisión para este problema de clasificación.

Algoritmo	Precisión
Árbol de decisión (Decision tree)	0.852459
K-ésimo vecino más cercano (K-Nearest Neighbour)	0.918033
Regresión logística (Logistic regression)	0.934426
SVM Radial (Radial SVM)	0.918033
SVM Lineal (Linear SVM)	0.918033
Bosques aleatorios (Random Forest)	0.8852
Naive-Bayes	0.8689
Potenciación de Gradiente (Gradient Boosting Machine)	0.76
Potenciación de Gradiente (Modelo XGBoost)	0.903225
Gradiente descendente aleatorio (Stochastic Gradient Descent)	0.818181
Cesamo	0.9874
One-hot encoding	0.845871

Tabla 5.6: Comparación de resultados reportados con algoritmos tradicionales y CESAMO.

5.1.2 **Aprendizaje no supervisado**

También se llevó a cabo un ejercicio de agrupamiento usando los mismos datos. En este caso, se entrenaron a varios mapas auto-organizados de Kohonen (SOM). Se realizó para 2,3,...,13 grupos.

La distancia media y máxima se reportan en la siguiente ilustración.

CLUSTERS	MEAN DISTANCE	MEAN DELTA	MAX DISTANCE	MAX DELTA
2	0.2151	0.019	0.6656	0.068
3	0.1964	0.009	0.5975	0.021
4	0.1871	0.008	0.5767	0.005
5	0.1786	0.008	0.5815	0.012
6	0.1706	0.004	0.5696	0.047
7	0.1667	0.005	0.5228	0.035
8	0.1614	0.003	0.4883	0.037
9	0.1583	0.004	0.5256	0.012
10	0.1546	0.004	0.5371	0.106
11	0.1505	0.002	0.4314	0.051
12	0.1485	0.003	0.4825	0.085
13	0.1457		0.3979	

Figura 5.9: Distancia media y máxima para los ejercicios de agrupación usando SOM's.

En donde, delta máxima se refiere al incremento relativo de la diferencia de error de aproximación máxima entre números consecutivos de grupos. Ahora, para determinar cuál de estos ejercicios contiene la "mejor" agrupación se recurre al criterio de Bezdek [36]. Este criterio sugiere que el número de agrupaciones se encuentra dentro de la región de mayor cambio. Graficando las diferencias mencionadas, se puede ver más claro el mayor cambio.

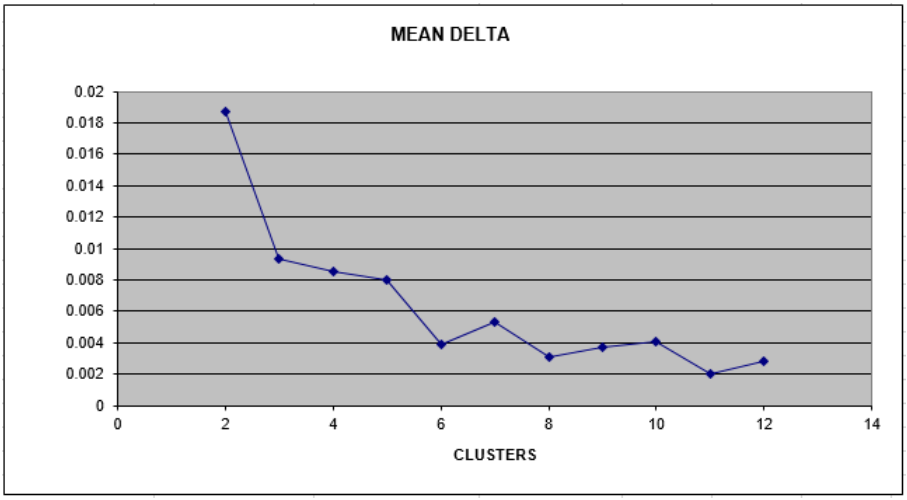

Figura 5.10: Determinación de números de grupos para la BDM Enfermedad del corazón.

A partir de la figura previa, se determina que 6 es el número de grupos. Debido a que la gráfica tiene un comportamiento decreciente antes del punto 6 y a partir de este punto existe un crecimiento hacia el punto 7. Por lo que, es notable una región de cambio en dicho punto. Cabe mencionar

que también pudo haber sido elegido 8 u 11 ya que también contienen regiones de cambio. Sin embargo, si eligiéramos algunas de éstas regiones, el cambio resultante sería que tendríamos más sub-grupos. Decidimos seleccionar 6 grupos para evitar la formación de estos sub-grupos.

Posteriormente, se procede a la caracterización de cada uno de los grupos. En la siguiente figura, se muestra el atributo sexo y la forma en la que se distribuye en los grupos.

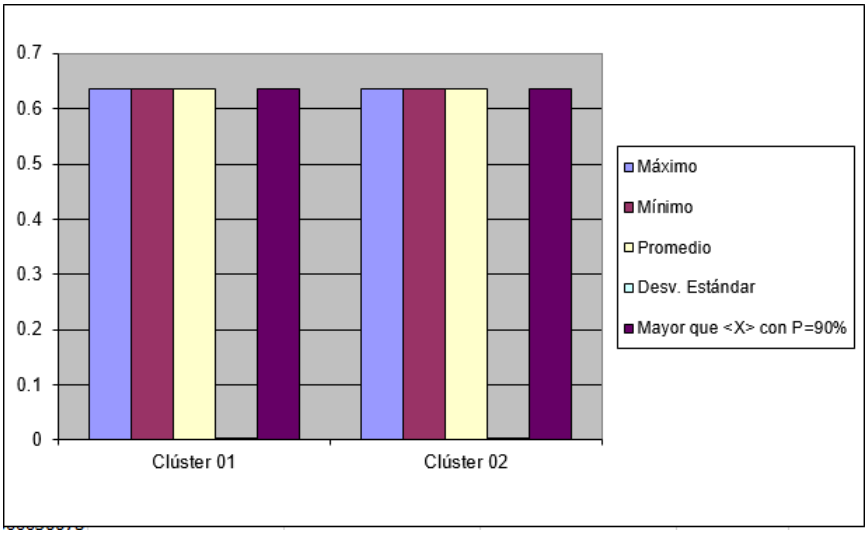

Figura 5.11: Caracterización de grupos para el atributo sexo.

Cabe mencionar que los atributos: sexo, dolor en el pecho, azúcar en sangre, etcétera, son categóricos. En este caso se eligió analizar sexo, sin embargo, se pudo haber elegido cualquier otro atributo categórico. Ya que lo que se pretende es mostrar que un atributo categórico, después de ser codificado numéricamente con CESAMO, puede ser caracterizado en grupos.

Se observa que solo hay 2 grupos (con una desviación estándar de 0), debido a que los demás grupos se encuentran vacíos. Con esto se obtiene un resultado interesante, ya que sugiere que debido a esta distribución en los grupos, la red neuronal es capaz de discernir sencillamente entre cuáles pacientes se encontraban enfermos y cuáles no. Resulta interesante ya que hay dos posibles clases para esta BDM y también dos grupos, por lo que, sugiere que cada clase se encuentra en un grupo particular.

Al analizar los grupos se pudo notar que cada uno de los grupos tenía tuplas de diferentes clases.

Es decir, el grupo 1 contiene tuplas tanto de pacientes enfermos y sanos, así como de pacientes de diferentes edades y de diferentes sexos. Un ejemplo de esto, se encuentra en la siguiente figura.

Figura 5.12: Atributo sexo distribuido en los grupos (cada grupo contiene instancias de ambos sexos).

Por último, se realizó esta misma agrupación pero a una BDM codificada mediante one-hot encoding. Los resultados se representan en la siguiente figura.

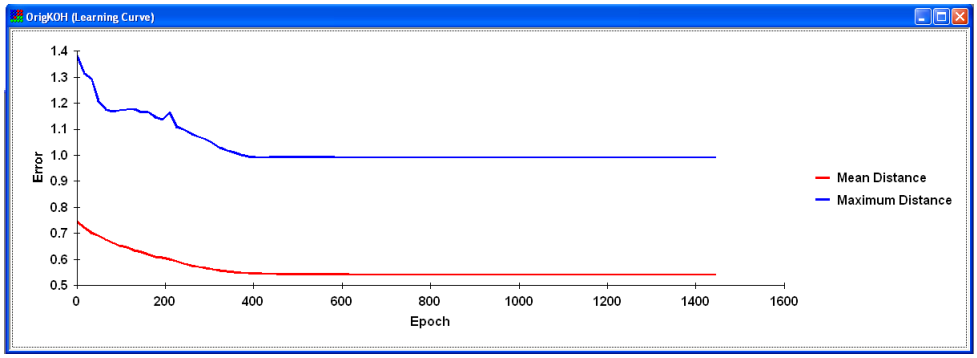

Figura 5.13: Curva de distancia media y máxima para BDM codificada con one-hot encoding.

Una vez que se obtuvieron estos resultados, con el objetivo de realizar una comparación, se elaboró una gráfica similar para la BDM que fue codificada por CESAMO.

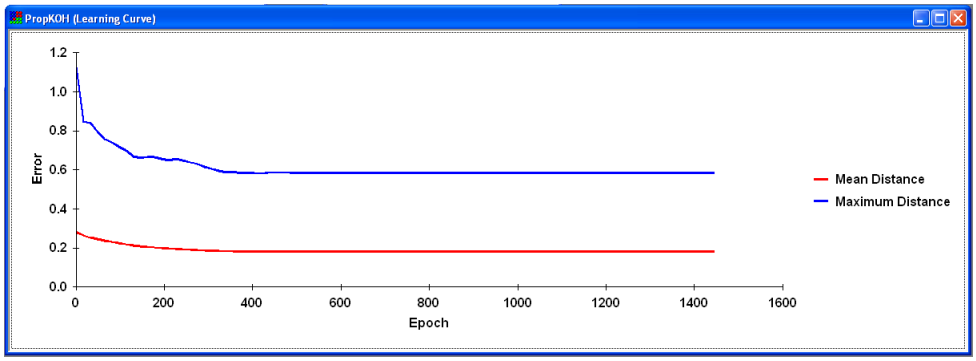

Figura 5.14: Curva de distancia media y máxima para BDM codificada con CESAMO.

La diferencia de distancia media y máxima entre las dos figuras previas es muy grande. El agrupamiento realizado a la BDM codificada con one-hot encoding tiene una distancia media de 0.5407. Mientras, que CESAMO tiene una distancia media de 0.1786, la cuál es mucho menor. Esto significa que los elementos que se encuentran en un grupo determinado, se encuentran menos *separados* entre sí. Se sabe que los mapas auto-organizados de Kohonen (SOM) tienen la característica de encontrar patrones inherentes en los datos. En las gráficas se puede notar que el comportamiento de los valores de distancia media y máxima es similar. Sugiriendo que la aplicación del SOM sobre la BD, detectó patrones similares después de la aplicación de cada uno de los algoritmos. Por lo que, se observa que al aplicar CESAMO, no hubo pérdida de patrones inherentes a la BDM Enfermedad del corazón.

5.2 Segundo caso: más tuplas

De <https://archive.ics.uci.edu/ml/datasets/abalone> se extrajo la base de datos mixta Abalone. En la BDM se tienen registros de un animal marino llamado ”Abalón”, del cual es difícil conocer la edad que tiene. La edad se encuentra relacionada con el número de anillos. De acuerdo a características como: longitud, peso, sexo, entre otros, se desea resolver este problema y conocer la edad de cada espécimen de Abalón. La BDM contiene 9 atributos: sexo (C), longitud (N), diámetro (N), altura (N), peso completo (N), peso en sacos (N), peso visceral (N), peso de la concha (N), anillos (N). Donde, (C) significa que es un atributo categórico y (N) atributo numérico. Son 4,177 tuplas

y 3 posibles instancias categóricas: femenino, masculino e indefinido para el atributo categórico sexo. Un pequeño segmento de la BDM se ilustra a continuación.

Sexo	Longitud	Diámetro	Altura	Peso completo	Peso en sacos	Peso visceral	Peso de la concha	Anillos
M	0.455	0.365	0.095	0.514	0.2245	0.101	0.15	15
M	0.35	0.265	0.09	0.2255	0.0995	0.0485	0.07	7
F	0.53	0.42	0.135	0.677	0.2565	0.1415	0.21	9
M	0.44	0.365	0.125	0.516	0.2155	0.114	0.155	10
I	0.33	0.255	0.08	0.205	0.0895	0.0395	0.055	7
I	0.425	0.3	0.095	0.3515	0.141	0.0775	0.12	8
F	0.53	0.415	0.15	0.7775	0.237	0.1415	0.33	20
F	0.545	0.425	0.125	0.768	0.294	0.1495	0.26	16
M	0.475	0.37	0.125	0.5095	0.2165	0.1125	0.165	9
F	0.55	0.44	0.15	0.8945	0.3145	0.151	0.32	19
F	0.525	0.38	0.14	0.6065	0.194	0.1475	0.21	14
M	0.43	0.35	0.11	0.406	0.1675	0.081	0.135	10

Figura 5.15: Muestra de la BDM Abalone.

CESAMO fue aplicado sobre esta BDM y en la siguiente imagen se ilustra un pequeño segmento de la BDM codificada.

Sexo	Longitud	Diámetro	Altura	Peso completo	Peso en sacos
0.75184022	0.55828221	0.56153846	0.0840708	0.181914705	0.150873656
0.75184022	0.42944785	0.40769231	0.07964602	0.079808883	0.06686828
0.7366921	0.65030675	0.64615385	0.11946903	0.23960361	0.172379032
0.75184022	0.5398773	0.56153846	0.11061947	0.182622545	0.144825269
0.75811824	0.40490798	0.39230769	0.07079646	0.07255353	0.060147849
0.75811824	0.52147239	0.46153846	0.0840708	0.124402761	0.094758065
0.7366921	0.65030675	0.63846154	0.13274336	0.275172536	0.159274194
0.7366921	0.66871166	0.65384615	0.11061947	0.271810299	0.197580645
0.75184022	0.58282209	0.56923077	0.11061947	0.180322067	0.145497312
0.7366921	0.67484663	0.67692308	0.13274336	0.316581136	0.211357527
0.7366921	0.64417178	0.58461539	0.12389381	0.214652274	0.130376344
0.75184022	0.52760736	0.53846154	0.09734513	0.143691382	0.112567204

Figura 5.16: Muestra de la BDM Abalone codificada.

Como se observa en la imagen anterior, ya no existen instancias categóricas. Las instancias del atributo sexo fueron reemplazadas de acuerdo a lo siguiente:

Los valores de los atributos restantes fueron escalados y se les asignó un valor que se encuentra en el rango entre $(0, 1]$.

Instancia categórica	Código numérico propuesto
Indefinido	0.75811824
Femenino	0.7366921
Masculino	0.751840221

Tabla 5.7: Codificación del atributo categórico *sexo* de la BDM Abalone.

5.2.1 Aprendizaje supervisado

El problema original es conocer el número de anillos que tiene el abalone de acuerdo a los atributos restantes. Se entrenó una red neuronal, para la cual la arquitectura fue definida de igual manera que el primer caso. La red está compuesta por 8 neuronas de entrada, una neurona en la capa oculta y una neurona en la capa de salida. Este problema de clasificación fue abordado con la técnica one-hot encoding para asignarle un valor numérico a las instancias del atributo sexo. Los resultados son los siguientes:

Entrenamiento de red neuronal	Precisión
Entrenamiento	0.9376
Máximo error de entrenamiento	0.5391
Prueba	0.9476
Máximo error de prueba	0.2832

Tabla 5.8: Resultados del entrenamiento de la red neuronal aplicando técnica one-hot encoding.

Una vez que se tiene la BDM Abalone codificada mediante CESAMO. De igual forma se entrenó una red neuronal con la misma arquitectura explicada anteriormente. Los resultados son los que se muestran a continuación:

A partir de esta BDM y en las posteriores se ha decidido comparar CESAMO solo con one-hot encoding, debido a que es un enfoque que ha sido ampliamente aplicado en proyectos de aprendizaje de máquina. Además, de la facilidad de aplicar este enfoque a las BDM.

Se puede observar que el error de entrenamiento y prueba disminuyeron con la ultima red neuronal entrenada, en comparación con la primera red neuronal. De hecho, el máximo error de prueba es

Entrenamiento de red neuronal	Precisión
Entrenamiento	0.9569
Máximo error de entrenamiento	0.5115
Prueba	0.9525
Máximo error de prueba	0.2629

Tabla 5.9: Resultados del entrenamiento de la red neuronal aplicando CESAMO.

más pequeño que el de entrenamiento, sugiriendo una buena generalización de la red. Entonces, la precisión de clasificación del modelo entrenado fue mejor cuando se aplicó CESAMO, como se observa en la siguiente tabla.

Algoritmo	Precisión
One-hot encoding	0.9476
Cesamo	0.9525

Tabla 5.10: Comparación de los resultados obtenidos para la BDM Abalone.

5.2.2 **Aprendizaje no supervisado**

También se llevó a cabo un ejercicio de agrupamiento usando los mismos datos. En este caso, se entrenaron varios mapas auto-organizados de Kohonen (SOM). Se realizó para 2,3,...,8 grupos. La distancia media y máxima se reportan en la siguiente ilustración.

GRUPOS	DISTANCIA MEDIA	DELTA MEDIA	DISTANCIA MÁXIMA	DELTA MÁXIMA
2	0.192	0.043	1.088	0.160
3	0.149	0.024	0.928	0.038
4	0.125	0.017	0.89	0.007
5	0.108	0.007	0.897	0.006
6	0.101	0.008	0.891	0.001
7	0.093	0.006	0.89	0.001
8	0.087		0.889	

Figura 5.17: Distancia media y máxima en los ejercicios de agrupación usando SOM's.

Para determinar cuál de estos ejercicios contiene la "mejor" agrupación se recurre al criterio de Bezdek como en el anterior caso de estudio. La siguiente gráfica ilustra la región de mayor cambio.

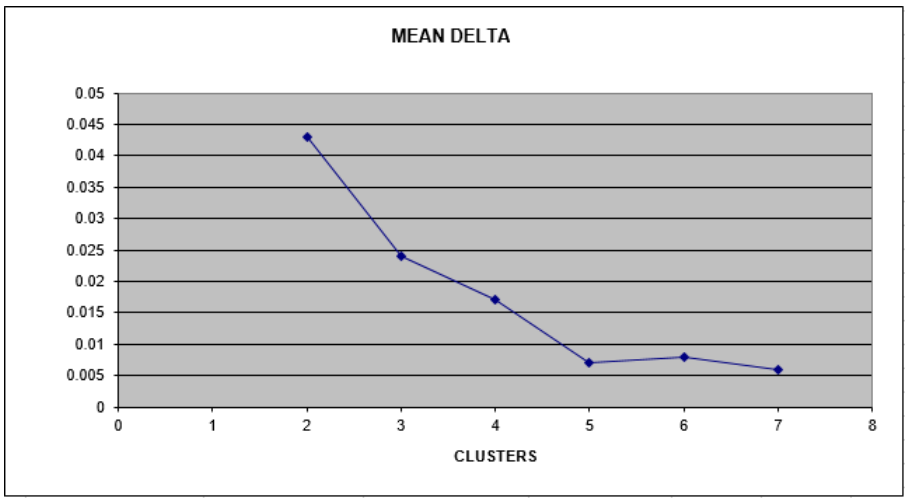

Figura 5.18: Determinación de números de grupos para la BDM Abalone.

Por lo que, se determina que 5 es el número de grupos. Posteriormente, se procede a la caracterización de cada uno de los grupos. Al caracterizar los grupos uno de los cinco se encontró sin elementos, por lo tanto se tienen 4 grupos a caracterizar. En la siguiente figura, se muestra el atributo ”altura de abalone” y la forma en la que se distribuye en los grupos.

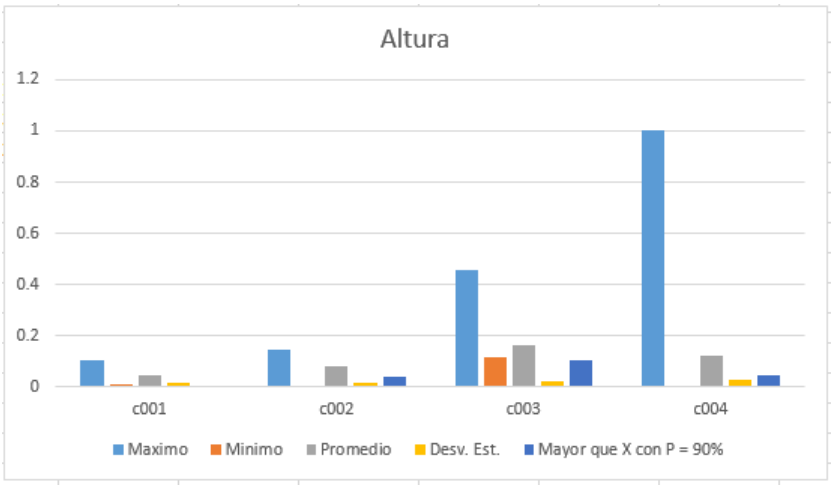

Figura 5.19: Atributo altura de abalone caracterizada por grupos.

Un resultado interesante sucede, ya que el valor máximo parece incrementar del primer al último grupo. En el primer grupo se nota un máximo de aproximadamente 0.10 y en el cuarto grupo un

máximo de 1.0. El agrupamiento fue capaz de diferenciar esta característica al definir los elementos pertenecientes a cada grupo.

Posteriormente, se realizó una gráfica similar para los atributos ”diámetro del abalone” y ”peso completo”.

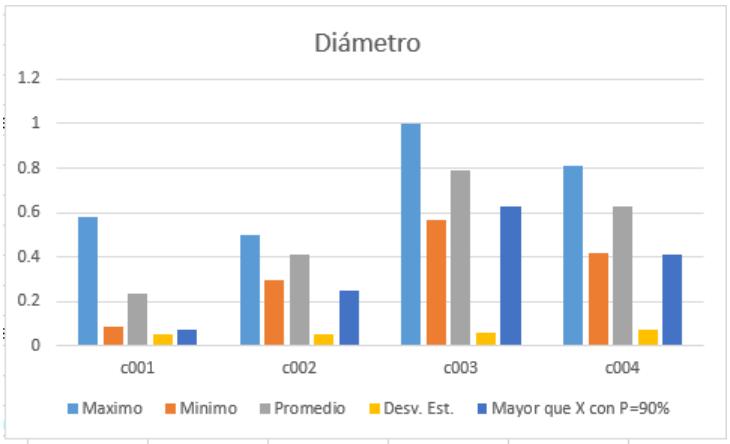

Figura 5.20: Atributo diámetro de abalone caracterizada por grupos.

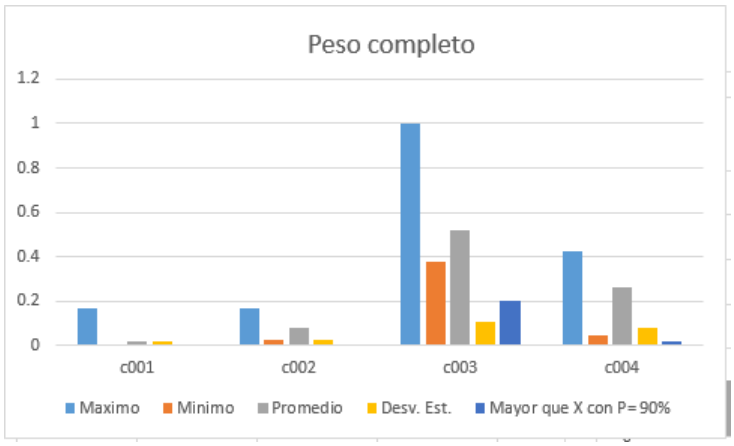

Figura 5.21: Atributo peso completo de abalone caracterizada por grupos.

Se puede notar en la caracterización de grupos que el atributo "diámetro del abalone", tiene un comportamiento diferente en sus valores promedio. El grupo 1 tiene un valor alrededor de 0.2, para el grupo 2 es de 0.4, para el grupo 3 es de 0.8 y para el grupo 4 es de 0.6, aproximadamente. Hay una diferencia de 0.2 entre ellos. De igual forma, para el atributo "peso completo de abalone",

el promedio se comporta de forma similar.

Por último, se realizó la agrupación con 5 grupos a la misma BDM pero codificada a través de la técnica one-hot encoding. Los resultados se ilustran en la siguiente imagen.

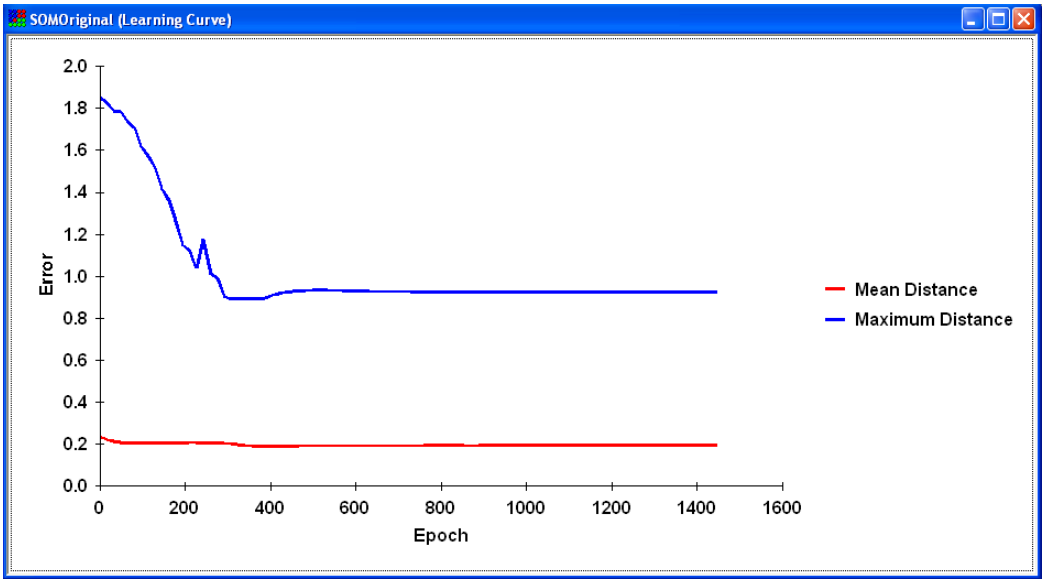

Figura 5.22: Curva de distancia media y máxima para BDM codificada con one-hot encoding.

Una vez obtenidos estos primeros resultados, con el objetivo de realizar una comparación, se elaboró una gráfica similar para la BDM que fue codificada por CESAMO.

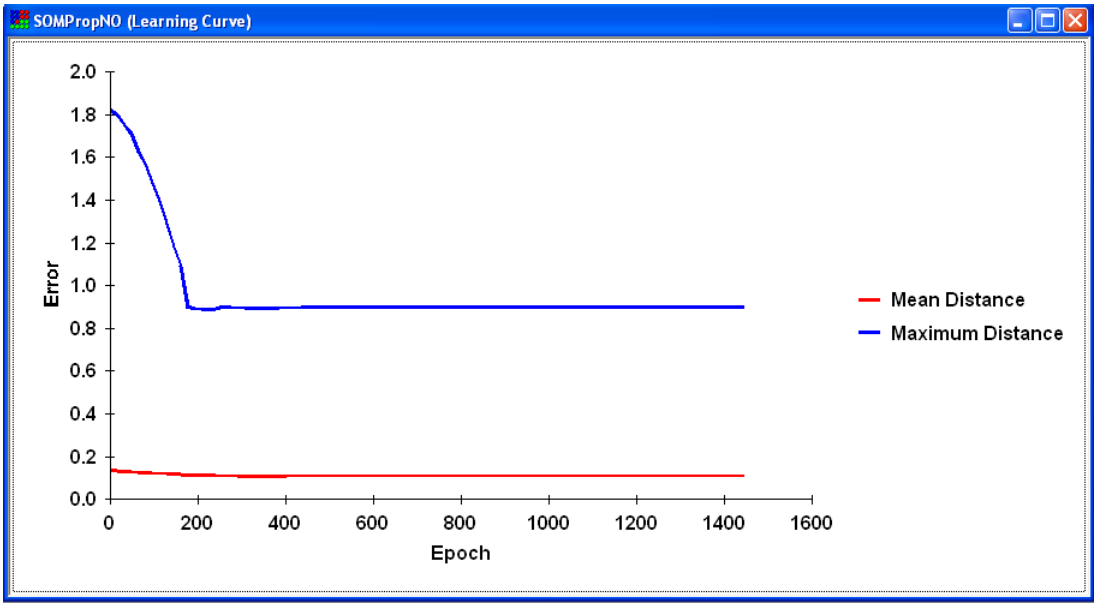

Figura 5.23: Curva de distancia media y máxima para BDM codificada con CESAMO.

Existe una diferencia en la distancia media y máxima entre las dos ilustraciones previas. El agrupamiento realizado a la BDM codificada con one-hot encoding tiene una distancia media de 0.19. Mientras, que con CESAMO tiene una distancia media de 0.10, la cuál es mucho menor. Esto significa que los elementos que se encuentran en un grupo determinado, se encuentran menos *separados* entre ellos dentro del grupo, es decir, son más similares entre sí y difieren en mayor medida con los miembros de un grupo diferente. Se sabe que los mapas auto-organizados de Kohonen tienen la característica de encontrar patrones inherentes en los datos. En las dos gráficas anteriores se puede notar que el comportamiento de los valores de distancia media y máxima es similar. Por lo que, se observa que al aplicar dicho algoritmo no hubo pérdida de patrones inherentes a la BDM Abalone.

5.3 Tercer caso: una prueba mayor

Se extrajo la BDM llamada ”Violencia con armas” de <https://www.kaggle.com/jameslko/gun-violence-data>. En dicha BDM se tienen datos de los crímenes que ocurrieron en Estados Unidos, desde Enero de 2013 hasta Marzo de 2018. Se tiene información del tipo de arma (pistola, 9 mm, etc.), número de personas lesionadas o fallecidas, fecha y lugar del crimen, entre otros. Los datos

de la BDM fueron registrados por varios años en distintos estados de Estados Unidos. Cada tupla representa las características de un crimen diferente. Lo que se necesita conocer es el distrito del estado en donde ocurrió, ya que los agentes policíacos buscan conocer las características específicas de los crímenes que suceden en cada distrito para poder preparar o reforzar una defensa personalizada para el distrito ante dichos crímenes.

El conjunto completo de atributos con los que cuenta la BDM son los siguientes.

Atributo	Descripción	Tipo
Identificador	Identificador del incidente	Numérico
Año del incidente	Año (2013,...,2018)	Numérico
Estado del incidente	Estado (Pensylvania, California, etc.)	Categórico
Dirección	Dirección del incidente (ciudad)	Categórico
Personas fallecidas	Número de personas fallecidas	Numérico
Personas lesionadas	Número de personas lesionadas	Numérico
URL incidente	Sitio web donde se describe el incidente	Categórica
Tipo de arma	Tipo de arma involucrada (22 mm, AR-15, etc.)	Categórica
Latitud	Coordenada de latitud	Numérico
Longitud	Coordenada de longitud	Numérico
Armas	Número de armas involucradas	Numérico
Edad del agresor	Grupos de edad del agresor (niños, jovenes, etc.)	Categórico
Estatus del agresor	Estatus (muerto, arrestado, etc.)	Categórico
Tipo de agresor	Tipo (sospechoso, víctima, etc.)	Categórico
Distrito	Distrito de estado donde ocurrió el incidente	Categórico

Tabla 5.11: Descripción de base de datos mixta Violencia con armas.

Se tiene un total de $239,678$ tuplas. Con un total de 126 instancias categóricas. Una pequeño muestra de la BDM se ilustra a continuación.

Identificador	Año	Estado	Ciudad	Dirección	Fallecidos	Lesionados
461105	01/01/2013	Pennsylvania	Mckeesport	1506 Versailles Avenue and Coursin Street	0	4
460726	01/01/2013	California	Hawthorne	13500 block of Cerise Avenue	1	3
478855	01/01/2013	Ohio	Lorain	1776 East 28th Street	1	3
478925	05/01/2013	Colorado	Aurora	16000 block of East Ithaca Place	4	0
478959	07/01/2013	North Carolina	Greensboro	307 Mourning Dove Terrace	2	2
478948	07/01/2013	Oklahoma	Tulsa	6000 block of South Owasso	4	0
479363	19/01/2013	New Mexico	Albuquerque	2806 Long Lane	5	0
479374	21/01/2013	Louisiana	New Orleans	LaSalle Street and Martin Luther King Jr. Boule	0	5
479389	21/01/2013	California	Brentwood	1100 block of Breton Drive	0	4
492151	23/01/2013	Maryland	Baltimore	1500 block of W. Fayette St.	1	6
491674	23/01/2013	Tennessee	Chattanooga	1501 Dodds Ave	1	3

Figura 5.24: Muestra de la BDM Violencia con armas.

Debido a que es de interés los atributos que *ayuden* a identificar el distrito en donde ocurrió el incidente. Entonces, previamente a la aplicación de CESAMO, se realizó pre-procesamiento de los datos. Por ejemplo, el atributo ”identificador” solo contiene el número de incidente, pero no provee información útil para la tarea de clasificación. De igual forma el atributo ”URL del incidente” no contiene información de utilidad. Una vez realizado el pre-procesamiento de datos, se procedió a la aplicación de CESAMO a la BDM. En la siguiente imagen se ilustra un pequeño segmento de la BDM codificada. El orden de las columnas es el mismo que se ilustró en la tabla 5.11, iniciando con Año del incidente.

0.99900744	0.7520194	0.000001	0.21052632	0.67901061	0.56587158	0.48267631	0.000001	0.7270262	0.72120342	0.73598434	0.000001
0.99900744	0.39760329	0.09090909	0.15789474	0.67901061	0.47558138	0.71524504	0.000001	0.7270262	0.72120342	0.74605023	0.37234043
0.99900744	0.61752338	0.09090909	0.15789474	0.67901061	0.58128249	0.49646829	0.00714286	0.7270262	0.72120342	0.73598434	0.13829787
0.99900744	0.71462796	0.36363636	0.000001	0.67901061	0.55612545	0.63345906	0.000001	0.7270262	0.72087562	0.74605023	0.29787234
0.99900744	0.59804881	0.18181818	0.10526316	0.57628626	0.506507	0.48328679	0.00714286	0.70048986	0.72087562	0.70076183	0.28723404
0.99900744	0.20200371	0.36363636	0.000001	0.67901061	0.50828119	0.58011653	0.000001	0.7270262	0.72087562	0.73598434	0.11702128
0.99900744	0.43401742	0.45454545	0.000001	0.40992189	0.49058977	0.64502792	0.00714286	0.73050398	0.72120342	0.74605023	0.14893617
0.99900744	0.55423534	0.000001	0.26315789	0.67901061	0.41996435	0.54449602	0.000001	0.73050398	0.72120342	0.70076183	0.05319149
0.99900744	0.39760329	0.000001	0.21052632	0.67901061	0.53247611	0.73570513	0.000001	0.70048986	0.72120342	0.70076183	0.07446809
0.99900744	0.89626945	0.09090909	0.31578947	0.67901061	0.55104972	0.46324557	0.000001	0.70048986	0.72120342	0.74605023	0.46808511
0.99900744	0.92050298	0.09090909	0.15789474	0.67901061	0.49119286	0.51539917	0.00357143	0.7270262	0.72120342	0.74605023	0.10638298
0.99900744	0.55316845	0.09090909	0.15789474	0.67901061	0.54287021	0.54549817	0.00357143	0.7270262	0.72120342	0.74605023	0.04255319
0.99900744	0.55423534	0.18181818	0.15789474	0.27285936	0.41909619	0.55320894	0.00357143	0.7270262	0.72120342	0.74605023	0.22340426
0.99900744	0.75298799	0.000001	0.26315789	0.57628626	0.54555043	0.46524322	0.00357143	0.7270262	0.72087562	0.70076183	0.000001

Figura 5.25: Muestra de la BDM codificada mediante CESAMO.

Como se observa en la imagen anterior, ya no existen instancias categóricas. Las instancias de cada atributo categórico fueron reemplazadas por códigos numéricos.

Los valores del resto de los atributos fueron escalados y se les asignó un valor que se encuentra en el rango entre $(0, 1]$.

5.3.1 **Aprendizaje supervisado**

El problema original es conocer el distrito en donde se sucitó el incidente, de acuerdo a los atributos restantes. Este problema de clasificación fue abordado con la técnica one-hot encoding para asignarle un valor numérico a las instancias de cada atributo categórico. Se entrenó una red neuronal con 13 neuronas de entrada, 7 neuronas en la capa oculta con función de activación sigmoidal, y una neurona en la capa de salida. Los resultados son los siguientes:

Entrenamiento de red neuronal	Precisión
Entrenamiento	0.7890
Máximo error de entrenamiento	0.8931
Prueba	0.7799
Máximo error de prueba	0.9315

Tabla 5.12: Resultados aplicando técnica one-hot encoding en BDM Violencia con armas.

Una vez que se tiene la BDM Violencia con armas codificada mediante CESAMO. Se entrenó una red neuronal con la misma arquitectura explicada anteriormente. Los resultados son los que se muestran a continuación:

Entrenamiento de red neuronal	Precisión
Entrenamiento	0.8880
Máximo error de entrenamiento	0.5423
Prueba	0.8902
Máximo error de prueba	0.5735

Tabla 5.13: Resultados aplicando CESAMO en BDM Violencia con armas.

Se puede notar que los errores de entrenamiento y de prueba disminuyeron, por lo que la precisión de clasificación del modelo entrenado fue mejor después de aplicar CESAMO. Además, se observa que la red neuronal no se sobre-ajusto y tampoco se sub-ajusto a los datos de entrenamiento. Por lo que, la generalización realizada por la red neuronal fue adecuada en los datos de prueba. A continuación se presenta una tabla comparativa con los resultados obtenidos.

Algoritmo	Precisión
One-hot encoding	0.7799
Cesamo	0.8902

Tabla 5.14: Comparación de los resultados obtenidos para la BDM Violencia con armas.

5.3.2 Aprendizaje no supervisado

De igual forma que en el primer caso de estudio, se llevó a cabo un ejercicio de agrupamiento usando la BDM. En este caso, se entrenaron a varios mapas auto-organizados de Kohonen (SOM). Se realizó para 2,3,...,10 grupos. La distancia media y máxima se reportan en la siguiente ilustración.

GRUPOS	DISTANCIA MEDIA	DELTA MEDIA	DISTANCIA MÁXIMA	DELTA MÁXIMA
2	0.1906	0.036	1.0543	0.002
3	0.1543	0.014	1.0525	0.031
4	0.1404	0.003	1.0214	0.000
5	0.1373	0.008	1.0215	0.000
6	0.1294	0.012	1.0217	0.000
7	0.1172	0.008	1.0215	0.001
8	0.1094	0.003	1.0221	0.013
9	0.1067	0.005	1.0092	0.000
10	0.1021		1.0092	

Figura 5.26: Distancia media y máxima en los ejercicios de agrupación usando SOM's.

Para determinar cuál de estos ejercicios contiene la "mejor" agrupación se recurre al criterio de Bezdek como en los anteriores casos de estudio.

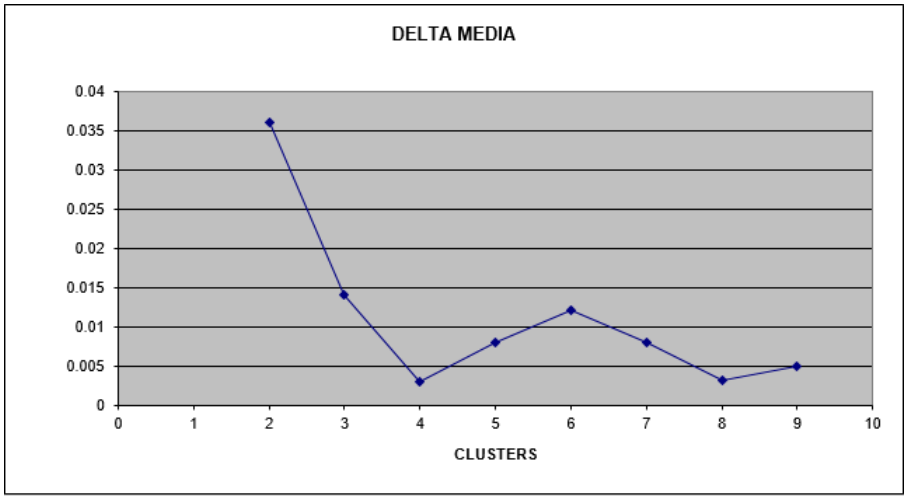

Figura 5.27: Determinación de números de grupos para la BDM Violencia con armas.

De acuerdo a la ilustración anterior, se determina que 4 es en donde se encuentra la mayor región de cambio, por lo que, es el número de grupos elegido.

Posteriormente, se procede a la caracterización de cada uno de los grupos. En la siguiente figura, se muestra el atributo ”personas fallecidas” y la forma en la que se distribuye en los grupos.

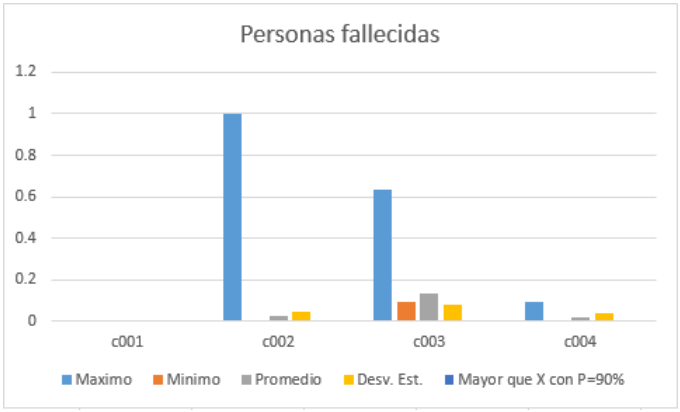

Figura 5.28: Atributo *personas fallecidas* caracterizada por grupos.

Analizando la figura anterior, es posible observar que el máximo de personas fallecidas en cada grupo es diferente. Para el grupo 1, de hecho, tiene un valor de 0 por lo que no aparece en la gráfica. El máximo del grupo 2 es todo lo contrario, alcanza el valor de 1.0 y en los dos restantes grupos también tiene un valor diferente. El agrupamiento discernió entre los crímenes a mano armada en donde si fallecieron personas, de los crímenes donde nadie falleció.

Se realizó el mismo procedimiento para el atributo "tipo de arma". En la siguiente figura se ilustran los resultados.

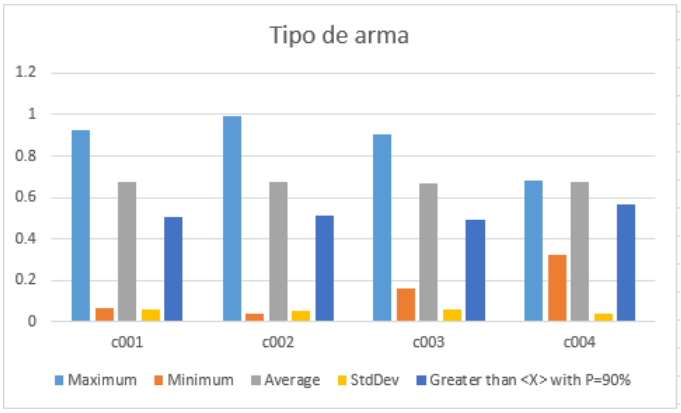

Figura 5.29: Atributo *tipo de arma* caracterizada por grupos.

Observando los siguientes valores: máximo y mínimo, cada grupo contiene diferentes pares de valores. Por lo que, el tipo de arma incriminada también influyó al realizar agrupamiento de elementos. Es decir, si el calibre del arma era mayor o menor, por ejemplificar algo, influye en el tipo de crimen suscitado y el algoritmo de agrupamiento hace notarlo.

Por último, se realizó la agrupación con 4 grupos a la misma BDM pero codificada a través de la técnica one-hot encoding. Los resultados se ilustran en la siguiente imagen.

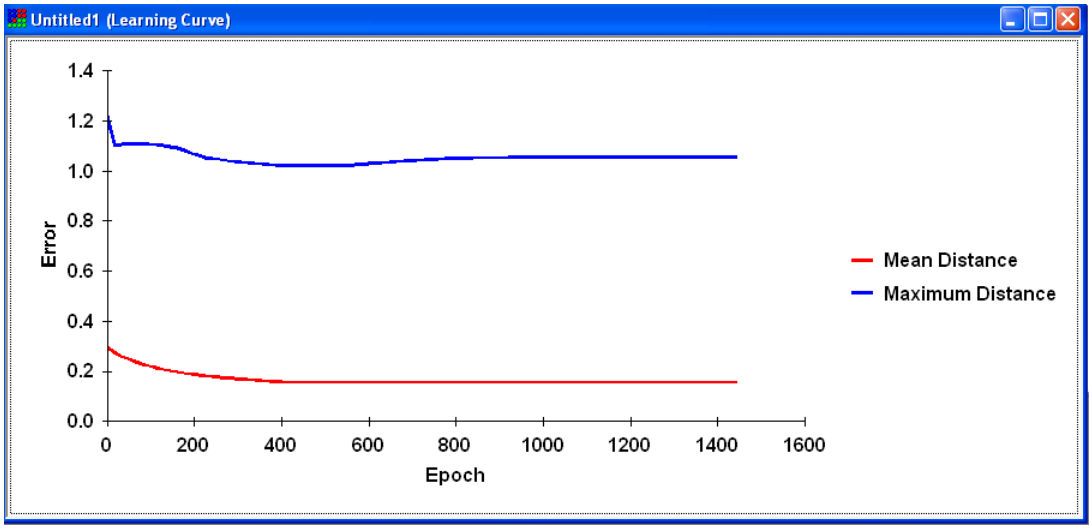

Figura 5.30: Curva de distancia media y máxima para BDM codificada con one-hot encoding.

Una vez obtenidos estos primeros resultados, con el objetivo de realizar una comparación, se elaboró una gráfica similar para la BDM codificada con CESAMO.

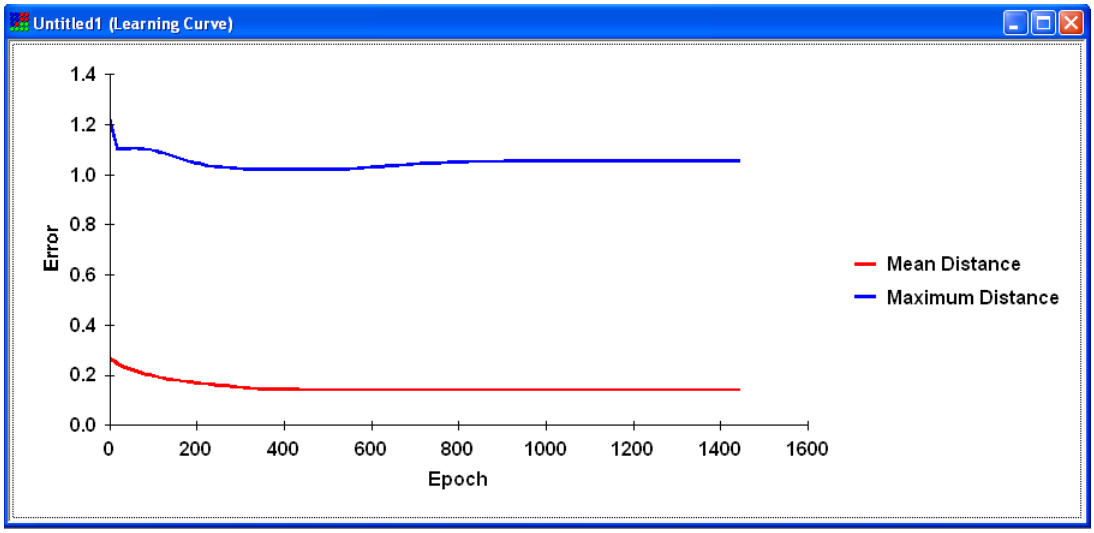

Figura 5.31: Curva de distancia media y máxima para BDM codificada con CESAMO.

Existe una diferencia en la distancia media y máxima entre las dos ilustraciones previas. Sin embargo, dicha diferencia es mínima. En las gráficas se puede notar que el comportamiento de los valores de distancia media y máxima es similar. Lo que significa que, al aplicar CESAMO no hubo pérdida de patrones inherentes a la BDM Violencia con armas.

5.4 Cuarto caso: otro caso de estudio

De <https://www.kaggle.com/fertilityRate> se extrajo la base de datos mixta llamada ”Tasa de fertilidad”. Los datos de la BDM fueron registrados por varios años en clínicas médicas de varios países del mundo (Eslovenia, Ucrania, Korea, etc.). En donde cada tupla representa el caso de un paciente que tiene entre 15 y 49 años de edad. Entonces, de acuerdo a los atributos tales como: tasa de reproducción de pacientes entre 15 y 19 años, entre otros, se desea conocer la tasa bruta de reproducción de los pacientes. La BDM contiene 13 atributos que se describen a continuación.

En la base de datos no hay tuplas con valores faltantes. Se cuenta con un total de $3,100$ tuplas. A continuación se muestra un pequeño segmento de la BDM.

Atributo	Descripción	Tipo
Identificador de país	Número que identifica al país	Numérico
Código de país	Código del país (KS para Korea, por ejemplo)	Categórico
Nombre de país	Nombre completo del país	Categórico
Año	Año del estudio	Numérico
Fertilidad 15 – 19	Tasa de fertilidad en pacientes de 15 a 19 años	Numérico
Fertilidad 20 – 24	Tasa de fertilidad en pacientes de 20 a 24 años	Numérico
Fertilidad 25 – 29	Tasa de fertilidad en pacientes de 25 a 29 años	Numérico
Fertilidad 30 – 34	Tasa de fertilidad en pacientes de 30 a 34 años	Numérico
Fertilidad 35 – 39	Tasa de fertilidad en pacientes de 35 a 39 años	Numérico
Fertilidad 40 – 44	Tasa de fertilidad en pacientes de 40 a 44 años	Numérico
Fertilidad 45 – 49	Tasa de fertilidad en pacientes de 45 a 49 años	Numérico
Fertilidad Total	Tasa total de fertilidad	Numérico
Tasa de reproducción	Tasa bruta de reproducción	Numérico

Tabla 5.15: Descripción de base de datos mixta Tasa de fertilidad.

Identificador	Código	Nombre	Año	Fertilidad 15-19	Fertilidad 20-24	Fertilidad 25-29	Fertilidad 30-34	Fertilidad 35-39	Fertilidad 40-44	Fertilidad 45-49	Fertilidad Total	Tasa de reproducción
1	SI	Slovenia	1989	7.3	54.1	114	100.9	40.3	10.3	3.2	1.6475	0.8081
2	SI	Slovenia	1991	4.9	44.2	110.8	104.2	43	9.3	1.3	1.5951	0.7824
3	SI	Slovenia	1992	5	44.8	111.1	103.9	42.9	9.3	1.5	1.5988	0.7842
4	SI	Slovenia	1993	5.1	45.5	111.4	103.8	42.7	9.3	1.6	1.6026	0.7861
5	SI	Slovenia	1994	5.2	45.9	111.5	103.7	42.2	9.4	1.8	1.6063	0.7878
6	SI	Slovenia	1995	5.4	46.4	111.7	103.6	41.7	9.5	1.9	1.6101	0.7898
7	SI	Slovenia	1996	5.7	46.9	111.9	103.3	41.3	9.6	2	1.6138	0.7916
8	SI	Slovenia	1997	5.9	47.3	112.2	102.8	41.2	9.7	2.1	1.6176	0.7934
9	SI	Slovenia	1998	6.1	47.8	112.4	102.5	41.2	9.7	2.3	1.6213	0.7952
10	SI	Slovenia	1999	6.3	48.6	112.5	102.3	41.2	9.7	2.4	1.6251	0.7971
11	SI	Slovenia	2000	6.4	49.6	112.6	102.1	41.2	9.8	2.5	1.6288	0.7989
12	SI	Slovenia	2001	6.6	50.8	112.8	101.8	41	9.8	2.7	1.6326	0.8008
13	SI	Slovenia	2002	6.7	51.8	112.9	101.5	40.7	9.9	2.9	1.6363	0.8026

Figura 5.32: Pequeño segmento de la BDM Tasa de fertilidad.

Debido a que es de interés los atributos que provean información útil para determinar con precisión la tasa de reproducción bruta, entonces previamente a la aplicación de CESAMO, se realiza un pre-procesamiento de los datos. Por ejemplo, el atributo ”identificador” solo contiene el número de incidente, pero no contiene información útil para la tarea de clasificación. De igual forma el atributo ”código del país” es innecesario, ya que con el nombre de país es suficiente y dichos atributos son redundantes entre sí. Al final, solo quedan 11 atributos con los cuales se trabajará en este caso de estudio.

Previo a la aplicación del algoritmo, se contó el número de instancias categóricas posibles en la BDM y se tiene la suma de 66. Es decir, al finalizar la aplicación del algoritmo se deben tener 66

códigos numéricos que reemplazarán las instancias categóricas. En la siguiente imagen se ilustra un pequeño segmento de la BDM codificada.

0.59229587	0.99317073	0.04154809	0.17513758	0.37401575	0.35678925	0.21300211	0.09961315	0.06765328	0.25152672	0.80148137
0.59229587	0.98634146	0.02788845	0.14308838	0.36351706	0.36845827	0.22727273	0.08994197	0.02748414	0.24352672	0.80148137
0.59229587	0.98682927	0.0284576	0.14503075	0.36450131	0.36739745	0.22674419	0.08994197	0.03171247	0.2440916	0.80140421
0.59229587	0.98731707	0.02902675	0.14729686	0.36548556	0.36704385	0.2256871	0.08994197	0.03382664	0.24467176	0.80148137
0.59229587	0.98780488	0.0295959	0.14859178	0.36581365	0.36669024	0.2230444	0.09090909	0.03805497	0.24523664	0.80155852
0.59229587	0.98829268	0.03073421	0.15021042	0.36646982	0.36633663	0.22040169	0.09187621	0.04016913	0.24581679	0.80140421
0.59229587	0.98878049	0.03244166	0.15182907	0.36712598	0.36527581	0.21828753	0.09284333	0.0422833	0.24638168	0.80140421
0.59229587	0.98926829	0.03357997	0.15312399	0.36811024	0.36350778	0.21775899	0.09381044	0.04439746	0.24696183	0.80140421
0.59229587	0.9897561	0.03471827	0.15474264	0.3687664	0.36244696	0.21775899	0.09381044	0.04862579	0.24752672	0.80140421
0.59229587	0.9902439	0.03585657	0.15733247	0.36909449	0.36173975	0.21775899	0.09381044	0.05073996	0.24810687	0.80140421
0.59229587	0.99073171	0.03642573	0.16056976	0.36942257	0.36103253	0.21775899	0.09477756	0.05285412	0.24867176	0.80148137
0.59229587	0.99121951	0.03756403	0.16445452	0.37007874	0.35997171	0.2167019	0.09477756	0.05708245	0.24925191	0.80148137
0.59229587	0.99170732	0.03813318	0.16769181	0.37040682	0.35891089	0.21511628	0.09574468	0.06131078	0.24981679	0.80140421
0.59229587	0.99219512	0.03927149	0.17060537	0.37073491	0.35785007	0.2140592	0.0967118	0.06342495	0.25038168	0.80148137
0.59229587	0.99268293	0.04040979	0.17319521	0.37237533	0.35714286	0.21353066	0.09767892	0.06553911	0.25096183	0.80148137

Figura 5.33: Pequeña muestra de la BDM codificada por CESAMO.

Como se observa en la imagen anterior, las instancias de cada atributo categórico fueron reemplazados por códigos numéricos.

Debido a que son demasiadas instancias categóricas, a continuación se muestran las primeras 20 y el código numérico que las reemplazó.

Instancia categórica	Código numérico propuesto
Eslovenia	0.592295875
Latvia	0.695261861
Mongolia	0.457825899
San Bartolomé	0.519936649
Ucrania	0.683590102
Chipre	0.980539074
Nauru	0.927375862
Curazao	0.361770932
Korea del Sur	0.162371567
Moldavia	0.490675297
Mónaco	0.472077231
Tuvalú	0.554034939
Sahara Occidental	0.407533143
Armenia	0.087261498
Wallis y Futuna	0.710654012
Nueva Zelanda	0.728382232
Islandia	0.866822812
República Dominicana	0.955029172
Isla de San Martín	0.397527563
Hungría	0.131333474

Tabla 5.16: Códigos numéricos propuestos para las instancias categóricas de la BDM Tasa de fertilidad.

Los valores del resto de los atributos fueron escalados y se les asignó un valor que se encuentra en el rango entre $(0, 1]$.

5.4.1 **Aprendizaje supervisado**

El problema original es conocer la tasa de reproducción, de acuerdo a los atributos restantes. Este problema de clasificación fue abordado con la técnica one-hot encoding para asignarle un valor numérico a las instancias de cada atributo categórico. Se entrenó una red neuronal con 10 neuronas de entrada, 3 neuronas en la capa oculta con función de activación tangencial hiperbólica, y una neurona en la capa de salida. Los resultados son los siguientes:

Entrenamiento de red neuronal	Precisión
Entrenamiento	0.6389
Máximo error de entrenamiento	0.7643
Prueba	0.6093
Máximo error de prueba	0.8353

Tabla 5.17: Resultados de red neuronal aplicando one-hot encoding para codificar la BDM Tasa de fertilidad.

Posteriormente, fue utilizada la BDM Tasa de fertilidad codificada mediante CESAMO. Se entrenó una red neuronal con la misma arquitectura explicada anteriormente. Los resultados son los que se muestran a continuación:

Entrenamiento de red neuronal	Precisión
Entrenamiento	0.9628
Máximo error de entrenamiento	0.4416
Prueba	0.9617
Máximo error de prueba	0.4635

Tabla 5.18: Resultados de red neuronal aplicando CESAMO para codificar la BDM Tasa de fertilidad.

Se puede notar que los errores de entrenamiento y de prueba disminuyeron significativamente. La precisión de clasificación del modelo entrenado fue mejor cuando CESAMO fue aplicado a la BDM. Además, se observa que la red neuronal no se sobre-ajusto y tampoco se sub-ajusto a los datos de entrenamiento.

Además, el máximo error tanto en entrenamiento y en prueba, fue menor a 0.5, lo cual es un error bastante pequeño en comparación con los máximos errores obtenidos al entrenar la red neuronal con one-hot encoding como técnica de codificación. A continuación se presenta una tabla comparativa con los resultados obtenidos.

Algoritmo	Precisión
One-hot encoding	0.6093
Cesamo	0.9617

Tabla 5.19: Comparación de los resultados obtenidos para la BDM Tasa de fertilidad.

5.4.2 Aprendizaje no supervisado

De igual forma que en los casos de estudios previos, se llevó a cabo un ejercicio de agrupamiento usando la BDM. En este caso, se entrenaron a varios mapas auto-organizados de Kohonen (SOM). Se realizó para 2,3,...,8 grupos. La distancia media y máxima se reportan en la siguiente ilustración.

GRUPOS	DISTANCIA MEDIA	DELTA MEDIA	DISTANCIA MÁXIMA	DELTA MÁXIMA
2	0.3057	0.049	1.295	0.055
3	0.2566	0.024	1.2401	0.019
4	0.2324	0.011	1.2215	0.541
5	0.2211	0.013	0.6806	0.013
6	0.2077	0.004	0.6677	0.030
7	0.2116	0.019	0.6381	0.013
8	0.193		0.6512	

Figura 5.34: Distancia media y máxima en los ejercicios de agrupación usando SOM's.

Para determinar cuál de estos ejercicios contiene la "mejor" agrupación se recurre al criterio de Bezdek como en el anterior caso de estudio. En la siguiente figura se puede observar la gráfica que servirá para determinar el número de grupos.

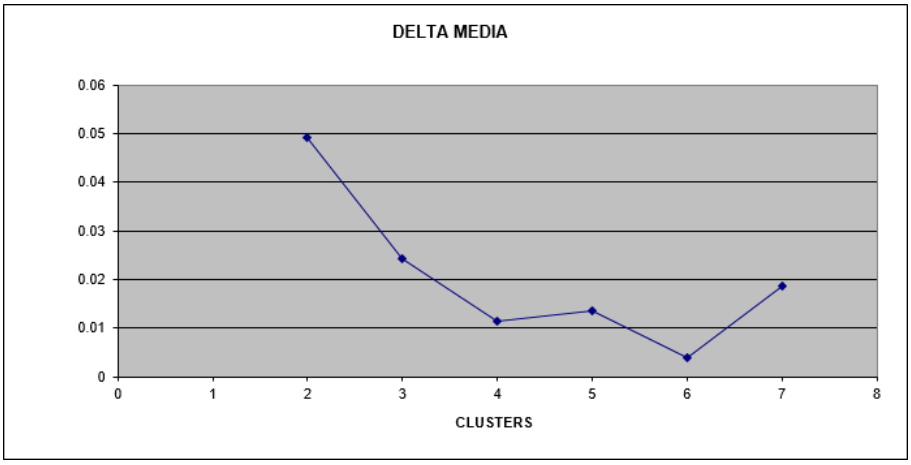

Figura 5.35: Determinación de números de grupos para la BDM Tasa de fertilidad.

De acuerdo a la gráfica anterior, se nota un cambio en el punto 4 y 6, sin embargo, debido a que se nota un cambio más pronunciado en el punto de 6 grupos. Se determina que 6 es el número de grupos. Debido a que la gráfica tiene un comportamiento decreciente entre el punto 5 y 6, entonces, a partir del punto 6 existe un crecimiento hacia el punto 7. Por lo que, es notable una región de cambio en dicho punto.

Posteriormente, se procede a la caracterización de cada uno de los grupos. En la siguiente figura, se muestra el atributo "tasa de fertilidad en pacientes entre 15 y 19 añosz la forma en la que se distribuye en los grupos.

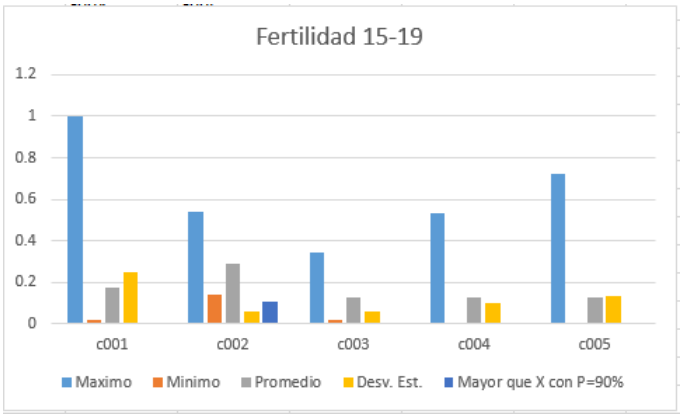

Figura 5.36: Atributo *fertilidad de* $15 - 19$ años caracterizada por grupos.

Se observa que se formaron 5 grupos ya que uno de los 6 iniciales resultó no tener elementos. En la figura anterior existe una tendencia en los máximos de cada grupo. Primero inicia con un valor mayor hasta llegar al tercer grupo donde alcanza su mínimo y después tiene una tendencia incremental. Esto parece indicar que la fertilidad entre 15 y 19 años depende de varios aspectos y por eso los grupos tienen un comportamiento diferente en cada uno de ellos.

Se realizó el mismo ejercicio para el atributo de tasa de fertilidad en pacientes de $30 - 34$ años.

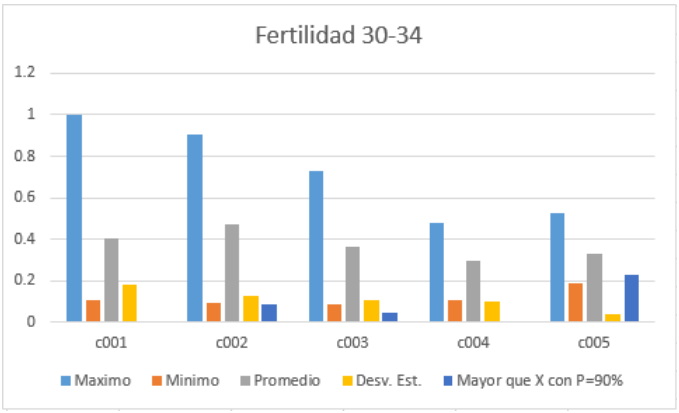

Figura 5.37: Atributo *fertilidad de* $30-34$ años caracterizada por grupos.

Otra vez, los máximos tienen una tendencia decremental del primer hasta el cuarto grupo. Los promedios tienen una tendencia similar. Por lo que en una edad más avanzada resulta mas sencillo agrupar valores máximos. Un grupo específico se formó al realizar el agrupamiento, esto se nota en el grupo 4, que contiene el menor valor para máximo y para promedio.

Por último, se realizó la agrupación con 6 grupos a la misma BDM pero codificada a través de la técnica one-hot encoding. Los resultados se ilustran en la siguiente imagen.

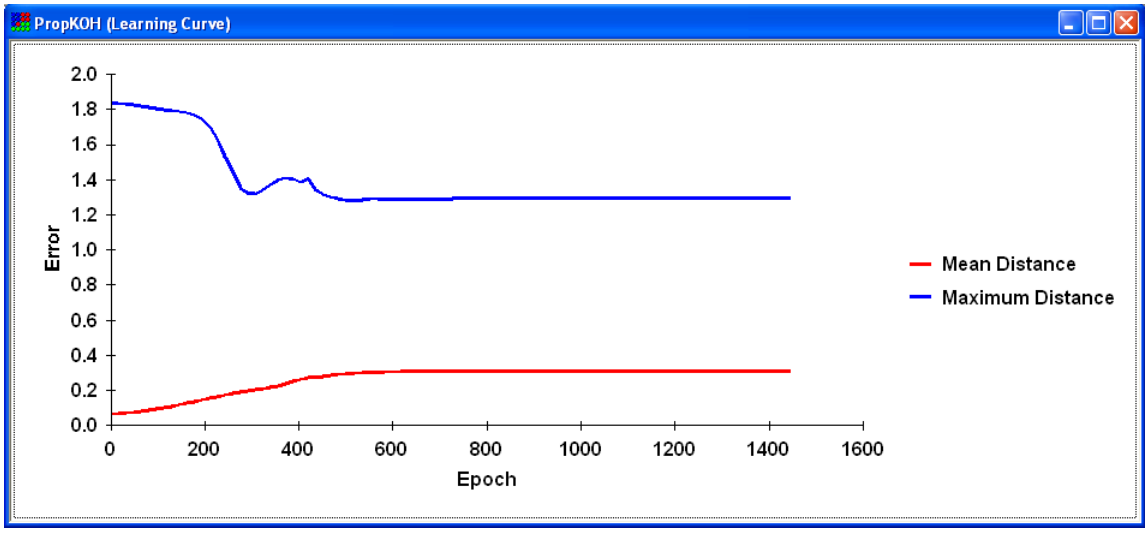

Figura 5.38: Curva de distancia media y máxima para BDM codificada con one-hot encoding.

Una vez obtenidos estos primeros resultados, con el objetivo de realizar una comparación, se ela-

boró una gráfica similar para la BDM que fue codificada por CESAMO.

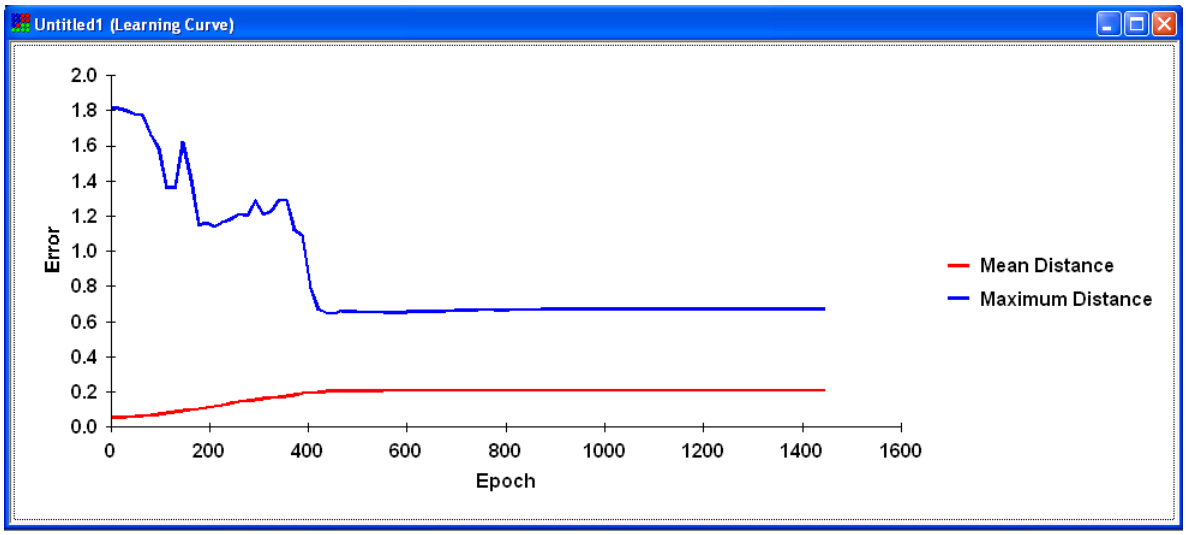

Figura 5.39: Curva de distancia media y máxima para BDM codificada con CESAMO.

Existe una diferencia en la distancia media y máxima entre las dos ilustraciones previas. El agrupamiento realizado a la BDM codificada con one-hot encoding tiene una distancia media de 0.226. Mientras que con CESAMO tiene una distancia media de 0.188, la cuál es menor. Esto significa al igual que en los casos de estudio previos, que los elementos que se encuentran en un grupo determinado están menos ”dispersos” entre ellos dentro del grupo. En las dos gráficas anteriores se puede notar que el comportamiento de los valores de distancia media y máxima es similar. De esta forma se observa que al aplicar CESAMO no hubo pérdida de patrones inherentes a la BDM Tasa de fertilidad.

A person well-trained in computer science knows how to deal with algorithms: how to construct them, manipulate them. This knowledge is preparation for much more than writing good computer programs; it is a general-purpose mental tool that will be a definite aid to the understanding of other subjects, whether they be chemistry, linguistics, etc. An attempt to formalize things as algorithms leads to a much deeper understanding than if we simply try to comprehend things in the traditional way.

Donald Knuth

CAPÍTULO

6

Conclusiones y trabajo futuro

Recordando el objetivo de la presente tesis, el cual es, abordar el problema de tener atributos categóricos en bases de datos mixtas y aplicar algoritmos de inteligencia computacional, basados en métricas. Debido a que no son variables numéricas, distintos algoritmos no pueden ser aplicables. Sin embargo, no solo se busca transformar los datos categóricos a datos numéricos, también se pretende preservar los patrones presentes en los datos.

Por lo que, la principal aportación de este trabajo es, codificar los atributos categóricos de una BDM, para habilitar la aplicación posterior de algoritmos basados en métricas. Para lograrlo, aplicamos una metodología diferente a las metodologías clásicas. Posteriormente se aplicaron algoritmos de inteligencia computacional con el objetivo de observar la preservación de los patrones inherentes en cada BDM.

Las conclusiones sobre cada caso de estudio analizado, se muestran en la siguiente tabla.

BDM	Conclusión
Enfermedad del corazón	En este primer caso de estudio, existen dos clases (enfermo o sano). Se obtuvieron resultados de precisión más altos que 10 algoritmos presentes en la literatura. Los pacientes enfermos fueron clasificados con un error muy pequeño menor al 0.04. En el campo de la medicina esto podría hacer más confiable las decisiones de los médicos para saber el proceso que se debe realizar para cada paciente.
Abalone	Se compararon los resultados contra one-hot encoding, en donde CESAMO obtuvo resultados de precisión visiblemente mejores. Caracterizando los grupos se puede observar que la altura, diámetro y peso influyen notablemente en el agrupamiento. Lo que significa que en dichos atributos, es donde se debe poner mayor atención al momento de realizar clasificación. Debido a que son atributos que podrían influir mayormente en la definición del sexo del Abalone.
Violencia con armas	Una base de datos más grande, fue clasificada con una precisión mayor al ser codificada por CESAMO. Con una precisión 0.1 mayor en comparación con one-hot encoding. Al tratarse de $239,678$ tuplas en la BDM, se sabe que, aproximadamente $213,313$ de ellas fueran clasificadas de forma correcta con CESAMO. Mientras que con one-hot encoding, aproximadamente $189,345$ tuplas se clasificaron correctamente. Por lo que, hay una diferencia de al menos $23,000$ tuplas clasificadas incorrectamente por one-hot encoding en comparación con CESAMO.
Tasa de fertilidad	Para el último caso de estudio, la diferencia de precisión de clasificación entre one-hot encoding y CESAMO fue grande. Con 0.9628 de precisión en CESAMO, lo que significa que casi las $3,100$ tuplas de la BDM fueron clasificadas de forma correcta. Con one-hot encoding aproximadamente, solo $1,980$ tuplas se clasificaron adecuadamente. De igual forma, esta precisión de clasificación puede ayudar en campos de estudio donde se necesita tomar decisiones fundamentadas. Y se puede determinar una decisión, con una mayor seguridad, al tener soporte en estos datos.

Tabla 6.1: Conclusiones sobre las BDM previamente analizadas.

El resultado de este trabajo es, por tanto, el diseño de una metodología que consta de dos partes esenciales. La primera es aproximar una variable con respecto de otra variable (ambas numéricas) y minimizar el error de aproximación mediante la aplicación del algoritmo de ascenso en interacción con el algoritmo genético ecléctico. Lo cual fue logrado satisfactoriamente, a través de un programa realizado en Java. Durante el trabajo de tesis, se realizó la optimización del programa

para realizar operaciones 2 órdenes menores en términos de complejidad computacional, y para evitar el agotamiento de memoria RAM del equipo donde fue programado.

Sin embargo, esta primera parte no se puede efectuar infinitas veces, debido a que el programa realizado en Java se encuentra en un espacio discreto. Para discretizar el espacio fue empleada una técnica estadística fundamentada en el teorema del límite central. Para lograrlo, se asignaron los errores de aproximación (provenientes de la primera parte) en un espacio dividido en 10 cuantiles. Esta parte fue satisfactoriamente aplicada y cuando se observaba una distribución normal en los errores de aproximación, el programa era detenido. Entonces, los mejores códigos numéricos eran obtenidos, con el objetivo de reemplazar las instancias categóricas de una de las variables de la BDM por dichos códigos numéricos.

Al terminar esta metodología, no bastó con tener códigos numéricos en donde antes existían instancias categóricas. El algoritmo de mapas auto-organizados de Kohonen (SOM) fue aplicado, con la finalidad de notar la preservación de patrones inherentes en la BDM. Se observó que el objetivo fue cumplido al analizar el comportamiento de la distancia media y máxima en la BDM antes y después de ser codificada.

En resumen, se ha demostrado que el algoritmo obtiene buenos resultados, incluso en bases de datos mixtas estructuralmente muy diferentes. Fue comparado con resultados que se obtuvieron en la aplicación de algoritmos existentes en la literatura, resultando que, el algoritmo propuesto obtuvo un menor error de entrenamiento y de prueba en el entrenamiento de una red neuronal.

Es así que, aplicamos el algoritmo mencionado (CESAMO) para la codificación categórica y su aplicación en diversas BDM. Y se observa que tendrá resultados competitivos en distintas BDM, en comparación con algoritmos existentes en la literatura.

El trabajo futuro consiste en realizar la aproximación entre más de dos variables de la BDM. Posiblemente realizar un ”uno contra todos”, lo que podría generar una codificación que permita preservar mayor parte de los patrones. En este proyecto no fue realizado debido al coste computacional que implica.

Así también, se espera observar la funcionalidad del algoritmo propuesto en una diversidad más

amplia de BDM. Con el objetivo de tener una idea más clara de los alcances que puede llegar a tener.

La prueba matemática del procedimiento mediante el cual, las relaciones funcionales son encontradas, están fuera del alcance de esta tesis y se establece como trabajo futuro.

Bibliografía

[1] Vazirgiannis M. Halkidi M., Batistakis Y. On clustering validation techniques. *Intelligent Information Systems*, page 107 – 145, 2001. 1, 8

[2] Eds. Le Cam L.M., Neyman J. Some methods for classification and analysis of multivariate observations. *In The Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1:281 – 297, 1967. 1

[3] Kohonen T. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59 – 69, 1982. 2

[4] J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer, 1981. 2

[5] J.C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal Cybernetics*, 3:32 – 57, 1973. 2

[6] J.C. Dunn. Well-separated clusters and optimal fuzzy partitions. *Journal Cybernetics*, 4:95 – 104, 1974. 2

[7] Haykin S. *Neural networks: a comprehensive foundation*. Prentice Hall, 1994. 2

[8] Vapnik V. *Statistical Learning Theory*. John Wiley y Sons, Inc., 1998. 2

[9] Kuri-Morales A. F. Transforming mixed data bases for machine learning: A case study. *Advances in Soft Computing (MICAI)*, 11288:157 – 170, 2018. 2, 11

[10] Cooper L. N. Intrator N. Objective function formulation of the bcm theory of visual cortical plasticity: Statistical connections, stability conditions. *Neural Networks*, 5:3 – 17, 1992. 10

[11] Kotsiantis S. B. *Supervised Machine Learning: A Review of Classification Techniques*. University of Peloponnese, 2007. 7

[12] Kohavi R. Bauer E. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105 – 139, 1999. XI, 8

[13] Jain A. K. Ross A. A., Nandakumar K. *Handbook of multibiometric*. Springer, 2006. 10

[14] A. F. Kuri-Morales. Categorical encoding with neural networks and genetic algorithms. *In Proceedings of the 6th Intl. Conf. on App. Info. and Comp. Theory*, page 167 – 175, 2015. 5, 12, 16

[15] Sagastuy-Breña J. Kuri-Morales A. F. A parallel genetic algorithm for pattern recognition in mixed databases. *In Mexican Conference on Pattern Recognition*, page 13 – 21, 2017. 12, 17

[16] Jagadish H. Labrinidis, A. Challenges and opportunities with big data. *Proceedings of the VLDB Endowment*, 5(12):2032 – 2034, 2012. 13

[17] Meyarivan T. Kalyanmoy D., Agrawal S. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. *In International Conference on Parallel Problem Solving From Nature*, page 849 – 858, 2000. 15

[18] Cybenko G. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303 – 314, 1989. 15

[19] Günter R. Convergence analysis of canonical genetic algorithms. *Neural Networks, IEEE Transactions on*, 5(1):96 – 101, 1994. 15

[20] Aldana-Bobadilla E. Kuri-Morales A. F. The best genetic algorithm i. *Advances in Soft Computing and Its Applications*, page 16 – 29, 2013. 15

[21] Lehr M. A. Widrow B. 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78(9):1415 – 1442, 1990. 15

[22] Kuri-Morales A. F. Lopez-Peña I. Multivariate approximation methods using polynomial models: A comparative study. *Artificial Intelligence (MICAI)*, 2015. 20, 25

[23] Cartas-Ayala A. Kuri-Morales A. F. Polynomial multivariate approximation with genetic algorithms. *Canadian Conference on Artificial Intelligence*, 2014. 20

[24] E. W. Cheney. *Introduction to approximation theory*. McGraw-Hill, 1966. 21

[25] Galaviz-Casas J. Kuri-Morales A. F. *Algoritmos Genéticos*. Fondo de CulturaEconómica/UNAM/IPN, 2002. 28

[26] Kincaid D. Cheney W. *Linear algebra: Theory and applications*. The Australian Mathematical Society, 2009. 30

[27] Gurland J. Ram D. C. Pearson chi-squared test of fit with random intervals. *Biometrika*, 1972. 36

[28] H. W. Lilliefors. On the kolmogorov-smirnov test for normality with mean and variance unknown. *Journal of the American Statistical Association*, 62(318), 1967. 37

[29] D. A. Darling. The kolmogorov-smirnov, cramer-von mises tests. *The Annuals of Mathematical Statistics*, 28(4):823 – 838, 1957. 38

[30] Yap B. W. Razali, N. M. Power comparisons of shapiro-wilk,kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of statistical modeling and analytics*, 2(1):21 – 33, 2011. 38

[31] P. Royston. Approximating the shapiro-wilk w-test for non-normality. *Statistics and Computing*, 2(3):117 – 119, 1992. XI, XV, 39, 41, 42

[32] H. Akaike. Akaike's information criterion. *International Encyclopedia of Statistical Science*, page 22 – 25, 2011. 39

[33] López-Peña I. Kuri-Morales A.F. Normality from monte carlo simulation for statistical validation of computer intensive algorithms. *Advances in Soft Computing (MICAI)*, 10062, 2017. 40

[34] K. Binder. Introduction: Theory and "technical.ªspects of monte carlo simulations. *Monte Carlo Methods in Statistical Physics*, page 1 – 45, 1986. 41

[35] A. F. Kuri-Morales. Closed determination of the number of neurons in the hidden layer of a multi-layered perceptron network. *Soft Computing*, 2017. 46

[36] S. H. Kwon. Cluster validity index for fuzzy clustering. *Electronics letters*, 34(22):2176 – 2177, 1998. 51

[37] R. R. Sokal. *The principles of numerical taxonomy: twenty-five years later*. Prentice Hall, 1985.

[38] Couto Y. Barbará, Daniel Li Y. Coolcat: an entropy-based algorithm for categorical clustering. *In Proceedings of the eleventh international conference on Information and knowledge management*, page 582 – 589, 2002.

[39] Kuri-Morales A. F. Aldana-Bobadilla E. A clustering method based on the maximum entropy principle. *Entropy*, 17(1):151 – 180, 2015.

[40] Cartas-Ayala A. Kuri-Morales A. F. Automatic closed modeling of multiple variable systems using soft computation. *Advances in Soft Computing (MICAI)*, 10632, 2018.

[41] A. F. Kuri-Morales. Minimum data base determination using machine learning. *International Journal of Web Services Research (IJWSR)*, 13(4):1 – 18, 2016.

[42] A. F. Kuri-Morales. Categorical encoding with neural networks and genetic algorithms. *Proceedings of the 6th International Conference on Applied Informatics and Computing Theory*, page 167 – 175, 2015.

[43] A. F. Kuri-Morales. Pattern discovery in mixed data bases. *Pattern Recognition (MCPR)*, 10880, 2018.