RNPS No. 2142
ISSN 2072-6287

SERIE AZUL

REPORTE TÉCNICO
Reconocimiento de Patrones

# Métodos de construcción de árboles de clasificaciónI.

Yenny Villuendas Rey, Anilú Franco Acerga,
José Ruiz Shulcloper

**RT_005** **Diciembre 2008**

CENATAV
Centro de Aplicaciones de Tecnologías de Avanzada
MINISTERIO DE LA INDUSTRIA BÁSICA

7ma. No. 21812 e/218 y 222,
Rpto. Siboney, Playa;
Ciudad de La Habana.
Cuba. C.P. 12200
**www.cenatav.co.cu**

RNPS No. 2142
ISSN 2072-6287

SERIE AZUL

CENATAV
Centro de Aplicaciones de Tecnologías de Avanzada
MINISTERIO DE LA INDUSTRIA BÁSICA

REPORTE TÉCNICO
**Reconocimiento de Patrones**

# Métodos de construcción de árboles de clasificación

Yenny Villuendas Rey, Anilú Franco Acerga,
José Ruiz Shulcloper

**RT_005** **Diciembre 2008**

7ma. No. 21812 e/218 y 222,
Rpto. Siboney, Playa;
Ciudad de La Habana.
Cuba. C.P. 12200
**www.cenatav.co.cu**

# Métodos de construcción de árboles de clasificación

Yenny Villuendas Rey, Anilú Franco Acerga, José Ruiz Shulcloper

Facultad de Informática, Universidad de Ciego de Ávila, Carretera a Morón Km. 9 ½, Ciego de Ávila, Cuba
Instituto Nacional de Astrofísica, Óptica y Electrónica, INAOE, Puebla, México
Centro de Aplicaciones de Tecnología de Avanzada, 7a #21812 e/ 218 y 222, Siboney, Playa, Habana, Cuba
yennyv@bioplantas.cu; anifranco6@yahoo.com.mx; jshulcloper@cenatav.co.cu

RT__005 CENATAV
Fecha del camera ready: diciembre del 2008

**Resumen:** Los árboles de clasificación han sido ampliamente utilizados como clasificadores supervisados, y numerosos algoritmos para su construcción se reportan en la literatura. En este reporte se exponen algunos de los métodos más significativos, y se analiza su desempeño en términos del trabajo con datos mezclados e incompletos. Se ofrecen además varias taxonomías que coadyuvan al mejor entendimiento y comparación de los diferentes métodos.

**Palabras clave**: Árboles de clasificación, clasificación supervisada, datos mezclados e incompletos.

**Abstract:** Classification trees have been widely used in literature as supervised classifiers, and a large amount of algorithms for constructing them has been reported. In this Technical Report we explain some of the most important of these algorithms, and we analyze their performance with mixed and incomplete data. We also offer several taxonomies which lead to a better understanding and comparison of the selected methods.

**Keywords: C**lassification trees, supervised classification, mixed and incomplete data.

## Introducción

Una de las estructuras de datos más utilizadas en la Computación es la estructura de árbol debido a sus numerosas aplicaciones. Los árboles se utilizan en algoritmos de ordenamiento (ej. *HeapSort*), en procesos de codificación (ej. código de Huffman), en la solución de problemas de optimización, entre otros. Dentro de la Inteligencia Artificial (IA) y el Reconocimiento de Patrones (RP) también se han utilizado estas estructuras con diversos fines. Algunos de ellos se listan a continuación:

- ✓ Para estimar las probabilidades condicionales de las clases [1].
- ✓ Para estimar valores incompletos en un conjunto de datos [2].
- ✓ Para generar reglas [3].
- ✓ Como clasificadores supervisados [4-6]. Es a este último uso que haremos referencia en este reporte.

En la clasificación supervisada en *k* clases, se tiene un conjunto de objetos que pertenecen respectivamente a dichas clases. A estos objetos se les denomina *muestra de entrenamiento* y a su representación matricial *matriz de entrenamiento*. El problema consiste en dado un nuevo objeto, asignarle la clase (o clases) a la que pertenece. Ejemplos de problemas de clasificación

supervisada son: dado un paciente, determinar si tiene cáncer (algún tipo de patología), dada una zona geológica saber si es perspectiva o no, entre otros.

Los árboles que son utilizados como clasificadores supervisados comúnmente se conocen como *árboles de clasificación*. Este uso de las estructuras de árbol data de la década de los setenta y es difícil precisar quién fue el primero en aplicarlos a la clasificación supervisada. Desde entonces, los árboles de clasificación constituyen un área de investigación activa dentro de la IA y el RP, y se han reportado numerosos algoritmos para su construcción.

Los árboles de clasificación han logrado un merecido reconocimiento dentro de la clasificación supervisada no paramétrica[1], debido entre otros aspectos, a los siguientes:

- ✓ Permiten explicar su comportamiento. Estos clasificadores, a diferencia de otros clasificadores supervisados como las Máquinas de Soporte Vectorial (SVM) [7] y las Redes Neuronales Artificiales (ANN) [8] pueden mostrar por qué a un determinado objeto se le asigna cierta clase. Esto permite a los usuarios o especialistas examinar las decisiones del clasificador hasta la asignación de la clase.
- ✓ Su representación gráfica generalmente resulta de fácil comprensión por los especialistas. Los árboles son representados por nodos y arcos, y en cada nodo se analizan uno (o varios) rasgos, y en los arcos se muestran los valores del (los) rasgo(s), hasta llegar a las clases (hojas) resultando en una estructura muy intuitiva (ver Fig. 1).
- ✓ No utilizan todos los rasgos para clasificar los objetos de todas las clases. Esta característica permite que clases menos complejas sólo utilicen un determinado conjunto de rasgos, mientras que otras más complejas usan otro conjunto de rasgos, por lo que no es necesario encontrar el conjunto de rasgos óptimo para discriminar entre todas las clases, sino que cada clase puede ser inferida a partir de su propio conjunto de rasgos. Nótese en la Fig. 1 cómo para clasificar objetos de la clase Nematelmintos sólo se utiliza el rasgo Celoma.
- ✓ El proceso de clasificación es muy rápido. Los árboles de clasificación operan en dos fases: entrenamiento y clasificación. En la primera (que se realiza una sola vez) se crea la estructura de árbol a partir de la matriz de entrenamiento. Luego, para clasificar un objeto, en dependencia de los valores de sus rasgos, éste recorre el árbol hasta llegar a una hoja (donde se le asigna una clase), por lo que el costo de clasificación es logarítmico.
- ✓ No utilizan funciones de comparación entre objetos. En numeroso problemas resulta difícil determinar de qué manera se comparan los objetos, y la eficacia de algunos clasificadores supervisados, por ejemplo el *k*-NN [9] depende en gran medida de la función de comparación que se utilice. Los árboles de clasificación, por otra parte, son capaces de inferir propiedades más generales de la matriz de entrenamiento, por lo que no necesitan comparar los objetos en el proceso de clasificación.

[1] Los clasificadores no paramétricos son aquellos que no asumen ningún conocimiento acerca de las distribuciones de los datos.

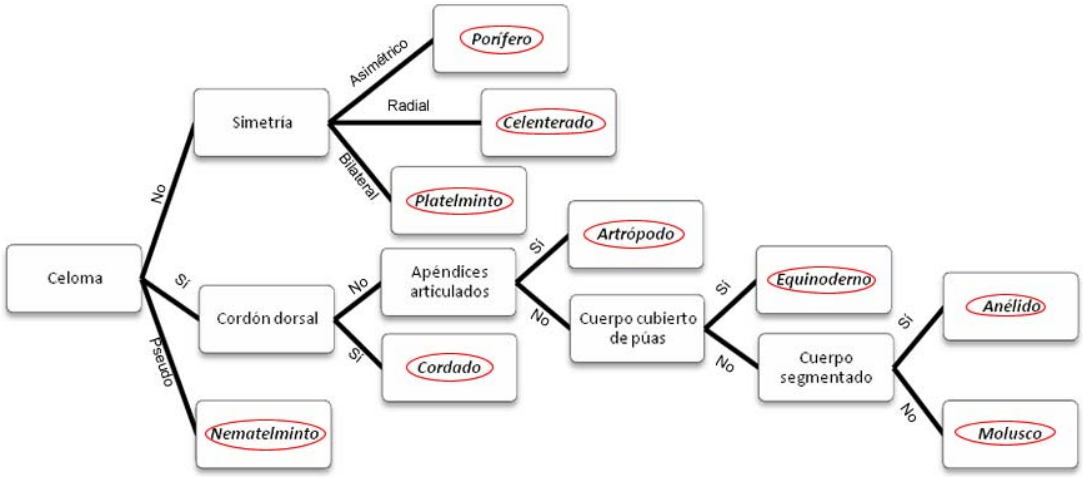


**Fig.1.** Árbol de clasificación de animales. Las clases se encuentran circuladas.

Entre los aspectos que intervienen en la construcción de un árbol de clasificación están los siguientes: selección del(los) rasgos(s) a evaluar en cada nodo, criterio de parada, tratamiento de diferentes tipos de datos (numéricos y no numéricos), tratamiento de las ausencias de información, etc. Especial significación reviste el tratamiento de datos mezclados e incompletos (MID), pues es sabido que en numerosos problemas los objetos están descritos simultáneamente por rasgos numéricos y no numéricos, y que en ocasiones se desconocen determinados valores en algunos rasgos.

La construcción de árboles de clasificación ha abordado el problema de los MID de varias formas diferentes:

- ✓ Discretizar los datos numéricos [10]. Esta estrategia tiene como desventaja que existen rasgos que no pueden ser discretizados (ej. la anomalía de Bouguer, en el pronóstico de las magnitudes máximas de terremotos [11]).
- ✓ Conversión de los datos no numéricos a numéricos. A cada valor de cada rasgo se le asigna un codificador numérico [12]. Esto tiene como desventaja que luego los datos cualitativos pueden ser sumados, multiplicados y promediados, perdiendo el sentido inicial del dato, ya que los códigos no son números.
- ✓ Analizar por separado los datos cuantitativos y los datos cualitativos [13]. En esta variante, no se toman en cuenta las posibles dependencias semánticas entre datos de diferente naturaleza. Por ejemplo, el mismo valor en el campo magnético será interpretado de forma diferente por un geofísico si el terreno es arenoso, rocoso o pantanoso. Al separar las variables numéricas del resto, no se analizan este tipo de situaciones.
- ✓ Trabajar directamente con los datos mezclados e incompletos [14]. Esta variante es considerada como la más cercana a la forma como trabajan los especialistas de las ciencias naturales y sociales en la realidad.

Como se mencionó anteriormente, numerosos algoritmos para la construcción de árboles de clasificación se han reportado en la literatura, y recientemente se han desarrollado nuevas estrategias de clasificación supervisada que utilizan estructuras de árbol. Entre ellas se encuentra la combinación de árboles de clasificación [15-18] y los árboles de clasificación difusos [19-21]. Sin embargo, estos temas están fuera del alcance de este reporte, que sólo hará referencia a los métodos para construir árboles de clasificación duros.

El reporte está estructurado como sigue: a continuación se presentan algunos conceptos relacionados con los árboles de clasificación, en la sección 3 se detallan algunos de los algoritmos para la construcción de árboles de clasificación más representativos de los reportados en la literatura, en la sección 4 se ofrece una taxonomía integrada de estos métodos y en la sección 5 aparecen las conclusiones.

## 1 Algunos conceptos necesarios

### 1.1 Conceptos generales

El concepto de objeto es primario y por tanto muy difícil de definir en función de otros conceptos más simples. En este trabajo se considera como objeto una entidad entendible por el ser humano, que interviene en un problema de RP. El conjunto de todos los objetos se llamará *universo* y se denotará con $U$. Ejemplos de objetos pueden ser pacientes, zonas geológicas, imágenes, entre otros.

Usualmente los objetos están descritos por un conjunto de $m$ rasgos $\chi = \{\chi_1, \chi_2, \ldots \chi_m\}$, que son extraídos del problema. Cada rasgo está definido en un dominio $D_i = dom(\chi_i)$. La función $\begin{matrix}\chi_i : U \to D_i \\ O \to \chi_i(O)\end{matrix}$ asocia a cada objeto el valor $\chi_i(O)$ de su atributo $\chi_i$. El dominio de definición de un atributo es su conjunto de valores admisibles. En dependencia de este conjunto se pueden tener atributos de diferentes tipos, por ejemplo: Booleanos ($D_i = \{0,1\}$); *k*-valentes ($D_i = \{0,1,\ldots k-1\}, k > 2$); reales ($D_i \subseteq \Re$); etc.

Como usualmente se trabaja con descripciones de los objetos, y no con los objetos como tal, se introduce la siguiente definición:

**Definición** 1. Una *descripción estándar* de un objeto $O$ con respecto a $\chi$: será un n-uplo,

$$\begin{matrix}I : U \to D_1 \times D_2 \times \ldots \times D_n \\ O \to I(O) = (\chi_1(O), \chi_2(O), \ldots, \chi_n(O))\end{matrix}$$

Usualmente no se trabaja directamente con los objetos, sino con su descripción en función de los rasgos seleccionados. En este documento utilizaremos la palabra “objeto” para referirnos tanto a los objetos, como a su descripción, pudiéndose seleccionar el significado real del contexto. El desconocimiento del valor de un atributo en un objeto se denota con el símbolo “?”, que se adiciona al conjunto de valores admisibles de dicho atributo. Es decir, si $? \in D_i$ puede haber desconocimiento del valor del atributo $\chi_i(O)$ en algún objeto.

### 1.2 Conceptos propios de los árboles de clasificación.

A continuación se detallan algunos conceptos específicos de los árboles de clasificación.

**Definición** 2. Un *árbol* es un grafo dirigido y acíclico. En una estructura de árbol, existe un nodo, denominado *raíz*, a partir del cual existe un camino hacia el resto de los nodos del árbol.

Si $(v, w)$ es un arco en un árbol ($v \neq w$), $v$ es considerado *padre* de $w$, y $w$ es considerado *hijo* de $v$. Si existe un camino de $v$ a $w$ entonces $v$ es *antecesor* de $w$ y $w$ es *descendiente* de $v$. Los nodos sin descendientes se denominas *hojas*. En un árbol de clasificación, a cada nodo se le asocia una regla de decisión y a cada hoja una clase.

**Definición** 3. Una *regla de decisión* es una proposición que divide a la matriz de entrenamiento en al menos dos subconjuntos de objetos no vacíos, en dependencia de los valores del (los) rasgo (s) que se tienen en cuenta en la proposición. En el caso de que se tenga en cuenta un rasgo que sólo tome dos valores (ej. Sexo), la proposición será booleana, y se obtendrán dos nuevos subconjuntos de objetos, si el rasgo que se analiza es *k*-valente (ej. Raza), la proposición será multivaluada, y se obtendrán *k* subconjuntos de objetos.

## 2 Algoritmos para la construcción de árboles de clasificación

Existen en la literatura numerosos algoritmos para la construcción de árboles de clasificación. En general, estos métodos utilizan diversas heurísticas para obtener una estructura de árbol que sea óptima o cercana al óptimo con respecto a un determinado criterio. Debido a la gran cantidad de métodos existentes que siguen estrategias diferentes, escogimos para su análisis a algunos de los más populares de las décadas de los 70, 80 y 90, y a algunos de los más recientes (2000 hasta la fecha). Tratamos, además, de mostrar en lo posible la mayor cantidad de estrategias diferentes, con el objetivo de obtener una visión general de la problemática que nos ocupa.

### 2.1 Algoritmos clásicos: el ID3 y el C4.5.

En las décadas del 70 y 80 hubo un gran auge en la construcción de árboles de clasificación, ejemplo de ello son los trabajos de [22-28]. Entre ellos, el ID3 [27] y el CART [23] son los más populares.

El ID3 es un algoritmo basado en la metodología de construcción (*Top-Down*), es decir, comienza a construir el árbol por la raíz, hasta llegar a las hojas. El ID3 funciona como sigue: inicialmente se selecciona un subconjunto aleatorio de la matriz de entrenamiento, llamado *window*, y utilizando estos objetos se construye el árbol. Luego, se clasifica toda la matriz de entrenamiento utilizando el árbol generado. Si no existen errores en la clasificación, el proceso termina, en caso contrario, se añaden al conjunto *window* los objetos mal clasificados y el proceso se repite hasta lograr clasificar correctamente toda la matriz de entrenamiento.

Para la construcción del árbol, en el ID3 sólo se utiliza un rasgo en la regla de decisión de cada nodo, y el proceso de construcción se basa en la Teoría de la Información, partiendo para ello de dos supuestos [27]:

Cualquier árbol de clasificación correcto[2] para la matriz de entrenamiento MI clasificará los objetos en la misma proporción que su representación en MI. Un objeto arbitrario será asignado a la clase P con probabilidad $p/p+n$ y a la clase N con probabilidad $n/p+n$, donde $n$ y $p$ representan la cantidad de objetos pertenecientes a las clases N y P, respectivamente. Cuando un árbol de clasificación se utiliza para clasificar un objeto, devuelve una clase. Así, un árbol de clasificación puede verse como una fuente que emite un mensaje "P" o "N", con la información esperada necesaria para generar este mensaje, la cual está dada por:

$$I(p,n) = -\frac{p}{p+n}\log_2(\frac{p}{p+n}) - \frac{n}{p+n}\log_2(\frac{n}{p+n}) \quad (1)$$

Al utilizar el rasgo $A = \{A_1, A_2, \cdots, A_m\}$ en un nodo, MI se particiona en $m$ subconjuntos $MI_i$, donde existen $n_i$ objetos de la clase N y $p_i$ objetos de la clase P. Luego, la información requerida para el subárbol de $MI_i$ está dada por $I(p_i, n_i)$. La información esperada que se requiere al utilizar el rasgo $A$ es la siguiente:

$$E(A) = \sum_{i=1}^{m} \frac{p_i + n_i}{p+n} I(p_i, n_i) \quad (2)$$

La información que se gana al dividir por $A$ es como sigue:

$$gain(A) = I(p,n) - E(A) \quad (3)$$

Así, en cada caso se escoge el rasgo que tiene una mayor ganancia de información para construir el árbol. Ya que $I(p,n)$ es constante para todos los rasgos, maximizar la ganancia es equivalente a minimizar la información esperada $E(A)$.

Para lidiar con la presencia de ruido en la matriz de entrenamiento, el ID3 utiliza una prueba de independencia a cada rasgo basada en el estadígrafo de Chi. De esta forma, no se tienen en cuenta los rasgos para los que no sea posible rechazar la hipótesis de independencia con respecto a las clases.

Para el manejo de datos incompletos, el ID3 realiza lo siguiente: Sea $p_u$ la cantidad de objetos con ausencia de información en el rasgo $A$, para cada posible valor $A_i$ se calcula la siguiente expresión:

[2] Un árbol de clasificación es *correcto* con respecto a un conjunto de objetos cuando clasifica correctamente a todos los objetos de ese conjunto.

$$ratio_i = \frac{p_i + n_i}{\sum_i p_i + n_i} \quad (4)$$

Luego, se calcula la ganancia como si el verdadero valor de $p_i$ estuviera dado por $p_i + p_u.ratio_i$. Así, se distribuyen los valores incompletos entre todos los posibles valores del rasgo, de forma proporcional a la frecuencia de aparición de los valores, y se penalizan los rasgos con mayor cantidad de ausencias de información, pues su ganancia se ve disminuida. Luego de seleccionado un atributo con valores incompletos, estos objetos se descartan y no pasan a formar parte del árbol.

Para clasificar un objeto con un árbol de clasificación ID3, el objeto va recorriendo el árbol en dependencia de sus valores hasta llegar a una hoja, donde se le asigna una clase. Para clasificar un objeto con valores incompletos, se utiliza la siguiente estrategia: si el valor del rasgo $A$ es desconocido, se exploran todas las ramas de ese rasgo, tomando en cuenta un valor dado por $T \cdot ratio_i$, donde $T$ es un valor arbitrario. El valor que se le pasa a una rama se va distribuyendo sucesivamente si aparecen más rasgos con valores incompletos. Como se tienen varios caminos, se suman los valores para cada clase y se asigna la clase de mayor valor.

Una de las principales desventajas de este método es que está concebido para el trabajo con dos clases solamente, y que originalmente no concibe el trabajo con datos numéricos. Otra desventaja es que el criterio de selección de rasgos (en este caso la ganancia de información), favorece a los rasgos con mayor cantidad de valores admisibles [10]. Además, al construir árboles correctos, las posibilidades de sobreentrenamiento del ID3 son elevadas, y pueden obtenerse árboles de un gran tamaño.

En la década del 90, se proponen nuevas estrategias para la construcción de árboles. Un estado del arte al respecto se encuentra en [29]. Entre las estrategias propuestas se destaca por su popularidad y simpleza el C4.5 [10]. El algoritmo C4.5 es una extensión del ID3. En este método se permite el trabajo con varias clases, modificando convenientemente las ecuaciones 1, 2 y 3 como se muestra a continuación.

$$I(T) = -\sum_{j=1}^{l} \frac{freq(K_j, T)}{|T|} \log_2 \left( \frac{freq(K_j, T)}{|T|} \right) \quad (5)$$

Al utilizar el atributo $A = \{A_1, A_2, \cdots, A_m\}$ en un nodo, se particiona $T$ en $m$ subconjuntos $T_i$ con $i = (1, \cdots, m)$. La información esperada al utilizar el rasgo $A$ es la siguiente:

$$E(A) = \sum_{i=1}^{m} \frac{|T_i|}{|T|} I(T_i) \quad (6)$$

La información que se gana al dividir por $A$ es como sigue:

$$gain(A) = I(T) - E(A) \quad (7)$$

Sin embargo, debido a que el criterio de la ganancia de información tiende a favorecer a los rasgos con mayor cantidad de valores [10], el C4.5 utiliza una nueva medida, la proporción de ganancia, que no es más que el cociente entre la ganancia y la información que se obtiene al dividir por el rasgo que se analiza.

$$gain\ ratio(A) = \frac{gain(A)}{E(A)} \quad \textbf{(8)}$$

La información esperada de dividir utilizando un rasgo $A$ se calcula como:

$$E(A) = \sum_{i=1}^{m} \frac{|T_i|}{|T|} log_2\left(\frac{|T_i|}{|T|}\right) \quad \textbf{(9)}$$

El C4.5, a diferencia del ID3, sí concibe el trabajo con rasgos numéricos. Para ello, cuando se analiza un rasgo numérico, éste es discretizado de la siguiente forma: sea un rasgo $A$ numérico, se toman todos los valores de éste que aparecen en los objetos del subconjunto que se analiza y se ordenan de forma ascendente. Luego, se calcula la proporción de ganancia que se obtendría al discretizar el rasgo $A$ utilizando cada uno de estos valores como punto de corte. Finalmente, se escoge el valor que maximice la proporción de ganancia (ver Fig. 1).

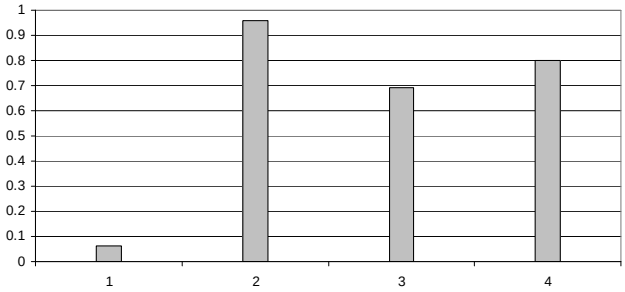


**Fig. 1.** Gráfico de valores contra proporción de ganancia.

El algoritmo C4.5 toma en cuenta las ausencias de información para decidir qué rasgo deberá formar el nodo a construir, esa ausencia de información reducirá la medida de proporción de ganancia de acuerdo al número de casos que la tengan. Para el tratamiento de los valores incompletos, el C4.5 modifica la fórmula de la ganancia. Sea $F$ la cantidad de objetos donde el valor del rasgo $A$ es conocido, la fórmula queda de la siguiente forma:

$$gain(A) = \frac{F}{|T|} I(T) - E(A) \quad \textbf{(10)}$$

Al dividir el conjunto de entrenamiento $T$, en el caso de tener ausencias de información, se asigna una probabilidad de pertenencia al subconjunto. Para ello, a cada objeto que pertenece a un subconjunto $T_i$, si su valor es conocido, se le asigna un peso $w(o)$ igual a 1, de lo contrario, se le asigna la probabilidad de obtener $A_i$ en ese subconjunto ($P(A_i)$). Como el proceso de división es iterativo, de manera general, a un objeto con peso $w(o)$ con un valor desconocido, será asignado a cada subconjunto $T_i$ con peso $w(o) = w(o) \cdot P(A_i)$.

$$P(A_i) = \frac{\sum_{o \in T, A(o)=A_i} w(o)}{\sum_{o \in T, A(o) \neq ?_i} w(o)} \quad (\mathbf{11})$$

Es decir, el peso del objeto estará determinado por la razón entre la suma de los pesos de los objetos que tienen valor $A_i$, y la suma de los pesos de los objetos que tienen un valor conocido para el atributo $A$.

Para clasificar un objeto con descripción incompleta, si se llega a un nodo donde el valor del rasgo que se analiza es desconocido en el objeto en cuestión, se explorarán todas las ramas del nodo y se combinarán los posibles resultados de la clasificación de forma aritmética, es decir, se calcula una tupla con las probabilidades frecuenciales de cada una de las posibles clases, y se asigna la clase de mayor probabilidad.

Una de las desventajas del ID3 era el sobreentrenamiento, para evitarlo, el C4.5 utiliza la poda basada en el error. Esta estrategia se basa en la estimación del error que se cometería en un nodo. Para ello, considera que el error está acotado por una distribución binomial para un nivel de confianza dado $\alpha$. Para calcular el error en una hoja, se estima una distribución binomial $U_\alpha(E, |T|)$ teniendo como argumentos la cantidad de objetos clasificados de forma incorrecta ($E$) y el total de objetos en la hoja. Así, el error predicho será $|T| \cdot U_\alpha(E, |T|)$. En cada subárbol, se calcula su error como la suma del error asociado a sus ramas. En caso de que un subárbol tenga un mayor error que si fuera reemplazado por una hoja, es podado. El procedimiento termina cuando no se pueden realizar más podas.

### 2.2 Propuestas posteriores al 2000

En el presente siglo, nuevas ideas se han desarrollado en cuanto a la construcción de árboles de clasificación [3, 13, 14, 30-32].

El *ModelTree* [13] es el primer algoritmo reportado en la literatura que permite el trabajo con rasgos numéricos sin utilizar técnicas de discretización. Este algoritmo genera un árbol de clasificación, incorporando máquinas de soporte vectorial suaves (*Smooth Support Vector Machines*SSVM - por sus siglas en inglés) al proceso de construcción del árbol. Las SSVM son utilizadas para tratar los rasgos numéricos, produciendo un atributo booleano sintético. Luego, se utiliza el criterio de proporción de ganancia [10] para seleccionar entre cada uno de los rasgos nominales y el atributo sintético, el más apropiado para dividir el árbol (ver Fig. 2). Se continúa dividiendo el árbol hasta que se cumpla un criterio de parada. En experimentaciones realizadas

en [13] utilizan como criterio de parada la mínima cantidad de objetos (si un nodo representa menos objetos que el número mínimo predefinido, no se divide).

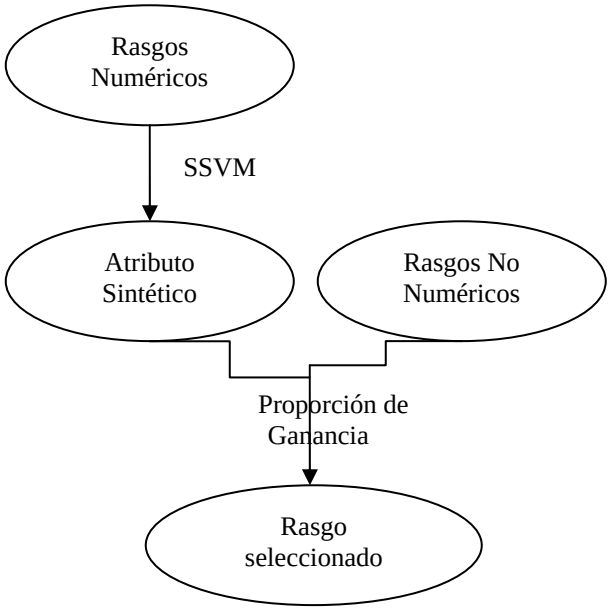


**Fig. 2.** Estrategia de selección de rasgos en el ModelTree.

La propuesta de [13] considera que no sólo las SSVM pueden ser utilizadas para el tratamiento de los rasgos numéricos, sino que también se pueden aplicar otros métodos como Redes Neuronales Artificiales, manteniendo el procedimiento de construcción del árbol. Por otra parte, sus autores consideran que es extremadamente importante diseñar un procedimiento de ajuste para disminuir el riesgo de sobreentrenamiento.

Una de las limitaciones que presenta este algoritmo es que realiza un cambio de espacio en las variables numéricas cuando usa SSVM, es decir, las procesa de tal manera que las convierte en variables booleanas. Además, para determinar qué atributo se utilizará en la construcción de los nodos, no se toman en cuenta, de manera simultánea, los datos mezclados, debido a que cada tipo es procesado de manera diferente [14]. Así, en problemas donde existan dependencias entre los rasgos, éstas no son tomadas en cuenta por el *ModelTree.*

Por otra parte, en el caso del manejo de descripciones incompletas de los datos, según sus autores es necesario “llenar” los valores incompletos para los rasgos numéricos, pues de lo contrario no es posible aplicar SSVM.

Una visión totalmente diferente del problema de la construcción de árboles de clasificación es la de Papagelis y Kalles [31]. Se trata de seleccionar el mejor árbol utilizando para ello Algoritmos Genéticos. La población la constituyen árboles binarios generados de forma aleatoria, y se modifican los operadores convencionales de mutación y cruzamiento para trabajar sobre estructuras jerárquicas. En el caso del operador de mutación, si se trata de mutar un nodo, se cambia el valor del rasgo por cualquiera de los posibles (si es un rasgo numérico se escoge un punto de corte), y si es una hoja, se cambia la clase por cualquier otra. Este proceso se muestra en la Fig. 4.

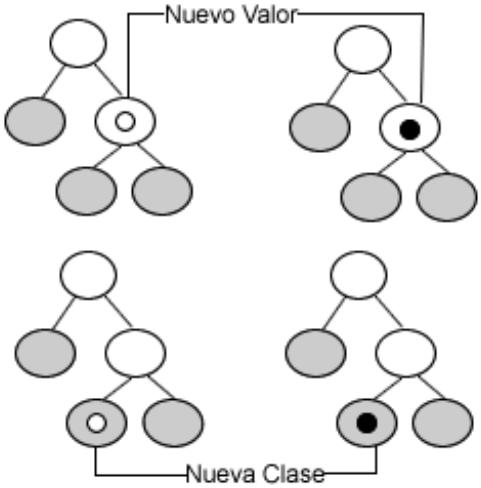


**Fig. 4.** Mutación en el GATree

La modificación realizada al operador de cruzamiento consiste en seleccionar dos nodos a cruzarse, y se intercambian los subárboles correspondientes a dichos nodos. Este proceso se muestra en la Fig. 5.

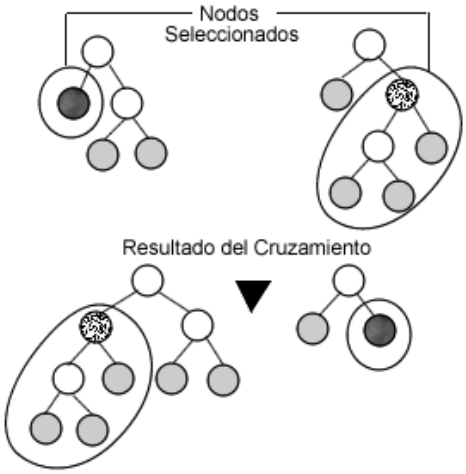


**Fig. 5.** Cruzamiento en el GATree

La función objetivo utilizada integra la eficacia del árbol al clasificar la matriz de entrenamiento y el tamaño del árbol, favoreciendo los más pequeños. Se incluye un factor de tamaño $x$ al cual se le asigna de forma arbitraria un valor elevado, de forma tal que se penalicen los árboles de menor tamaño.

$$payoff(T_i) = CorrectClassified_i^2\left(\frac{x}{size_i^2 + x}\right) \quad (12)$$

Para optimizar los parámetros del algoritmo, se utilizó un algoritmo genético, y se obtuvieron los siguientes resultados: la probabilidad de mutación igual a 0.005, la probabilidad de cruzamiento igual a 0.93, el uso de una función de optimización escalada para minimizar la similaridad entre individuos, Se utiliza una estrategia de selección con poblaciones solapadas, cada generación reemplaza el 25% de los peores individuos de la población anterior.

Debido al alto costo computacional de los Algoritmos Genéticos, en [31] se proponen algunas alternativas para aumentar la eficiencia del algoritmo propuesto.

Este algoritmo tiene como desventaja que sus resultados dependen fuertemente del azar, no se garantiza la convergencia del algoritmo en un número prefijado de generaciones y que en caso de tener los objetos descritos solamente por rasgos numéricos, este procedimiento sólo obtendría un árbol de un nodo y dos hojas, pues al aplicarse los operadores de cruzamiento y mutación no sería posible obtener otro tipo de árboles. No trabaja con datos incompletos. En presencia de ruido tiende a obtener clasificadores sobre entrenados [31].

Entre sus ventajas podemos mencionar que trata de lograr un óptimo desde el punto de vista global, lo que lo hace superior a estrategias golosas (*greedy*), y capaz de clasificar conceptos complejos con árboles pequeños. Debido a que busca la obtención de un óptimo global de acuerdo a la eficacia del clasificador, puede evitar los máximos locales y obtener combinaciones de rasgos en el árbol que logran mejores resultados, y al penalizar el tamaño del árbol, generalmente obtiene árboles pequeños. Otro enfoque es seguido en [14]. Este método, llamado ADT, construye el árbol basándose en los conceptos generados por el algoritmo RGC[33]. El RGC es un algoritmo de agrupamiento conceptual que devuelve no sólo los agrupamientos, sino los conceptos que los caracterizan. El RGC tiene dos etapas: en la primera, forma los agrupamientos y en la segunda halla los conceptos de cada agrupamiento.

Para hallar los conceptos que caracterizan a un agrupamiento, el RGC se basa en la construcción de la estrella de cada agrupamiento, que no es más que un conjunto de l-complejos maximales que son excluyentes[3] para el agrupamiento en cuestión. Un l-complejo booleano es una conjunción de selectores, y un selector está formado por un rasgo, un operador lógico y un conjunto de referencia. Ejemplos de selectores son los siguientes: $color = rojo$, $tamaño > 1.80$, $peso \in [55,70]$.

Un l-complejo entonces pudiera ser por ejemplo el formado por los selectores anteriores $(tamaño > 1.80) \wedge (peso \in [55,70]) \wedge (color = rojo)$. Luego de tener los l-complejos que forman la estrella, éstos son generalizados. Para ello se utilizan diferentes heurísticas en dependencia de los tipos de rasgos. El objetivo de la generalización es obtener un conjunto de l-complejos que sean caracterizantes del agrupamiento que se analiza, y que incluyan la información de todos los l-complejos que conforman la estrella.

En el ADT, se utiliza solamente la segunda etapa del RGC, para obtener así los conceptos que caracterizan a las clases del problema. El ADT simplifica estos conceptos, creando así los conceptos de cobertura mínima. En [14] se presenta un algoritmo para la obtención de estos

[3] Un l-complejo es excluyente para un agrupamiento cuando no existe en ningún otro agrupamiento un objeto que satisfaga al l-complejo.

conceptos de cobertura mínima (un concepto para cada clase) a partir de los conceptos obtenidos por el RGC.

Para la selección del rasgo (o rasgos) a utilizar en cada nodo, se introduce un nuevo criterio, que permite elegir entre las conjunciones (rasgos, valores) presentes en los conceptos de cobertura mínima, y que encuentra la separabilidad entre las clases del problema.

Para el cálculo de la separabilidad de una determinada conjunción $CJ$ se utiliza la siguiente expresión, donde $R_i^{CJ}$ es el conjunto de valores de la conjunción $CJ$ en el concepto de la clase $i$.

$$ST(CJ) = \begin{cases} CL(CJ)\ si \exists R_i^{CJ} \cap R_j^{CJ} = \phi, i \neq j \\ S(CJ)\ en\ otro\ caso \end{cases} \quad (13)$$

Para el cálculo de $CL(CJ)$ se utiliza:

$$CL(CJ) = \sum_{i=1}^{c} V_i \quad (14)$$

Donde $c$ es la cantidad de clases y $V_i$ se define como:

$$V_i = \begin{cases} 1\ si \exists R_i^{CJ} \cap R_j^{CJ} = \phi, i \neq j \\ 0\ en\ otro\ caso \end{cases} \quad (15)$$

De esta forma, si existe al menos una clase que se separa de las demás, $V_i$ toma valor 1.

En otro caso, si la conjunción $CJ$ no separa ninguna clase, se aplica la segunda parte de la ecuación (13). En caso de tratarse de un rasgo cualitativo, $S(CJ)$ se calcula como sigue:

$$S(CJ) = \frac{1}{c} \sum_{i=1}^{c} \left( \frac{\left| R_i^{CJ} \cap R_j^{CJ} \right|}{\left| R_i^{CJ} \cup R_j^{CJ} \right|} \right), i \neq j \quad (16)$$

En caso de tratarse de un rasgo cuantitativo, $S(CJ)$ se aplica la siguiente ecuación:

$$S(CJ) = \frac{1}{c} \sum_{i=1}^{c} \left( \sum_{v=1}^{n} \left( \frac{InD_v(CJ)}{InT_v(CJ)} \right) \right), i \neq j \quad (17)$$

Donde $n$ representa el número de elementos presentes en $R_i^{CJ}$ multiplicado por el número de elementos de $R_j^{CJ}$ y:

$$InD_v(CJ) = \left| \min\left\{R_i^{CJ}\right\} - \min\left\{R_j^{CJ}\right\} \right| + \left| \max\left\{R_i^{CJ}\right\} - \max\left\{R_j^{CJ}\right\} \right| \quad (18)$$

$$InT_v(CJ) = \max\left\{\min\left\{R_i^{CJ}\right\}, \min\left\{R_j^{CJ}\right\}\right\} - \min\left\{\max\left\{R_i^{CJ}\right\}, \max\left\{R_j^{CJ}\right\}\right\} \quad (19)$$

El árbol de clasificación en el ADT se construye de forma *Top-Down*. Esto se hace mediante una función recursiva que permite ir creando cada uno de los nodos (inicial, internos y terminales) que contendrá el árbol de clasificación final. Comienza por formar la raíz, para luego generar los nodos intermedios, hasta llegar a las hojas. Cada clase tendrá un camino único en el árbol, ya que está descrita por un solo concepto [14].

Este método, según sus autores, es robusto en presencia de ruido, pues no se basa directamente en las descripciones de los objetos de entrenamiento, sino en conceptos que permiten discriminar entre las clases. Tiene en cuenta durante el procesamiento las interacciones entre los rasgos y no realiza cambios en el espacio de representación de éstos. Sin embargo, el método no permite el trabajo con datos incompletos, ni con bases de datos en las cuales los conceptos generados por RGC tengan l-complejos elementales asociados.

Otra propuesta para la construcción de árboles de clasificación es la ofrecida por Alhammady y Ramamohanarao en [4]. Este método se basa en asignarle un peso por clase a cada objeto. De esta forma, el peso representará la fuerza de relación del objeto con cada una de las clases. Cada objeto $i$ tendrá asignado un peso $\delta_{C_j}(i)$, que representará su relación con la clase $C_j$, cumpliendo que $\sum_{j=1}^{c} \delta_{C_j}(i) = 1$, donde $c$ es el total de clases.

Para la construcción del árbol, se sigue un enfoque similar al del C4.5, aunque modificando las ecuaciones para tener en cuenta los pesos. La probabilidad para la clase $C_j$ de un conjunto de objetos $T$ estará dada por:

$$\bar{p}_j(T) = \frac{\sum_{i \in T} \delta_{C_j}(i)}{|T|} \quad (20)$$

La información para el conjunto de objetos $T$ quedaría:

$$Info_{WDT}(T) = -\sum_{j=1}^{c} \bar{p}_j(T) * \log_2\left(\bar{p}_j(T)\right) \quad (21)$$

De manera similar, la información que se espera al dividir utilizando un rasgo $A$ sería:

$$E_{WDT}(A) = \sum_{l=1}^{m} \frac{|T_l|}{|T|} * Info_{WDT}(T_l) \quad (22)$$

La ganancia de información en este esquema estaría dada por:

$$gain_{WDT}(A) = Info_{WDT}(T) - E(A) \quad \textbf{(23)}$$

Y la proporción de ganancia sería:

$$gainRatio_{WDT}(A) = \frac{Info_{WDT}(T)}{E_{WDT}(A)} \quad \textbf{(24)}$$

Para asignarle peso a los objetos, es necesario calcular primero todos los EP[4] (*Emerging Patterns*) [4] de la matriz de entrenamiento. Sea $E_{C_i}$ el conjunto de los EP para la clase $C_j$, y $e \in E_{C_j}$. La contribución de $e$ a la clase $C_j$ se define como:

$$\alpha_{C_j}(e) = \frac{gr_{C_j}(e)}{1 + gr_{C_j}(e)} * S_{C_j}(e) \quad \textbf{(25)}$$

Para determinar la contribución de un objeto $y$ a la clase $C_j$ se suman las contribuciones de los EP que el objeto contiene.

$$\beta_{C_j}(y) = \sum_{e \subseteq y, e \in E_{C_j}} \alpha_{C_j}(e) \quad \textbf{(26)}$$

Así, el peso de un objeto $y$ en la clase $C_j$ está dado por:

$$\omega_{C_j}(y) = \frac{\beta_{C_j}(y)}{Median_{C_j}} \quad \textbf{(27)}$$

Donde $Median_{c_j}$ es la mediana de las contribuciones de los EP que contiene el objeto. Los pesos para cada clase son normalizados como sigue:

$$\delta_{C_j}(y) = \frac{\omega_{C_j}(y)}{\sum_{f=1}^{c} \omega_{C_f}(y)} \quad \textbf{(28)}$$

En este método, el árbol es construido hasta que se obtengan nodos puros o se haya alcanzado un cierto factor de confianza. En esta propuesta, no se hace referencia al tratamiento

[4] Un patrón emergente es un conjunto de pares (rasgo, valor) que aparecen muy frecuentemente en una clase y poco frecuentemente en el resto de las clases.

de la ausencia de información, ni a la utilización de métodos de discretización para datos continuos.

En [34] se propone un método para la construcción de árboles que utiliza una estrategia *Bottom-Up*. El método, llamado ODT, sólo trabaja con rasgos numéricos en el intervalo [0,1]. El ODT construye un árbol que es óptimo globalmente con respecto a una función de ajuste, que pueden ser los errores de clasificación. Los autores definen una celda como uno de los paralepípedos que se obtiene de cortar recursivamente un hipercubo $[0,1]^d$ en dos mitades iguales, de forma perpendicular a uno de los ejes coordenados y en el punto medio de los valores de dicho eje.

Para la construcción de la estructura de árbol se obtiene para cada objeto (en este caso son puntos) la celda mínima que lo contiene. Para ello, el algoritmo tiene prefijado un valor máximo de cortes ($k_{\max}$), que indica la cantidad máxima de veces que se puede cortar utilizando cada eje. Así, se busca la partición óptima con las celdas mínimas y se almacenan en un diccionario, en la profundidad $dk_{\max}$. En cada paso, para cada una de las celdas, se calcula su hermana en cada eje, y se van uniendo o no en dependencia del valor de la función de optimización. Las celdas que se unen forman una nueva celda que sube un nivel ($dk_{\max}-1, dk_{\max}-2$) y así sucesivamente. El proceso termina cuando se llegue a la profundidad cero.

Luego de calculada la partición óptima, se construye el árbol, y en cada celda (hoja) se asigna la clase mayoritaria de los objetos que contiene. En caso de que la celda no contuviera ningún objeto, los autores plantean que pueden utilizarse diversas estrategias, como asignación aleatoria, o voto mayoritario de todas las celdas vecinas. Sin embargo, le asignan la clase de la celda padre en sus experimentaciones.

Este método tiene como ventajas que obtiene un óptimo global con respecto al error de clasificación, pero no trabaja con datos cualitativos ni permite en manejo de ausencias de información.

## 3 Taxonomías

En esta sección se presentan varias taxonomías que permiten agrupar a los diferentes métodos de construcción de árboles de acuerdo a sus características.

### 3.1 Diseño de la estructura del árbol (Taxonomía #1)

1) Encaminado a la solución de un problema particular [22, 24]. En estos casos, se generan árboles utilizando el conocimiento y las especificidades del problema en particular que se desea resolver con el objetivo de obtener una mejora en el proceso de clasificación.

2) Encaminado a óptimos globales [26]. Estos métodos generalmente utilizan estrategias de optimización de la programación matemática, y su objetivo es la construcción de árboles óptimos con respecto a un determinado criterio. De manera general son más complejos que los que utilizan heurísticas para la construcción del árbol, pero permiten obtener el mejor árbol de forma global con respecto al criterio que se desee.

3) Encaminado a óptimos locales [3, 10]. Este enfoque construye árboles de forma independiente del problema que se pretende resolver, se basa en heurísticas y en cada paso se trata de obtener nodos que optimicen un cierto criterio (ej. Proporción de ganancia en el C4.5). Dentro de este enfoque se agrupa una gran cantidad de algoritmos, y ha resultado uno de los más utilizados debido a su versatilidad. Generalmente son más sencillos que los que buscan óptimos globales.

### 3.2 Dirección de la construcción (Taxonomía #2)

1) De abajo hacia arriba (*Bottom Up*). En este enfoque, se construye un nivel del árbol en cada paso, comenzando por las hojas. En cada iteración, algunas hojas se unen para formar niveles superiores del árbol, hasta llegar a la raíz.

2) De arriba hacia abajo (*Top Down*). En este enfoque, inicialmente todos los objetos de entrenamiento se encuentran en la raíz, luego, en cada nodo se evalúa una regla de decisión y en dependencia de ésta los objetos son separados. Cuando se satisface un determinado criterio de parada, se le asignan las clases a las hojas del árbol.

### 3.3 Tipo de árbol que se genera (Taxonomía #3)

1) Árboles binarios. En ocasiones son preferidos por su simplicidad a la hora de construir el árbol, además, se han realizado varios desarrollos teóricos para la obtención de árboles óptimos utilizando árboles binarios (ej. [35]). Sin embargo, en el caso de rasgos que no sean Booleanos, los árboles binarios pierden en interpretabilidad y tienden a producir estructuras más profundas que las producidas por árboles n-arios.

2) Árboles n-arios. Son más complejos de construir, pero tienden a obtener árboles más sencillos y más fáciles de interpretar. Sin embargo, en nuestro conocimiento, no se han realizado trabajos para la obtención de árboles que satisfagan óptimos globales utilizando árboles n-arios, sino que se utilizan fundamentalmente en algoritmos basados en heurísticas.

### 3.4 Cantidad de rasgos que se utilizan en las reglas de decisión (Taxonomía #4)

1) Un rasgo. En cada nodo se utiliza un solo rasgo en la regla de decisión. Esto tiene como ventaja que sólo es necesario analizar un rasgo y sus valores en cada caso, lo que es poco costoso desde el punto de vista computacional, pero con esta estrategia pueden obtenerse árboles más profundos que si se utilizaran varios rasgos. Por ejemplo, en la Fig. 6, si se utilizan los rasgos $x$ y $y$ de conjunto, es posible discriminar las clases utilizando un solo nodo. En caso contrario, se obtiene un árbol más profundo.

2) Varios rasgos. Pueden utilizarse varios rasgos para conformar la decisión en un nodo. Esto tiene como ventaja que pueden construirse árboles más pequeños que reflejen

mejor la naturaleza del problema, pero por otra parte analizar combinaciones de rasgos y valores tiende a ser costoso desde el punto de vista computacional.

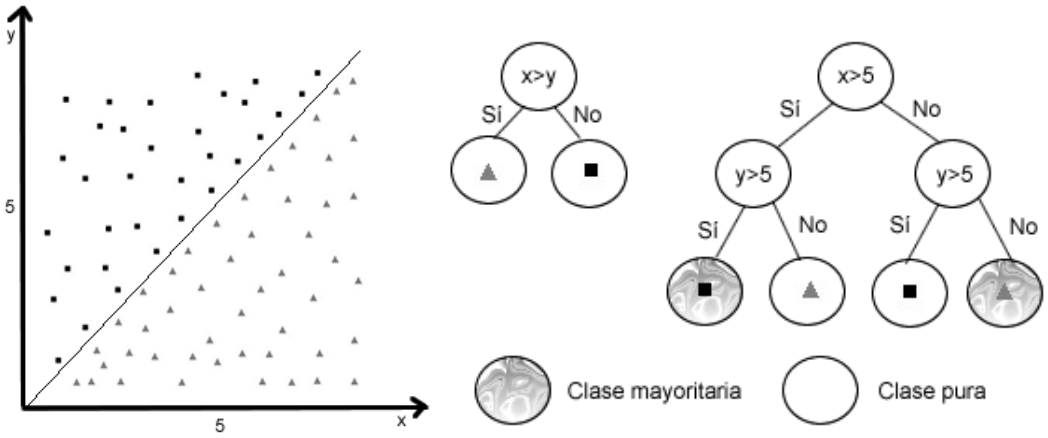


**Fig. 6.** Ejemplo 2D. Nótese la diferencia en los árboles construidos. En el primer caso, con un solo nodo se obtienen clases puras. En el segundo, es necesario utilizar tres nodos y no se clasifican correctamente todos los objetos.

### 3.5 Clasificación de los métodos analizados

| Método | T1 | T2 | T3 | T4 |
|---|---|---|---|---|
| ID3 | Óptimo local | Top Down | n-ario | Un rasgo |
| C4.5 | Óptimo local | Top Down | n-ario | Un rasgo |
| ModelTree | Óptimo local | Top Down | n-ario | Ambos |
| GATree | Óptimo global | Top Down | binario | Un rasgo |
| ADT | Óptimo local | Top Down | n-ario | Varios rasgos |
| WDT | Óptimo local | Top Down | n-ario | Un rasgo |
| ODT | Óptimo global | Bottom Up | binario | Varios rasgos |

## 4 Conclusiones

En este trabajo se exponen métodos para la construcción de árboles de clasificación. Se seleccionaron para su estudio algunos de los más populares y de los más recientes. Se brinda una explicación general del funcionamiento de cada método, haciendo referencia además a sus ventajas y desventajas. Además, en el Reporte se brindan varias taxonomías de los métodos de construcción de árboles de clasificación, que son de utilidad a la hora de realizar comparaciones entre diferentes métodos.

De manera general, las limitantes encontradas en los métodos de construcción de árboles de clasificación se muestran a continuación (no todos los métodos tienen todas las limitantes) y nuestro trabajo futuro estará encaminado a la solución de estas limitantes:

- ✓ Sólo se trabaja con datos numéricos (ej. [34]).
- ✓ Sólo se trabaja con datos no numéricos (ej. [27]).
- ✓ Se discretizan los datos numéricos (ej. [10]).
- ✓ No se manejan datos incompletos (ej. [14]).
- ✓ Los rasgos numéricos y no numéricos son procesados de forma independiente (ej.[13]).

## Referencias bibliográficas


[1] G. Blanchard, C. Schäfer, Y. Rozenholc, and K. R. Müller, "Optimal dyadic decision trees," *Machine Learning*, vol. 66, pp. 209-241, 2007.

[2] L. Hawarah, A. Simonet, and M. Simonet, "Evaluation of a probabilistic approach to classify incomplete objects using decision trees," *Lecture Notes on Computer Science*, vol. 4080, pp. 193-202, 2006.

[3] F. Berzal Galiano, "ART. Un método alternativo para la construcción de árboles de decisión," Tesis de Doctorado. *Departamento de Ciencias de la Computación e Inteligencia Artificial. E.T.S. Ingeniería Informática.*: Granada, España, 2002.

[4] H. Alhammady and K. Ramamohanarao, "Using emerging patterns to construct weighted decision trees," *IEEE Transactions on Knowledge Discovery and Data Engeneering*, vol. 18, pp. 865-876, 2006.

[5] P. Vermeesch, "Tectonic discrimination of basalts with classification trees," *Geochimica et Cosmochimica*, vol. 70, pp. 1839-1848, 2006.

[6] P. Knab, M. Pinzger, and A. Bernstein, "Predicting defect densities in source code files with decision tree learners," in *MSR'06*, Shanghai, China, 2006, pp. 119 - 125

[7] R. Duda, P. Hart, and D. Stork, *Pattern Classification*: John Wiley and Sons, Inc, 2000.

[8] T. Kohonen, *Self-organizing maps*, 3rd ed. Berlin; New York: Springer, 2001.

[9] B. V. Dasarathy, *Nearest neighbor (NN) norms: NN pattern classification techniques*. Los Alamitos, Calif.: IEEE Computer Society Press, 1991.

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

[11] J. Ruiz-Shulcloper, E. Alba-Cabrera, and M. Lazo-Cortés, *Introducción al Reconocimiento de Patrones (Enfoque Lógico - Combinatorio)*. México, D. F.: CINVESTAV-IPN, 1995.

[12] M. Chacón and O. Luci, "Patients Classification using Cluster Analysis and Genetic Algorithms," *Lecture Notes on Computer Science*, vol. 2905, pp. 350-358, 2003.

[13] H.-K. Pao, S.-C. Chang, and Y.-J. Lee, "Model Trees for Classification of Hybrid Data Types," in *Intelligent Data Engineering and Automated Learning - IDEAL: 6th International Conference*, Brisbane, Australia, 2005, pp. 32-39.

[14] A. Franco Acerga, "Un método para la generación de árboles de decisión basado en algoritmos conceptuales," Tesis de Maestría. *Instituto de Ciencias Básicas e Ingeniería. Centro de investigación en Tecnologías de la información y Sistemas* vol. M. Sc. Pachuca de Soto, Hidalgo: Universidad Autónoma del estado de Hidalgo, 2006.

[15] R. E. Banfield, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, "A comparison of decision tree ensemble creation techniques," *iEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 173-180, 2007.

[16] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *23th International Conference on Machine Learning*, Pitsburg, PA, USA, 2006, pp. 161 - 168.

[17] G. Martínez-Muñoz and A. Suárez, "Comité de árboles IGP," in *I simposio de inteligencia computacional - CEDI 2005*, 2005, pp. 277-283.

[18] D. Mease, A. J. Wyner, and A. Buja, "Boosted classification trees and class probability/quantile estimation," *Journal of Machine Learning Research*, vol. 8, pp. 409-439, 2007.
[19] X. Liu and W. Pedrycz, "The development of fuzzy decision trees in the framework of Axiomatic Fuzzy Set logic," *Applied Soft Computing* vol. 7, pp. 325-342, 2007.
[20] Z. Qin and J. Lawry, "Fuzziness and performance: An empirical study with Linguistic Decision Trees," *Lecture Notes on Artificial Intelligence*, vol. 4529, pp. 407-416, 2007.
[21] X. Wang, D. Nauck, S. Martin, and R. Kruse, "Intelligent data analysis with fuzzy decision trees," *Soft Comput*, vol. 11, pp. 439-457, 2007.
[22] P. Argentiero, R. Chin, and P. Beaudet, "An automated approach to the design of decision tree classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, pp. 51-57, 1982.
[23] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Montrey, CA: Wadsworth & Brooks, 1984.
[24] Y. X. Gu, Q. R. Wang, and C. Y. Suen, "Application of multi-layer decision tree in a computer recognition of Chinese characters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 83-89, 1983.
[25] J. Mingers, "An empirical comparison of selection measures for decision-tree induction," *Machine Learning*, vol. 3, pp. 319-342, 1989.
[26] H. Payne, W. Meisel, and R. B. Irani, "An algorithm for constructing optimal binary decision trees," *iEEE Transactions on Computing*, vol. 26, pp. 905- 916, 1977.
[27] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, pp. 81-106, 1986.
[28] I. K. Sethi and G. Sarvarayudu, "Hierarchical classifier design using mutual information," *iEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, pp. 441-445, 1982.
[29] S. Murthy, "Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey " *Data Minimg and Knowledge Discovery*, vol. 2, pp. 345-389, 1998.
[30] A. Atramentov, H. Leiva, and V. Honavar, "A Multi-relational Decision Tree Learning Algorithm - Implementation and Experiments," *Lecture Notes on Artificial Intelligence*, vol. 2835, pp. 38-56, 2003.
[31] A. Papagelis and D. Kalles, "GATree: Genetically Evolved Decision Trees," in *12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'00)*, 2000, pp. 203-208.
[32] W. N. Street, "Oblique Multicategory Decision Trees Using Nonlinear Programming," *INFORMS Journal on Computing* vol. 17, pp. 25-31, 2005.
[33] J. Martínez-Trinidad, A. Pons-Porrata, and J. Ruiz-Shulcloper, "Refunion - generalization - conceptual clustering algorithm," in *V Simposio Iberoamericano de Reconocimiento de Patrones*, Lisboa, Portugal, 2000, pp. 679-690.
[34] G. Blanchard, c. Schafër, Y. Rozenholc, and K. R. Müller, "Optimal dyadic decision trees," *Machine Learning*, vol. 66, pp. 209-241, 2007.
[35] C. Scott and r. D. Nowak, "Minimax-optimal classififcation with dyadic decision trees," *IEEE Transactions on Information Theory*, vol. 52, pp. 1335-1353, 2006.

RT _005, Diciembre 2008
Aprobado por el Consejo Científico CENATAV

**Editor:** Lic. Miriela Santos Toledo
**Diseño de Portada:** DCG Matilde Galindo Sánchez
RNPS No. 2142
ISSN 2072-6287
**Indicaciones para los Autores:**
Seguir la plantilla que aparece en www.cenatav.co.cu
C E N A T A V
7ma. No. 21812 e/218 y 222, Rpto. Siboney, Playa;
Ciudad de La Habana. Cuba. C.P. 12200
*Impreso en Cuba*