Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA

ACTA DE EXAMEN DE GRADO

No. 00057
Matrícula: 2143805398

UN SISTEMA CLASIFICADOR NO SUPERVISADO UTILIZANDO COLORACIÓN DE GRÁFICAS SUAVES.

JORGE FLORES CRUZ
ALUMNO

REVISÓ

LIC. JULIO CESAR DE LARA ISASSI
DIRECTOR DE SISTEMAS ESCOLARES

En la Ciudad de México, se presentaron a las 8:00 horas del día 8 del mes de noviembre del año 2016 en la Unidad Iztapalapa de la Universidad Autónoma Metropolitana, los suscritos miembros del jurado:

DR. SERGIO GERARDO DE LOS COBOS SILVA
MTRO. MARIA DEL CARMEN FERNANDEZ GARCIA
DR. PEDRO LARA VELAZQUEZ

Bajo la Presidencia del primero y con carácter de Secretario el último, se reunieron para proceder al Examen de Grado cuya denominación aparece al margen, para la obtención del grado de:

MAESTRO EN CIENCIAS (CIENCIAS Y TECNOLOGIAS DE LA INFORMACION)

DE: JORGE FLORES CRUZ

y de acuerdo con el artículo 78 fracción III del Reglamento de Estudios Superiores de la Universidad Autónoma Metropolitana, los miembros del jurado resolvieron:

Aprobar

Acto continuo, el presidente del jurado comunicó al interesado el resultado de la evaluación y, en caso aprobatorio, le fue tomada la protesta.

DIRECTOR DE LA DIVISIÓN DE CBI

DR. JOSE GILBERTO CORDOBA HERRERA

PRESIDENTE

DR. SERGIO GERARDO DE LOS COBOS SILVA

VOCAL

MTRO. MARIA DEL CARMEN FERNANDEZ GARCIA

SECRETARIO

DR. PEDRO LARA VELAZQUEZ

UNIVERSIDAD AUTÓNOMA METROPOLITANA-IZTAPALAPA
DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA

UN SISTEMA CLASIFICADOR NO SUPERVISADO UTILIZANDO COLORACIÓN DE GRÁFICAS SUAVES

Tesis que presenta
Jorge Flores Cruz
Para obtener el grado de
Maestro en ciencias y tecnologías de la información

Asesores: Dr. Pedro Lara Velázquez
Dr. Miguel Ángel Gutiérrez Andrade

Jurado calificador:
Presidente: Dr. Sergio Gerardo de los Cobos Silva
Secretario: Dr. Pedro Lara Velázquez
Vocal: M. I. María del Carmen Fernández García

México, Ciudad de México Noviembre 2016

Resumen

Los seres humanos son buenos para clasificar los elementos de su entorno, la acción de etiquetar o colocar un objeto dentro de un grupo de similares permite un mejor estudio de los mismos. El poder automatizar ese proceso con el menor procesamiento de datos y la menor intervención humana es el objetivo del presente trabajo. Los clasificadores no supervisados plantean un método de agrupamiento con una menor intervención humana que los clasificadores supervisados, para ello es necesario perfeccionar la técnica de agrupado bajo el menor filtrado de datos. En este documento se propone un sistema clasificador no supervisado que utiliza el modelo de coloración de gráficas suaves. El método se evalúa con algunas instancias clásicas de la literatura especializada y se comparan los resultados obtenidos con las clasificaciones esperadas. Los resultados obtenidos, en conjunto, son tan buenos como los de otros clasificadores supervisados y no supervisados y en algunos casos proporcionan información adicional no considerada previamente en la clasificación realizada por seres humanos. Acompañando al documento, se ha enviado un artículo a una revista indexada en Scopus y Zentralblatt MATH con nombre "Revista Matemática Teoría y Aplicaciones" el cual al momento de la conclusión de la Idónea Comunicación de Resultados (ICR) se encuentra listo para ser publicado en el volumen 21-1 de 2017.

Agradecimientos

A mis asesores por su orientación, apoyo y consejos en este proyecto. A la Universidad Autónoma Metropolitana por darme la oportunidad de ingresar al posgrado y proporcionarme las herramientas necesarias durante mi estadía en el mismo.

Extiendo el agradecimiento a la empresa por permitirme estudiar mientas me encontraba trabajando.

A mi familia por siempre estar presente en cada momento de mi vida y especialmente a mi mujer por haberme convencido, impulsado y animado a concluir esta nueva meta en mi vida.

Contenido

Lista de Figuras

Lista de Tablas

1. Introducción

Una de las aplicaciones de la Inteligencia Artificial consiste en el reconocimiento de patrones, el cual se puede entender como clasificar grandes cantidades de objetos físicos o abstractos con el propósito de extraer información útil y que permita establecer propiedades entre agrupaciones de dichos objetos. Las razones para automatizar este proceso van desde hacer actividades autoadaptables, es decir, aplicaciones capaces de adaptarse a distintos tipos de problemas y que van aprendiendo a lo largo de su ejecución sin requerir de la intervención humana para su aprendizaje, hasta actividades tediosas, tardadas o hasta imposibles para una persona o un grupo de personas.

Los patrones se obtienen de un proceso de segmentación, extracción de características y reseña de cada objeto como una colección de descriptores tal y como se muestra en la figura 1.1

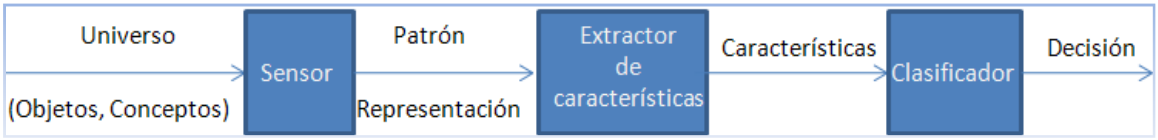

Figura 1.1 Proceso de reconocimiento de patrones.

La ciencia y la tecnología se han valido del reconocimiento de patrones para diagnosticar enfermedades, detectar seres submarinos con un sonar, identificar clientes que es posible que no paguen créditos o den información falsa, etc.

Los sistemas clasificadores son un tipo especial de reconocimiento de patrones, éstos colocan una etiqueta a un objeto de acuerdo a sus características; ya sea para personas, otros seres vivos u objetos inanimados. Los seres humanos hacemos reconocimiento de patrones cotidianamente incluso de forma inconsciente. El reto de un modelo de reconocimiento de patrones es enseñarle a una máquina, en este caso una computadora, a hacerlo de forma eficiente sin la ayuda humana.

En un sistema clasificador, la etiqueta puede ser determinada previamente, luego se tiene que decidir dado un objeto en qué clase particular dadas las opciones existentes. Por ejemplo, un servidor debe decidir si un correo entrante es “spam” o se dirige a la bandeja de entrada. Hay veces donde las clases no están definidas con

anterioridad y es necesario encontrar la forma de agrupar los elementos de forma óptima.

La coloración de gráficas suaves es una generalización del problema de coloración (Diestel, 2000) en el que se busca encontrar una coloración que minimice la dureza en la gráfica, o dicho de otra forma, reducir la suma de distancias entre vértices con colores idénticos. Este modelo se utiliza en la programación de eventos susceptibles de cambios, asignación estable de frecuencias del espectro electromagnético, calendarización de actividades, asignación de recursos en organizaciones, etc. Se ha demostrado que es un problema de tipo NP-Duro, aunque para grafos de orden menor o igual a 20, se pueden utilizar algoritmos exactos que resuelven el problema; caso contrario es necesario el uso de técnicas heurísticas con aproximaciones bastante aceptables (Gutiérrez-Andrade, *et al.*, 2011).

El reconocimiento de patrones se refiere al problema de encontrar la estructura interna de los objetos etiquetados. Para llevar a cabo la tarea es necesario definir una distancia entre dos objetos, de esta manera si dos objetos están muy cerca respecto a una métrica dada, es muy probable que estén en la misma clase. Si los objetos están lejos uno del otro, van a estar en clases diferentes, la métrica más utilizada entre otras muchas técnicas (Jacquard, Manhattan, logarítmica, cuadrados independientes, coeficientes de correlación de Pearson, etc.) es la distancia euclidiana (Diestel, 2000).

En el presente documento se explicarán los conceptos de sistema clasificador, aprendizaje computacional y métricas de similitud o disimilitud, acompañado de distintos ejemplos de clasificadores supervisados y no supervisados (capítulo 2). En la siguiente sección se abordará el problema de coloración de gráficas suaves así como los conceptos utilizados para la propuesta de solución (capítulo 3). La propuesta esclarece el método utilizado para implementar el modelo de coloración de gráficas suaves en el clasificador no supervisado (capítulo 4), seguido de la evaluación del modelo en distintos tipos de bases de datos (capítulo 5). Finalizando con las conclusiones y las referencias.

2. Marco de referencia

Un sistema clasificador es la abstracción informática de la habilidad humana para reconocer clases o categorías de varios objetos, ya sean caras, voces, letras, seres vivos, etc. Para la tarea que implica el reconocimiento de patrones se busca clasificar ciertos objetos en una de *k* categorías preestablecidas. Resultando la clase a la que pertenece el objeto o en algunos casos no poder determinar la clase a la que se llega.

Una clase, también llamada categoría, puede entenderse como una característica que exhibe ciertas regularidades, sirve como modelo y representa un concepto de lo observado. Un patrón es un conjunto de clases o características observadas relacionadas en el espacio y/o tiempo, y que muestra una estructura indicativa del concepto subyacente de un objeto (Rodríguez, 2004). Formalmente un patrón es un par ordenado de observación y concepto.

2.1. Diseño de un Sistema Clasificador

La construcción de un sistema clasificador consiste en diseñar un algoritmo que a partir de un conjunto de objetos, éste sea capaz de asignarles una etiqueta de un conjunto de clases preestablecidas o a partir del conjunto de objetos, crear las clases que lo agrupen a partir de sus características comunes, estableciendo una métrica de similitud.

Por ejemplo, un sistema que tratase de emular a una rana debe contener las características más importantes de los objetos percibidos por ella. De acuerdo al movimiento, tamaño, localización, olor y otros atributos del objeto, la rana debe discernir entre atacarlo, perseguirlo o ignorarlo (Holland, 1992).
Por ejemplo:

- Si el objeto es grande, alargado y se mueve, huir.
- Si el objeto se mueve en el aire, es pequeño y cercano, atacarlo.
- Si el objeto se mueve, es del tamaño de la rana y parece atractiva, perseguirla.

Los ejemplos anteriores representarían una serpiente, una mosca y una rana hembra respectivamente, por supuesto que las características proporcionadas son insuficientes y pueden dar lugar a ambigüedades, por lo que la obtención de mejores resultados dependerá de la robustez del algoritmo y la cantidad de características o atributos que puedan extraerse de los objetos.

2.2. Aprendizaje Computacional

Al buscar problemas cuya solución óptima resulta poco factible o muy complicada, se recurren a métodos heurísticos, con los cuales se busca encontrar soluciones parciales o aproximadas pero que en la práctica dan buenos resultados. Para alcanzar una buena solución, los programadores deben introducir un conocimiento previo del problema para facilitar la solución, reduciendo el costo computacional de la búsqueda de solución (Moreno-Montiel, 2009).

En muchos casos cuando un programa da una solución a un problema específico, si ésta se quiere generalizar es necesario reconstruir el programa y agregar nuevo conocimiento, limitándose a la incapacidad de resolver problemas para los que no fueron programados.
Todo lo anterior marca la pauta para desarrollar una solución capaz de auto adaptarse y de agregar nuevo conocimiento de forma automática, resolviendo programas distintos al que fueron programados.

El Aprendizaje Computacional es una rama de la Inteligencia Artificial que busca construir un modelo matemático para modelar el proceso cognitivo (Anthony, 1997). Descrito como una macro-teoría que provee el entorno para estudiar el comportamiento de los algoritmos bajo distintos tipos de entrenamiento o conjunto datos de entrada.

Aprendizaje supervisado.

Es un tipo de aprendizaje computacional, el cual consiste en encontrar una aproximación de un problema o función objetivo usando datos etiquetados (Zhu, *et al*., 2003), es decir, los ejemplos proporcionados ya tienen clases asociadas previamente. La meta es conseguir que los demás datos que no se usen como ejemplo puedan ser clasificados dentro de las clases de ejemplo. Análogo a que un profesor le da ejemplos a un alumno.

Aprendizaje no supervisado.

Contrario al aprendizaje supervisado, en el aprendizaje no supervisado no hay clases predeterminadas. El objetivo es encontrar la clase más adecuada para el conjunto sin etiquetar. Utilizando el grado de similitud en sus atributos podemos determinar si los objetos van a una misma clase o en caso de encontrarse diferencias, colocarlos en distintas clases.

Aprendizaje semisupervisado.

El aprendizaje semi-supervisado contiene una combinación de los aprendizajes supervisado y no supervisado. En adición a los datos sin etiquetar, se provee al algoritmo de algo de información supervisada, no necesariamente a todos los objetos o información de todas las clases. Los métodos semiautomáticos asumen que los datos sin etiquetar y que sean similares deben de ubicarse siempre en la misma clase (Zhu, *et al*., 2003).

2.3. Tipos de clasificadores

Se han construido varios clasificadores aplicando diversos tipos de aprendizaje, entre los cuales destacan:

Clasificador de Bayes Ingenuo

Aplica el teorema de Bayes de probabilidad condicional como modelo de clasificación con algoritmo simplificado pero que contiene las características más importantes. Se le llama clasificador ingenuo porque asume que hay independencia probabilística entre cada atributo o característica de los objetos; un valor de característica no influye en el valor de otra característica o atributo. Aún con esa limitante es un clasificador muy efectivo que en la práctica da resultados bastante aceptables y no calcula probabilidades cero siempre y cuando los valores de cada clase se presenten en el conjunto de entrenamiento (Joachims, 1996).

Sea el vector $X = (x_1, x_2, x_3, \ldots, x_n)$ un ejemplo para clasificar dentro de cualquiera de las clases C_1 y C_2 (Larrañaga, *et al*., 2007) Usando el teorema de Bayes para expresar la probabilidad condicional $P(X|C_1)$ de los atributos independientes entre sí de la siguiente manera:

$$P(X|C_1)= P(x_1|C_1)* P(x_2|C_1)*\ldots *P(x_n|C_1)$$

$$P(X|C_1) = \prod_{i=1}^{n} P(x_i|C_1)$$

De igual forma se puede realizar para una segunda, tercera o enésima clase:

$$P(X|C_k)= P(x_1|C_k)* P(x_2|C_k)*\ldots *P(x_n|C_k)$$

$$P(X|W_k) = \prod_{i=1}^{n} P(x_i|W_k)$$

Tanto las clases como la distribución de los atributos en los datos se pueden aproximar usando las frecuencias relativas del conjunto de entrenamiento por lo que una clase se puede calcular estimando la probabilidad de clase del conjunto de entrenamiento.

Para la distribución de los parámetros o características en el clasificador Bayes Ingenuo se utilizan comúnmente la distribución binomial y la distribución de Bernoulli cuando se trabaja con datos discretos, cuando los datos son continuos se asume que la distribución de los datos se comporta como una distribución Normal.

Clasificador Parzen

Mientras algunos clasificadores buscan establecer distancias entre los objetos mediante métricas −distancias entre los objetos− de similitud o disimilitud, hay clasificadores que buscan encontrar las distancias entre objetos a las líneas de características. El concepto de líneas de características permite generalizar el espacio de similitudes o disimilitudes y una vez obtenido el espacio estimar densidades de probabilidad mediante el método Parzen (Parzen, 1962) y hacer uso de ellas para la clasificación de patrones.

Otra de sus características es que sostiene un aprendizaje no paramétrico. En el aprendizaje paramétrico se trata de determinar la densidad de probabilidad de las características, es decir, qué tanto influye una característica en que un objeto entre en una clase. En el aprendizaje no paramétrico, el conocimiento previo de las distribuciones de probabilidad condicional de cada característica no está disponible o no se muestra de forma explícita, es aquí donde entra el clasificador Parzen basado en la estimación de densidades de probabilidad (Trujillo-Pulgarín, 2012).

El camino planteado es particionar el espacio de métricas en cajas disjuntas R_i y se obtienen las muestras que caen en cada una de esas cajas.

$N_{k,n}$ expresa el número de muestras de una clase C_k entonces la densidad de probabilidad dentro de la *n-ésima* caja, por lo que la densidad de probabilidad se calcula como:

$$P(x_n|W_k) = \frac{N_{k\ n}}{\text{volumen}(R_n) * N_k}$$

Para cada clase, el número $N_{k,n}$ tiene una distribución multinomial denominada histograma con parámetros:

$P_{k,n} = \int_{z \epsilon R_n} P(z|W_k)$ Para *n* perteneciente al número de cajas.

Clasificador Red Neuronal

Tratando de simular el comportamiento de una neurona biológica, se sintetizan los componentes principales que son las dendritas o entradas, el cuerpo de la neurona y los axones o salidas. En la parte final de cada axón se encuentra la conexión con dendritas de otra neurona, a este proceso de comunicación se llama sinapsis. Utilizando el Modelo de Perceptrón Multicapa (PMC por sus siglas en inglés) para clasificación tenemos un conjunto de señales de entrada multiplicadas por sus pesos correspondientes y luego sumadas a cada una de las nuevas entradas en el proceso de sinapsis o nivel de activación de la neurona (Jain, *et al.*, 1999).

La estructura de un PMC consiste en una capa de entrada, una o más capas ocultas y una o más capas de salida. Primero se inicializan aleatoriamente los pesos, luego se ingresan los datos en las entradas, las capas ocultas se encargan de procesar la información mediante las operaciones de multiplicación y de suma y finalmente se muestran los resultados en la(s) capa(s) de salida. Como se trata de un modelo de aprendizaje supervisado, cuando un resultado no es satisfactorio se repite el proceso ajustando los pesos hasta que el modelo sea capaz de clasificar satisfactoriamente el conjunto de entrenamiento.

2.4. Proceso de agrupamiento

Los algoritmos de agrupamiento o clústering son métodos que dividen un conjunto de n datos dentro de k grupos de tal modo que los miembros de un mismo grupo sean más parecidos entre ellos que los miembros de distintos grupos (Ripley, 1996). El número de grupos g puede estar predeterminado o puede ser decidido por el algoritmo. Formalmente un algoritmo de agrupamiento produce un mapeo C:{1,…,k} →{1,…,n} asociando cada objeto con un grupo.

Un agrupamiento o clúster se describe como un grupo o comunidad con homogeneidad de características internas y una separación explícita de las otras comunidades (Gordon, 1999). El análisis de información por agrupamiento entra en el terreno del aprendizaje no supervisado y consiste en la búsqueda a través de una gran variedad de comunidades. La diversidad de la información obliga a desarrollar técnicas y herramientas que permitan procesar tal cantidad en el menor tiempo posible; se han utilizado técnicas estadísticas, de ciencia computacional y de aprendizaje computacional.

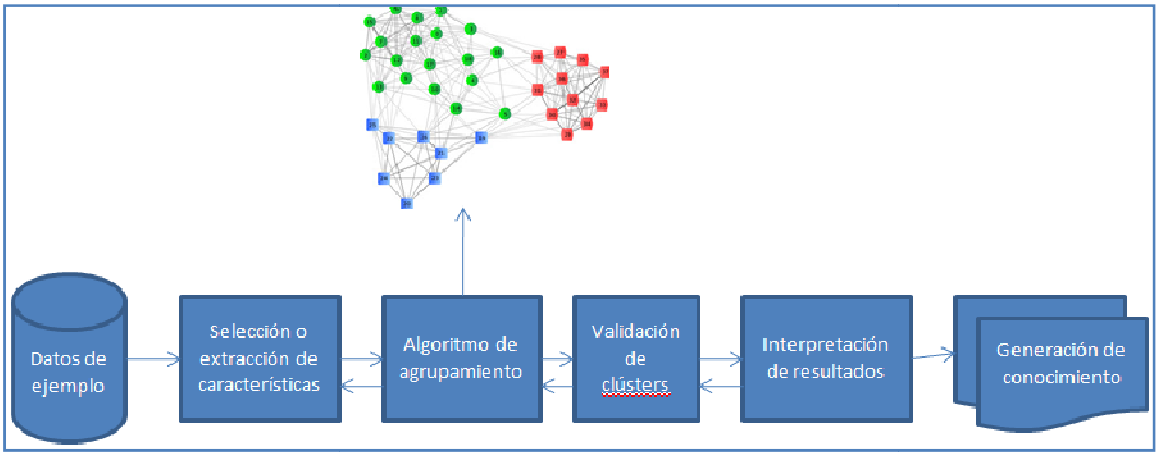

Figura 2.1 Proceso de análisis por agrupamiento.

El proceso de análisis por agrupamiento mostrado en la Figura 2.1 es un camino bidireccional que consiste en (Xu, Wunsch, 2005):

- **Selección o extracción de características:** Se escogen las características más representativas de un conjunto de candidatos, también se utilizan transformaciones para generar características nuevas a partir de las originales.
- **Diseño y selección de un algoritmo agrupador:** Usualmente combinado con la selección de una medida de proximidad y la construcción de una función de criterio, establece la métrica de similitud entre un conjunto de datos.
- **Validación de clústers:** Se analiza si la división en subgrupos es confiable, acorde a unos estándares de evaluación neutrales, sin preferencia por algún algoritmo.
- **Interpretación de resultados:** Finalmente se provee de la información sustancial que se obtuvo de la cantidad de datos original, pasando por un juicio de expertos que analizan e interpretan la partición de los datos.

En el siguiente ejemplo se puede observar el proceso de agrupamiento a grandes rasgos. Se tiene un conjunto de datos dispersos donde se calculan *k* centroides, de ahí se deriva el nombre de *k*-medias, como se puede observar en la figura 2.2:

Figura 2.2 Cálculo de centroides.

Cada uno de los datos se asocia a la media más cercana que tienen formando *k* grupos o clases como se puede apreciar en la figura 2.3.

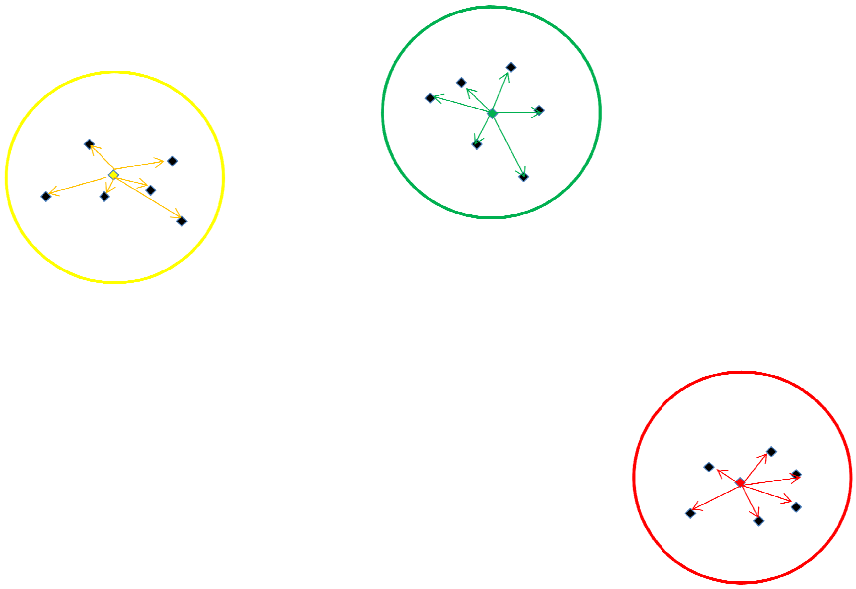

Figura 2.3 Formación de los grupos.

Se repiten los procesos de las figuras 2.2 y 2.3 (cálculo del centroide y asociación de los elementos) varias veces hasta que se obtiene la mejor clasificación mostrada en la figura 2.4:

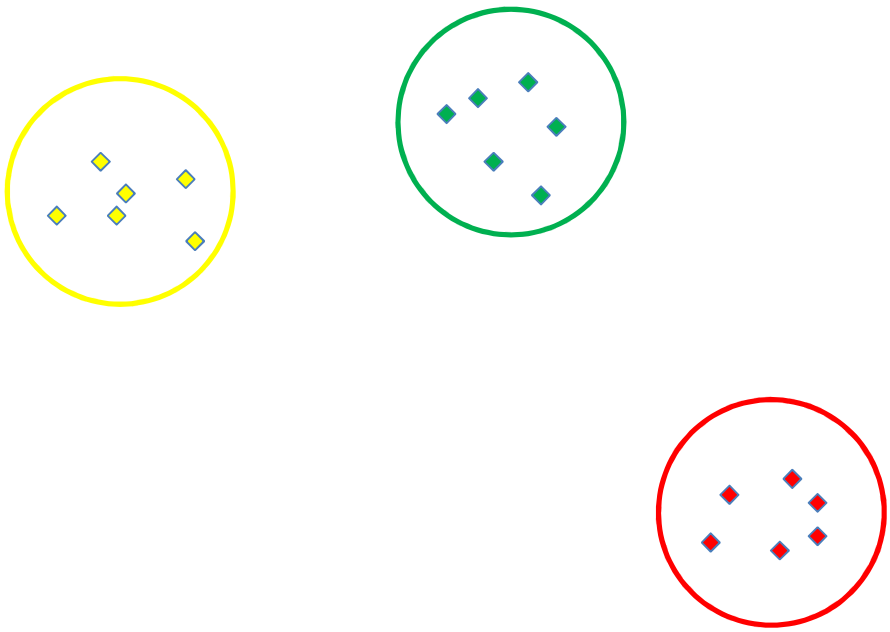

Figura 2.4 Clasificación final propuesta por el algoritmo.

2.5. Tipos de agrupadores

Se mencionan a continuación los tipos de algoritmos agrupadores de mayor relevancia, posteriormente se explicarán a detalle:

a) Por medidas de similitud y distancia.
b) Jerárquico
    a. Aglomerativos.
    b. Divisivos.
c) Basados en error cuadrático.
d) Estimación por mezcla de densidades.
e) Basados en teoría de grafos.
f) Basados en técnicas de búsqueda combinatoria.
g) De lógica borrosa.
h) Basados en redes neuronales.

i) Agrupamiento basado en núcleo.
j) Agrupamiento de datos secuenciales.
k) Agrupamiento de gran cantidad de datos.
l) Exploración de datos multidimensionales.

2.6. Algoritmos no supervisados

Un marco de clasificación ampliamente aceptado es la clasificación de técnicas de agrupamiento en agrupamiento jerárquico y agrupamiento por partición (Jain, *et al*., 1999). El agrupamiento jerárquico forma grupos de datos ordenándolos de forma secuencial y estableciendo relaciones de herencia entre ellos, mientras que el agrupamiento por partición divide los datos en un número predeterminado de grupos sin relaciones de herencia. A continuación se revisarán diferentes tipos de algoritmos no supervisados como los basados en una medida de proximidad, algoritmos de agrupamiento jerárquico, algoritmos por partición, además de incluir una amplia variedad de técnicas y teorías como teoría de grafos, búsqueda combinatoria, teoría de conjuntos difusos, redes neuronales y técnicas de granularidad (Xu, Wunsch, 2005).

2.7. Métricas de Similitud y Disimilitud.

Parten de la problemática de encontrar los estándares a usar para determinar la similitud y disimilitud entre dos objetos, entre un objeto y un grupo o entre dos grupos (Everitt, *et al*., 2001). Algunos enfoques para encontrar esas métricas sugieren establecer vínculos jerárquicos entre grupos y con el uso de prototipos que representan un grupo para ser procesados como entidades individuales.

Un objeto se describe como un conjunto de características comúnmente representadas por un vector multidimensional (Kaufman, Rousseeuw, 1990). Dichas características pueden ser cuantitativas o cualitativas, continuas o discretas. La condición *D* de disimilitud o distancia en un conjunto de datos *X* se satisface con las condiciones de:

1. Simetría: $D(x_i, x_j)=D(x_j,x_i)$.
2. Positividad: $D(x_i, x_j) \geq 0$ para todo x_i,x_j.
3. Desigualdad del triángulo: $D(x_i, x_j) \leq D(x_i, x_l) + D(x_l,x_j)$ para todo x_i,x_j, x_l.
4. Reflexividad: $D(x_i, x_j)=0$ sii $x_i= x_j$

Análogamente, unas condiciones parecidas se establecen para la condición *S* de similitud (Xu, Wunsch, 2005).

1. Simetría: $S(x_i, x_j)=S(x_j,x_i)$.

2. Positividad: $0 \leq S(x_i, x_j) \leq 1$ para todo x_i, x_j
3. $S(x_i, x_j)\ S(x_j, x_l) \leq [S(x_i, x_j) + S(x_j, x_l)]\ S(x_i, x_l)$ para todo x_i, x_j, x_l.
4. $S(x_i, x_j)=1$ sii $x_i = x_j$

En un conjunto de *k* atributos, se define una matriz simétrica de *k x k* llamada matriz de proximidad cuyo par *k*(*i,j*) representa la medida de similitud o disimilitud entre los atributos *i* y *j* (*i,j =1, …, k*). Las métricas de disimilitud o distancia se usan más para atributos continuos, mientras que las métricas de similitud se utilizan más para atributos cualitativos. En la tabla 2.1 se muestran las métricas más comunes para atributos cuantitativos.

Métrica	**Ecuación**	**Descripción**	**Ejemplo**
Distancia de Minkowsky	$D_{ij} = \left(\sum_{l=1}^{k} \lvert x_{il} - x_{jl} \rvert^{a} \right)^{1/a}$	Atributos con grandes valores y varianzas tienden a dominar sobre los demás.	Principalmente para conjuntos difusos. (Höppner, *et al.*,1999)
Distancia Euclidiana	$D_{ij} = \left(\sum_{l=1}^{k} (x_{il} - x_{jl})^{2} \right)^{1/2}$	Métrica más utilizada. Caso particular de Minkowsky cuando *n=2*. Tiende a formar grupos hiperesféricos.	Para algoritmo de operador núcleo. (Schölkopf, *et al.*, 1998)
Distancia Manhattan	$D_{ij} = \sum_{l=1}^{k} \lvert x_{il} - x_{jl} \rvert$	Caso especial de Minkowsky para *n=1.* Tiende a formar grupos hiperrectangulares.	Conjuntos difusos ART. (Baraldi, Alpaydin, 2002)
Distancia Sup	$D_{ij} = \max_{1 \leq l \leq d} \lvert x_{il} - x_{jl} \rvert$	Caso especial de Minkowsky para $n \rightarrow \infty$	Conjuntos difusos *c-means.*
Distancia de Mahalanobis	$D_{ij} = (x_i - x_j)^{T} * A^{-1}(x_i - x_j)$, Donde *A* es la matriz de covarianza para el grupo.	Tiende a formar grupos hiperelipsodiales. Cuando los atributos no están correlacionados es equivalente a la	ART elipsoidal, algoritmo de agrupamiento hiperelipsoidal.

		distancia Euclidiana. Puede elevar el costo computacional.	
Correlación de Pearson	$D_{ij} = \frac{1 - r_{ij}}{2}, donde$ $r_{ij} = \frac{\sum_{l=1}^{k}(x_{il} - \overline{x_i})(x_{jl} - \overline{x_j})}{\sqrt{\sum_{l=1}^{k}(x_{il} - \overline{x_i})^2 \sum_{l=1}^{k}(x_{il} - \overline{x_j})^2}}$	Coeficiente de correlación entre dos atributos que no establece métricas.	Agrupamiento por algoritmos genéticos
Distancia de punto de simetría	$D_{ir} = \min_{j=1,\ldots,n \,\&\, j \neq i} \frac{\|(x_i - x_r) + (x_j - x_r)\|}{\|(x_i - x_r)\| + \|(x_j - x_r)\|}$	Establece la distancia entre un objeto x_i y un punto de referencia común x_r. No es una métrica como tal.	Simetría basada en operador núcleo.
Similitud por coseno	$S_{nm} = \cos \propto = \frac{x_n^T x_m}{\|x_n\| \|x_m\|}$	Independiente de la longitud del vector e invariante en la rotación pero no para transformaciones lineales.	Métrica más utilizada para agrupamiento de documentos.

Tabla 2.1 (Xu, Wunsch, 2005) Resumen de las principales métricas para atributos cuantitativos.

2.8. Agrupamiento Jerárquico.

El agrupamiento jerárquico organiza los datos en una estructura jerárquica, normalmente un árbol binario o dendrograma, acorde a una matriz de proximidad (Everitt, *et al*., 2001). El nodo raíz representa todo el conjunto de datos y cada hoja representa un objeto de los datos. Los nodos intermedios describen la similitud que tienen los objetos entre sí y la profundidad del árbol describe la distancia entre cada objeto o grupo. Esta representación provee mucha información sobre la disposición de los datos y sus relaciones entre ellos. Los agrupamientos jerárquicos son métodos aglomerativos y divisivos.

Un algoritmo aglomerativo parte de un conjunto de n grupos con un objeto cada uno, llamados grupos unitarios, después realiza una serie de operaciones de unión para ir colocando los demás objetos dentro del grupo que considere más adecuado, mientras que un algoritmo divisivo parte de un conjunto que contiene a

todos los datos para subdividirlo en varios grupos que contendrán a los objetos (Kaufman, Rousseeuw, 1990).

Un agrupamiento jerárquico puede explicarse de forma gráfica mediante un dendrograma donde se genera una clase padre y a partir de ella se va subdividiendo en clases hijas, tal y como se observa en la figura 2.5:

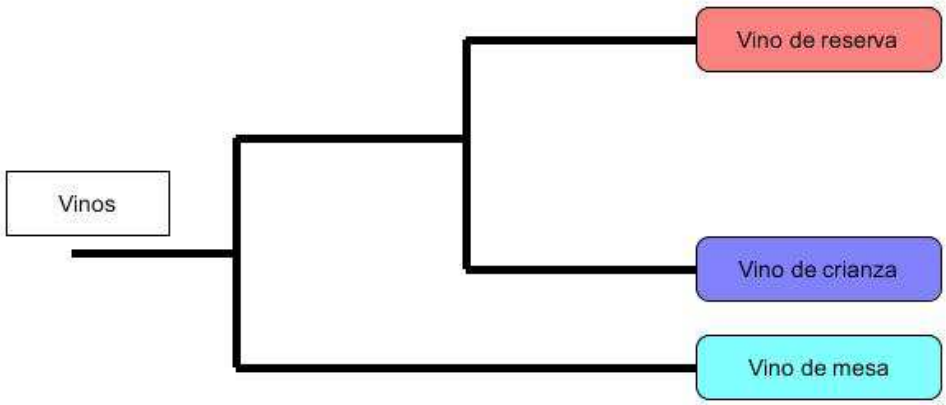

Figura 2.5 Ejemplo de agrupamiento jerárquico.

2.9. Agrupamiento basado en error cuadrático.

A diferencia del agrupamiento jerárquico que va formando varios niveles de grupos, el agrupamiento divisivo crea un conjunto de *n* grupos sin relación jerárquica, necesitando de heurísticas para resolverlo por ser un problema NP-Duro al haber una enorme cantidad de combinaciones incluso para grupos pequeños (Hansen, Jaumard , 1997).

Lo primero que hay que establecer es el criterio de separación y uno de los más usados es el de la suma del error cuadrático. Teniendo un conjunto de valores continuos x_n, n=1,…,m hay que organizarlos en k grupos C={C_1,…,C_k} se define criterio del error cuadrático como (Duda, *et al*., 2001):

$$J(\Gamma, M) = \sum_{i=1}^{k} \sum_{j=1}^{n} \gamma_{ij} \left\| x_j - m_i \right\|^2$$

$$y_i = \frac{1}{n_i} \sum_{j=1}^{n} \gamma_{ij}\, x_j$$

Donde $\boldsymbol{\Gamma}$ es la matriz de partición con elementos γ_{ij}tal que:

$$\gamma_{\mathrm{ij}} = \begin{cases} 1 \text{ si } x_{\mathrm{j}} \in \text{grupo } C_{\mathrm{i}} \\ 0 \text{ en otro caso} \end{cases} \text{ con } \sum_{i=1}^{k} \gamma_{ij} = 1 \text{ para todo } j$$

M=prototipo de centroide para la matriz de partición.
m_i es la media de la muestra para el grupo *i*.
n_i es el número de objetos en el grupo *i*.

2.10. Estimación por mezcla de densidades.

Tiene un enfoque probabilístico de la solución, asume que los objetos son generados obedeciendo a alguna distribución de probabilidad (Zhuang, *et al*., 1996). Si las distribuciones de los datos son conocidas, crear los grupos es equivalente a estimar los parámetros de la función de densidad con que se distribuyeron los datos (e.g. Gaussiana multivariable, *t* de Student). Teniendo la probabilidad a priori, o la probabilidad mezclante, P(C_i) para el grupo C_i, i=1,…,k y la densidad de probabilidad condicional $p(x|C_i, \theta_i)$ donde θ_i es el parámetro de la función de densidad que se busca (Everitt, *et al*., 2001).

2.11. Agrupamiento por teoría de grafos.

Modela a los vértices *V* de un grafo ponderado *G* como datos en un espacio de patrones y aristas *E* donde se visualizan las proximidades o similitudes y las diferencias o disimilitudes, definiendo la siguiente matriz (Jain, *et al*., 1999).

$$D_{nm} = \begin{cases} 1 \text{ cuando } D(x_n, x_m) < d_0 \\ 0 \text{ en otro caso} \end{cases}$$

Donde d_0 es el valor del peso del grafo. Aquí es donde la similitud se obtiene normalizando la suma de los pesos de las aristas que interconectan los grupos.

La teoría de grafos ha ayudado al desarrollo de algoritmos de agrupamiento tanto jerárquicos: Camaleón (Karypis, *et al*., 1999), AMOEBA (Estivill-Castro, Lee, 1999), como no jerárquicos: CLICK (Sharan, Shamir, 2000), CAST (Ben-Dor, *et al*., 1999).

En el caso del agrupamiento jerárquico, éste puede representarse como un árbol donde se pueden estudiar jerarquías locales en los subgrafos y la jerarquización completa utilizando el grafo completo. Camaleón (Karypis, *et al*., 1999) es un algoritmo aglomerativo jerárquico basado en encontrar a los vecinos más cercanos y la relación entre 2 vértices se elimina si uno de ellos no cumple con los criterios de cercanía establecidos. Lo que termina generando un conjunto de subgrafos donde

cada uno debe contener suficientes nodos para que el costo computacional sea similar. Camaleón establece la jerarquización a partir de la interconectividad y relaciones de los subgrafos, lo que flexibiliza al algoritmo al momento que generar las relaciones entre los grupos. La cercanía de los subgrupos se obtiene normalizando la suma de los pesos de los bordes entre cada subgrafo.

La teoría de grafos también es utilizada en los agrupadores no jerárquicos, encontrando componentes relacionados mediante la técnica del árbol de expansión mínima (Jain, *et al*., 1999). Otro algoritmo no jerárquico llamado CLICK (Sharan, Shamir, 2000), calcula los pesos de los nodos combinando probabilidad con teoría de grafos, es decir, el grado de cercanía entre un nodo y otro como la probabilidad de que un nodo pertenezca a un nodo o a otro. Una vez generado los subgrupos, el algoritmo vuelve a revisar las características más importantes de cada grupo para poder incluir a los nodos que se encuentran solos. Al ser recursivo el algoritmo, se requieren de metaheurísticas para reducir los costos.

Para el caso de este documento, el modelo de coloración de gráficas suaves plantea los datos como una gráfica completa ponderada, los pesos de las aristas se establecen por medio de una métrica, más adelante en el capítulo 4 se explicará la métrica utilizada, por lo que la representación de los datos en un grafo queda de la siguiente manera:

$$D_{nm} = \begin{cases} \text{Valor de la métrica cuando } D(x_n, x_m) < d_0 \\ 0 \text{ en otro caso} \end{cases}$$

2.12. Técnicas de búsqueda combinatoria.

El problema de optimización combinatoria (De Los Cobos, *et al*., 2010) define un conjunto finito de soluciones posibles de una función objetivo o función costo, tratando de encontrar para el caso de minimización x_{opt} que pertenece al dominio de la función que cumpla con la condición:

$$f(x_{\text{opt}}) \leq f(x) \text{ para toda } x \in F$$

Donde *F* es el espacio de soluciones. Si se trata de un problema de maximización x_{opt} debe satisfacer la condición de:

$$f(x_{\text{opt}}) \geq f(x) \text{ para toda } x \in F$$

El problema de optimización combinatoria también define una estructura de vecindades o de soluciones cercanas a la instancia *x*. Entonces un mecanismo de

generación es un medio para seleccionar una solución *z* dentro de la vecindad F_i de la solución *w.*

La búsqueda combinatoria es una técnica que busca la mejor aproximación al óptimo para problemas de optimización combinatoria con complejidad NP-Dura con altos tiempos de ejecución. Modelando el problema de agrupación, tenemos un conjunto de datos x_m, m = 1, …, n se busca organizar en k subgrupos C={C_1,…,C_k}. Con esto se obtiene la siguiente fórmula para particionar todo el conjunto de datos (Jain, *et al*., 1999)**.**

$$P(n,k) = \frac{1}{k!}\sum_{i=1}^{k}(-1)^{k-i}C_k^i i^n$$

Basados en la búsqueda combinatoria se encuentran en los algoritmos de agrupamiento genético GGA por sus siglas en inglés (Hall, *et al*., 1999), CLUSTERING (Tseng, Yang, 2001), y el Recocido Simulado SA por sus siglas en inglés (Kirkpatrick, *et al*., 1983).

Búsqueda Tabú TS por sus siglas en inglés, (Glover, Laguna, 1997) es una técnica de búsqueda combinatoria que usa la lista tabú para encaminar la ruta a seguir. Un conjunto de soluciones candidatas son generadas de la solución actual. Estas soluciones representan las asignaciones de *n* objetos en *k* grupos, la mejor solución de las candidatas se convierte en la nueva solución actual y es agregada a la lista tabú. En caso de que haya un empate con las soluciones candidatas se vuelven a generar todas las soluciones candidatas. El procedimiento durará el máximo número de iteraciones posible.

Otra solución al problema de optimización combinatoria consiste en el algoritmo de búsqueda local (De Los Cobos, *et al*., 2010), planteando una instancia de un problema con su respectiva estructura de vecindades *F*, entonces *o* es un óptimo local con respecto a *F* si *o* es mejor que o igual a todas las soluciones vecinas con respecto a la función objetivo. El pseudo-código del algoritmo de búsqueda local queda de la siguiente manera.

Inicio
  ESTABLECE ($o_{inicial}$)
  $o := o_{inicial}$
  Haz
    GENERA ($p \epsilon F_o$)
    Si $f(p) \leq f(o)$ entonces $o := p$

Hasta $f(p) \geq f(o)$, para toda $p \in F_o$
Fin

Los algoritmos de búsqueda local frecuentemente se quedan estancados en un óptimo local, por lo que se han desarrollado técnicas de optimización como algoritmos genéticos y programación evolutiva, todas las anteriores hacen uso de las operaciones de selección, combinación y mutación. Sin embargo hay soluciones para atenuar las desventajas del algoritmo de búsqueda local como son:

- Ejecutar el algoritmo de búsqueda para varias soluciones iniciales.
- Generar estructuras de vecindades complejas, para que se abarque lo más posible del conjunto de soluciones.
- Descartar para futuras ejecuciones las soluciones que incrementen (para el caso de minimización) la función objetivo.

Aplicando el problema al modelo de coloración de gráficas suaves se vuelve un problema de optimización combinatoria porque dependiendo del número de nodos, las combinaciones de coloración con *k* colores lo convierten en un problema NP-Duro. Partiendo de una solución de coloración inicial el algoritmo buscará todas las soluciones a la función objetivo hasta que se encuentre el menor valor de coloración; satisfaciendo así el problema de minimización.

Es importante para las técnicas de búsqueda una buena selección de parámetros para evitar elevar el costo computacional del algoritmo. También cobra relevancia encontrar el balance entre tener un costo computacional bajo, contra la garantía de encontrar el óptimo global.

2.13. Agrupamiento difuso.

A diferencia de las otras técnicas de agrupamiento donde un objeto tiene que ser colocado en uno y sólo un grupo, el agrupamiento difuso es más flexible y un objeto puede pertenecer a todos los grupos con cierto grado de pertenencia (Zadeh, 1965). Útil cuando las diferencias entre los grupos no están bien delimitadas o se encuentran ambigüedades, sin embargo los objetos pueden establecer relaciones más complejas que con otros algoritmos de agrupamiento.

El algoritmo difuso de *c medios* FCM por sus siglas en inglés es uno de los algoritmos difusos más populares, en él se trata de encontrar una división o grupos difusos para un conjunto de datos x_m para m=1,…,n minimizando la función de costo (Höppner, *et al*., 1999).

El algoritmo FCM es poco efectivo con la presencia de ruido y se le dificulta encontrar la división inicial. FCM alterna el cálculo del grado de pertenencia con la matriz prototipo, lo cual causa un elevado costo en el procesamiento para grandes volúmenes de datos.

Las teorías difusas también pueden ser utilizadas para crear estructuras jerárquicas y explorar los diferentes niveles de la estructura de los datos mientras se establecen las conexiones entre cada objeto y grupo con los demás miembros, también pueden aplicarse para redes neuronales.

2.14. Agrupamiento por redes neuronales.

El agrupamiento basado en redes neuronales utiliza una técnica de resonancia adaptativa (ART por sus siglas en inglés) donde las neuronas se comportan de forma competitiva. (Baraldi, Alpaydin, 2002) Las neuronas activas refuerzan su vecindario desde el interior mientras reprimen las actividades de las neuronas que se encuentran en frontera (comportamiento encendido-centro/apagado-envoltura).

La auto-organización de mapas de características SOFM por sus siglas en inglés (Kohonen, 2001) tiene como objetivo representar las entradas de la red neuronal como un conjunto de vectores entrelazados en forma de matriz o enrejado. Cada unidad del enrejado es tratado como una neurona y se interconecta con las neuronas adyacentes, convirtiendo a la red neuronal en el espacio de dimensiones. Los datos de entrada se conectan a toda la red neuronal a través de los pesos y durante el proceso de entrenamiento se va construyendo el enrejado con las similitudes encontradas, haciendo del algoritmo SOFM un método efectivo para mostrar la estructura que comparten los datos y no tanto como un algoritmo de agrupamiento.

La arquitectura básica de ART1 (Moore, 2001) consiste en dos capas de nodos, la representación de los atributos K_1 y la representación de las clases C_2, éstos están conectados con pesos adaptativos de abajo hacia arriba W^{12} y de arriba hacia abajo W^{21}. Los prototipos de los grupos se almacenan en la capa C_2. Después de activar la red neuronal se genera un candidato de agrupación en la capa C_1 y se compara con los datos de entrada para determinar si se han clasificado correctamente con base en cierto parámetro de vigilancia z ($0 \leq z \leq 1$). Una vez terminado el primer ciclo los pesos se adaptan al mismo tiempo (resonancia). Tal y como muestra la figura 2.2.

2.15. Agrupamiento de k-medias.

Los agrupamientos de *k*-medias se basan en el teorema de la cubierta, en el cual transformaciones no lineales de un conjunto de atributos dentro de un espacio multidimensional, se busca separar los atributos de forma lineal (Haykin, 1999). La dificultad consiste en el manejo del espacio multidimensional, es entonces donde entra un producto interno de núcleo, para evitar el alto procesamiento computacional al mapear y transformar las entradas en el espacio multidimensional.

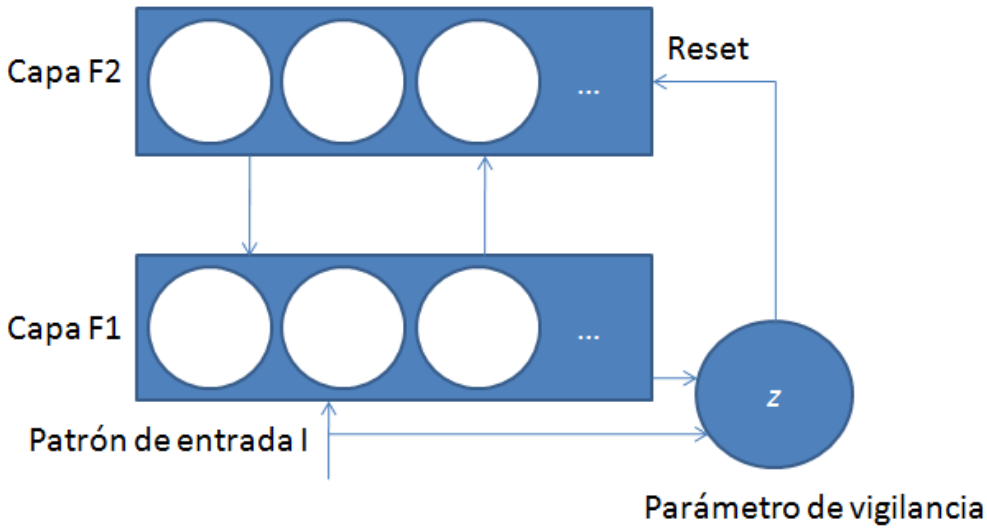

Figura 2.6 Arquitectura ART1 (Moore, 2001): Dos capas se interconectan mediante los pesos adaptativos. Sus interacciones son controladas a través de un parámetro de vigilancia.

En el siguiente algoritmo supóngase un conjunto de datos x_m con m=1,…,n y un mapeo no lineal $\phi : \Re^d \rightarrow F$ donde F es el nuevo espacio de dimensiones arbitrarias. El objetivo es encontrar k centros que puedan minimizar la distancia entre el atributo mapeado y su centro más cercano (Schölkopf, *et al.*, 1998).

$$\|\phi(x) - m_l\|^2 = \left\| \phi(x) - \sum_{j=1}^{N} \tau_{lj} \phi(x_j) \right\|^2$$
$$= k(x,x) - 2\sum_{j=1}^{N} \tau_{lj} k(x, x_j) + \sum_{i,j=1}^{N} \tau_{li}\tau_{lj} k(x_i, x_j)$$

Donde m_l es el centro del grupo *l* y se encuentra en un intervalo $\phi(x_1),\ldots, \phi(x_N)$ y el producto núcleo interno se define como $k(x, x_j) = \phi(x) * \phi(x_j)$.

Al igual que los algoritmos anteriores el agrupamiento por núcleo depende de la elección correcta de los parámetros y puede aumentar el costo computacional para grandes cantidades de datos. Se abre un importante panorama para plantear variantes no lineales a distintos tipos de algoritmos y al operador núcleo para la simplificación de los mismos.

2.16. Agrupamiento de datos secuenciales.

Los datos secuenciales son sucesiones con longitud variable y características distintivas como comportamientos dinámicos, restricciones temporales y gran volumen. Los datos secuenciales se obtienen de procesamiento del lenguaje natural, minería de datos, diagnósticos médicos, bolsa de valores, transacciones bancarias, minería de datos global, análisis de sensibilidad, etc. En la actualidad los datos secuenciales han crecido de manera explosiva y se necesita de una herramienta confiable y robusta que pueda procesar toda esa información. Las técnicas de agrupamiento buscan patrones dentro de la enorme cantidad de datos secuenciales en el contexto del aprendizaje no supervisado (Sun, Giles, 2000).

2.17. Agrupamiento a gran escala.

La escalabilidad de un algoritmo de agrupamiento cobra gran relevancia por el vertiginoso crecimiento de los datos y la complejidad que representa el poder procesar tal cantidad de información. Con los avances en las bases de datos y en tecnologías de Internet los algoritmos tienen que adecuar sus técnicas de optimización para poder manejar grandes volúmenes de datos en un tiempo razonable al igual que la complejidad computacional no crezca a la misma tasa de los datos (Xu, Wunsch, 2005).

2.18. Exploración de datos multidimensionales.

Los clasificadores revisados anteriormente pueden trabajar grandes cantidades de datos bajo ciertas condiciones ideales, aunque no son muy efectivos para analizar datos multidimensionales, por el crecimiento exponencial que representa el agregar más atributos o dimensiones a los datos. Se ha probado que la distancia entre los datos más cercanos no difiere de los otros datos o puntos cuando las dimensiones del espacio son suficientemente altas (Beyer, *et al*., 1999). Eso quiere decir que los algoritmos basados en métricas tampoco son muy efectivos para separar datos multidimensionales. Una reducción en las dimensiones es importante para el análisis

grupal tanto para reducir el costo computacional como para poder direccionar la información, afrontando por otro lado la pérdida de información y pueda alterar la interpretación de los datos; una reducción de dimensiones mal planteada puede incluso distorsionar el sentido de la información.

El análisis de componentes principales (PCA por sus siglas en inglés) (Dony, 2001) consiste en una transformación para la reducción de dimensiones mediante la extracción de las componentes de los datos originales construyendo una combinación lineal de un conjunto de vectores que describan las variaciones de los datos.

La reducción de dimensiones es utilizada también en los algoritmos de análisis de componentes independientes (ICA por sus siglas en inglés) (Cherkassky, Mulier, 1998), escalamiento multidimensional (MDS) (Duda, *et al*., 2001), incursión linealmente local (LLE) (Roweis, Saul, 2000).

3. Coloración de Gráficas suaves

La coloración de gráficas suaves es una generalización del problema de coloración de grafos. Consiste en colocar una etiqueta de coloración a un vértice de un grafo completo y ponderado en aristas de tal modo que la suma de las aristas pintadas con el mismo color en los vértices es mínima (Ramírez, 2001). Al ser una generalización del problema de coloración robusta, se sabe que este problema es del tipo NP-Duro y se necesita de técnicas metaheurísticas para gráficas con más de 20 vértices.

3.1. Problemática

Se busca reducir la dureza de la coloración, es decir, una coloración que minimice la suma de las penalizaciones entre las aristas adyacentes cuyos vértices tienen el mismo color. Una penalización, para fines de este proyecto, se considera como la distancia entre los vértices. Sea el grafo completo no dirigido *G = (V,E)* con un conjunto de vértices *V= {1,2,…,n}* y un conjunto de aristas *E* comportándose de la siguiente manera (Lara-Velázquez, *et al*., 2015).

$$G= (V,E)\ ;\ |V|=n;\ |E|=n(n-1)/2$$

Existe una penalización por arista (*i,j*), denotada por p_{ij} tal que:

$$p_{ij} \geq 0, \text{Para todo } i,j \in E$$

Una función de coloración de vértices del grafo con *k* colores que sirven como etiquetado del vértice *i* se define como:

$$C^k : V \rightarrow \{1,2,\ldots,k\}$$

Para una coloración C^k en el grafo, la función de dureza de la suma de las penalizaciones con el mismo color en los extremos está dada por:

$$H(C^k) = \sum_{(i,j)\in E,\, C^k(i)=C^k(j)} p_{ij}$$

El objetivo es encontrar la coloración C^k_{op} que minimice la dureza $H(C^k_{op})$.

3.2. Propiedades

Solidez de una coloración.

Dada una coloración de *k* colores sobre un grafo completo con *n* vértices el promedio de los vértices con el mismo color *m* viene dado por *m=n/k* y el número de aristas que comparten los *m* vértices viene dado por *C(m,2)=m(m-1)/2* con un número promedio de penalizaciones o valores de aristas que contribuyen a la dureza proporcional al promedio de vértices pintados con el mismo color multiplicado por el número de colores. Entonces la función de solidez de una coloración se define como la dureza de un grafo dividido por el número medio de aristas que contribuyen a la dureza:

$$S\left(C_{op}^{k}\right) = \frac{H\left(C_{op}^{k}\right)}{km(m-1)/2} = \frac{2H\left(C_{op}^{k}\right)}{k\frac{n}{k}\left(\frac{n}{k}-1\right)} = \frac{2kH\left(C_{op}^{k}\right)}{n(n-k)}$$

Resiliencia de una coloración.

La resiliencia de una coloración C^k consiste en el porcentaje en que disminuye la solidez de una coloración con *k*-1 colores respecto a una con *k* colores, expresado como:

$$R\left(C_{op}^{k}\right) = \frac{S\left(C_{op}^{k-1}\right) - S\left(C_{op}^{k}\right)}{S\left(C_{op}^{k}\right)}$$

Si obtenemos la resiliencia de todas las coloraciones posibles desde 1 hasta *k* los valores más grandes permitirán las mejores opciones de número de clases que se pueden utilizar para clasificar un conjunto.

3.3. Modelo Binario Entero.

El modelo de programación binaria es utilizado para obtener la coloración óptima C^k_{op} sobre el grafo no dirigido *G*=(*V,E*) de orden |*V*|=*n* y de tamaño |*E*|=*n*(*n*-1)/2, definido como:

$$\min z = \sum_{(i,j)\in E} p_{ij}\, y_{ij}$$

Sujeto a:

$$\sum_{l=1}^{k} x_{il} = 1 \; \forall \; i \in \{1, \dots, n\}$$

$$x_{il} + x_{jl} - 1 \leq \; y_{ij} \; \forall \; (i,j) \in E \; y \; \forall \; l \; \in \{1, \dots, k\}$$

$$y_{ij} \in \{0,1\} \; \forall (i,j) \in E$$

Donde:

$$x_{il} = \begin{cases} 1 \; si \; C(i) = l \text{ para todo } i \in \{1, \dots, n\} \; y \; \forall \; l \; \in \{1, \dots, k\}, \\ 0 \text{ en otro caso} \end{cases}$$

$$y_{ij} = \begin{cases} 1 \text{ si para alguna } l \; \in \{1, \dots, k\} \text{ se cumple que } x_{il} = x_{jl} = 1 \\ 0 \text{ en otro caso} \end{cases}$$

Las ecuaciones anteriores garantizan que cada vértice sea etiquetado con uno y sólo un color; las desigualdades asignan valor de uno a y_{ij} si se colorean con el mismo color a los vértices *i*, *j* y cero cuando se coloreen con colores diferentes. Activando para el primer caso la penalización p_{ij} en la función objetivo *z*. El modelo encuentra como solución la coloración mínima C^k_{op} que vale *l* si $x_{il} = 1$ donde *i* y *l* son las variables de decisión a partir de la solución óptima y el valor mínimo de la función objetivo *f(x)* es igual a $H(C^k_{op})$.

El número total de variables en el modelo binario es *nk + n(n-1)/2*. El número de ecuaciones es *n* y el número de desigualdades es *kn(n-1))/2* haciendo un total de restricciones igual a $n + kn(n-1)/2$.

Ejemplo.

Dada la siguiente gráfica se utilizarán las métricas: Euclidiana, Euclidiana elevada al cuadrado y Manhattan para establecer las distancias entre vértices.

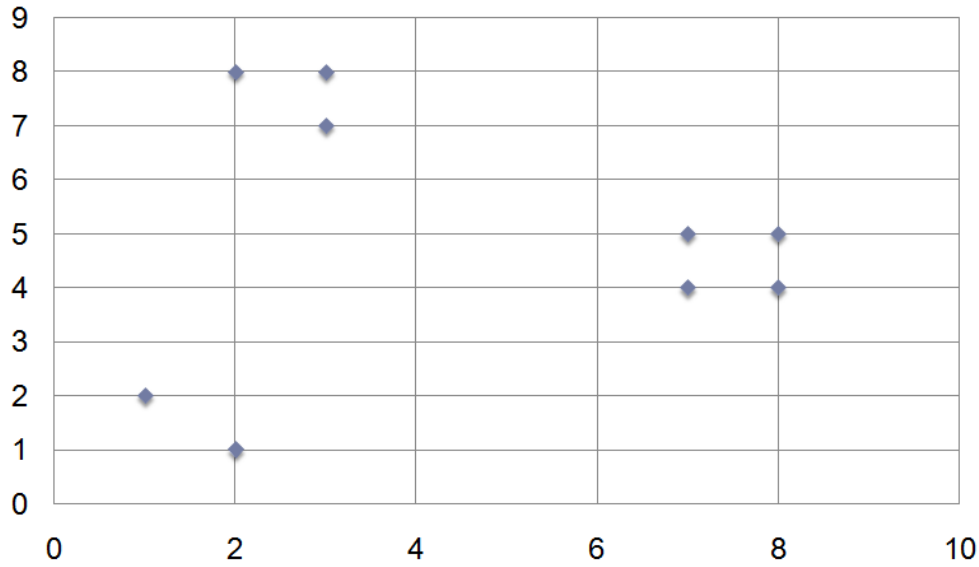

Figura 3.1 Ejemplo de gráfica con cuadrícula para ejemplificar las distancias.

Número de colores	Dureza	Solidez	Resiliencia
1	226	6.2778	
2	58	3.6825	0.7047
3	14	1.5556	1.3673
4	8	1.4222	0.0938
5	5	1.3889	0.0240
6	3	1.3333	0.0417

Tabla 3.1 Resultados de Resiliencia con la distancia Manhattan.

Número de colores	Dureza	Solidez	Resiliencia
1	173.4	4.8172	--
2	49.6	3.1494	0.5295
3	11.66	1.2952	1.4316
4	6.828	1.2139	0.0669
5	4.414	1.2262	-0.0100
6	3	1.3333	-0.0804

Tabla 3.2 Resultados de Resiliencia con la distancia Euclidiana.

Número de colores	Dureza	Solidez	Resiliencia
1	1036	28.7778	--
2	256	16.2540	0.7705
3	14	1.5556	9.4490
4	8	1.4222	0.0938
5	5	1.3889	0.0240
6	3	1.3333	0.0417

Tabla 3.3 Resultados de Resiliencia con la distancia Euclidiana cuadrática.

Como se puede observar en las tablas anteriores, utilizar 2 colores es una buena solución, sin embargo…

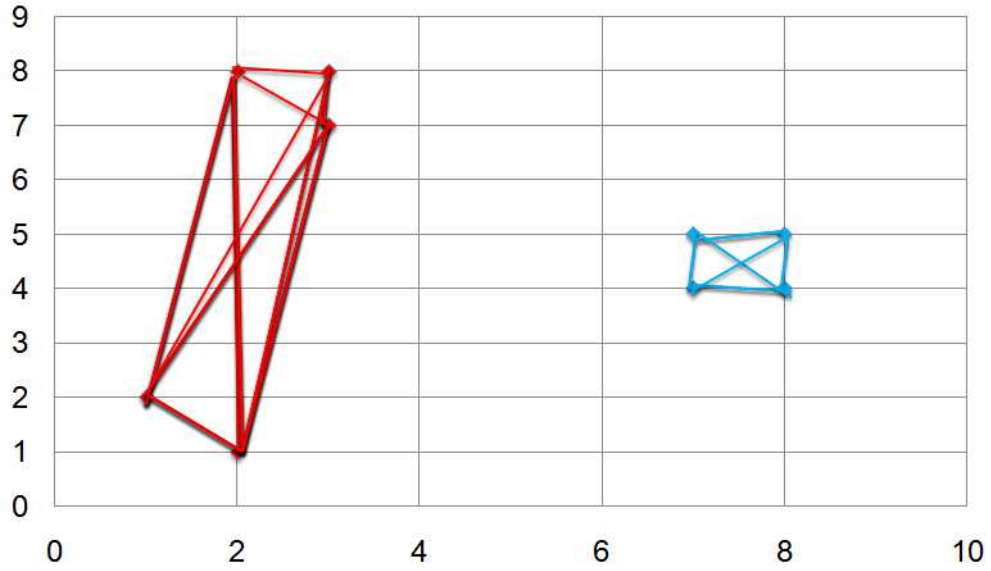

Figura 3.2 Clasificación de una gráfica con 2 colores.

Cuando se utilizan los 3 colores recomendados se obtiene el mayor valor de resiliencia, hasta un 944% de mejora para la distancia euclidiana cuadrática.

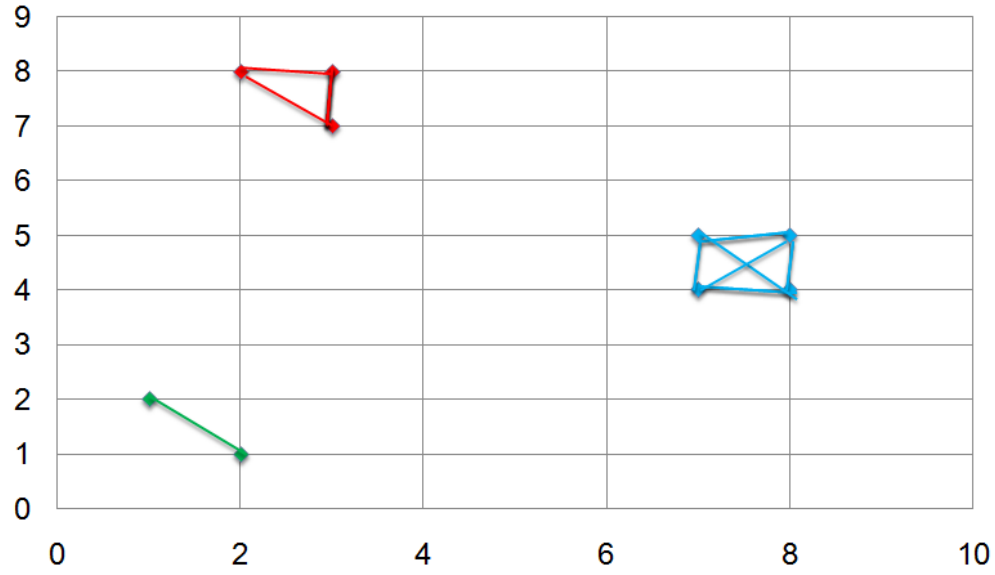

Figura 3.3 Clasificación de una gráfica con 3 colores.

3.4. Solución mediante Recocido Simulado.

Recocido Simulado es una técnica metaheurística para solucionar problemas de optimización combinatoria, está basado en el proceso de recocido de sólidos de la metalurgia y produce buenas soluciones en un tiempo polinomial (De Los Cobos, *et al*., 2010).

El proceso de recocido de un sólido consiste en primero elevar un sólido a altas temperaturas seguido de un proceso de enfriamiento paulatino, el efecto que produce es que las partículas se acomodan siguiendo una estructura de cristal, debe tenerse cuidado en establecer una temperatura adecuada para el equilibrio térmico, si se baja demasiado rápido la temperatura el sólido puede adoptar una estructura amorfa consecuencia de una mala cristalización.

El equilibrio térmico está definido por la siguiente ecuación donde la probabilidad del estado actual *x* del sólido con energía E_x a la temperatura *T* se muestra a continuación:

$$\text{Probabilidad}_T\{X = x\} = \frac{1}{Z(T)} exp\left(\frac{-E_x}{k_B T}\right)$$

Donde k_B es la constante de Boltzman y *Z(T)* es conocida como función partición dada por la siguiente ecuación:

$$Z(T) = \sum_y exp\left(\frac{-E_y}{k_B T}\right)$$

La suma comprende todos los posibles estados *y* del sólido, como se trata de una distribución de probabilidad, la probabilidad de cada estado es mayor o igual que cero y la suma de todas las probabilidades de todos los estados es igual a uno. Lo que resume el proceso de recocido simulado a dos etapas fundamentales (De Los Cobos, *et al*., 2010):

- Incrementar la temperatura a un valor máximo.
- Disminuir la temperatura lentamente hasta que las partículas se reacomoden en un estado de mínima energía llamado estado fundamental del sólido.

Si la energía del estado *y* es menor o igual que la energía del estado *x* entonces se acepta como el estado actual el estado *y* con una probabilidad dada por la ecuación:

$$E_x - E_y$$

En caso contrario el estado *y* se acepta con una probabilidad que está dada por la ecuación:

$$exp\left(\frac{E_x - E_y}{k_B T}\right)$$

Este proceso de evaluación se recomienda llevarlo a cabo en un número grande de iteraciones.

Haciendo la analogía al flujo computacional se toman las siguientes consideraciones:

- Los estados físicos del material se pueden tomar como las diferentes soluciones de un problema de optimización combinatoria.
- El costo es equivalente a la energía del estado.
- El criterio para aceptar la solución del estado *y* respecto a *x* viene definido por:

$$\text{Probabilidad}\{\text{aceptar } y\} = \begin{cases} 1 & si\ f(y) \leq f(x) \\ exp^{\left(\frac{f(x)-f(y)}{T}\right)} & si\ f(y) > f(x) \end{cases}$$

Donde *f(x)* y *f(y)* son los costos de la solución y *T* es la temperatura o parámetro de control definido por el programador.

La implementación del algoritmo consiste en definir la solución inicial x_0, el parámetro de control inicial t_0 y el número de iteraciones L_0 necesarias para alcanzar el equilibrio térmico (x_0, t_0, L_0), el ciclo finalizará cuando se cumpla(n) la(s) condición(es) de paro establecidas por el programador.

Inicio

Inicializar (x_0, t_0, L_0)

i=0

t_i=t_0

x= x_0

L_i= L_0

Haz

Para l=1 hasta L_i

Generar (*y* de F_x) donde F_x representa el espacio de soluciones vecinas

Si $f(y) \leq f(x)$ entonces *x*=*y*

Si no

Si $e^{\left(\frac{f(x)-f(y)}{t_i}\right)}$ > que un número aleatorio entre [0, 1) entonces *x*=*y*

Fin Para

i=i+1

CALCULAR LONGITUD (L_i)

CALCULAR CONTROL (t_i)

Hasta criterio de paro

Fin

3.5. Solución mediante Análisis de Regresión.

En los casos donde el modelo no ofreció una mejor solución que los clasificadores con entrenamiento, para solucionarlo se optó por un análisis de regresión porque ofrece una solución probada y estandarizada de análisis de datos en la ingeniería, y así poder determinar qué columnas influyen más en el modelo. En estos casos el modelo pasó de no supervisado a semisupervisado.

El análisis de regresión es una técnica muy utilizada para explorar las relaciones entre dos o más variables de un modelo empírico (Montgomery, Runger, 2003), también es utilizada para construir un modelo que prediga el comportamiento de las variables y para procesos de optimización. Se tienen datos dispersos y sabiendo que no existe una curva simple que pase por todos los puntos, hay una tendencia de que los puntos se distribuyen aleatoriamente pero cercanos a una línea recta que pueda describir su comportamiento.

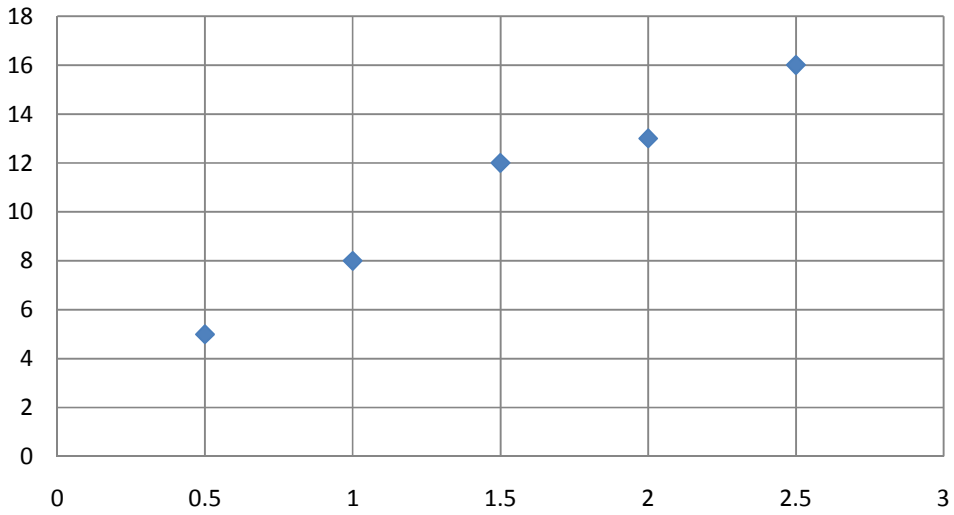

Figura 3.4 Conjunto de datos dispersos.

En la figura 3.4 puede apreciarse un conjunto de datos dispersos pero que tienden a seguir un patrón de distribución, la regresión lineal busca establecer un modelo lineal que pueda describir con una alta probabilidad de incluir la mayoría de los datos obtenidos empíricamente.

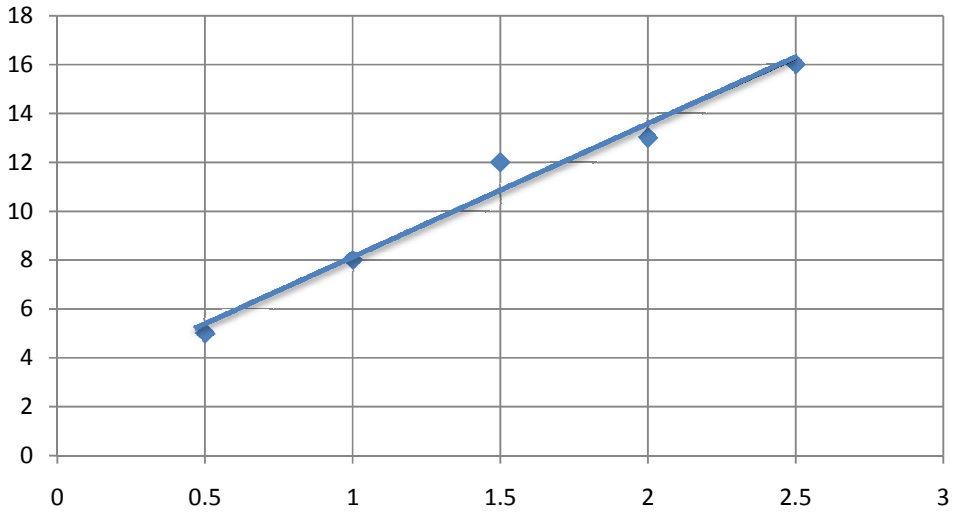

Figura 3.5 Modelo lineal que busca describir la correlación de los datos.

Una vez construido el modelo se encuentra una recta que describe una tendencia de mayor agrupamiento de los datos, asumiendo que el comportamiento de la variable dependiente es una función lineal de la variable independiente, el modelo matemático encuentra los valores de los coeficientes de regresión, de error y la varianza del error.

Existen indicadores que permiten establecer la exactitud del modelo de análisis de regresión, es decir, qué tan efectivo es el modelo lineal para describir el comportamiento de los datos (Montgomery, Runger, 2003), en ellos se puede encontrar qué tanto influyen las columnas en el resultado así como la correlación de los datos.

Ejemplo.

Dado un conjunto de datos, se utilizará la herramienta de Microsoft Excel para efectuar los cálculos del análisis de regresión, mediante la herramienta de Análisis de Datos.

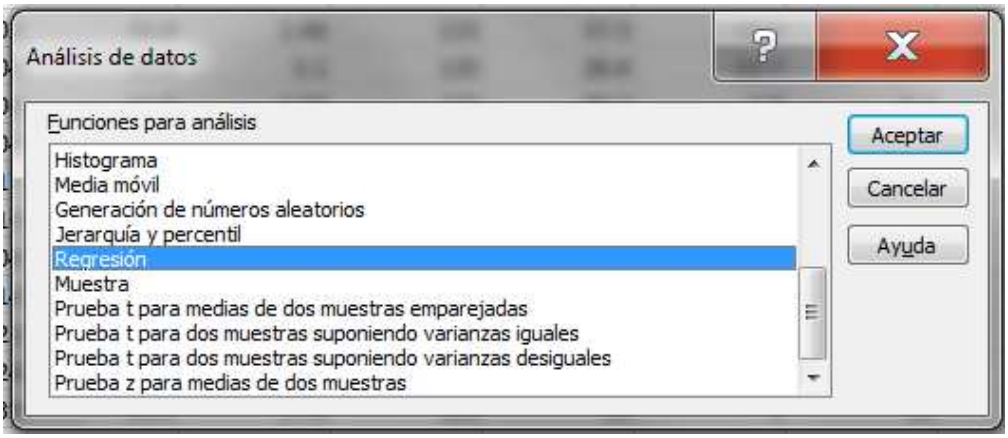

Figura 3.6 Herramienta de análisis de regresión en Microsoft Excel.

La opción requiere especificar el rango de entradas del modelo o rango *X* de entrada, así como el rango de salidas del modelo o rango *Y* de salida, se puede especificar el nivel de confianza y si los datos de entrada incluyen rótulos o encabezados, también se puede especificar si los resultados del análisis de regresión se pueden mostrar en la misma hoja de cálculo o en una nueva.

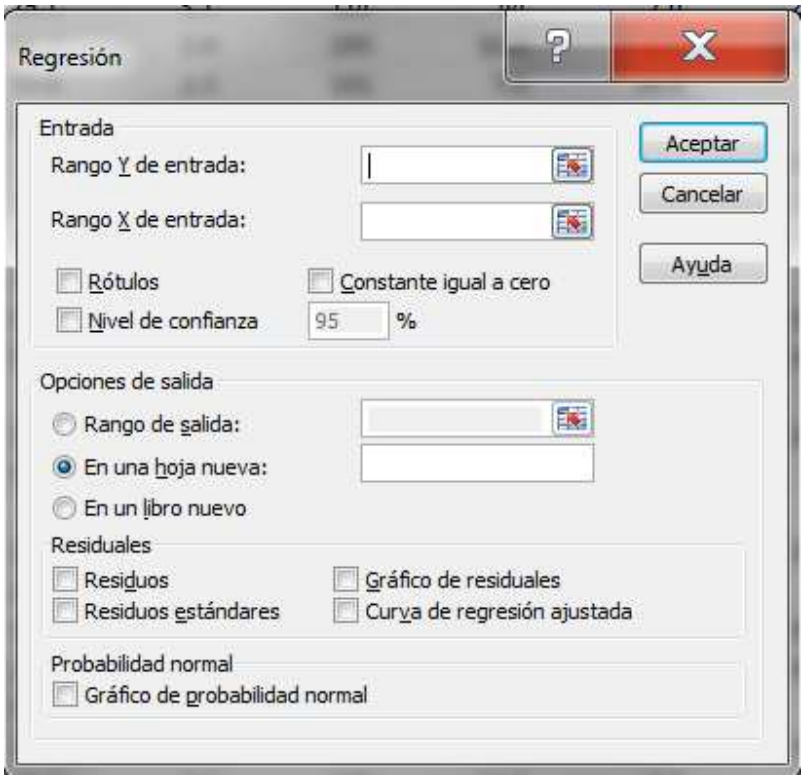

Figura 3.7 Opciones de la herramienta de análisis de regresión en Microsoft Excel.

Finalmente se mostrará una hoja con los resultados del análisis de regresión, la interpretación de los datos se explicará en el siguiente capítulo de este documento.

Estadísticas de la regresión									
Correlación múltiple	0.987217773								
R^2	0.97459893								
R^2 ajustado	0.966131907								
Error típico	0.795822426								
Observaciones	5								
ANÁLISIS DE VARIANZA									
	Grados de libertad	*Suma de cuadrados*	*Promedio de los cuadrados*	*F*	*Valor crítico de F*				
Regresión	1	72.9	72.9	115.1052632	0.001731447				
Residuos	3	1.9	0.633333333						
Total	4	74.8							
	Coeficientes	*Error típico*	*Estadístico t*	*Probabilidad*	*< 95%*	*> 95%*	*< 95.0%*	*> 95.0%*	
Clase	2.7	0.834665602	3.234828409	0.048039217	0.04372154	5.35627846	0.04372154	5.35627846	
Atributo 1	5.4	0.503322296	10.72871209	0.001731447	3.79820382	7.00179618	3.79820382	7.00179618	

Figura 3.8 Resultados del análisis de regresión en Microsoft Excel.

4. Propuesta

Para generar la matriz de distancias de una cierta base de datos (BD) se diseñó el siguiente procedimiento:

Es necesario evaluar la eficiencia del modelo de coloración de gráficas suaves y para ello se requiere probar su funcionamiento con diferentes tipos de bases de datos. Las bases de datos se obtuvieron del repositorio de Aprendizaje Automático de la Universidad de California Irvine (University of California, 2015).

Una base de datos se compone de registros o tuplas en el que se describen sus características o atributos y distribuidos en varias clases según sea el caso, porque como se verá en el siguiente capítulo, en el caso de la base de datos Stone Flakes sólo se hace una sugerencia de clasificación. Debido a que la ausencia de los datos disminuye la eficacia del método, se decidió eliminar los datos faltantes como se explica a continuación.

Las bases de datos del repositorio siguen un estándar: están disponibles en un archivo de texto plano, todas las características están separadas por comas y todos los datos faltantes se representan con el símbolo "*?*".

```
1,30,2,1,2,2,2,2,1,2,2,2,2,2,1,85,18,4,?,1
1,50,1,1,2,1,2,2,1,2,2,2,2,2,0.9,135,42,3.5,?,1
1,78,1,2,2,1,2,2,2,2,2,2,2,2,0.7,96,32,4,?,1
1,31,1,?,1,2,2,2,2,2,2,2,2,2,0.7,46,52,4,80,1
1,34,1,2,2,2,2,2,2,2,2,2,2,2,1,?,200,4,?,1
1,34,1,2,2,2,2,2,2,2,2,2,2,2,0.9,95,28,4,75,1
2,51,1,1,2,1,2,1,2,2,1,1,2,2,?,?,?,?,?,1
1,23,1,2,2,2,2,2,2,2,2,2,2,2,1,?,?,?,?,1
1,39,1,2,2,1,2,2,2,1,2,2,2,2,0.7,?,48,4.4,?,1
1,30,1,2,2,2,2,2,2,2,2,2,2,2,1,?,120,3.9,?,1
1,39,1,1,1,2,2,2,1,1,2,2,2,2,1.3,78,30,4.4,85,1
1,32,1,2,1,1,2,2,2,1,2,1,2,2,1,59,249,3.7,54,1
1,41,1,2,1,1,2,2,2,1,2,2,2,2,0.9,81,60,3.9,52,1
1,30,1,2,2,1,2,2,2,1,2,2,2,2,2.2,57,144,4.9,78,1
1,47,1,1,1,2,2,2,2,2,2,2,2,2,?,?,60,?,?,1
1,38,1,1,2,1,1,1,2,2,2,2,1,2,2,72,89,2.9,46,1
1,66,1,2,2,1,2,2,2,2,2,2,2,2,1.2,102,53,4.3,?,1
1,40,1,1,2,1,2,2,2,1,2,2,2,2,0.6,62,166,4,63,1
1,38,1,2,2,2,2,2,2,2,2,2,2,2,0.7,53,42,4.1,85,2
```

Figura 4.1 Ejemplo de un archivo de BD, en él se muestra la representación de un dato faltante.

Las bases de datos utilizadas para el modelo tuvieron que cumplir con las siguientes características para que fueran consideradas:

1) **Atributos de características numéricas:** El modelo requiere que todos los datos sean numéricos, para el caso de datos alfanuméricos se estableció una ponderación y se reemplazaron dichos valores por un número.
2) **No deben haber datos faltantes:** Es necesario que para cada ejemplo se tengan características de todas las tuplas, ya que todo modelo es sensible a datos faltantes y queremos obtener la mejor eficiencia posible.
3) **Los datos deben estar normalizados:** Significa que la distribución de las características de todos los datos debe estar entre 0 y 1. Esto con el objetivo de que todos los datos tengan la misma importancia a priori.

Estando conscientes de que es sumamente difícil que hayan bases de datos que cumplan con las características por defecto, las bases de datos a evaluar pasaron por un proceso de pre-procesado (limpieza, ponderación y normalización) antes de que fueran incluidas en el modelo.

4.1. Limpieza de datos faltantes.

La primera fase del proceso consiste en excluir la tupla completa si en ella se encontraba al menos un dato faltante, esta fase es muy importante dado que el modelo es sensible al ruido. El proceso de limpieza se describe a continuación en pseudocódigo:

```
Abrir archivo de origen

  Mientras no se alcance el final del archivo

        Capturar renglón del archivo

        Si cadena no contiene "?" entonces

              ///////****Proceso de captura de datos******///////

        Fin Si

  Fin Mientras

Cerrar archivo de origen
```

4.2. Ponderación de columnas alfanuméricas.

En algunos casos hay columnas con valores alfanuméricos que tienen un orden intrínseco, por ejemplo escalas de malo, bueno y excelente o paciente vivo y paciente muerto. Como el modelo trabaja exclusivamente con datos numéricos se otorgó de un valor numérico a cada uno de los datos, por ejemplo, malo se sustituyó por un 0 y bueno con uno y excelente con un 2. Por otra parte se codificó vivo por un 1 y muerto

por un 0. Este proceso se efectuó de forma manual, debido a que las bases de datos tenían diferente información.

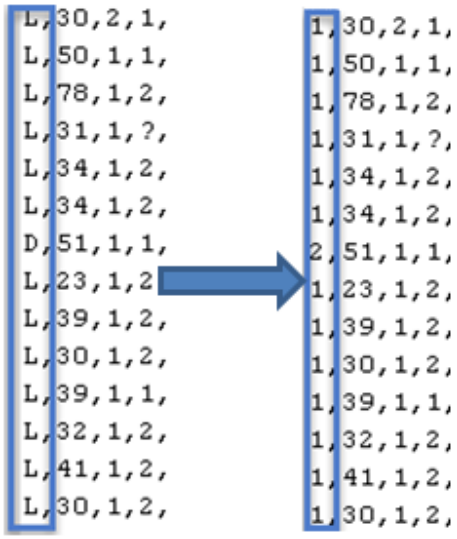

Figura 4.2 Proceso de sustitución de datos alfanuméricos.

4.3. Normalización de los datos.

Algunas columnas cuentan con escalas mayores a otras, por ejemplo hay columnas con métrica en miligramos (dosis) y otras columnas con métrica en kilogramos (peso del paciente), el modelo considera que todas las columnas tienen la misma influencia para la clasificación, así que para evitar que la magnitud de cualquier atributo influya más que los otros, todas las bases de datos se sometieron a un proceso previo de normalización.

La normalización consiste en tomar los valores mínimos y máximos de cada columna, una vez obtenido los límites, los demás valores se sustituirán con la siguiente fórmula, garantizando que todos los valores del atributo tengan una distribución entre 0 y 1.

$$x_i = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

Donde:

x_{min} es el valor más pequeño de una columna de la base de datos.

x_{max} es el valor más grande de una columna de la base de datos.

Si los valores máximos y mínimos son los mismos, se dará valor de 1 a todos los valores de la columna, evitando así la división entre cero.

4.4. Matriz de distancias.

La última fase de la estandarización de los datos consiste en obtener la matriz de distancias. Todas las distancias de las instancias de prueba fueron evaluadas con la distancia Euclidiana cuadrática porque después de probar con varios tipos de métricas explicadas en el capítulo 2, ésta nos dio mejores resultados porque acentúa la distancia de los nodos más lejanos al elevarlos al cuadrado y minimiza la distancia de los nodos más cercanos igualmente por elevarlos al cuadrado. La fórmula de la distancia Euclidiana al cuadrado se describe a continuación:

$$D_{ij} = \sum_{l=1}^{k} \left(x_{il} - x_{jl}\right)^2$$

Donde:

D_{ij} es la distancia euclidiana al cuadrado entre un nodo y otro, cabe aclarar que se asume que la base de datos es una gráfica completa con aristas ponderadas.

k es el número de características o atributos.

x_{il} y x_{jl} son los valores de dos nodos respecto a una misma característica.

El resultado es una matriz *nxn* simétrica con diagonal principal compuesta por ceros, donde *n* es el número de instancias trabajadas por el número de clases distintas del conjunto de prueba, por ejemplo si una base de datos es de 10 instancias por cada clase y tiene 2 clases, se obtiene una matriz de distancias de 20 *x* 20.

Con la matriz de distancias representando a la gráfica completa de los datos y sus correlaciones, se puede poner en marcha la técnica de coloración de gráficas suaves.

4.5. GAMS.

Una vez concluido el proceso de preparación hay que ejecutar el modelo sobre la matriz de distancias, para ello se utilizaron 2 tipos de procedimientos, para los casos en los que las Bases de Datos no excedían los 20 registros por ejemplo las tomas parciales y graduales de una base de datos tal y como se explica en el capítulo de Evaluación, se utilizó el Software de Licencia: *Sistema de Modelado Algebraico General* GAMS por sus siglas en inglés (GAMS Development Corporation, 2016). En el cual se usó el algoritmo binario explicado en el capítulo 3.

GAMS es un sistema de modelado algebraico para programación matemática y optimización, éste se compone de un lenguaje, un compilador y un módulo de ejecución optimizado, diseñado para el desarrollo de modelos de optimización y de programación matemática.

La ventaja de utilizar GAMS radica en que garantiza encontrar el óptimo para el problema que se haya ingresado, y su principal desventaja es que para las bases de datos grandes el tiempo y los costos de ejecución crecen demasiado.

Para poder enfrentarse a la desventaja del costo computacional se usó de una técnica metaheurística que redujera el tiempo de ejecución, para ello nos valimos de la técnica de Recocido Simulado que se explica en el capítulo 3.

4.6. Recocido Simulado.

Cuando la cantidad de datos es demasiado grande, el proceso de coloración puede tardar demasiado e implicar un costo computacional muy alto, por lo que se optó por aplicar una técnica metaheurística que permita resolver el problema reduciendo los costos computacionales, para este caso se optó por recocido simulado porque es una técnica de propósito general, permite definir las iteraciones, la estructura de las vecindades, el momento de evaluar la función objetivo y la facilidad para implementarla en la computadora.

4.7. Análisis de Regresión.

Tal y como se explicó en el capítulo 3, la técnica utilizada para mejorar la eficiencia del modelo en bases de datos complicadas fue el análisis de regresión y el software encargado de efectuar los cálculos de regresión lineal para las bases de datos que lo requirieron fue Microsoft Excel en su versión 2007. A continuación se muestra un ejemplo del análisis de regresión calculado en Excel junto con su interpretación.

Estadísticas de la regresión	
Coeficiente de correlación múltiple	0.964584039
Coeficiente de determinación R^2	0.930422368
R^2 ajustado	0.928502985
Error típico	0.219053816
Observaciones	150

Tabla 4.1 Modelo de regresión calculado por MS Excel.

La tabla 4.1 del análisis de regresión muestra los coeficientes de regresión, así como el coeficientes de error.

ANOVA					
	GL	*SS*	*MC*	*F0*	*F*
Regresión	4	93.04223675	23.26055919	484.7507687	8.21218E-83
Residuos	145	6.957763247	0.047984574		
Total	149	100			

Tabla 4.2 Análisis de Varianza (ANOVA) calculado por MS Excel.

En la tabla 4.2 se muestra el análisis de varianza ANOVA (Montgomery, Runger, 2003) como un modelo más certero para establecer la confianza del modelo, se incluyen la suma de los cuadrados (SS), los grados de libertad (GL), la media de los cuadrados (MC), F_0 obtenido y *F* como coeficiente de distribución de Fisher para comparar.

	Coeficiente	*Error Típico*	*t*	*p*		
Y	1.192083995	0.204698118	5.823619721	3.56594E-08		
X1	-0.109741463	0.057758651	-1.900000449	0.059418327		
X2	-0.044240447	0.059963135	-0.737794097	0.461831907		
X3	0.227001382	0.0569904	3.983151224	0.000107245		
X4	0.60989412	0.094474472	6.455649959	1.51962E-09		
<95%	*>95%*	*<95%*	*>95%*			
0.787506449	1.59666154	0.787506449	1.59666154			
-0.2238991	0.004416173	-0.2238991	0.004416173			
-0.162755156	0.074274262	-0.162755156	0.074274262			
0.114362163	0.339640602	0.114362163	0.339640602			
0.42316915	0.79661909	0.42316915	0.79661909			

Tabla 4.3 Estadísticos de las columnas calculado por MS Excel.

Finalmente en la tabla 4.3 se muestran los valores estadísticos de las variables dado un intervalo de confianza, en este caso con 4 variables independientes contra una variable dependiente, es aquí donde se explica la utilidad del análisis de regresión para este documento:

- Mientras más grande sea el valor de *t*, mayor es su influencia en la columna de resultado. Un valor negativo de *t* establece que mientras el valor de la variable dependiente crece, el valor de esa variable independiente decrece.
- El valor de *p* representa la probabilidad de que los valores de la columna valgan cero y no se altere el valor de la columna resultado, es decir, prescindir del atributo en el modelo.

Con la información proporcionada por el análisis de regresión, se podrán tomar acciones correctivas para mejorar la eficiencia del modelo, por ejemplo en la base de datos Iris que se verá en el siguiente capítulo serán:

- Invertir el signo de la característica longitud de sépalo (hoja que envuelve a la flor).
- Eliminar la columna ancho de sépalo.

5. Evaluación

La evaluación de las instancias se realizó en una PC con Windows 7 Professional de 64 bits con Service Pack 1, procesador Intel Xeon a 2.27 GHz y 4 GB de memoria RAM.

5.1. Base de datos Hepatitis (Chow, 2006).

Reúne los datos del tratamiento y los síntomas de 155 pacientes de hepatitis, cuenta con 19 características tomadas como atributos que son:

1. Edad del paciente, los datos oscilan entre los 7 y 78 años.
2. Sexo del paciente, hombre (1), mujer (2).
3. Tratamiento con esteroides, no (1) si (2).
4. Tratamiento con antivirales, no (1) si (2).
5. Síntoma de Fatiga, no (1) si (2).
6. Malestar general en el paciente, no (1) si (2).
7. El paciente presentaba un cuadro de anorexia, no (1) si (2).
8. Hígado grande, no (1) si (2).
9. Hígado duro, no (1) si (2).
10. Bazo fácilmente palpable, no (1) si (2).
11. Araña vascular o aparición de manchas rojizas en la piel, no (1) si (2).
12. Ascitis, es decir, presión alta en los vasos sanguíneos del hígado, no (1) si (2).
13. Várices en el paciente, entiéndanse como hinchazón de las venas, no (1) si (2).
14. Cantidad de Bilirrubina como consecuencia de una inflamación aguda del hígado, los datos se distribuyen entre 0.3 y 8 de forma continua.
15. Cantidad anormal de Fosfatasa alcalina, una enzima muy sensible a problemas del hígado o los huesos, fluctuando entre 26 y 295 en números enteros.
16. Presencia de Transaminasa sérica de glutamato-oxalacetato (SGOT por sus siglas en inglés), cuando esta enzima se encuentra en la sangre hay alerta de daños en el corazón o en el hígado, desde 14 hasta 648 todos números enteros.
17. Albúmina, una baja cantidad de esta proteína en la sangre es señal de células hepáticas dañadas, ubicados los datos entre 2.1 y 6.4.
18. Tiempo de Protrombina, un monitoreo de la coagulación sanguínea comúnmente usado, distribuido entre 0 y 100 en cantidades enteras.
19. Histología o si el paciente cuenta con expediente clínico, no (1) si (2).

La base tenía datos faltantes, así como valores alfanuméricos por lo que se tuvo que efectuar una previa ponderación. Los datos están divididos en 2 clases y están distribuidos de la siguiente manera:

- Vivo: 123 instancias.
- Muerto: 32 instancias.

Debido a que se trata de un clasificador no supervisado y no se conoce a priori la cantidad de clases, es necesario apoyarse en la propiedad de resiliencia de la coloración de gráficas suaves para poder obtener el óptimo de colores en la clasificación, para después compararlo con las clases predefinidas en la base de datos.

Nodos	Número de colores	Dureza	Solidez	Resiliencia
20	1	1083.0941	5.700495263	
20	2	398.5664	4.428515556	0.287224848
20	3	231.7599	4.089880588	0.082798253
20	4	149.2803	3.7320075	0.095892918
20	5	97.0654	3.235513333	0.153451436
20	6	70.4842	3.020751429	0.071095524
20	7	52.8542	2.845995385	0.061404191
20	8	41.8342	2.788946667	0.020455292
20	9	31.3302	2.56338	0.087995797
20	10	23.95254	2.395254	0.070191303

Tabla 5.1 Resultados de Resiliencia para la BD Hepatitis.

Por tiempos de ejecución se utilizó una muestra pequeña con 20 elementos y así poder reportar el óptimo de la función objetivo que es la del Modelo Binario Entero con el software GAMS. Se puede observar en la tabla 5.1 que el proceso sugiere 2 colores como óptimo para agrupar el conjunto de datos, tal y como sugiere la clasificación supervisada.

Se aplicaron los procesos de estandarización (limpieza, ponderación, normalización), se generó la matriz de distancias y se ejecutó el algoritmo de

coloración a varios casos con el proceso de recocido simulado obteniendo los siguientes resultados:

Instancias por clase	Número de colores	Tiempo de ejecución (segundos)	Porcentaje de eficiencia
2 vivos, 2 muertos	2	0.9165	100%
4 vivos, 4 muertos	2	1.9520	87.5%
5 vivos, 5 muertos	2	2.3223	90%
10 vivos, 10 muertos	2	9.4864	85%
20 vivos, 13 muertos	2	23.6332	84.8%

Tabla 5.2 Resultados de Clasificación de la BD Hepatitis.

La eficiencia se entiende como el número de "aciertos" del algoritmo de Coloración de Gráficas Suaves (CGS) respecto a la clasificación predeterminada de la base de datos, por ejemplo con una base de datos de 50 instancias y el algoritmo coincidió en 48 instancias con la clasificación predeterminada, la eficiencia es de 48/50 es decir de un 96%.

Para poder observar mejor el comportamiento del proceso de coloración, se fue incrementando gradualmente el número de instancias hasta abarcar toda la base de datos. Para este caso, la base de datos no tenía los datos uniformemente distribuidos en las clases predeterminadas, tal y como puede observarse en la tabla 5.2 sólo 13 instancias de una clase cumplieron con todos los filtros previos mientras que 20 instancias de la otra clase pasaron todos los filtros, la eficiencia de clasificación es buena incluso para una cantidad dispar de instancias por clase, en cuanto al factor tiempo, se pueden observar resultados buenos.

A continuación se compararán los resultados obtenidos con otras técnicas de clasificación, el elemento sombreado es el algoritmo de clasificación desarrollado en este documento.

Método y métrica	Eficiencia	Tipo de prueba
21 NN, Manhattan k=1	90.3	leave-one-out
Featuring Space Mapping	90	leave-one-out
14-NN, Euclidiana k=1	89	leave-one-out
LDA	86.4	leave-one-out
Coloración Gráficas Suaves , Euclidiana k=2	84.8	Subconjuntos sin entrenamiento.
CART (árbol de decisión)	82.7	leave-one-out
MLP+backprop	82.1	leave-one-out

Tabla 5.3 Fuente (Universidad Nicolás Copérnico Polonia, 2010) Comparación de clasificadores para la BD Hepatitis en pruebas leave-one-out.

Método y métrica	Eficiencia	Tipo de prueba
9NN con pesos	92.9	10 x cross validation
18NN Manhattan	90.2±0.7	10 x cross validation
FSM con rotaciones	89.7	10 x cross validation
15 NN Euclidiana k=1	89±0.5	10 x cross validation
FSM sin rotaciones	88.5	10 x cross validation
VSS 4 neuronas, 5 it	86.5±8.8	10 x cross validation
LDA,análisis lineal discriminante	86.4	10 x cross validation
Bayes ingenuo y semi NB	86.3	10 x cross validation
IncNet	86	10 x cross validation
QDA, Análisis cuadrático discrimiante	85.8	10 x cross validation
1-NN	85.3±5.4	10 x cross validation
VSS 2 neuronas, 5 it	85.1±7.4	10 x cross validation
ASR	85	10 x cross validation
Coloración GS, Euclidiana k=2	84.8	Subconjuntos sin entrenamiento.
Análisis discrimiante de Fisher	84.5	10 x cross validation
LVQ	83.2	10 x cross validation
CART (árbol de decisión)	82.7	10 x cross validation
MLP con BP	82.1	10 x cross validation
ASI	82	10 x cross validation
LFC	81.9	10 x cross validation
RBF	79	10 x cross validation
MLP+BP	77.4	10 x cross validation

Tabla 5.4 (Universidad Nicolás Copérnico Polonia, 2010) Comparación de clasificadores para la BD Hepatitis con prueba 10 x cross validation.

Como se puede observar en las tablas 5.3 y 5.4 el clasificador se encuentra bien posicionado respecto a los demás, lo que significa que los resultados de eficiencia son bastante buenos.

5.2. Base de datos Wine.

Un conjunto de resultados del análisis químico de 178 instancias de vinos de una misma región, cuenta con 13 características tomadas como atributos que son (Fernandez, *et al*., 2009) (Waterhouse, Ebeler, 1998):

1. Porcentaje de alcohol, cuyos valores oscilan entre 11.03 y 14.83.
2. Cantidad de Ácido Málico, responsable del sabor ácido de la fermentación de las frutas, el grado de acidez se encuentra entre 0.74 y 5.8.
3. Cenizas, como resultado de la calcinación del vino a 500°C, los valores se dispersan entre 1.36 y 3.23.
4. Alcalinidad de las cenizas, tratándose de la suma de los cationes de amonio mezclados en los ácidos del vino cuyos valores se encuentran entre 10.6 y 30.
5. Magnesio, importante para determinar las condiciones de almacenamiento del vino, las concentraciones varían entre 70 y 162 en números enteros.
6. Fenoles totales, o pigmentación del vino, distribuidos entre 0.98 y 3.88.
7. Flavonoides o pigmentos amarillos que aumentan al envejecer el vino blanco, cuyos valores se colocan entre 0.34 y 5.08.
8. Fenoles no flavanoides, otro tipo de elemento que influye en la coloración del vino, con valor mínimo de 0.13 y máximo de 0.66.
9. Proantocianinas, sustancia más importante de la uva para este caso, otorga las propiedades beneficiosas del vino a la salud humana, donde los valores oscilan entre 0.41 y 3.58.
10. Intensidad del color sin importar si el vino es blanco o tinto con datos desde 1.28 hasta 13.
11. Matiz del color del vino, de igual forma éste no distingue si el vino es blanco o tinto, su información de distribuye entre 0.48 y 1.71.
12. OD280/OD315, concentración de esas proteínas en vinos diluidos, los datos fluctúan entre 1.27 y 4.
13. Prolina, un aminoácido importante del metabolismo del nitrógeno de las levaduras, distribuido entre 278 y 1680 en número entero.

No hay datos faltantes, todos los valores son numéricos y los datos están distribuidos en 3 clases de la siguiente manera:

- Vino de mesa: 59 instancias.

- Vino de crianza: 71 instancias.
- Vino de reserva: 48 instancias.

En la siguiente tabla se muestra la cantidad óptima de colores que sugiere el algoritmo, para ello se utilizó la propiedad de resiliencia.

Nodos	Número de colores	Dureza	Solidez	Resiliencia
12	1	163.374	2.475363636	
12	2	49.4852	1.649506667	0.500669071
12	3	17.2711	0.959505556	0.719121538
12	4	10.8214	0.901783333	0.06400897
12	5	7.1476	0.850904762	0.059793497
12	6	4.5769	0.762816667	0.115477413
12	7	3.1643	0.738336667	0.033155607
12	8	2.1423	0.7141	0.033940158
12	9	1.3831	0.69155	0.03260791
12	10	0.715	0.595833333	0.160643357

Tabla 5.5 Resultados de Resiliencia para la BD Wine.

Por tiempos de ejecución se utilizó una pequeña muestra de las instancias, y así garantizar el óptimo de la función objetivo con el software GAMS. El modelo sugiere 3 colores como óptimo para agrupar el conjunto de datos, tal y como ya se sabía con anterioridad.

Se estandarizaron los datos y se usó el algoritmo de coloración a varios casos obteniendo los siguientes resultados:

Instancias totales	Número de colores	Tiempo de ejecución (segundos)	Porcentaje de eficiencia
6	3	1.6493	100%
9	3	2.5924	100%
12	3	3.9579	100%
15	3	5.8813	100%
30	3	17.4964	96.6%
60	3	59.6774	96.6%
90	3	126.4094	95.5%
Toda la BD	3	471.2716	93.25%

Tabla 5.6 Resultados de clasificación para la BD Wine.

Cabe aclarar que se utilizó una parte proporcional de cada clase para las pruebas preliminares, al momento de utilizar toda la base de datos, la proporción cambió a la original.

Analizando la tabla 5.6 tanto la eficiencia de clasificación como el tiempo de ejecución son bastante buenos incluso para una cantidad dispar de instancias por clase, aunque para analizar la eficiencia del algoritmo a mayor detalle fue necesario compararlo con otros clasificadores similares, como a continuación se detalla.

Método y métrica	Eficiencia	Tipo de Prueba
RDA	100	leave-one-out
QDA	99.4	leave-one-out
LDA	98.9	leave-one-out
kNN, Manhattan k=1	98.7	leave-one-out
1NN	96.1	leave-one-out
kNN, Euclidiana k=1	95.5	leave-one-out
kNN, Chebyshev k=1	93.3	leave-one-out
Coloración GS, Euclidiana k=2	93.2	Subconjuntos sin entrenamiento.

Tabla 5.7 (Universidad Nicolás Copérnico Polonia, 2010) Comparación de clasificadores para la BD Wine con prueba de leave-one-out.

Método y métrica	Eficiencia	Tipo de Prueba
kNN, Manhattan, k de 1-10	98.9 ±2.3	10 x cross validation
IncNet, 10CV, Gauss	98.9 ±2.4	10 x cross validation
10 CV SSV, con podado	98.3 ±2.7	10 x cross validation
10 CV SSV, 7 nodos	98.3 ±2.7	10 x cross validation
kNN, Euclidiana, k=1	97.8 ±2.8	10 x cross validation
kNN, Manhattan, k=1	97.8 ±2.9	10 x cross validation
kNN, Manhattan, k de 1-10	97.8 ±3.9	10 x cross validation
kNN, Eucliana, k=3, pesos alterados	97.8 ±4.7	10 x cross validation
IncNet, 10CV, bicentral	97.2 ±2.9	10 x cross validation
kNN, Euclidiana k de 1-10	97.2 ±4.0	10 x cross validation
10 CV SSV, nodos optmizados	97.2 ±5.4	10 x cross validation
FSM a=0.99	96.1 ±3.7	10 x cross validation
FSM 10CV, Gauss, a=0.999	96.1 ±4.7	10 x cross validation
FSM 10CV, triangular, a=0.99	96.1 ±5.9	10 x cross validation
kNN, Euclidiana k=1	95.5 ±4.4	10 x cross validation
Coloración GS, Euclidiana k=2	93.2	Subconjuntos sin entrenamiento.
10 CV SSV, nodos optimizados, BFS	92.8 ±3.7	10 x cross validation
10 CV SSV, nodos optimizados, BS	91.6 ±6.5	10 x cross validation
10 CV SSV, con podado, BFS	90.4 ±6.1	10 x cross validation

Tabla 5.8 (Universidad Nicolás Copérnico Polonia, 2010) Comparación de clasificadores para la BD Wine con prueba 10 x cross validation.

Para este caso también se obtuvieron buenos resultados, cabe mencionar que la fuente no especifica las características del modelo de clasificación que utilizaron.

5.3. Base de datos Iris.

Ésta es posiblemente la base de datos más conocida en la literatura de reconocimiento de patrones. Contiene 3 clases de 50 instancias cada una y cada clase corresponde a un tipo de planta. Los atributos de la base de datos son:

1. Longitud del sépalo (hoja que envuelve a la flor desde sus fases tempranas de desarrollo) en cm.
2. Ancho del sépalo en cm.
3. Longitud del pétalo (antófilo que forma parte de la corola de una flor) en cm.
4. Ancho del pétalo en cm.

Los tipos de planta se distribuyen en tres especies:

- Iris setosa.
- Iris versicolor.
- Iris virginica.

Se especifica de antemano que 1 clase puede ser separada linealmente de las otras dos, mientras que las últimas clases no pueden ser separadas de forma lineal, tomando en cuenta esa información, el primer color será compuesto por la Iris setosa, mientras que el segundo color corresponderá a la Iris versicolor y la Iris virginica.

Mientras se siga tomando en cuenta que sólo se pueden separar 2 tipos de planta por método lineal, los resultados son bastante favorables. En la siguiente tabla se muestra el óptimo de colores que sugiere el método.

Nodos	Número de colores	Dureza	Solidez	Resiliencia
15	1	99.5758	0.94834095	
15	2	13.7321	0.2816841	2.36668255
15	3	4.8804	0.16268	0.73152264
15	4	2.3024	0.11163152	0.45729456
15	5	1.4355	0.0957	0.16647351
15	6	0.7959	0.07074667	0.35271391
15	7	0.5276	0.06155333	0.14935557
15	8	0.3129	0.04768	0.29096756
15	9	0.2139	0.04278	0.1145395
15	10	0.1237	0.03298667	0.29688763

Tabla 5.9 Resultados de Resiliencia para la BD Iris.

Por cuestión de tiempo de ejecución se escogió un 10% de la BD para poder encontrar el óptimo con GAMS.

El algoritmo sugiere 2 colores como el óptimo de la coloración, lo que quiere decir que es incapaz de ver diferencias claras entre los otros 2 tipos de planta, confirmando lo explicado previamente. Las pruebas de agrupamiento arrojaron los siguientes resultados de eficiencia:

Instancias totales	Número de colores	Tiempo de ejecución (segundos)	Porcentaje de eficiencia
6	2	1.0535	100%
9	2	2.1621	100%
12	2	3.7661	91.6%
15	2	5.8036	93.3%
30	2	19.2118	93.3%
60	2	68.3645	96.6%
90	2	145.4169	95.5%
Toda la BD	2	383.8048	95.3%

Tabla 5.10 Resultados de Clasificación para la BD Iris.

Al igual que en todas las bases de datos de prueba, no se forzó a que ambos colores tuvieran la misma cantidad de datos, por lo que la proporción se mantuvo original ya que al grupo 2 lo componen 2 tipos de planta.

Enfrentándose al problema de los 2 tipos de planta mezclados, se hizo un análisis de regresión (Montgomery, Runger, 2003), de esta manera podremos decidir que columnas influyen más en el tipo de planta. En la tabla siguiente se muestran los resultados del análisis de regresión.

Columna	Estadístico t	Probabilidad
Tipo de flor	5.823619721	3.56594x10-8
Longitud del sépalo	-1.900000449	0.059418327
Ancho del sépalo	-0.737794097	0.461831907
Longitud del pétalo	3.983151224	0.000107245
Ancho del pétalo	6.455649959	1.51962x10-9

Tabla 5.11 Análisis de Regresión de la BD Iris.

Para interpretar los valores, es necesario saber que mientras más grande sea el valor de *t*, mayor es su influencia en la columna de resultado, la siguiente columna

representa la probabilidad de que los valores de la columna valgan cero y no se altere el valor de la columna resultado, es decir, prescindir del atributo en el modelo. Entonces como se aprecia en la tabla 5.11 la columna de ancho de sépalo tiene una alta probabilidad de valer cero, mientras que la columna de longitud de sépalo tiene valor negativo en su estadístico t, una vez teniendo esta información se tomaron las siguientes medidas:

- Invertir el signo de la característica longitud de sépalo.
- Eliminar la columna ancho de sépalo.

Ya tomadas las medidas correctivas se volvió a ejecutar el modelo de coloración.

Nodos	Número de colores	Dureza	Solidez	Resiliencia
150	1	4504.2856	0.40306806	
150	2	517.0014	0.09315341	3.32692783
150	3	113.9274	0.03100065	2.00488526
150	4	60.2439	0.0220069	0.40867852
150	5	51.1956	0.02353821	-0.06505605
150	6	44.6389	0.02479939	-0.05085537
150	7	40.9661	0.02673778	-0.07249637
150	8	35.8347	0.02691808	-0.00669823
150	9	32.8099	0.02792332	-0.03599983
150	10	31.8684	0.03035086	-0.07998252

Tabla 5.12 Resultados de Resiliencia para la BD Iris.

La resiliencia se calculó con el proceso de recocido simulado y así poder incluir todos los elementos de la BD. El proceso sigue sugiriendo 2 colores como óptimos, sin embargo ya ve más claramente que 3 colores es una buena clasificación. A continuación se ejecutó el proceso de clasificación ya con todos los nodos de la BD usando tres clases.

Instancias totales	Número de clases	Tiempo de ejecución (segundos)	Porcentaje de eficiencia
150	3	350.753	96%

Tabla 5.13 Resultado de clasificación para la BD Iris con 3 colores.

En el proceso de clasificación es donde más se puede apreciar la mejora, el modelo ya es capaz de ver los 3 tipos de planta y las clasifica a un porcentaje de eficiencia muy alto, en el siguiente capítulo se podrá comparar el modelo con otros clasificadores ya publicados y probados.

Como se mencionó anteriormente, Iris es una de las bases de datos más utilizadas, los autores también han tenido que enfrentarse al problema de separar los otros 2 tipos de planta con varias técnicas como son algoritmos evolutivos, redes neuronales o árboles de decisión (Universidad de Irvine California, 2016), la comparación de resultados se hará sobre la clasificación de los 3 tipos de plantas.

Método y métrica	Eficiencia	Tipo de prueba
Coloración GS, Euclidiana k=2	96	Sin entrenamiento.
C4.5	93.6	Algoritmo k-medias, cross validation
C4.5+m	93.1	Algoritmo k-medias, cross validation
C4.5+m+cf	93.1	Algoritmo k-medias, cross validation
C4.5+cf	91.6	Algoritmo k-medias, cross validation

Tabla 5.14 Fuente (Demšar, 2006) Comparación de Clasificadores para la BD Iris.

Tal y como se muestra en la tabla 5.14 el clasificador está muy bien colocado respecto a otros trabajos y aunque la muestra de trabajos es muy pequeña, se obtuvo de un artículo dedicado a la comparación de distintos clasificadores.

5.4. Base de datos Car Evaluation.

1728 modelos distintos de automóviles evaluados sin datos faltantes que incluye entre atributos subjetivos y atributos técnicos. Las características se describen a continuación:

1. Precio. (Evaluado como muy alto, alto, medio y bajo, se reemplazaron por 4, 3, 2, 1 respectivamente)
2. Mantenimiento. (Muy alto, alto, medio y bajo, se reemplazaron por 4, 3, 2, 1 respectivamente)
3. Puertas. (2, 3, 4, 5 o más)
4. Capacidad. (2, 4 o más personas)
5. Cajuela. (Pequeña, mediana y grande, se sustituyeron esos valores por 1, 2 y 3 respectivamente)
6. Seguridad. (baja, media y alta, reemplazándose por 1, 2 y 3 respectivamente)

Los valores de precio, mantenimiento, cajuela y seguridad, además de ser subjetivos no son numéricos, por lo que fue necesario establecer la ponderación de los valores descrita. Todos los datos se encuentran distribuidos entre las siguientes categorías a continuación:

- Inaceptable: 1210 instancias.
- Aceptable: 384 instancias.
- Bueno: 69 instancias
- Muy bueno: 65 instancias.

Suponiendo que no sabemos que la agencia ha dividido los autos en estas cuatro clases, se calculó la resiliencia en una muestra pequeña para conocer el número de clases en los que trabajará el agrupador.

Nodos	Número de colores	Dureza	Solidez	Resiliencia
20	1	367.2775	1.93303947	
20	2	96.4999	1.07222111	0.80283661
20	3	48.8055	0.86127353	0.24492519
20	4	22.1111	0.5527775	0.55808355
20	5	13.2778	0.44259333	0.24895126
20	6	8.3055	0.35595	0.24341434
20	7	5.5555	0.29914231	0.1899019
20	8	3.8889	0.25926	0.15383132
20	9	3.1389	0.25681909	0.00950439
20	10	2.5278	0.25278	0.01597868

Tabla 5.15 Resultados de Resiliencia para la BD Car Evaluation.

Por tiempos de ejecución se utilizó una cantidad pequeña de instancias, y así garantizar el óptimo de la función objetivo con el software GAMS. En la tabla 5.15 se sugieren 2 colores como óptimos, es decir, para el proceso la mejor clasificación es sólo entre carros buenos y malos, hay que resaltar que el proceso ve como buena clasificación el uso de 4 colores, por lo que el algoritmo de coloración ve adecuada también la clasificación original.

Estos son los resultados luego de estandarizar y ejecutar la coloración suave óptima con dos clases:

Instancias totales	Número de clases	Tiempo de ejecución (segundos)	Porcentaje de eficiencia
8	2	1.8176	100%
12	2	4.07	100%
16	2	5.7669	100%
20	2	8.7322	100%
40	2	27.7042	100%
80	2	102.1054	100%
120	2	228.0418	95.83%
200	2	618.1844	98.5%

Tabla 5.16 Resultados de Clasificación para la BD Car Evaluation.

Debido al tiempo de ejecución no fue posible utilizar toda la base de datos, por lo que las pruebas se acotaron hasta 50 instancias por clase. Esta base de datos es la que cuenta con el mayor éxito de clasificación, teniendo resultados de 100% en la mayoría de las pruebas, por lo que podemos concluir que mientras se use la cantidad de colores sugerida por el proceso, se obtienen excelentes resultados.

En la tabla 5.17 se pueden observar los datos de eficiencia del proceso de coloración de Gráficas Suaves contra otros clasificadores, cabe resaltar que la comparación se hace con 2 colores que es donde se obtuvieron los mejores resultados de clasificación.

Método y métrica	Eficiencia	Tipo de prueba
Coloración GS, Euclidiana k=2	98.5	Subconjuntos sin entrenamiento.
TAN	94.10±0.48	Entrenamiento previo con subconjunto
BAN	94.04±0.44	Entrenamiento previo con subconjunto
Bayes Ingenuo	86.58±1.78	Entrenamiento previo con subconjunto
GBN	86.11±1.46	Entrenamiento previo con subconjunto

Tabla 5.17 (Cheng, Greiner, 1999) Comparación de Clasificadores para la BD Car Evaluation.

Con lo mostrado en la tabla anterior, el agrupador se muestra bien posicionado respecto a clasificadores supervisados, incluso contra el Bayes Ingenuo, lo que parece indicar que el clasificador es bastante eficiente para distintos tipos de bases de datos.

5.5. Base de datos Stone Flakes.

Se trata de una base de datos publicada en 2014 y se compone de la recopilación de 79 objetos que sirvieron como auxiliares para crear puntas de flechas, lanzas o cuchillos en la Europa Occidental durante el periodo Paleolítico. Se busca encontrar una relación de progreso tecnológico con estas herramientas de trabajo o algún cambio en la técnica de creación de armas de la Prehistoria (Weber, 2009). Hay presentes 8 atributos de la base de datos que se describen a continuación:

1. Ancho del objeto distribuido en un intervalo desde 1.02 hasta 1.69 cm.
2. Grosor del objeto entre 16.5 y 43.7 cm.
3. Profundidad del objeto oscilando entre 1.66 y 4.9 cm.
4. Ángulo de golpeo entre el objeto y la superficie en la que se talló desde los 105° hasta los 131°.
5. Frecuencia de uso como herramienta primaria cuyos datos se encuentran entre 0 y 67.2.
6. Frecuencia de uso como herramienta multiusos con valores desde 0 hasta 55.3.
7. Área de mayor desgaste, distribuyéndose los valores entre 5 y 94.1 cm^2.
8. Proporción de desgaste respecto a la superficie total que varía entre 30% y 98%.

La base de datos tiene la siguiente clasificación sugerida por los arqueólogos:

- Edad y tipo de homínido, desde el Homo ergaster en el Bajo Paleolítico entre 600,000 y 300,000 años de antigüedad, pasando por la técnica *Levallois* de hace 200,000 años, seguido del grupo de neandertales del Medio Paleolítico entre 120,000 y 60,000 años de antigüedad, llegando finalmente al Homo sapiens del Alto Paleolítico, hace unos 40,000 años.

Una vez estandarizados los datos y aplicando el proceso de coloración, se obtuvieron los siguientes resultados:

Nodos	Número de colores	Dureza	Solidez	Resiliencia
73	1	1909.4213	0.72656823	
73	2	564.74	0.43584025	0.6670517
73	3	309.4432	0.3633384	0.1995436
73	4	201.9049	0.32067485	0.13304302
73	5	149.0708	0.30030379	0.06783484
73	6	120.4644	0.29555772	0.01605801
73	7	101.0152	0.29352694	0.00691854
73	8	84.661	0.28547439	0.0282076
73	9	76.99	0.29662243	-0.03758326
73	10	64.3584	0.27987997	0.05982013

Tabla 5.18 Resultados de Resiliencia para la BD Stone Flakes.

Se utilizaron las 73 instancias que cumplían con los criterios de limpieza y de datos faltantes. Aplicando el método de recocido simulado dado que se utilizaron todos los nodos de la base de datos. En la tabla 5.18 se sugieren 2 colores como óptimos, sin embargo, no descarta la posibilidad de que se utilicen 3 colores. En la siguiente tabla a diferencia de las anteriores se ejecutó el modelo de clasificación tanto con el software GAMS como con el proceso de recocido simulado. La eficiencia se tomó respecto a la clasificación sugerida de edad y tipo de homínido:

Modelo de ejecución	Número de clases	Tiempo de ejecución (segundos)	Valor de la función objetivo	Porcentaje de eficiencia
GAMS	2	8042.54	564.74	89.04%
Recocido Simulado	2	100.6817	564.74	89.04%

Tabla 5.19 Clasificación de la BD Stone Flakes con 2 Colores.

Se ejecutó el Modelo Binario Entero en GAMS garantizando el óptimo, pero como se puede observar en la tabla 5.19 el tiempo de ejecución es mucho mayor que mediante el proceso de recocido simulado; el proceso de recocido simulado es más rápido aunque no garantiza encontrar el óptimo. En este caso alcanzó la misma solución que GAMS, pero no se puede tener la certeza de que siempre encuentre el óptimo.

Respecto a la clasificación, el modelo es capaz de distinguir las piezas entre los homínidos más antiguos de los más recientes, lo que podría confirmar de manera matemática la hipótesis de que existe una evolución en las técnicas de tallado de piedras en el Paleolítico (Weber, 2009). A continuación se muestran los resultados del proceso de clasificación ahora con 3 colores.

Modelo de ejecución	Tiempo de ejecución (segundos)	Valor de la función objetivo	Porcentaje de eficiencia
GAMS	50704.21	--	Desconocido
Recocido Simulado	102.9938	309.4432	82.19%

Tabla 5.20 Clasificación de la BD Stone Flakes con 3 colores.

En esta ocasión GAMS fue incapaz de encontrar una solución tras 14 horas aproximadas en ejecución por lo que nos apoyamos en los resultados del recocido simulado encontrando un porcentaje de eficiencia bastante aceptable tomando en cuenta que dado que se usó una metaheurística y por el tamaño de la instancia, es poco probable haber encontrado la coloración óptima que sugiere el modelo. En este caso el proceso de clasificación ve claramente la separación de los homínidos más

antiguos, pero le cuesta trabajo diferenciar las piedras talladas entre Neandertales y Homo Sapiens.

Al momento de comparar los resultados con otro clasificador propuesto (Ritter, 2015), éste también realiza el agrupamiento de los datos haciendo uso de la clasificación geo-cronológica propuesta.

Método y métrica	**Eficiencia**	**Tipo de prueba**
Algoritmo de Agrupación recortada	84.93%	Probabilidad aplicada en análisis de agrupamiento y selección de variables.
Coloración GS, Euclidiana k=3	82.19%	Sin entrenamiento.

Tabla 5.21 Comparación de Clasificadores para la BD Stone Flakes.

Los resultados son muy similares en la ejecución a 3 colores. Se puede percibir una ventaja importante del modelo de coloración de gráficas suaves respecto al modelo de Ritter, la cual consiste en que el método de coloración no conlleva un análisis tan exhaustivo de los datos ni consideró previamente varios tipos de separación (lineal, Fisher, p-valor, etc.). Lo que permite obtener resultados similares con un modelo mucho más simple de resolver.

Conclusiones

Este trabajo tuvo como fin explicar el desarrollo de un clasificador no supervisado utilizando el modelo de coloración de gráficas suaves, primero se dio un breve repaso de la base teórica sobre la que se fundamentó, seguido de la construcción de la propuesta, su evaluación en algunas bases de datos las cuales se buscaron de distintas ramas del conocimiento y finalmente se compararon los resultados con otros modelos ya publicados y aceptados por la comunidad.

Se validó el modelo utilizando un conjunto de datos sin procesar obtenidos del repositorio de la Universidad de California Irvine, se seleccionaron tanto bases numéricas como alfanuméricas que se podían ponderar y volver numéricas. La información se procesó en tres fases:

- Estandarización de los datos (Limpieza, ponderación y normalización).
- Creación de la matriz de distancias.
- Ejecución del modelo de coloración de gráficas suaves aplicando la técnica metaheurística de recocido simulado.

Para resolver las instancias con hasta 20 elementos a clasificar, se desarrolló un modelo de programación entera binaria. Este modelo se resolvió utilizando el software comercial GAMS que, como se explicó en el capítulo de propuesta, el software cuenta con sus propias bibliotecas de optimización por lo que sólo se le implementó el modelo de coloración de gráficas suaves a la biblioteca CPLEX 12.6.

Para instancias de más de 20 elementos, el tiempo de cálculo en GAMS se volvía prohibitivamente grande debido a que el problema de coloración de gráficas suaves es del tipo NP-Duro y era necesario reducir el tiempo de ejecución por lo que se desarrolló un modelo metaheurístico de recocido simulado el cual se implementó en *FreeBASIC* porque es un lenguaje cuyos programas se ejecutan más rápido que en casi todos los demás lenguajes de alto nivel.

El resultado fue un clasificador no supervisado de propósito general (por ello se buscaron distintos tipos de bases de datos) eficiente y con tiempos de ejecución excelentes para una cantidad pequeña de datos y buenos para una gran cantidad de datos si se acompaña de una técnica metaheurística, permitiendo la mejora de resultados utilizando análisis de regresión. El clasificador desarrollado es bastante eficiente bajo condiciones de limpieza de ruido y de normalización de datos, esto lo posiciona muy bien entre clasificadores ya publicados y usados frecuentemente.

Consideramos que la solución como primer proceso de filtrado debía ser rápida y fácil de utilizar por lo que un manejo complejo de los datos va en contra del

paradigma original de la propuesta. Nuestro algoritmo se desempeña bien incluso en las bases de datos complicadas para los clasificadores más utilizados por la comunidad.

Una de las diferencias a destacar entre un clasificador supervisado y un clasificador no supervisado consiste en que en al clasificador supervisado se le indica qué debe encontrar mediante un entrenamiento, mientras que un clasificador no supervisado tiene su propia inteligencia y sus propios criterios sin necesidad de ser entrenado. Si los resultados de clasificación arrojados por el agrupador no coinciden con los que se conocen a priori, no necesariamente se trata de un error, sino de otro enfoque del problema que debería ser tomado en cuenta, por ejemplo el caso de la evaluación de carros, donde el clasificador distingue con gran claridad un auto bueno de uno malo.

Dentro del trabajo a futuro se espera que el clasificador pueda ser usado para el reconocimiento de patrones en el que no se tiene claro cuál sería la mejor forma de agrupar, facilitando muchas actividades tales como (Journal Pattern Recognition, 2015):

- Previsión meteorológica con conocimiento a priori de datos meteorológicos previos creando mapas de predicción automática.
- Reconocimiento de caracteres como son símbolos de escritura a mano o a máquina.
- Reconocimiento de voz y del lenguaje natural.
- Aplicaciones médicas como el análisis de signos vitales, detección de tumores, marcas en la piel.
- Reconocimiento de huellas dactilares para la autenticación.
- Reconocimiento de rostros comúnmente utilizados por las cámaras para la detección de sonrisas o de personas en una imagen.
- Interpretación de fotografías aéreas y de satélite para la agricultura, geología, geografía.
- Predicción de magnitudes de terremotos.
- Ayuda a personas con discapacidad visual mediante el reconocimiento de objetos.
- Reconocimiento de música identificando el género musical y/o la canción concreta.

Respecto a las bases de datos utilizadas se puede concluir que los resultados son de muy buena calidad comparados con modelos supervisados, los modelos supervisados suelen estar orientados a aprender a ver lo que el investigador quiere que se vea por lo que su eficiencia casi siempre es superior.

En nuestro caso, el modelo no supervisado tiene una eficiencia tan buena y frecuentemente superior que los supervisados. Por ejemplo en Hepatitis no hay una diferencia mayor del 6% de aciertos respecto al algoritmo supervisado mejor posicionado que se ha encontrado.

En la base Wine los resultados son comparables con técnicas estándar de clasificación supervisada e incluso con clasificadores estructurados en redes neuronales incrementales (IncNet por sus siglas en inglés).

Para la base de Iris, una base de datos históricamente difícil de clasificar, coloración de gráficas suaves fue el mejor clasificador por más de 2% a los mejores clasificadores supervisados que se conocen.

Una historia similar se tiene con los datos de Car Evaluation donde la eficiencia de nuestro algoritmo es mejor por más de 4%, cabe mencionar que debido a que la base es muy grande, se tomó una muestra de 200 datos respecto a la base original de más de 1700 elementos.

La base de datos Stone Flakes ha sido estudiada a profundidad por Gunter Ritter (Ritter, 2015). Esta referencia desarrolla un algoritmo sofisticado donde los datos son filtrados, ponderados y los datos restantes son linealizados y en general los datos son procesados intensivamente con casi todas las técnicas de clasificación que publica en su libro. Por nuestra parte utilizando un algoritmo general en la misma base se obtuvo una clasificación con casi la misma eficiencia (<3%) la cual no requirió un estudio tan complejo. En un trabajo futuro se podría desarrollar un algoritmo especializado para Stone Flakes únicamente.

Se cumplió el objetivo de reducir la intervención humana, obteniendo una solución muy eficaz para el ámbito de los algoritmos no supervisados; y al implementarle una técnica metaheurística se logró que el clasificador resultara escalable, es decir, . Además de la influencia de las características del equipo en la eficiencia de un algoritmo de agrupamiento, el lenguaje de programación utilizado tiene injerencia importante en los tiempos de ejecución, por ello también se uso un lenguaje sin middleware, sacrificando la portabilidad por la rapidez en el tiempo de ejecución a gran escala. Para una cantidad pequeña de datos el clasificador ofrece el óptimo sin gran consumo de recursos, y con la técnica de recocido simulado se superó la prueba de escalabilidad para el modelo de coloración de gráficas suaves. Finalmente al ser la coloración de grafos una técnica poco explotada, este trabajo mostró que ésta es una opción sólida y confiable dentro el inmenso abanico de opciones dentro de los algoritmos de reconocimiento de patrones.

Con base en el tema de esta Idónea Comunicación de Resultados se escribió con un artículo enviado a una revista indexada en Scopus y Zentralblatt MATH llamada "Revista Matemática Teoría y Aplicaciones" el cual al momento de la conclusión de este documento se encuentra listo para ser publicado en el volumen 21-1 de 2017.

Referencias

Anthony M. H. G. (1997) *Computational learning theory*. Cambridge University Press, Cambridge.

Baraldi A., Alpaydin E. (2002). "Constructive feedforward ART clustering networks.", *I. Neural Networks, IEEE Transactions on*, 13(3).

Barbará D., Chen P. (2000) "Using the fractal dimension to cluster datasets," *in Proc. 6th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*.

Ben-Dor A., Shamir R., Yakhini Z. (1999) "Clustering gene expression patterns*," J. Comput. Biol.*, 6.

Beyer K., Goldstein J., Ramakrishnan R., Shaft U. (1999) "When is nearest neighbor meaningful," *in Proc. 7th Int. Conf. Database Theory*.

Cheng J., Greiner R. (1999), "Comparing Bayesian Network Classifiers," *Proceedings of UAI*.

Cherkassky V.,Mulier F. (1998) *Learning From Data: Concepts, Theory,and Methods.* Wiley, New York.

Chow, James H., and Cheryl Chow. (2006) T*he Encyclopedia of Hepatitis and Other Liver Diseases.* Infobase Publishing.

De Los Cobos S. G., Goddard J., Gutiérrez M.A., Martínez A.E.,"Búsqueda y Exploración Estocástica," *Universidad Autónoma Metropolitana*,2010.

Demšar, J. (2006) "Statistical comparisons of classifiers over multiple data sets." *The Journal of Machine Learning Research 7*.

Diestel R. (2000) *Graph Theory.* Springer-Verlag, New York.

Dony, R. (2001). "Karhunen-Loeve Transform." *The transform and data compression handbook. CRC Press, Boca Raton, London, New York, Washington, DC.*

Duda R., Hart P., Stork D. (2001), *Pattern Classification*,Wiley,NewYork.

Ester M., Kriegel H., Sander J., Xu X. (1996) "A density-based algorithm for discovering clusters in large spatial databases with noise," *in Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining (KDD'96)*.

Estivill-Castro V., Lee I. (1999) “AMOEBA: Hierarchical clustering based on spatial proximity using Delaunay diagram,” *in Proc. 9th Int. Symp. Spatial Data Handling (SDH’99)*.

Everitt B., Landau S., Leese M. (2001), *Cluster Analysis.* Arnold, London.

Fernández, V, Berradre, M, Sulbarán, B, Ojeda de Rodríguez, G, & Peña, J. (2009) “Caracterización química y contenido mineral en vinos comerciales venezolanos.” *Revista de la Facultad de Agronomía*, 26(3).

GAMS Development Corporation, http://www.gams.com/ obtenido en Agosto de 2015.

Glover F. Laguna M. (1997), *Tabu Search*. Kluwer Academic Publishers.

Gordon A.D. (1999) *Classification.* Chapman & Hall, London.

Guha S., Rastogi R., Shim K. (1998) “CURE: An efficient clustering algo- rithm for large databases,” *in Proc. ACM SIGMOD Int. Conf. Manage-ment of Data*.

Gutiérrez-Andrade M.A., Lara-Velázquez P., López-Bracho R., Ramírez-Rodríguez J. (2011) “Heuristics for the Robust Coloring Problem”, *Revista de Matemática: Teoría y Aplicaciones* 18(1).

Hall L., Özyurt I., Bezdek J. (1999) “Clustering with a genetically optimized approach,” *IEEE Trans. Evol. Comput.*, 3(2).

Hansen P., Jaumard B. (1997), “Cluster analysis and mathematical programming,” *Math. Program.*, (79).

Haykin S. (1999) *Neural Networks: A Comprehensive Foundation*, 2nd ed. Prentice-Hall, Englewood Cliffs.

Holland, J. H. (1992) “Algoritmos genéticos”, *Investigación y Ciencia* (192).

Höppner F., Klawonn F., Kruse R. (1999) *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis, and Image Recognition.* Wiley, New York.

Jain A., Murty M., Flynn P. (1999). “Data Clustering: A Review”, *ACM Computing Surveys,* 31(3).

Jain A. K., Duin, R. P. W., & Mao, J. (2000). “Statistical pattern recognition: A review. Pattern Analysis and Machine Intelligence”, *IEEE Transactions on*, 22(1).

Jain A., Murty M., Flynn P. (1999), "Data clustering: A review," *ACM Comput. Surv.*, 31(3).

Joachims T. (1996). "A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization", *Carnegie-Mellon Univ Pittsburgh Pa Dept Of Computer Science CMU(CS).*

Journal Pattern Recognition JPRR, http://www.jprr.org/index.php/jprr revisado el 19 de Mayo de 2015.

Karypis G., Han E., Kumar V. (1999) "Chameleon: Hierarchical clustering using dynamic modeling," *IEEE Computer*,32(8).

Kaufman L., Rousseeuw P. (1990), *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley & Sons, Hoboken.

Kirkpatrick S., Gelatt C., Vecchi M. (1983) "Optimization by simulated annealing," *Science*, 220(4598).

Kohonen. T. (2001). *Self-organizing maps*. Springer Science & Business Media, New York.

Kuhn F. (2009) "Weak Graph Colorings: Distributed Algorithms and Applications", in ACM: *Proceedings of the twenty-first annual symposium on Parallelism in algorithms and architectures, ACM*.

Lara-Velázquez P., Gutiérrez-Andrade M.A., De-los-Cobos-Silva S. G., Rincón-García E. A. (2015) "Coloración de Gráficas Suaves", *Revista de Matemática: Teoría y Aplicaciones* 22(2).

Larrañaga P., Inza, I., Moujahid, A. (2007): "Clasificadores Bayesianos", Preprint, Universidad del País Vasco-Euskal, Herriko Unibertsitatea.

Meyer W. (1973). "Equitable Coloring"*, American Mathematical Monthly (Mathematical Association of America)* (80).

Moore B. (1989) "ART1 and pattern clustering," *in Proc. 1988 Connectionist Models Summer School*.

Montgomery D.C., Runger G.C. (2003). *Applied Statistics and Probability for Engineers*, John Wiley and Sons, Inc. Third Edition.

Moreno-Montiel B. (2009) *Minería Sobre Grandes Cantidades de Datos*. Tesis de maestría, Universidad Autónoma Metropolitana, Ciudad de México.

Ng R., Han J. (2002) “CLARANS: A method for clustering objects for spatial data mining,” *IEEE Trans. Knowl. Data Eng.*, 14(5).

Parzen E. (1962). “On estimation of a probability density function and mode”, *The annals of mathematical statistics* 33(3).

Ramírez J. (2001) *Extensiones del Problema de coloración de grafos,* Tesis de doctorado, Universidad Complutense de Madrid, Madrid.

Ripley B. D. (1996) *Pattern Recognition and Neural Networks,* Cambridge University Press, Cambridge.

Ritter G., (2015) “Robust cluster analysis and variable selection.” *CRC Press*.

Rodríguez E. V. (2004) *Aprendizaje evolutivo de reglas fuzzy en un sistema clasificador modificado para control de agentes móviles.* Tesis de doctorado, Universitat Politècnica de València, Valencia.

Roweis S., Saul L. (2000) “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, 290(5500).

Sharan R., Shamir R. (2000) “CLICK: A clustering algorithm with applications to gene expression analysis,” *in Proc. 8th Int. Conf. Intelligent Systems for Molecular Biology*.

Schölkopf B., Smola A.,Müller K. (1998) “Nonlinear component analysis as a kernel eigenvalue problem,” *Neural Computat.*, 10(5).

Sun R., Giles C. (2000) “Sequence learning: Paradigms, algorithms, and applications,” *in LNAI*.

Trujillo-Pulgarín C. A. (2012) *Clasificación basada en la estimación de Parzen en espacios generalizados de disimilitudes*, Tesis de maestría, Universidad Nacional de Colombia, Medellín.

Tseng L., Yang S. (2001) “A genetic approach to the automatic clustering problem,” Pattern Recognit., 34.

Universidad Nicolás Copérnico Polonia, Datsets Classifier: http://www.is.umk.pl/projects/datasets.html revisado el 25 de Noviembre de 2015.

University of California Machine Learning Repository UCI, http://archive.ics.uci.edu/ml/datasets.html revisado entre Mayo de 2015 y Enero de 2016.

Waterhouse A. L., Ebeler. S. E. (1998) “Chemistry of wine flavor.” *American Chemical Society*; Distributed by Oxford University Press.

Weber T. (2009), “The Lower/Middle Palaeolithic transition - is there a Lower/Middle Palaeolithic transition?,” *Preistoria Alpina*, 44.

Xu R., Wunsch D. (2005). “Survey of clustering algorithms” in *Neural Networks, IEEE Transactions on*, *IEEE*.

Yañez J.,Ramirez J (2003), “The Robust Coloring Problem”, *European Journal of Operational Research*, 148(3).

Zadeh L. (1965) “Fuzzy sets,” *Inf. Control*, 8.

Zhang T., Ramakrishnan R., Livny M. (1996) “BIRCH: An efficient data clustering method for very large databases,” *in Proc. ACM SIGMOD Conf. Management of Data*.

Zhu X., Ghahramani Z., Lafferty J. (2003) “Semi-supervised learning using gaussian fields and harmonic functions” ,in: T. Fawcett & N. Mishra (Eds.) *International Conference on Machine Learning*, *AAAI Press*.

Zhuang X., Huang Y., Palaniappan K., Zhao Y. (1996), “Gaussian mixture density modeling, decomposition, and applications,” *IEEE Trans. Image Process.*, 5(9).