CLASIFICACIÓN SUPERVISADA
INDUCCIÓN DE ARBOLES DE DECISIÓN, ALGORITMO k-d

Alfonso García, agarcia@geminis.cic.ipn.mx
Gilberto L. Martínez, lluna@geminis.cic.ipn.mx
Gustavo Núñez, gnunez@pollux.cic.ipn.mx
Adolfo Guzmán, aguzman@pollux.cic.ipn.mx
Centro de Investigación en Computación (CIC), Instituto Politécnico Nacional (IPN)

Resumen. El descubrimiento de conocimiento en base de datos (*Knowledge Discovery in Database*, KDD) y la minería de datos (*Data Mining*, DM) a partir de grandes bases de datos es una tecnología reciente para el análisis automático de grandes colecciones de información. En este artículo, se presenta el desarrollo de un nuevo algoritmo de clasificación supervisada para la minería de datos, el cual genera un árbol de decisión llamado k-*dimensional* (k-d), en este árbol, cada nodo es expandido, eligiendo el atributo que tiene menos confusión. Aquí el concepto de confusión se formaliza en base a la matriz de aprendizaje, la cual se obtiene directamente del conjunto de entrenamiento, los datos se describen por una colección finita de caracteristicas o atributos. El concepto de confusión permite generar arboles de decisión más eficientemente que la mayoría de los algoritmos reportados en la literatura. Así mismo, se ha comprobado, experimentalmente que el arbol k-d, con conjuntos de entrenamiento suficientemente grandes clasifica, al menos tan bien, como otros algoritmos. Además, se presenta una implementación del algoritmo y un procedimiento sencillo para calcular su eficiencia.

Palabras Claves. Descubrimiento de conocimiento, minería de datos, clasificación, clasificación supervisada, clasificación no supervisada, sistema experto, árboles de decisión, bases de datos

1. Introducción

El problema de la clasificación de objetos definidos por atributos, es uno de los enfoques más basicos del aprendizaje automático

El método de clasificación basado en árboles de decisión [17], [18], ha sido utilizado con éxito en el aprendizaje automático. Los árboles de decisión reconstruyen a partir de un conjunto de ejemplos. La calidad de un árbol construido depende de tanto la exactitud de la clasificación y el tamaño del árbol. El método usa una muestra de datos llamada generalmente conjunto de entrenamiento para formar un árbol de decisión. Si el árbol no clasifica correctamente todos los objetos, una selección de las excepciones es adicionada a los subconjuntos de entrenamiento y el proceso continua hasta que el subconjunto correcto de decisión se encuentra. La salida eventual es un árbol en el cual cada hoja lleva un nombre de clase, y cada nodo interior específica un atributo con una correspondiente rama a cada posible valor de este atributo.

Existen dos tipos de clasificaciones, cuando el experto indica las clases en las que debe dividirse el dominio se llama clasificación supervisada; cuando el procedimiento de clasificación genera automaticamente las clases, sin intervención del experto, se llama clasificación no supervisada.

Una de las principales tareas de minería de datos, es la de agrupar datos identificando grupos o regiones densamente populares, de acuerdo a algunas medidas de distancia o semejanza, en un gran, conjunto de datos multidimensional y poco estructurado. El agrupamiento de datos identifica los lugares de dispersión y de acumulamiento, y con esto se descubren patrones o importante de distribución de dichos conjuntos. Este es evidentemente un problema de clasificación no supervisada.

La *Clasificación Supervisada de datos*, es el proceso que se lleva a cabo para encontrar propiedades comunes entre un conjunto de datos y clasificarlos dentro de diferentes *clases*, de acuerdo a un modelo de clasificación. El objetivo de la clasificación es primero desarrollar una descripción o modelo para cada clase usando las características disponibles en los datos. Tales descripciones de las clases son entonces usadas para clasificar futuros datos de prueba en la base de datos o para desarrollar mejores descripciones (llamadas reglas de descripción) para cada clase en la base de datos. Las aplicaciones de la clasificación incluyen diagnóstico médico, predicción de rendimiento, mercadotecnia selectiva, por nombrar unas cuantas.

La referencia obligada en clasificación supervisada es el algoritmo ID-3 [17] este usa un

enfoque de teoría de información para minimizar el número esperado de pruebas para clasificar a un objeto. La selección de atributos en ID-3 está basada en la suposición de que la complejidad del árbol de decisión, está fuertemente relacionada con la cantidad de información que expresa el atributo. ID-3 selecciona el atributo que proporciona la más alta ganancia de información, es decir, aquel que minimiza la información necesaria en los árboles resultantes para clasificar los elementos. Una extensión para ID-3, C4.5 [18], extiende el dominio de la clasificación de atributos categóricos a numéricos.

Existen muchas otras funciones de evaluación, tales como, la *Gini index*, *Chi square test*, y así por el estilo [19], [20], [21], [22]. Por ejemplo, para Gini index [19], [23], si un conjunto de datos T contiene ejemplos de n clases, *gini(T)* esta definida como:

$$gini(T) = 1 - \sum pi^{2.}$$

donde p_i es la frecuencia relativa a la clase i en T. Más aún, existen otros enfoques para transformar árboles de decisión dentro de reglas [18] y reglas de transformación y árboles dentro de estructuras comprensivas de conocimiento [9].

En general la clasificación ha sido estudiada con éxito; en la parte de clasificación no supervisada en: estadística [10], [11], aprendizaje con máquinas [12], [13], bases de datos espaciales [14], y la minería de datos [10], [15], [16], áreas con enfoques diferentes; en la parte de clasificación supervisada los estudios son en: estadística [10] [24], [21] (regresión lineal y el análisis discriminante lineal son modelos clásicos estadísticos [26]), conjuntos difusos [25], técnicas de clasificación dentro del contexto de grandes bases de datos [26], [27], métodos para algoritmos de aprendizaje con máquinas escalares por medio de combinación de clasificadores base para conjuntos de datos particionados [10], clasificadores de intervalo [26] para reducir el costo de generación de árboles de decisión, redes neuronales para la clasificación y la extracción de reglas en bases de datos [28]; y en general en sistemas expertos [8] y la Minería de Datos[9].

Estructura del Trabajo

Este trabajo presenta un algoritmo de clasificación supervisada, utilizando el concepto de *árbol k-d* [2] y [3]**.** En esta sección 1 se presentarón antecedentes y aspectos de clasificación relacionados al trabajo. En la sección 2 se hace una breve descripción de los conceptos básicos que utiliza nuestro algoritmo. En la sección 3 se muestra el seudocódigo del algoritmo k-d y se muestra un ejemplo. En la sección 4 se describe una implementación computacional del algoritmo k-d, un Modelado de su Proceso utilizando la metodología IDEF0 y se muestra el entorno de la interfaz gráfica. La sección 5 se detalla caso de estudio del algoritmo aplicado a la C.F.E.

2. Conceptos Básicos de la Clasificación de k-d.

Las siguientes definiciones ayudaran en la formulación del algoritmo k-d

árbol k-d es un árbol de decisión que sirve para hallar una respuesta o curso de acción, dadas k variables o *entradas*. Cada nodo suyo hace una pregunta sobre una variable; para k variables, puede haber más de k preguntas antes de emitir una respuesta. Los hojas (nodos finales) no hacen pregunas, sino que "dan la respuesta" o curso a tomar.

Universo de objetos U. Son los objetos "admisibles" descritos mediante un conjunto de atributos. Simbólicamente **U** = {$\mathbf{O_j}$ / $\mathbf{O_j}$ es un objeto, j = 1,...,m}

Atributo. Es una característica de un objeto

Valores admisibles de un atributo A. Son los valores que puede tomar un atributo Simbólicamente
V(A). = {$\mathbf{V_g}$ / $\mathbf{V_g}$ es un valor del atributo **A**, g = 1,...,h}.

Objeto. Es una entidad que se describe mediante una **n-ada** de valores de atributos. Simbólicamente **O** = ($\mathbf{V_{1,..,}V_{g,..,}V_n}$) donde $\mathbf{V_g} \in \mathbf{V(A_k)}$ □ k = 1,..,m. con g = 1,..,n.

Tipo del Valor de un Atributo. Este puede ser numérico (enteros o reales) y no numéricos (booleano o k-valente).

Conjunto de Atributos A. Son los atributos mediante los cuales se describen los objetos admisibles pertenecientes a **U**. Simbólicamente
A={ $\mathbf{A_k}$ / $\mathbf{A_k}$ es un atributo, k = 1,...,m}

Clases. Sea **P**= {$\mathbf{C_i}$ **/** $\mathbf{C_i} \subset$ **U**} una partición del universo. Cada $\mathbf{C_i} \in$ **P** es una clase de **U**.

Confusión inducida por un atributo. Dos objetos en diferentes clases generan una confusión en la muestra, si estos dos objetos tienen el mismo valor en algun atributo.

Confusión inducida por un conjunto de atributos. Dos objetos de diferentes clases generan una confusión inducida por un conjunto de atributos en la muestra, si estos dos objetos tienen los mismos valores en cada atributo.

Testor T. Se dice que un **T** $\prec$ **A** es un testor, si **T** no induce confusión en ningun par de objetos**.**

Testor Típico T. Es aquel testor, tal que si se le quita cualquier atributo deja de serlo.

Conjunto de Entrenamiento CE. Es una muestra de objetos **CE**, que se representa mediante una matriz bidimensional en la cual cada renglón representa un objeto y cada columna representa un valor de un atributo. Si el **CE** consta de **m** objetos descritos por **n** atributos entonces la celda (j, k) contiene el valor del atributo k perteneciente al objeto j (figura 1). Donde $\mathbf{V(A_k)_j}$ representa el valor del atributo $\mathbf{A_k}$ en el objeto j

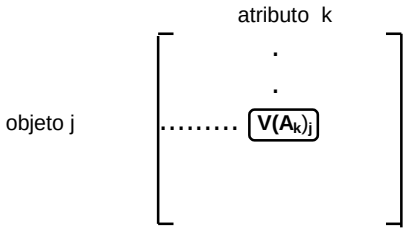

Figura 1. Conjunto de Entrenamiento.

Matriz de Aprendizaje (MA)

Es el conjunto de entrenamiento con una columa adicional que indica a que clase pertenece cada objeto (figura 2). Donde $\mathbf{C_j}$ es la clase a la que pertenece el objeto j. La **MA** se obtiene a partir de un testor típico en el cual no existe duplicación (objetos iguales) y por conveniencia todos los objetos de una clase aparecen juntos. Una MA puede contener varios testores típicos, de distinta longitud, incluso una sola columna (atributo) puede ser testor típico :

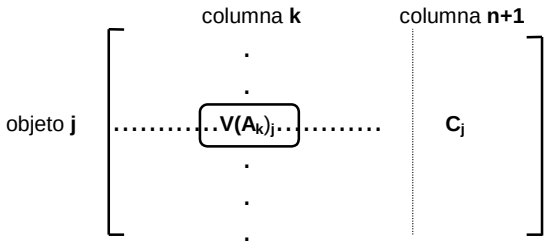

Figura 2. Matriz de Aprendizaje

Atributo que produce menos confusión en la MA Es aquel que tiene el menor número de pares de objetos que tienen el mismo valor en ese atributo pero que pertenecen a clases distintas. Por ejemplo, dada la **MA** de la figura 3, encontramos que el atributo *Color* tiene una confusión de 2 (de rojo) + 1 (de morado) + 2 (de verde) + 1 (de azul) + 1 (amarillo) = **7 (**Figura 2.1); la de la *Altura* es 11(de alto) + 1 (de mediano) + 5 (de chaparro) = **17**; la de *Peso* es 15 (de ligero) + 8 (de pesado) = **23**. Por lo tanto el atributo que produce menos confusión en la **MA** es *Color*.

rojo	alto	ligero	militar	**objeto 1**
rojo	chaparro	pesado	médico	**objeto 7**
rojo	alto	pesado	militar	**objeto 2**
rojo	chaparro	pesado	médico	**objeto 7**
morado	alto	pesado	militar	**objeto 3**
morado	alto	ligero	médico	**objeto 9**
verde	mediano	ligero	militar	**objeto 4**
verde	chaparro	pesado	abogado	**objeto 10**
verde	chaparro	ligero	militar	**objeto 5**
verde	chaparro	pesado	abogado	**objeto 10**
azul	mediano	ligero	médico	**objeto 6**
azul	alto	ligero	abogado	**objeto 11**
amarillo	alto	ligero	médico	**objeto 8**
amarillo	chaparro	pesado	abogado	**objeto 12**

Fígura 2.1. Pares de objetos que contribuyen a la confusión del Atributo (*Color*)

MA =

rojo	alto	ligero	militar
rojo	alto	pesado	militar
morado	alto	pesado	militar
verde	mediano	ligero	militar
verde	chaparro	ligero	militar
azul	mediano	ligero	médico
rojo	chaparro	pesado	médico
amarillo	alto	ligero	médico
morado	alto	ligero	médico
verde	chaparro	pesado	abogado
azul	alto	ligero	abogado
amarillo	chaparro	pesado	abogado

Figura 3. Matriz de Aprendizaje

Cálculo de la eficiencia del algoritmo k-d. Es el porcentaje de aciertos. Simbólicamente

Eficiencia k-d es el porcentaje de:

$$\frac{\sum_{j=1}^{n} (\text{No. } \mathbf{O} \text{ hallados de } C_j \text{ del total de } \mathbf{O} \text{ de } C_j)}{\text{Total de } \mathbf{O}}$$

por ejemplo tomando la información de la figura 16 tenemos que:

Eficiencia k-d = ((42+ 14) / 96) * 100
= (56 / 96) * 100 =(0.58 * 100)
= 58 %

3.- Algoritmo k-d.

El *algoritmo k-d* construye un árbol de decisión a partir de una **MA** , expandiendo cada nodo apartir del atributo que produce menos confusión [2].

Seudocódigo del Algoritmo k-d

Aqui **#T** indica la cardinalidad del testor **T**.

```
si #T > 1
 {
  atributo = SelAtrib-Confusión(MA,T)
  raíz = CreaRaíz(atributo)
  para i = 1 hasta No. de valores de atributo
       CreaRama(nodo_i,atributo,valor_i_atributo,MA,T)
}
```

donde

```
CreaRama(nodo_i,atributo,valor_i_atributo,MA,T)
{
   etiquetar la rama atributo
  si todos los objetos que tienen ese valor_i_atributo como
       atributo pertenecen a la misma clase.
       hoja = CreaNodo(nodo_i,clase_pertenecen_objetos)
  de otro modo
  {
       SubMatriz =
       CreaSubmatriz(MA,T,atributo,valor_i_atributo)
       T =T - {atributo}
       si #T > 1
       {
            atributo = SelAtrib-Confusión(SubMatriz,T)
            nodo = CreaNodo(nodo_i,atributo)
            para j = 1 hasta No. de valores de atributo

CreaRama(nodo_j,atributo,valor_j_atributo,SubMatriz,T)
      }
  }
}

SelAtrib-Confusión(MA,T)
{
     encontrar el atributo que menos confusión produce en la
     MA
     retornar atributo encontrado
}

CreaSubmatriz(MA,T,atributo,valor_i_atributo){
     seleccionar todos los objetos que tienen valor_i_atributo
     en atributo  y eliminar la columna atributo de estos
     objetos
     retornar matriz compuesta de los objetos seleccionados
}
```

Ejemplo: Considerese un universo de seres humanos, en el cual se tengan tres clases de profesionmes, como militar, médico y abogado. Sea el testor típico formado por los atributos: *Color*, *Altura* y *Peso*. Sean rojo, morado, verde, azul y amarillo los valores para el atributo color; alto, mediano y chaparro los valores para el atributo altura; ligero y pesado los valores para el atributo peso.

Aplicando el algoritmo k-d a la Matriz de Aprendizaje de la Figura 3 :

Paso 1: como la cardinalidad de **T** es mayor que uno (**#T** = 3), entonces se encuentra el atributo que menos confusión produce en **MA.** Resultando ser este el atributo *Color* (ver fig. 2.1). Con el atributo *Color* se crea la raíz del arbol k-d (figura 4).

Figura 4.

para cada uno de los valores del atributo *Color* , (ejemplificandose para el valor rojo).

Paso 2: Se aplica la función CreaRama a partir del nodo raíz

Se etiqueta la rama con el valor rojo (figura 4.1).

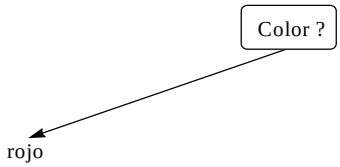

Figura 4.1

Como no se cumple que todos los objetos que tienen el color rojo pertenecen a la misma clase, encontramos la SubMatriz (figura4.2) y decrementando la cardinalidad de **T** (**#T = #T** -1).

$$\text{SubMatriz} = \begin{bmatrix} \text{alto} & \text{ligero} & \text{militar} \\ \text{alto} & \text{pesado} & \text{militar} \\ \text{chaparro} & \text{pesado} & \text{médico} \end{bmatrix}$$

Figura 4.2

Como la cardinalidad de **T** es mayor que uno entonces, se encuentra el atributo que menos confusión produce en SubMatriz**.** Resultando ser el atributo *Altura* (ver fig. 4.2). Con el atributo *Altura* se crea un nodo como se muestra (figura 4.3):

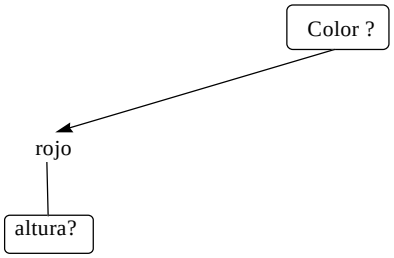

Figura 4.3

Para cada uno de los valores del atributo *Altura* (alto y chaparro), al aplicar la función CreaRama al atributo alto resulta la siguiente secuencia de figuras: 4.4, 4.5 y 4.6 En esta última figura los nodos terminales son hojas y tienen la etiqueta de la la clase que los identifica.

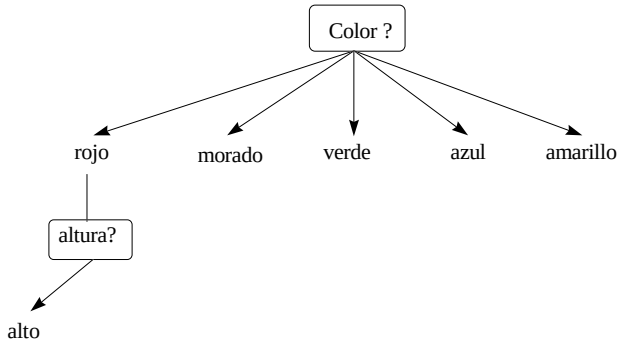

Figura 4.4

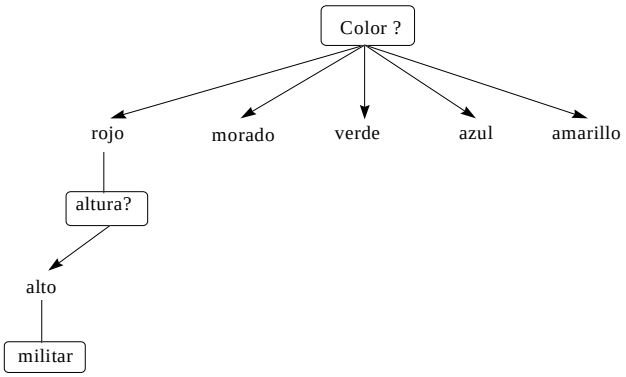

Figura 4.5

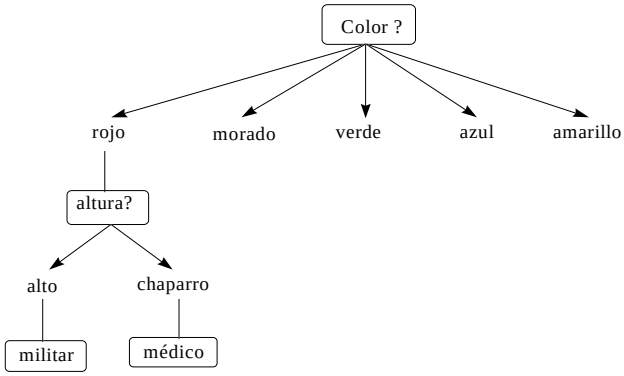

Figura 4.6

Paso 3: aplicando el paso 2, para cada valor restante del atributo *Color* (morado, verde, azul y amarillo) la función CreaRama tenemos finalmente el arbol k-d construido, tal y como se muestra (figura 4.7).

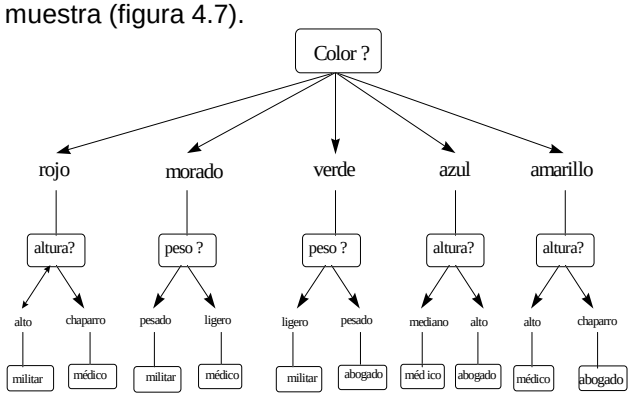

Figura 4.7

Características. Es importante notar:

1.- Que el árbol anterior tiene nodos que no estan completos, por ejemplo los azules preguntan por **altura**? pero no ofrecen alternativa si la respuesta es "**chaparro**", esto se debe a que algunas combinaciones legales no estuvieros presentes en la MA. Si en el momento de clasificar un objeto no pertenece a ninguna clase se etiqueta con "**no clasificado**", por lo tanto concluimos que **mientras más objetos diferentes tenga la MA mejor clasificará el algoritmo k-d.**

2.- El algoritmo k-d no utiliza una función de semejanza al momento de clasificar, para poder clasificar un objeto que no pertenece a ninguna clase, dentro de una clase existente. Es decir, poder determinar con la función de semejanza si este objeto esta cercano a alguna clase existente. En el caso de que el objeto no este cercano a otro lo etiquete con "**no clasificado".**

3.- El algoritmo k-d solo trata con atributos que poseen un número pequeño de valores: booleanos, k-valentes, intervalos y enteros pequeños. Si algun atributo tiene un número grande de valores, o es real, habrá que dividirla en intervalos para transformarla a k-valente (el experto define estos intervalos - pueden no ser igualmente espaciados y dependen del problema a resolver-)

4.- Se puede apreciar, que el algoritmo k-d "aprende perfectamente" todos los objetos de la MA, siendo 100% la **eficiencia** (porcentaje de aciertos) en ellos.

4.- Modelo Computacional.

4.1 Arquitectura del Modelo

La arquitectura del modelo computacional construido (figura 5), tiene los siguientes componentes de software que se describen brevemente a continuación:

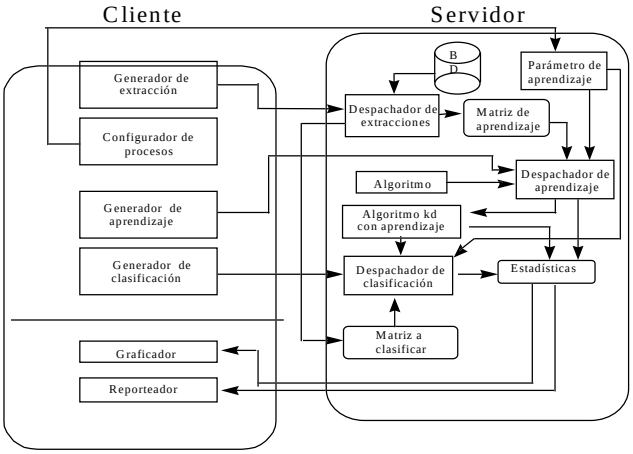

Figura 5.

Generador de Extracciones. Generador de peticiones de extracción de datos para formar la **MA** o la matriz a clasificar.

Despachador de Extracciones. Programador y ejecutor de las peticiones de extracción.

Configurador de Procesos. Constructor de los parámetros de los procesos de aprendizaje y clasificación, entre los cuales está la definición del testor, normalización de atributos, la matriz de aorendizaje y la matriz de clasificación..

Generador de Aprendizaje. Generador de peticiones de aprendizaje con algoritmo k-d.

Despachador de Aprendizaje. Programador y ejecutor de las peticiones de aprendizaje.

Generador de Clasificación. Generador de peticiones de clasificación con algoritmo k-d.

Despachador de Clasificación Programador y ejecutor de las peticiones de clasificación.

Graficador. Presentador de resultados de los procesos de Aprendizaje y Clasificación en forma gráfica bidimensional o tridimensional.

Reporteador. Presentador y generador de reportes; entre los que se encuentran: Parámetros de procesos, MA, matriz a clasificar y eficiencia de la clasificación.

Algoritmo k-d. Implementación computacional del algoritmo k-d.

Algoritmo k-d con Aprendizaje. Algoritmo k-d ajustado al dominio del problema.

4.2 Interface Gráfica.

El software construido tiene como objetivo, facilitar el uso del algorimo implementado. Como se observa en el punto 4.1, nuestro software se conforma de una parte en el Servidor y otra en el Cliente.

La parte del Servidor, es donde se lleva a cabo especificamente los procesos de aprendizaje y de clasificación de objetos.

La creación de la interfaz gráfica en el Cliente tiene el objetivo de suministrar un ambiente sencillo e intuitivo para la utilización del algoritmo k-d. En esta interfaz el usuario realiza peticiones de:
1) Extracción de objetos (Datos en un DBMS).
2) Configuración de procesos
3) Aprendizaje.
4) Clasificación

También genera los reportes y gráficas de los resultados del aprendizaje y de la clasificación, asi como el calculo de la eficiencia del aprendizaje o de la clasificacion. Esta interfaz gráfica permite hacer el trabajo del experto, debido a que el usuario puede seleccionar el testor típico que utilizará el algoritmo k-d, además de normalizar los atributos que tienen valores reales o un número muy grande de valores.

Las figuras 8, 9, 10, 11, 12, 13 y 14 son algunas de las pantallas de la interfaz gráfica en el Cliente.

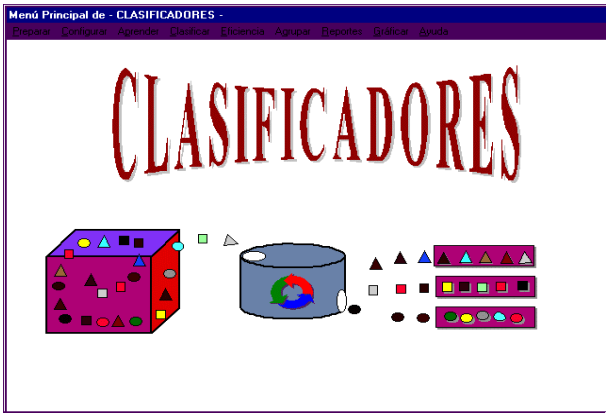

Figura 8 Entrada al Sistema.

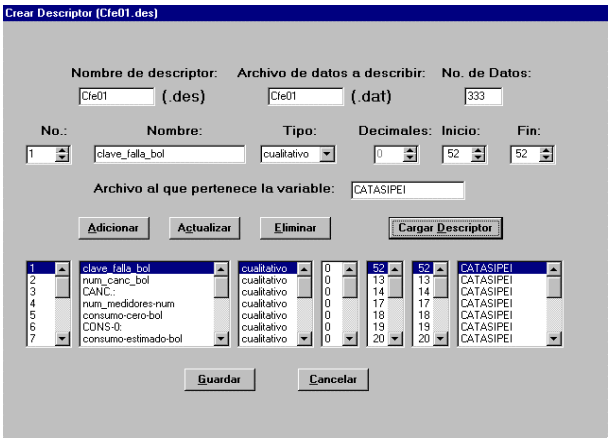

Figura 9. Construcción de testores tipicos

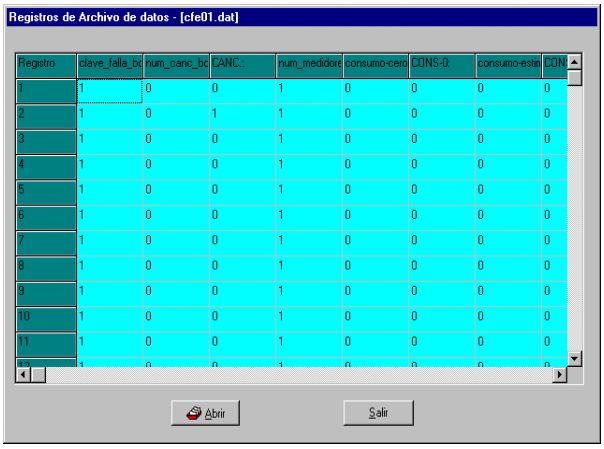

Figura 10 Visualicador de MA y MC

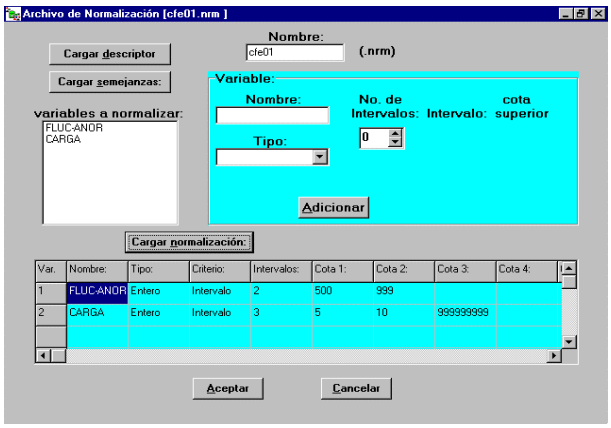

Figura 11, Normalización de Atributos.

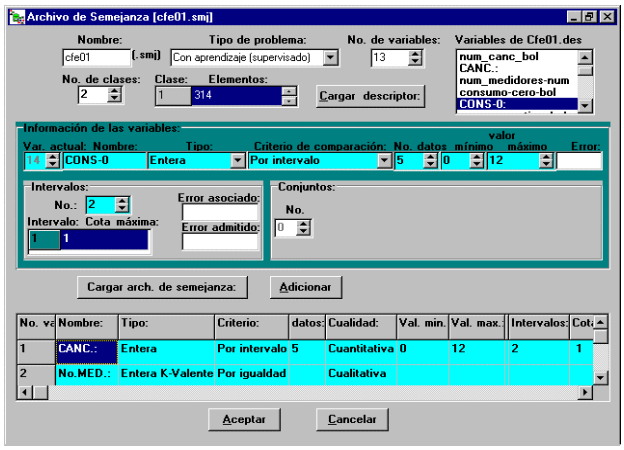

Figura 12, Construcción de Función de Semejanza.

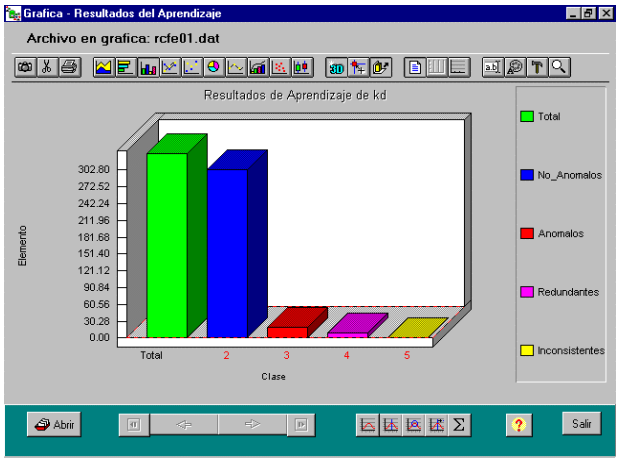

Figura 13, Visualizador Gráfico de Resultados de Aprendizaje y Clasificación.

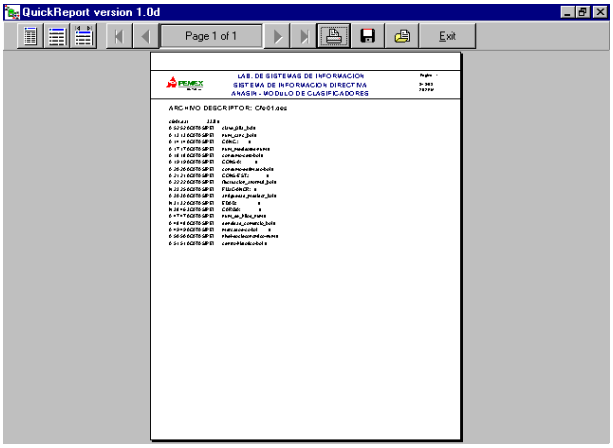

Figura 14, Visualizador Gráfico de Resultados de Aprendizaje y Clasificación.

5.- Caso de Estudio, Comisión Federal de Electrícidad.

El algoritmo k-d integrado en el Sistema de Pruebas Externas e Internas de la Comisión Federal de Electricidad (C.F.E.), División Centro-Oriente, del estado de Puebla, se utilizo para detectar anomalías en los medidores del servicio de suministro de energia electrica. La detección se baso en clasificar medidores en normales y anómalos. El algoritmo se aplico a tres conjuntos de datos. Para cada conjunto se construyo la matriz de aprendizaje y así extraer ciertos parámetros (etapa de aprendizaje), el algoritmo con aprendizaje se aplico a su respectivo grupo de medidores para determinar cuáles ameritan pasar a formar parte del plan de inspección de los servicios.

La primera muestra fue de 210 servicios (figuras 15 y 16) que se dividieron en dos matrices, la MA consta de 114 servicios de los cuales 72 eran buenos (normales) y 42 malos(anómalos), con la MA se entrenó al algoritmo k-d, y con la MC se vio qué tan bien trabajaba, si clasificaba los servicios malos efectivamente como malos, o si había algunos errores.

Primer Conjunto de Datos

APRENDIZAJE		
TOTAL	BUENOS	MALOS
114	72	42

Figura 15. Estadística de Aprendizaje.

CLASIFICACION						
TOTAL	BUENOS			MALOS		
96	60			36		
	B	M	N	B	M	N
	42	15	3	17	14	5

Figura 16. Estadística de Clasificación

El algoritmo k-d halló 42 de los 60 buenos y 14 de los 36 malos, por lo que su eficiencia fue de 57%

Segundo Conjunto de Datos

APRENDIZAJE		
TOTAL	BUENOS	MALOS
114	72	42

Figura 17, Estadística de Aprendizaje

CLASIFICACION						
TOTAL	BUENOS			MALOS		
361	332			29		
	B	M	N	B	M	N
	215	105	12	21	6	2

Figura 18. Estadísticas de Clasificación

El algoritmo k-d halló 215 de los 332 buenos y 6 de los 29 malos, por lo que su eficiencia fue de 61%

Tercer Conjunto de Datos

APRENDIZAJE		
TOTAL	BUENOS	MALOS
270	251	19

Figura 19. Estadísticas de Aprendizaje

CLASIFICACION						
TOTAL	BUENOS			MALOS		
361	332			29		
	B	M	N	B	M	N
	319	13	0	7	22	0

Figura 20. Estadísticas de Clasificación

El algoritmo k-d halló 319 de los 332 buenos y 22 de los 29 malos, por lo que su eficiencia fue de 95%

En un esfuerzo por aumentar la eficiencia de los resultados anteriores y de otros obtenidos, se analizaron los datos de la matriz de

aprendizaje y de control. Este análisis consistio en aplicar los criterios de semejanza (valores que se consideran iguales) de las atributos escogidos. Para comparar cuáles servicios son equivalentes o semejantes, pero estan en clases distintas, contribuye con 1 a la confusion. La confusión se determina contando el número de pares que se confunden.

Bibliografía

[1] Aprendizaje automático, Antonio Moreno Rivas, EDICIONS UPC, 1994

[2] Método para el uso de clasificadores supervisados en problemas complejos, Adolfo Guzmán Arenas, Dic. 2, 1994

[3] Uso de árboles k-d como clasificadores supervisados y para sustituir a sistemas expertos; Adolfo Guzmán Arenas, SoftwarePro International, Junio 22, 1995.

[4] Introducción al reconocimiento de patrones (enfoque logico-combinatorio), José Ruíz Shulcloper, Eduardo Alba Cabrera, serie verde No. 51, CINVESTAV-IPN, Noviembre 9, 1995.

[5] Sistema SIPEI Subsistema de detección de serviciós anomalos, Manual Tecnico, CFE Gerencia de Informatica y Telecomunicacines, Manual tecnico versión 1.0, 1994.

[6] Sistema SIPEI Subsistema de detección de serviciós anomalos, Manual Usuario, CFE Gerencia de Informatica y Telecomunicacines, Manual del usuario versión 1.0, 1994.

[7] *Integration Definition for function Modeling* (IDEFF0), Federal Information Processing Standards Publications (FIPS PUBS) are issued by the National Institute of Standars and Technology after approval by the Secretary of Commerce pursuant Section 111(d) of the Federal Property and Administrative Services Act of 1949 as amended by the Computer Security Act of 1987, Public Law 100-235. 1993 december 21, cunsulte la dirección www.idef.com.

[8] S.M. Wiss and C.A. Kulilowski, "*Computer Systems that learn: Classification and Prediction Methods from Statistic, Neuronal Nets, Machine Learning, and Experts Systems."* Morgan Kaufman, 1991.

[9] U.M. Fayyad, G. Piatetsky-Shapiro, P.Smyth, and R. Uthurumasy, *"Advances in Knowledge Discovery an Data Mining."* AAAI/MIT Press, 1996.

[10] P. Cheeseman and J. Stutz, "Bayesian Classification (AutoClass): Theory and Results," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining,* pp. 153-180, AAAI/MIT Press, 1996.

[11] J. Han, Y. Fu, W. Wang, J. Chiang J. Han, Y. Fu, W. Wang, J. Chiang, W. Gong, K. Koperski, D. Li, Y. Lu, A. Rajan,N. Stefanovic, B. Xia, and O.R. Zaiane, "DBMiner: A System for Mining Knowledge in Large Relitional Databases," *Proc, Int'l Conf. Data Mining and Knowledge Discovery (KDD '96),* pp. 250-255, Portland, Ore., Aug. 1996.

[12] D. Fisher, "lmproving Inference Through Conceptual Clustering," *Proc. AAAI Conf.,* pp. 461-465, Seattle, July 1987.

[13] D. Fisher, "Optimization and Simplification of Hierarchical Clusterings," *Proc. First Iint'l Conf. Knowledge Discovery and Data Mining (KDD'95),* pp. 118-123, Montreal, Canada, Aug.1995.

[14] N. Beckmann, H.-P..Kriegel, R. Schneider, and B. Seeger, *"The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles,"* Proc. ACM SIGMOD Int'l Conf.Management Data, pp. 322-331, Atlantic City, N.J., june 1990.

[15] M. Ester, H.-P. Kriegel, and X. Xu, "Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification," *Proc. Fourth Int'l Symp. Large Spatial Databases (SSD'95), pp.* 67-82, Portland, Maine, Aug. 1995.

[16] R. Ng and J. Han, "Efficient and Effective Clustering Method for Spatial Data Mining," *Proc. Int'l Conf. Very Large Data Bases,* pp. 144-155, Santiago, Chile, Sept. 1994.

[17] J.R. Quinlan, "Induction,of Decision Trees," *Machine Learning,* vol. 1, pp. 81-106, 1986.

[18] J.R. Quinlan, *C4.5:Programs for Machine Learning.* Morgan Kaufmann, 1993.

[19] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification of Regression Trees.* Wadsworth, 1984.

[20] W. Klosgen "Explora: A Multipattern and Multistrategy Discovery Assistant," U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining,* pp. 249-271. AAAI/MIT Press, 1996.

[21] G. Piatetsky-Shapiro, "Discovery, Analysis, and Presentation of Stron, Rules," G. Piatetsky-Shapiro and W. J. Frawley, eds., *Knowledge* Discovery *in Databases, pp.* 229-238. AAAI/MIT Press, 1991.

[22] S.M. Weiss and C.A. Kulikowski, *Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems.* Morgan Kaufman, 1991.

[23] M. Mehta, R. Agrawal, and J. Rissanen, "SLIQ: A Fist Secalable Classifier for Data Mining," *Proc. Int'l Conf. Extending Database Technology (EDBT'96),* Avignon, France, Mar. 1996.

[24] J. Elder IV and D. Pregibon, "A Statistical Perspective on Knowledge Discovery in Databases," U.M. Fayyad, G. PiatetskyShapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Minin,* pp. 83-115. AAAI/MIT Press,1996.

[25] W. Ziarko, *Rough Sets, Fuzzy Sets and Knowledge Discovery.*Springer-Verlag, 1994.

[26] R. Agrawal, S. Ghosh, T. Imielinski, B. Iyer, and A. Swami, "An Interval Classifier for Database Mining Applications," *Proc. 18th Int'l Conf. Very Large Data Bases,* pp. 560-573, Aug. 1992.

[27] T.M. Anwar, H.W. Beck, and S.B. Navathe, "Knowledge Mining by Imprecise Querying: A Classification-Based Approach," Proc. *Eighth Int'l Conf. Data Eng.,* pp. 622-630, Feb. 1992.

[28] H. Lu, R. Setiono, and H. Liu, 'NeuroRule: A Connectionist Approach to Data Mining," *Proc. 2lth Int'l Conf. Very Large Data Bases,* pp. 478-489, Sept. 1995.