Capítulo 3

El modelo de clasificación ART

...
cincel que el bloque muerde
la estatua modelando,
...
atmósfera en que giran
con orden las ideas,
cual átomos que agrupa
recóndita atracción;

GUSTAVO ADOLFO BÉCQUER
Rimas

El acrónimo TDIDT hace referencia a todos los algoritmos "divide y vencerás" que construyen árboles de decisión desde la raíz hasta las hojas *recursivamente* (sección 2.1). En cierto modo, el modelo de clasificación que se presenta en este capítulo puede considerarse un algoritmo más de la familia TDIDT, pues construye un árbol de decisión comenzando por su nodo raíz.

El modelo aquí presentado, sin embargo, no utiliza una medida de impureza como la entropía, el criterio de proporción de ganancia o el índice de Gini. En vez de emplear una medida heurística, se utiliza la información obtenida a través de un algoritmo de extracción de reglas de asociación para decidir cómo ramificar el árbol. Esta estrategia da nombre al modelo propuesto en esta memoria: *Association Rule Tree* (ART).

ART construye un modelo de clasificación parcial con reglas de asociación y utiliza un subconjunto del conjunto de reglas obtenido para ramificar el árbol de decisión. Las reglas seleccionadas puede que no cubran algunos de lo ejemplos del conjunto de entrenamiento, por lo que dichos ejemplos se agrupan todos en una rama 'else' para ser procesados conjuntamente. El proceso se repite para descubrir reglas aplicables a los ejemplos de la rama 'else' mientras queden ejemplos de entrenamiento por clasificar.

Al finalizar la construcción del clasificador, el modelo obtenido por ART tiene la forma del mostrado en la figura 3.1. En esta figura se muestra un clasificador construido por ART cuyo objetivo es clasificar las uniones de genes en secuencias de ADN a partir de un conjunto de entrenamiento que contiene secuencias de nucleótidos. Este problema se describirá con mayor detalle en el apartado 3.3 de esta memoria.

En las siguientes secciones se analizará en profundidad el modelo de clasificación ART. La sección 3.1 estudia la construcción de clasificadores con ART, tras lo cual se muestra un ejemplo detallado en la sección 3.2. Las dos secciones siguientes describen el uso y propiedades de los clasificadores obtenidos utilizando el modelo propuesto en este capítulo. La sección 3.4 comenta algunos detalles relacionados con el uso de clasificadores ART, mientras que en la sección 3.5 se destacan las cualidades más interesantes. Finalmente, la sección 3.6 recopila los resultados experimentales que se han obtenido al utilizar ART en las resolución de distintos problemas de clasificación.

```
P30 = A : TYPE = N (473|62)
P30 = C : TYPE = N (441|24)
P30 = T : TYPE = N (447|57)
else
  P28 = A and P32 = T : TYPE = EI (235|33)
  P28 = G and P32 = T : TYPE = EI (130|20)
  P28 = C and P32 = A : TYPE = IE (160|31)
  P28 = C and P32 = C : TYPE = IE (167|35)
  P28 = C and P32 = G : TYPE = IE (179|36)
  else
    P28 = A : TYPE = N (106|14)
    P28 = G : TYPE = N (94|4)
    else
      P29 = C and P31 = G : TYPE = EI (40|5)
      P29 = A and P31 = A : TYPE = IE (86|4)
      P29 = A and P31 = C : TYPE = IE (61|4)
      P29 = A and P31 = T : TYPE = IE (39|1)
      else
        P25 = A and P35 = G : TYPE = EI (54|5)
        P25 = G and P35 = G : TYPE = EI (63|7)
        else
          P23 = G and P35 = G : TYPE = EI (40|8)
          P23 = T and P35 = C : TYPE = IE (37|7)
          else
            P21 = G and P34 = A : TYPE = EI (41|5)
            else
              P28 = T and P29 = A : TYPE = IE (66|8)
              else
                P31 = G and P33 = A : TYPE = EI (62|9)
                else
                  P28 = T : TYPE = N (49|6)
                  else
                    P24 = C and P29 = A : TYPE = IE (39|8)
                    else
                      TYPE = IE (66|39)
```

Figura 3.1: Un modelo de clasificación construido con ART.

3.1. Construcción del clasificador

El algoritmo ART de construcción de modelos de clasificación construye un árbol de decisión utilizando una estrategia de control irrevocable, igual que la mayor parte de los algoritmos TDIDT; es decir, emplea un algoritmo greedy en el cual una decisión no se revoca una vez que ha sido tomada.

A continuación se ofrece una visión general del algoritmo ART, describiendo a grandes rasgos el proceso de construcción del clasificador y omitiendo detalles que serán analizados en los apartados siguientes de este capítulo.

ART comienza el proceso de construcción del clasificador buscando reglas de asociación simples como $\{A_i.a_i\} \Rightarrow \{C.c_j\}$, donde A es un atributo, a es el valor de ese atributo, C es el atributo de la clase y c es una de las posibles clases del problema. Para ramificar el árbol, ART selecciona el atributo A con el *mejor conjunto de reglas adecuadas* $\{A_i.a_i\} \Rightarrow \{C.c_j\}$. Acto seguido, ART genera u nodo hoja para cada *regla adecuada* y todos los ejemplos del conjunto de entrenamiento no cubiertos por ninguna de esas reglas se agrupan en una rama 'else' para ser tratados a continuación aplicando recursivamente el mismo algoritmo sobre los datos que aún no han sido clasificados.

En este contexto, una *regla adecuada* es una regla precisa; esto es, una regla de asociación con confianza lo suficientemente alta para ser útil en la construcción de un buen clasificador. El *mejor conjunto de reglas adecuadas* es el conjunto de reglas más prometedor, el conjunto de reglas que parece ser mejor para construir un modelo predictivo desde un punto de vista heurístico, obviamente. Estos conjuntos de reglas se construyen de modo que todas las reglas de asociación incluidas en un conjunto dado compartan los atributos que aparecen en sus antecedentes. De este modo nos aseguramos de que las distintas ramas del árbol construido serán mutuamente excluyentes.

Cuando no se encuentra ninguna regla de asociación adecuada con un único par atributo-valor $A.a$ en su antecedente, ART busca reglas de asociación más complejas añadiendo atributos a sus antecedentes. Se empieza generando reglas de la forma $\{A_1.a_1\ A_2.a_2\} \Rightarrow \{C.c_j\}$. Si no se consiguen buenas reglas de asociación, ART prosigue su búsqueda con reglas que contengan tres pares atributo-valor en sus antecedentes, $\{A_1.a_1\ A_2.a_2\ A_3.a_3\} \Rightarrow \{C.c_j\}$, y

así sucesivamente.

En consecuencia, ART, a diferencia de otros algoritmos TDIDT, es capaz de utilizar varios atributos simultáneamente para ramificar el árbol de decisión. Esta característica posibilita una mejora en la capacidad del clasificador, incrementando su precisión y disminuyendo su complejidad [163].

Es aconsejable, no obstante, establecer una cota superior para el tamaño de los antecedentes de las reglas de asociación. Esta cota sirve para detener la búsqueda si no se encuentran reglas adecuadas y se designa por *MaxSize*. El máximo valor posible para esta cota es igual al número de atributos predictivos en el conjunto de datos de entrenamiento. Este valor máximo para el parámetro *MaxSize* viene dictado por la Primera Forma Normal, que establece que los valores de un atributo han de ser atómicos: un caso de entrenamiento no puede tener, por tanto, más pares atributo-valor que atributos predictivos haya en el conjunto de entrenamiento.

> *Que sea sencillo: lo más sencillo posible, pero no más.*
>
> ALBERT EINSTEIN

Tal como se ha comentado en los párrafos anteriores, ART comienza utilizando hipótesis sencillas para intentar clasificar los ejemplos del conjunto de entrenamiento y sólo recurre a hipótesis más complejas cuando no se pueden encontrar hipótesis más simples que se adapten a los datos. Por tanto, ART incorpora un criterio explícito de preferencia por modelos sencillos utilizando el Principio de Economía de Occam como sesgo inductivo. A pesar de las críticas que la "navaja" de Occam ha recibido en algunos trabajos de extracción de conocimiento en bases de datos [47] [48], este criterio que aún sigue siendo adecuado y útil para resolver problemas de clasificación de muy diversa índole. De cualquier forma, siempre es necesario algún tipo de sesgo inductivo para construir clasificadores, ya que el número de hipótesis que implican unos hechos dados es potencialmente infinito [63]. En este contexto, el criterio de Occam sigue siendo tan válido como cualquier otro.

El diagrama de la página siguiente muestra el flujo de control del algoritmo ART, cuyo pseudocódigo aparece esbozado en la figura 3.3 (página 68).

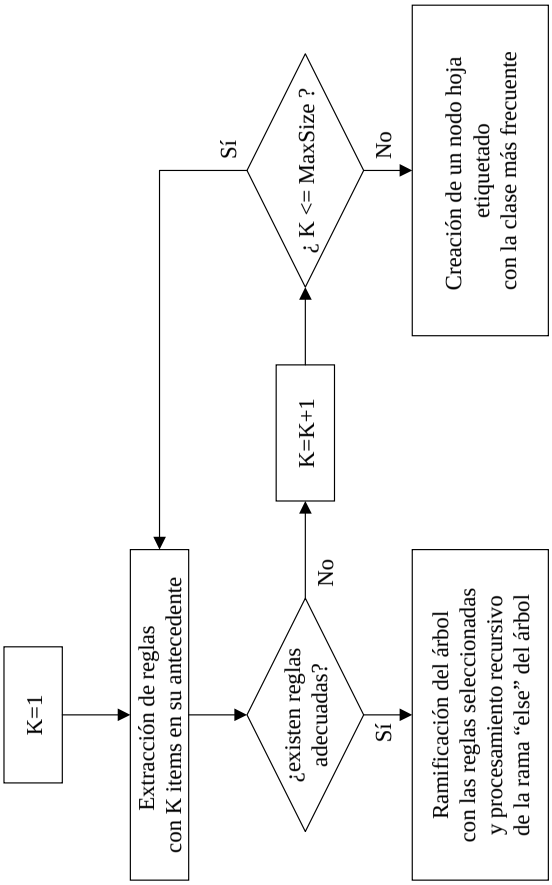

Figura 3.2: Algoritmo ART

En el pseudocódigo de ART mostrado en la figura 3.3 aparece indicado el uso de distintos parámetros. La tabla 3.1 describe la función de estos parámetros, que son necesarios para acotar el espacio de búsqueda explorado al construir el árbol de decisión. Las implicaciones del uso de estos parámetros serán objeto de estudio en los siguientes apartados, conforme se discutan los distintos aspectos relacionados con la construcción de clasificadores ART.

3.1.1. Selección de reglas: Criterio de preferencia

Cuando se construye un árbol de decisión, se ha de elegir cómo ramificar el árbol de entre las distintas particiones posibles del conjunto de datos de entrada, para lo que se han propuesto distintas reglas de división (apartado 2.1.1). Por ejemplo, ID3 y C4.5 utilizan reglas derivadas de la Teoría de la Información mientras que CART emplea el índice de Gini. De hecho, todos los algoritmos clásicos de construcción de árboles de decisión utilizan alguna medida de impureza o diversidad que tratan de minimizar.

El enfoque de ART es diferente. Una vez obtenidas algunas reglas adecuadas para clasificar los datos de entrada (un modelo de clasificación parcial), se seleccionan las mejores de ellas para construir un nivel del árbol de decisión (que una vez acabado será un modelo de clasificación completo por sí mismo).

Nuestro algoritmo comprueba la existencia de, al menos, un conjunto de atributos $\{A_1\ A_2\ ..\ A_k\}$ que sea adecuado para ramificar el árbol de decisión. El conjunto de atributos seleccionado debe proporcionar un conjunto de reglas precisas de la forma $\{A_1.a_1\ A_2.a_2\ ..\ A_k.a_k\} \Rightarrow \{C.c_j\}$. Cada regla de ese tipo se empleará para etiquetar los datos de entrada que verifiquen su antecedente como pertenecientes a la clase c_j.

El propósito de ART es aprovechar la información obtenida en forma de reglas de asociación para construir un buen clasificador. Todas las reglas extraídas en cada etapa del algoritmo incluyen *k*-itemsets en sus antecedentes. Cada *k*-itemset corresponde a un conjunto de k atributos diferentes (recuérdese nuevamente la Primera Forma Normal). Las reglas correspondientes a cada conjunto distinto de k atributos se agrupan y se recurre a heurísticas para seleccionar, de entre todos los conjuntos de k atributos, el mejor candidato para ramificar el árbol.

función ART (datos, MaxSize, MinSupp, MinConf): clasificador;

k = 1; *// Tamaño del antecedente*
árbol = $\varnothing$; *// Árbol resultante*

mientras ((árbol = $\varnothing$) **y** ($k \leq$ MaxSize))

// Extracción de reglas
Encontrar reglas en el conjunto de datos de entrada con
k items en el antecedente y la clase en el consecuente
v.g. $\{A_1.a_1$ .. $A_k.a_k\} \Rightarrow \{C.c_j\}$

si existen reglas buenas

// Selección de reglas
Seleccionar el mejor conjunto de reglas con el mismo
conjunto de atributos $\{A_1..A_k\}$ en el antecedente
de acuerdo con el criterio de preferencia.

// Ramificación del árbol
árbol = A partir de las reglas seleccionadas
- Cada regla $\{A_1.a_1$ .. $A_k.a_k\} \Rightarrow \{C.c_j\}$
se utiliza para crear una nueva hoja (etiquetada c_j).
- Todos los ejemplos de entrenamiento no cubiertos
por las reglas seleccionadas se agrupan en una rama
'else' que se construye recursivamente:
árbol.else = ART (datos, MaxSize, MinSupp, MinConf);

si no
k = k + 1;

si árbol = $\varnothing$ *// No se ha construido árbol alguno*
árbol = Nodo hoja etiquetado con la clase más frecuente;

devuelve árbol;

Figura 3.3: Pseudocódigo del algoritmo ART

Parámetro	*Descripción*
MaxSize	Tamaño máximo del antecedente de las reglas empleadas para construir el árbol de decisión. Por defecto es igual al número de atributos predictivos presentes en el conjunto de entrenamiento, si bien conviene establecer un valor menor para reducir el tiempo necesario para construir el árbol.
MinSupp	Umbral de soporte mínimo utilizado por el algoritmo de extracción de reglas de asociación. Su valor por defecto puede establecerse en torno a 0.1 (10 %). Este parámetro permite restringir el factor de ramificación del árbol de decisión y ajustar la granularidad del modelo construido (sección 3.5).
MinConf	Umbral de confianza mínimo para las reglas de asociación. Por defecto se utiliza una heurística que permite seleccionar automáticamente este parámetro, por lo que, normalmente, el usuario no tiene por qué especificarlo (véase la discusión sobre la selección del umbral en el apartado 3.1.3).

Tabla 3.1: Descripción de los parámetros de ART.

NOTA: A estos parámetros hay que añadirles el conjunto de datos de entrenamiento utilizado para construir el clasificador.

Por ejemplo, podríamos haber obtenido los siguientes conjuntos de reglas (todas las cuales satisfacen las restricciones de soporte y confianza en nuestro problema hipotético):

`Conjunto 1`:	`Regla 1`:	if W=w_4 and Y=y_2 then C=c_2
	`Regla 2`:	if W=w_3 and Y=y_3 then C=c_1
`Conjunto 2`:	`Regla 3`:	if X=x_1 and Z=z_1 then C=c_4
	`Regla 4`:	if X=x_1 and Z=z_5 then C=c_2

Como ART emplea una estrategia de control irrevocable, tenemos que seleccionar el `Conjunto 1` o el `Conjunto 2` y olvidarnos del otro.

Una heurística sencilla consiste en escoger el conjunto de atributos que clasifica correctamente más ejemplos del conjunto de entrenamiento (utilizando las reglas asociadas al conjunto de atributos seleccionado, obviamente). Dicha heurística hace máximo el número de ejemplos clasificados correctamente y, por tanto, hace mínimo el número de ejemplos que quedan por clasificar tras construir un nivel del árbol de decisión. Por tanto, indirectamente tiende a reducir la altura del árbol de decisión siguiendo el Principio de Economía de Occam.

Continuando el ejemplo anterior, si el primer conjunto de reglas (`Regla 1` y `Regla 2`) cubre a 200 ejemplos del conjunto de entrenamiento mientras que el segundo, formado por (`Regla 3` y `Regla 4`), sólo abarca 150 ejemplos, optaríamos por utilizar el `Conjunto 1` para ramificar el árbol.

3.1.2. Topología del árbol: Ramas 'else'

Una vez que hemos seleccionado un conjunto de reglas con el mismo conjunto de atributos en todos sus antecedentes, tenemos que ramificar el árbol utilizando la información obtenida durante el proceso de extracción de reglas de asociación:

- Cada una de las reglas seleccionadas conduce a una nueva hoja del árbol. Cada una de estas hojas se etiqueta con la clase que aparece en el conse-

cuente de la regla correspondiente. Siguiendo la notación de Quinlan en C4.5 [131], las etiquetas de las hojas pueden venir acompañadas por dos números $(n|e)$. El primero de ellos, n, indica el número de ejemplos de entrenamiento que caen en cada hoja (es decir, el soporte del antecedente de la regla correspondiente); mientras que el segundo, e, es el número de ejemplos del conjunto entrenamiento mal clasificados por la etiqueta de la hoja.

- Todos los ejemplos del conjunto de entrenamiento que no verifican el antecedente de ninguna de las reglas seleccionadas se agrupan en una rama 'else' para continuar por ella la construcción del árbol de decisión.

Utilizando el ejemplo de la sección anterior, se crearían dos nuevos nodos hoja al incluir dos reglas diferentes el conjunto seleccionado. Todos los ejemplos que verifiquen el antecedente de cualquiera de las dos reglas (`1` y `2`) se envían a las dos hojas recién creadas y los restantes casos del conjunto de entrenamiento se agrupan en una rama 'else':

W=w_4 and Y=y_2: C=c_2 (92|2)
W=w_3 and Y=y_3: C=c_1 (108|5)
`else`
...

ART difiere de los algoritmos TDIDT tradicionales porque utiliza ramas 'else' para agrupar todas aquellas tuplas que no correspondan a ninguna hoja de las generadas en cada nivel del árbol. En otras palabras, el conjunto de datos que queda sin cubrir tras ramificar el árbol se trata como un todo. Por tanto, ART no necesita crear una nueva rama para cada posible valor de los atributos seleccionados. ART centra su atención únicamente en aquéllos valores que generan resultados interesantes, reduciendo así el factor de ramificación del árbol. Este hecho, a su vez, permite que ART utilice simultáneamente varios atributos para ramificar el árbol, en vez del único atributo o test binario que usan casi todos los algoritmos TDIDT.

La utilización de ramas ‘else’ conduce a la construcción de un árbol de decisión cuya topología se asemeja a la de una lista de decisión. Este modelo de representación del conocimiento es más difícil de interpretar que los árboles de decisión tradicionales, pues el significado de una regla vendrá determinado por su posición en el modelo de clasificación construido. No obstante, el uso de ramas ‘else’ ayuda a construir mejores clasificadores agrupando pequeños subconjuntos de datos que, de otra forma, corresponderían a distintos subárboles en algoritmos como C4.5. Además, dichos conjuntos de datos llegarían eventualmente a ser tan pequeños que resultarían inútiles para construir modelos de clasificación exactos. Agrupando esos pequeños conjuntos de datos se puede conseguir una mayor precisión porque el efecto del ruido se reduce y se evita el problema del sobreaprendizaje.

En cierto sentido, ART sigue la filosofía de los algoritmos propuestos por Liu y sus colaboradores [101][102], quienes intentan organizar y resumir el conjunto de todas las reglas descubiertas mediante reglas más generales y excepciones a esas reglas. El objetivo es reemplazar un conjunto de potencialmente muchas reglas por una representación más concisa del conocimiento obtenido. El enfoque de ART a la hora de construir árboles de decisión permite alcanzar este objetivo conforme se crea el modelo de clasificación, mientras que Liu y sus colaboradores lo logran a posteriori.

Un efecto secundario del uso de ramas ‘else’ es que ART puede construir un árbol de decisión iterativamente, ya que la recursión de cola del algoritmo recursivo puede eliminarse para obtener un algoritmo iterativo equivalente, tal como se muestra en la figura 3.4. Los demás algoritmos TDIDT son inherentemente recursivos y requieren una implementación algo más compleja. De hecho, una implementación iterativa de nuestro algoritmo permite un aprendizaje más eficiente de árboles de decisión cuando el conjunto de datos de entrada es enorme.

Además, existen algoritmos eficientes de extracción de reglas de asociación y se pueden utilizar distintas técnicas para mejorar el rendimiento de ART. Por ejemplo, el tamaño del conjunto de entrenamiento se puede ir reduciendo conforme se profundiza en el árbol (e, independientemente, conforme se van generando itemsets [120]).

```
función ARTiterativo
        (datos, MaxSize, MinSupp, MinConf): clasificador;

var árbol: clasificador;                    // Árbol de decisión

nodo = árbol.raíz;                          // Comenzar por la raíz del árbol
entrenamiento = datos;                      // Conjunto de entrenamiento actual

mientras entrenamiento ≠ ∅                  // Construir nivel actual

        k = 1;                              // Tamaño del antecedente
        nuevoNodo = ∅;                      // Nuevo subárbol

        mientras ( (nuevoNodo=∅) y (k ≤ MaxSize) )

                Extraer reglas con k items en el antecedente

                si existen reglas con las cuales ramificar el árbol

                   Seleccionar del mejor conjunto de reglas
                   nuevoNodo = Nuevo nodo interno con:
                                - Una hoja por cada regla seleccionada.
                                - Una rama 'else' inicialmente vacía.
                   Reemplazar 'nodo' con 'nuevoNodo' en el árbol
                   nodo = nuevoNodo.else;
                   Eliminar los ejemplos de entrenamiento cubiertos
                             por alguna de las reglas seleccionadas.

                si no
                   k = k + 1;

        si nuevoNodo = ∅ // No se ha construido subárbol alguno
           Reemplazar 'nodo' con un nodo hoja
              (etiquetado con la clase más frecuente en 'entrenamiento')
           entrenamiento = ∅;

devuelve árbol;
```

Figura 3.4: Implementación iterativa de ART (completamente equivalente a la implementación recursiva del algoritmo que aparece en la figura 3.3).

3.1.3. Extracción de reglas: Hipótesis candidatas

La primera etapa de ART consiste en descubrir aquellas reglas derivadas del conjunto de entrenamiento que puedan ser potencialmente útiles para clasificar. Dichas reglas se utilizan para ramificar el árbol de decisión que ART construye de la forma que hemos visto en los apartados anteriores. Para completar el proceso de construcción del clasificador ART, por tanto, resulta imprescindible el uso de alguna técnica que nos permita descubrir tales reglas en el conjunto de datos de entrenamiento.

En el capítulo anterior se comentaron distintas técnicas de inducción de reglas y su uso en la construcción de clasificadores. Entre las técnicas analizadas se encontraban múltiples algoritmos de extracción de reglas de asociación, los cuales destacan por ser altamente eficientes y escalables. Los algoritmos tradicionales de extracción de reglas de asociación están ideados para extraer reglas de bases de datos transaccionales y no son adecuados para trabajar con conjuntos de datos densos (esto es, conjuntos de datos con un número relativamente pequeño de valores desconocidos). Desgraciadamente, los conjuntos de datos que se utilizan para resolver problemas de clasificación suelen ser densos al provenir de bases de datos relacionales. Por suerte, existen a nuestra disposición algoritmos diseñados para resolver este problema. Tal como se verá en el capítulo siguiente de esta memoria, un algoritmo como TBAR [19] resulta especialmente adecuado, pues está pensado explícitamente para extraer reglas de asociación en bases de datos relacionales.

El proceso de extracción de reglas de asociación usual consiste en descubrir todas las reglas de asociación que verifiquen las restricciones impuestas por el usuario mediante los umbrales de soporte y confianza, *MinSupp* y *MinConf* respectivamente. Evidentemente, a la hora de construir modelos de clasificación sólo nos interesan, de todas las reglas posibles, aquellas reglas que incluyan el atributo correspondiente a la clase en su consecuente.

Al emplear un algoritmo de extracción de reglas de asociación para construir un modelo de clasificación parcial, ART requiere, al menos en principio, que el usuario especifique los parámetros que guían el proceso de extracción de reglas de asociación: el umbral de soporte mínimo para los itemsets frecuen-

tes (*MinSupp*) y un valor mínimo para la confianza de las reglas de asociación (*MinConf*), aunque este último podrá omitirse, como se verá posteriormente.

3.1.3.1. El umbral de soporte mínimo: *MinSupp*

El umbral de soporte mínimo *MinSupp* puede fijarse como una cantidad determinada de tuplas o como un porcentaje sobre el tamaño del conjunto de entrenamiento (que va disminuyendo de tamaño conforme el algoritmo avanza). En el primer caso, se establece de antemano un umbral absoluto y se mantiene durante todas las etapas de construcción del árbol de decisión. Cuando se utiliza un umbral relativo, no obstante, el umbral real se va adaptando al tamaño del conjunto de datos.

Si, por ejemplo, el umbral de soporte relativo *MinSupp* se fija en 0.1 y comenzamos con 1000 tuplas en el conjunto de entrenamiento, el umbral de soporte absoluto será igual a 100 tuplas en la raíz del árbol e irá disminuyendo conforme el algoritmo avance. Cuando queden N tuplas en el conjunto de entrenamiento se utilizará 0.1*N como umbral de soporte para obtener los itemsets frecuentes en el conjunto de entrenamiento que nos queda, itemsets que quizá no sean frecuentes en el conjunto de entrenamiento completo.

Al establecer *MinSupp* como un porcentaje sobre el tamaño del conjunto de casos que nos queda por procesar, este parámetro se ajusta al tamaño de ese conjunto de forma que el usuario no tiene que afinarlo. Un valor de este umbral relativo en torno a 0.1 es razonable para construir árboles de decisión que no sean ni demasiado frondosos ni excesivamente escuetos.

Implícitamente, el umbral de soporte restringe el factor de ramificación del árbol. De esta forma, ART se asegura de que nunca se escogerán claves primarias ni claves candidatas para ramificar el árbol de decisión, un problema bastante común en otros algoritmos TDIDT.

En la sección 3.5 se puede encontrar una discusión algo más formal acerca del efecto que causa el umbral *MinSupp* en las propiedades del árbol de decisión construido por ART.

3.1.3.2. El umbral de confianza mínima: *MinConf*

> *...es característico de una mente instruida descansar satisfecha con el grado de precisión permitido por la naturaleza en cada asunto y no buscar la exactitud cuando sólo es posible una aproximación de la verdad...*
>
> ARISTÓTELES
> *330 a.C.*

Veamos, a continuación, cómo trabajar con el otro umbral que interviene en el proceso de extracción de reglas, el que establece la confianza mínima de las reglas de asociación: *MinConf*. Este parámetro influye decisivamente en el conjunto de reglas que se considerarán candidatas para ramificar el árbol y, por tanto, debería tener un valor cercano a 1 para asegurar que sólo se tendrán en cuenta reglas muy precisas (0.9 produce buenos resultados en muchos problemas). En nuestros experimentos hemos encontrado, no obstante, que el parámetro *MinConf* no debería ajustarlo el usuario. Como mostraremos, permitir que ART lo ajuste automáticamente suele conducir a mejores resultados.

De cualquier modo, si el usuario así lo exige, puede establecer manualmente un valor para el umbral *MinConf*. Un problema típico en el que el usuario puede estar interesado en fijar un umbral ad hoc es el conjunto de datos MUSHROOM del Machine Learning Repository de la Universidad de California en Irvine. Este conjunto de datos se utiliza para construir clasificadores que ayuden a determinar si una seta es venenosa o no a partir de sus propiedades organolépticas. Dado que un simple error de clasificación podría tener consecuencias dramáticas, un falso positivo no es permisible: Etiquetar una seta venenosa como comestible podrían conllevar graves problemas de salud para el degustador de setas, incluso su muerte. Por tanto, este conjunto de datos requiere un estricto umbral de confianza mínima igual a 1 (100 %).

En problemas como el mencionado en el párrafo anterior, es imprescindible la utilización de un umbral de confianza fijo para que el modelo de clasificación construido por ART tenga las propiedades deseadas por el usuario.

En tales situaciones, incluso, se podría plantear la conveniencia de establecer un umbral de confianza independiente para cada clase del problema, dependiendo de su semántica. En el problema concreto de las setas, que requiere la ausencia absoluta de falsos positivos para garantizar que no se ponga en peligro la salud de ningún aficionado a la micología, un falso negativo no tendría mayores consecuencias: etiquetar como venenosa una seta comestible sólo evita su posible degustación y no acarrea mayores complicaciones.

A pesar de la discusión recogida en los párrafos anteriores, un conjunto de datos como MUSHROOM es un caso extremo. Establecer a priori un umbral de confianza mínima puede ser contraproducente si dicho umbral no es realista, ya que impediría obtener modelo de clasificación alguno si su valor fuese excesivamente alto para un conjunto de datos particular. En esta situación, no se extraería ninguna regla de asociación del conjunto de datos y no se podría seguir ramificando el árbol de decisión utilizando ART (si bien siempre se puede recurrir a un algoritmo TDIDT tradicional es estos casos). Por otro lado, elevar el valor del umbral *MinConf* no implica necesariamente obtener un porcentaje de clasificación mayor (véase el apartado 3.6.4) y, desde un punto de vista práctico, el tiempo de ejecución del algoritmo se podría incrementar notablemente.

Visto lo anterior, resulta conveniente, si no necesario, incluir en ART algún mecanismo automático de selección del umbral de confianza. Una vez que se haya descubierto la regla de asociación más precisa posible (aquélla con mejor valor de confianza), sólo se deberían tener en cuenta reglas similares en cuanto a su precisión a la hora de construir el árbol de decisión, con el objetivo de que la precisión del modelo obtenido no se degrade innecesariamente.

Una heurística que hemos encontrado y parece funcionar bastante bien consiste en considerar en cada etapa del algoritmo sólo aquellas reglas cuya confianza supere el valor *MaxConf* - Δ, donde *MaxConf* es valor máximo de confianza obtenido de entre todas las reglas de asociación descubiertas. Esta heurística, parecida al criterio utilizado por las funciones de evaluación lexicográficas de los algoritmos STAR (página 37), nos sirve para considerar equivalentes aquellas reglas cuyo grado de cumplimiento es similar, estableciendo un "margen de tolerancia" dado por el intervalo [*MaxConf* - Δ, *MaxConf*]. En

concreto, se pueden obtener buenos resultados si se emplea el umbral de soporte mínimo como valor para el parámetro Δ (Δ = *MinSupp*) y 100 % como valor inicial de *MaxConf* (esto es, cuando aún no se ha descubierto ninguna regla, aspiramos a conseguir un modelo de clasificación perfecto).

Al utilizar la heurística anterior, se selecciona la mejor regla posible de las descubiertas y sólo se incluyen en el conjunto de reglas buenas aquéllas que son ligeramente peores que la mejor, las reglas cuya confianza queda dentro del intervalo [*MaxConf* - Δ, *MaxConf*]. Esta táctica restrictiva evita que el algoritmo ART tenga en cuenta reglas no muy buenas a la hora de construir hojas del árbol de decisión y deje abierta la posibilidad de encontrar mejores reglas en los siguientes niveles del árbol. Además, en el peor de los caso, aunque una regla adecuada no se seleccione como buena en el nivel actual del árbol, dicha regla se acabará seleccionando si no se encuentran reglas mejores al procesar los datos restantes (los de la rama 'else' del árbol de decisión).

La experimentación realizada, que aparece recogida en la sección 3.6 de este capítulo, nos ha mostrado que la selección automática del umbral de confianza obtiene soluciones casi óptimas. En nuestra opinión, el umbral de confianza sólo debería establecerlo manualmente el usuario cuando la semántica del problema lo requiera y sea estrictamente necesario (como sucedía con el conjunto de datos MUSHROOM). La incorporación de un umbral de confianza preestablecido en el mecanismo de construcción de árboles de ART es, en esos casos, positiva. En definitiva, el usuario normalmente no lo utilizará, aunque siempre tendrá la posibilidad de ajustarlo si su problema lo necesita.

3.2. Un ejemplo detallado

En esta sección utilizaremos el sencillo conjunto de datos de la tabla 3.2 para ilustrar el funcionamiento de nuestro algoritmo. Dicho conjunto de datos podría ser una versión simplificada del conjunto de datos usado para construir un sistema de ayuda a la decisión que permitiese conceder o denegar créditos. En este pequeño conjunto de datos, X, Y y Z son los atributos predictivos mientras que C indica la clase para nuestro problema.

	X	*Y*	*Z*	*C*
Cliente	¿desempleado?	¿con fondos?	¿con inmuebles?	¿crédito?
I1	0	0	0	0
I2	0	0	1	0
I3	1	1	0	0
I4	1	0	0	0
I5	1	1	1	1
I6	1	0	1	1
I7	0	1	0	1
I8	0	1	1	1

Tabla 3.2: Un conjunto de datos sencillo

Supongamos que utilizamos la selección automática de umbral de confianza de ART y establecemos el umbral de soporte relativo al 20 % (para requerir que, en la raíz del árbol, las reglas cubran al menos dos tuplas). Por último, el parámetro *MaxSize* es igual al número de atributos predictivos por defecto (3 en este sencillo ejemplo).

- *Nivel 1: Extracción de reglas*

  Comenzamos con el conjunto de datos completo e intentamos encontrar reglas con un único item en su antecedente (esto es, $k = 1$). Se obtienen los siguientes conjuntos de reglas:

```
S1: if (Y=0) then C=0 with confidence 75%
    if (Y=1) then C=1 with confidence 75%

S2: if (Z=0) then C=0 with confidence 75%
    if (Z=1) then C=1 with confidence 75%
```

  Si hubiésemos establecido un umbral de confianza rígido, aceptaríamos o rechazaríamos cualquiera de los conjuntos anteriores en función del valor que le hubiésemos asignado al umbral. Cualquier valor por encima

del 75 % haría que ART rechazase las reglas descubiertas al no considerarlas suficientemente buenas y buscase hipótesis más complejas para clasificar los datos. Si el umbral fuese igual o inferior al 75 %, ART escogería bien `S1` o bien `S2` para ramificar el árbol y el algoritmo terminaría en este punto.

Si embargo, estamos utilizando la selección automática de umbral proporcionada por ART y aspiramos a conseguir un clasificador mejor (con una precisión del 100 % a ser posible). Comprobamos que todas las reglas están por debajo del 80 % de confianza, valor de corte que proviene de 100 % (valor deseado inicialmente) - 20 % (margen de tolerancia fijado por el soporte mínimo). Al no haber ninguna regla lo suficientemente buena, ART busca conjuntos de reglas más complejas; esto es, reglas con más atributos en sus antecedentes. Con $k = 2$ se obtienen las siguientes reglas:

```
S1: if (X=0 and Y=0) then C=0 with confidence 100%
    if (X=0 and Y=1) then C=1 with confidence 100%

S2: if (X=1 and Z=0) then C=0 with confidence 100%
    if (X=1 and Z=1) then C=1 with confidence 100%

S3: if (Y=0 and Z=0) then C=0 with confidence 100%
    if (Y=1 and Z=1) then C=1 with confidence 100%
```

- *Nivel 1: Selección de reglas*

  A partir de los conjuntos de reglas de arriba, ART puede escoger cualquiera de los tres pares de reglas anteriores al producirse un empate técnico. Tanto `S1` como `S2` o `S3` tiene un 100 % de confianza y un soporte del 50 %, cubriendo el 50 % del conjunto de entrenamiento y dejando la otra mitad para otro nivel del árbol.

- *Nivel 1: Ramificación del árbol*

  Si se selecciona el primer par de reglas, el primer nivel del árbol quedaría como se muestra a continuación:

	X	*Y*	*Z*	*C*
Cliente	¿desempleado?	¿con fondos?	¿con inmuebles?	¿crédito?
I3	1	1	0	0
I4	1	0	0	0
I5	1	1	1	1
I6	1	0	1	1

Tabla 3.3: El conjunto de datos reducido

```
X = 0 and Y = 0 : C = 0 (2)
X = 0 and Y = 1 : C = 1 (2)
else
  ...
```

Recuérdese que la cantidad entre paréntesis indica el número de ejemplos de entrenamiento cubiertos por cada hoja del árbol. Los ejemplos de entrenamiento pueden eliminarse del conjunto de datos de entrenamiento una vez que han sido cubiertos por las reglas seleccionadas para ramificar el árbol, tal como se muestra en la tabla 3.3. A continuación se construye el siguiente nivel del árbol de decisión.

- *Nivel 2: Extracción de reglas*

  ART busca de nuevo reglas simples que permitan clasificar los datos que quedan en el conjunto de entrenamiento de la tabla 3.3 y encuentra el siguiente conjunto de reglas:

```
S1: if (Z=0) then C=0 with confidence 100%
    if (Z=1) then C=1 with confidence 100%
```

- *Nivel 2: Selección de reglas*

  Al descubrir un conjunto de reglas buenas, ART no necesita formular hipótesis más complejas para construir el segundo nivel del árbol.

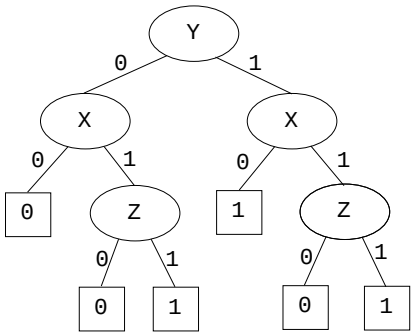

Figura 3.5: Clasificador TDIDT estándar para el conjunto de datos mostrado en la tabla 3.2

- *Nivel 2: Ramificación del árbol*

  Se ramifica el árbol y, como no quedan ejemplos del conjunto de entrenamiento por clasificar, el proceso de aprendizaje del clasificador concluye. El clasificador obtenido por ART es el siguiente:

```
X = 0 and Y = 0 : C = 0 (2)
X = 0 and Y = 1 : C = 1 (2)
else
  Z = 0 : C = 0 (2)
  Z = 1 : C = 1 (2)
```

ART ha logrado un árbol de decisión compacto mientras que cualquier otro algoritmo TDIDT estándar (tal como C4.5) construiría el árbol ligeramente más complejo que se muestra en la figura 3.5. Dicho árbol requeriría una etapa de post-procesamiento para igualar el resultado que ART consigue directamente.

En cualquier caso, en esta sección únicamente se ha presentado un sencillo ejemplo para ilustrar el funcionamiento de ART. Más adelante se discutirán los resultados experimentales obtenidos al aplicar ART en problemas reales.

3.3. Un caso real: SPLICE

Al comienzo de este capítulo se mostró, como adelanto, un ejemplo de clasificador construido por ART a partir del conjunto de datos SPLICE del UCI Machine Learning Repository (véase la página 63). El problema para cuya resolución se recopiló este conjunto de datos consiste en determinar el tipo de uniones de empalme de genes en secuencias de ADN (exon/intron, intron/exon, o ninguna de las dos) dada una secuencia de ADN de primates de 60 nucleótidos en torno a una posible unión de empalme. La siguiente descripción del problema es una traducción directa de la documentación que se puede encontrar en el UCI Machine Learning Repository:

> Las uniones de empalme son puntos en una secuencia de ADN en los cuales el ADN 'superfluo' se elimina durante el proceso de creación de proteínas en los organismos superiores. El problema planteado en este conjunto de datos consiste en reconocer, dada una secuencia de ADN, las fronteras entre exones (las partes de la secuencia de ADN que se conservan tras el empalme) y los intrones (las partes de la secuencia de ADN que quedan fuera del empalme). Este problema consiste en dos subproblemas: reconocer fronteras exon/intron (EI) y reconocer fronteras intron/exon (IE). En Biología, a las fronteras IE se les hace referencia como "receptoras" mientras que se alude a las fronteras EI como "donantes".

En este problema particular, el algoritmo ART construye un clasificador con una precisión bastante buena y un tamaño manejable. Otros algoritmos TDIDT, tal como C4.5, obtienen una precisión similar (entre el 85 y el 90 por ciento utilizando validación cruzada), si bien los árboles de decisión que construyen tienen siempre muchos más nodos y la complejidad adicional de esos árboles de decisión los hace más difíciles de comprender para el usuario final. Utilizando este conjunto de datos, el árbol de decisión construido por C4.5 posee cuatro veces más nodos que el generado por ART.

Si bien es cierto que C4.5 obtiene un porcentaje de clasificación ligeramente superior, ART consigue un modelo de clasificación más sencillo e inte-

	ART	*C4.5*
Precisión del clasificador	87.09 % ± 2.76 %	89.42 % ± 2.02 %
- Clase 1	76.64 %	77.82 %
- Clase 2	90.33 %	89.80 %
- Clase 3	90.41 %	94.60 %
Complejidad del árbol de decisión		
- Hojas	34.1 ± 3.7	143.1 ± 4.3
- Nodos internos	18.2 ± 3.4	62.0 ± 2.3
- Profundidad media	3.81 ± 0.67	3.13 ± 0.03
Tiempo de entrenamiento (en segundos)	84.81 ± 20.4	19.3 ± 0.7

Tabla 3.4: Resultados obtenidos por ART frente a C4.5 utilizando validación cruzada con 10 particiones del conjunto de datos SPLICE.

ligible para el usuario final. A pesar de que el número medio de nodos internos que un ejemplo del conjunto de entrenamiento ha de visitar antes de llegar a un nodo hoja (valor que aparece como profundidad media en la tabla 3.4) es ligeramente mayor en ART porque ART construye árboles de decisión nada balanceados, el número total de nodos internos, y también de hojas, es muy inferior en nuestro modelo al del árbol obtenido con C4.5.

En cuanto al tiempo requerido para construir el clasificador, C4.5 es más eficiente que ART porque no busca relaciones 'complejas' entre los atributos predictivos y el atributo que indica la clase. ART, sin embargo, explota las simetrías existentes en torno a las uniones (que se hallan entre los nucleótidos P30 y P31) para obtener un modelo más compacto Obsérvense, por ejemplo, las parejas P29-P31, P28-P32 y P25-P35 que se utilizan para ramificar el árbol de decisión. Los algoritmos TDIDT típicos no pueden utilizar esas relaciones y, por tanto, su aplicabilidad queda limitada en problemas para los cuales las interconexiones entre variables desempeñan un papel importante.

La tabla 3.4 compara los resultados obtenidos por ART y C4.5 sobre este conjunto de datos utilizando validación cruzada. En la sección 3.6 se pueden encontrar resultados adicionales relativos al conjunto de datos SPLICE, así como un estudio experimental más detallado del comportamiento de ART frente a otros métodos de construcción de clasificadores.

3.4. Notas acerca del clasificador ART

Antes de mostrar los resultados experimentales obtenidos con ART, resulta conveniente discutir algunos aspectos relacionados con el uso de los clasificadores construidos utilizando el modelo descrito en este capítulo: el proceso seguido por ART para clasificar datos, la forma en que ART maneja valores desconocidos y la interpretación del árbol construido por ART como un conjunto de reglas.

3.4.1. Clasificación con ART

El árbol de decisión que se obtiene con ART puede utilizarse para clasificar ejemplos no etiquetados como cualquier otro árbol. Comenzando en la raíz del árbol, un ejemplo dado sigue distintas ramas del árbol dependiendo de los valores de sus atributos. Cualquier ejemplo acabará alcanzando una hoja y será etiquetado con la clase más común en esa hoja.

Por tanto, cuando se nos presenta un caso concreto, se compara ese caso con cada una de las reglas empleadas para construir el nivel actual del árbol de decisión. Si el caso satisface el antecedente de alguna de ellas, entonces lo etiquetamos con la clase que aparece en el consecuente de la regla. Si, por el contrario, el caso no satisface ninguno de los antecedentes de las reglas del nivel actual, seguimos explorando el árbol por la rama 'else'.

Podría darse el caso, incluso, de que un ejemplo no corresponda a ninguna de las hojas del árbol ART porque no verifique el antecedente de ninguna de las reglas del último nivel del árbol. Esta situación puede aparecer cuando las reglas de asociación seleccionadas para ramificar el árbol cubren todos los ejemplos del conjunto de entrenamiento pero no todos los valores posibles de los atributos que aparecen en sus antecedentes. En tal escenario, el ejemplo se etiqueta utilizando la clase más común en el subárbol actual (más concretamente, la clase más común en el conjunto de entrenamiento que dio lugar a dicho subárbol).

El pseudocódigo correspondiente a los pasos seguidos por el método de clasificación ART para clasificar un ejemplo aparece en la figura 3.6.

función clasificar (art, ejemplo): clase;
// *Entrada: art = Árbol de clasificación ART*
// *ejemplo = Ejemplo no etiquetado*
// *Salida: clase = Clase asignada*

Emparejar el/los valor(es) de los atributos del ejemplo con
el/los valor(es) de los atributos utilizados para ramificar el árbol

si el/los valor(es) corresponde(n) a cualquiera
de las ramas que conducen a una hoja
devuelve la clase que etiqueta la hoja correspondiente
en otro caso, si existe una rama 'else' // *seguirla*
devuelve clasificar (art.else, ejemplo);
si no
devuelve la clase por defecto;

Figura 3.6: Clasificación de ejemplos no etiquetados

3.4.2. Manejo de valores nulos

Al construir el clasificador ART, los valores nulos pueden enviarse directamente a la rama 'else', ya que no están cubiertos por ninguna de las reglas de asociación descubiertas. Cuando lo que queremos es clasificar un caso para el cual no está definido un atributo de los usados en el nivel actual del árbol o dicho atributo tiene un valor desconocido, ART aplica la misma técnica que durante la etapa de construcción del árbol: al no poder emparejar el caso con el antecedente de ninguna de las reglas del nivel actual del árbol, se pasa dicho caso a la rama 'else' para buscar reglas aplicables al caso en los niveles inferiores del árbol.

No obstante, hay que mencionar que existen otras alternativas a la hora de clasificar datos que incluyan valores desconocidos:

- El algoritmo C4.5, por ejemplo, aplica una estrategia diferente a la de ART. Si nos encontramos en un nodo interno del árbol cuyo test incluye un atributo para el cual desconocemos su valor, se clasifica el caso actual utilizando todos y cada uno de los subárboles que cuelgan del nodo

interno. C4.5 construye una distribución de probabilidad de acuerdo a la frecuencia relativa de cada clase en los resultados provenientes de la clasificación del caso en cada uno de los subárboles del nodo. De hecho, al carecer de ramas 'else' en las que agrupar los datos que no se sabe muy bien a qué subárbol asignar, C4.5 se ve obligado a 'fraccionar' los casos utilizando pesos proporcionales a la frecuencia de cada rama del árbol (tanto al entrenar el clasificador como al utilizarlo para clasificar). De esta forma, un caso puede corresponder a múltiples caminos en el árbol, lo que dificulta la interpretación del modelo de clasificación cuando aparecen valores desconocidos.

- CART, por su parte, emplea una tercera estrategia que consiste en seleccionar tests alternativos cuya partición del conjunto de entrenamiento sea similar a la del test realizado en un nodo interno del árbol. Si el test original no es aplicable a un caso concreto, CART emplea el test alternativo (el cual, a su vez, puede tener otro test alternativo). Con este método las distintas ramas de un árbol siguen siendo mutuamente exclusivas, si bien puede ser difícil encontrar tests alternativos adecuados. Por desgracia, esta estrategia de búsqueda de tests alternativos es difícilmente aplicable en la construcción de árboles n-arios como los construidos por ART y C4.5. De hecho, en CART es viable porque este algoritmo se restringe a la construcción de árboles de decisión binarios.

La estrategia empleada en ART permite tratar adecuadamente la presencia de valores nulos (desconocidos o inaplicables), sin tener que añadir mecanismos adicionales (como en el caso de C4.5) ni restringir la topología del árbol (tal como sucede en CART), todo ello gracias a la utilización de ramas 'else' en el árbol de decisión.

3.4.3. Conversión del árbol en reglas

La popularidad de los árboles de decisión se debe, en gran medida, a que se pueden derivar reglas IF-THEN a partir de ellos de una forma trivial. Los árboles de decisión construidos por ART, de hecho, se pueden convertir en reglas empleando el mismo método que cualquier otro algoritmo TDIDT. Sólo

hay que tener en cuenta una salvedad: la utilización de ramas ‘else’ causa la aparición de negaciones en los antecedentes de las reglas.

Consideremos de nuevo el ejemplo de la sección 3.2. A partir del árbol obtenido en dicha sección, se puede obtener el siguiente conjunto de reglas:

```
if X=0 and Y=0 then C=0 (2)
if X=0 and Y=1 then C=1 (2)

if not(X=0) and Z=0 then C=0 (2)
if not(X=0) and Z=1 then C=1 (2)
```

Las reglas del conjunto anterior son independientes y mutuamente excluyentes, lo que permite su utilización inmediata en cualquier sistema basado en el conocimiento; esto es, el significado de cada regla es absoluto y no depende de su contexto.

De forma alternativa, se puede interpretar el modelo de clasificación construido por ART como una lista de decisión. En este caso, el árbol quedaría traducido a la siguiente lista ordenada de reglas:

```
1. if X=0 and Y=0 then C=0 (2)

2. if X=0 and Y=1 then C=1 (2)

3. if Z=0 then C=0 (2)

4. if Z=1 then C=1 (2)
```

Las reglas anteriores forman una lista de decisión en la cual el significado de cada regla individual depende de su posición dentro de la secuencia de reglas. Al final de esta secuencia se suele añadir una clase por defecto (para evitar la posibilidad de que un caso no verifique ninguna de las reglas).

La conversión del árbol ART en una lista de decisión nos permite reducir la complejidad de la formulación de las reglas (eliminando las negaciones correspondientes a las ramas ‘else’) a costa de empeorar la inteligibilidad del modelo de clasificación obtenido (pues las reglas hay que interpretarlas en su contexto, lo cual puede resultar difícil si la lista de reglas es larga).

En cualquier caso, ha de destacarse una diferencia de ART con respecto a los algoritmos tradicionales de inducción de listas de decisión: en vez de añadir reglas de una en una, ART añade directamente conjuntos de reglas en cada iteración (los conjuntos correspondientes a cada nivel del árbol de decisión). Este hecho permite que la implementación de ART sea más eficiente que la de otros algoritmos de inducción de listas de decisión.

3.5. Propiedades del clasificador ART

Esta sección complementa a la anterior tratando en profundidad varias propiedades clave del algoritmo ART como método de construcción de modelos de clasificación.

3.5.1. Estrategia de búsqueda

ART construye árboles muy poco balanceados porque intenta clasificar directamente tantos ejemplos como sea posible en cada nivel del árbol de decisión. No obstante, esta característica de ART también ayuda, al menos en parte, a tomar mejores decisiones locales al construir el árbol de decisión.

ART realiza una búsqueda exhaustiva de reglas potencialmente interesantes en el conjunto de datos de entrenamiento, aunque esa búsqueda se restringe al conjunto de datos que queda por clasificar conforme se profundiza en el árbol. De esta forma, la eficiencia, que caracteriza la estrategia de búsqueda greedy heurística de otros algoritmos TDIDT, se combina con la potencia ofrecida por la búsqueda exhaustiva realizada por el algoritmo de extracción de reglas de asociación.

En cierto sentido, ART sigue la metodología STAR de Michalski y sus colaboradores [140, capítulo 3] (igual que CN2 [35] [34]), ya que genera reglas iterativamente hasta que se construye un modelo de clasificación completo. Es digno de mención que, no obstante, ART elimina del conjunto de datos tanto los ejemplos positivos como los ejemplos negativos cubiertos por las reglas seleccionadas para ramificar el árbol, mientras que otros algoritmos STAR han de conservar todos los ejemplos negativos para construir nuevas hipótesis. Además, este hecho conduce a la obtención de reglas más complejas.

Aunque se ha escogido un algoritmo greedy para construir los clasificadores ART, también son factibles otros enfoques. Por ejemplo, una estrategia de búsqueda dirigida podría encontrar mejores clasificadores. Ese tipo de estrategia de búsqueda se utiliza en algoritmos como AQ o CN2, entre otros, los cuales siguen la filosofía de la metodología STAR. No obstante, la eficiencia es estrictamente necesaria en problemas de Data Mining y los algoritmos greedy constituyen la mejor alternativa en estos casos.

3.5.2. Robustez (ruido y claves primarias)

El uso de técnicas propias de la extracción de reglas de asociación ayuda a construir clasificadores más robustos al minimizar los efectos del ruido en los datos de entrada.

El umbral de soporte mínimo hace que sea completamente inofensiva la presencia de datos erróneos aislados (*outliers*), ya que no se tienen en cuenta a la hora de decidir cómo construir el clasificador. La aparición de este tipo de errores, usualmente debidos a problemas durante la adquisición de datos (p.ej. errores en la toma de medidas), es algo bastante común en cualquier ámbito y afecta negativamente a los algoritmos de aprendizaje supervisado. ART, no obstante, funciona adecuadamente incluso en presencia de ruido.

Por otra parte, el umbral de soporte también elimina los problemas que aparecen en otros algoritmos TDIDT cuando algunos atributos predictivos son casi claves (v.g. ID3 siempre escoge ese tipo de atributos para ramificar el árbol porque minimizan la entropía en los subárboles resultantes). Este problema se elimina fácilmente en ART sin necesidad de introducir mecanismos artificiales adicionales (como el criterio de proporción de ganancia en C4.5).

En definitiva, ART trata de una manera uniforme tanto a los outliers (que no contribuyen significativamente en las reglas de asociación) como a las claves primarias o candidatas (cuyos valores no tienen soporte suficiente para generar reglas de asociación por sí mismos).

El árbol construido por ART, a diferencia de los construidos por otros algoritmos TDIDT, no requiere poda alguna a posteriori porque los outliers no degradan su rendimiento. De hecho, los algoritmos de extracción de reglas de asociación funcionan perfectamente con datos que incluyen ruido.

3.5.3. Complejidad del árbol

Tanto la altura como la anchura del árbol de decisión construido por ART están restringidas por el umbral de soporte mínimo.

Dado un umbral absoluto de soporte mínimo *MinSupp* como un valor normalizado entre 0 y 1, el árbol de decisión construido por ART tendrá como máximo *1/MinSupp* niveles, porque en cada nivel se clasificarán al menos *#D*MinSupp* ejemplos, siendo *#D* el número de ejemplos en el conjunto de datos de entrenamiento.

Cuando se utiliza un umbral absoluto de soporte mínimo, nunca serán necesarias más de *MaxSize*(1/MinSupp)* pasadas sobre el conjunto de datos de entrada para construir el clasificador completo; ya que todas las reglas de asociación pueden obtenerse con, como mucho, *MaxSize* recorridos sobre el conjunto de datos de entrenamiento utilizando algoritmos como Apriori. En otras palabras, ART es *O(n)* sobre el tamaño del conjunto de datos de entrenamiento. Esta cualidad es esencial para realizar tareas de Data Mining.

El factor de ramificación del árbol también lo determina el umbral de soporte mínimo. *1/MinSupp* es una cota superior del factor de ramificación del árbol en cada nivel, independientemente de si el umbral de soporte *MinSupp* es absoluto o relativo, porque no puede haber más de *1/MinSupp* reglas de asociación seleccionadas simultáneamente. El caso extremo ocurre cuando todos los ejemplos del conjunto de entrenamiento se clasifican en el mismo nivel del árbol utilizando la máxima cantidad de reglas de asociación, todas ellas con el soporte mínimo permitido.

3.6. Resultados experimentales

Se ha implementado ART en el lenguaje de programación Java 2 utilizando el kit de desarrollo de Sun Microsystems. Durante los experimentos también se empleó AspectJ [90], una extensión “orientada a aspectos” del lenguaje Java. El uso de aspectos permite monitorizar la ejecución de un algoritmo, lo que resulta especialmente útil para tomar algunas medidas (en concreto, las relacionadas con la realización de operaciones de entrada/salida).

Para acceder a los datos, que se pueden almacenar en cualquier base de datos relacional, se emplea JDBC [Java Database Connectivity], el interfaz estándar de acceso a bases de datos de Java.

Para la implementación de ART se ha utilizado el algoritmo TBAR [19], un algoritmo de extracción de reglas de asociación de la familia de Apriori [7]. TBAR será objeto de estudio en el siguiente capítulo de esta memoria.

Los experimentos se realizaron en un PC con microprocesador Intel Pentium III a 800MHz con 128 MB de memoria RAM sobre el sistema operativo Windows NT 4.0 WorkStation de Microsoft. La base de datos a la que se accedía era InterBase 6, aunque cualquier otra base de datos podría haber sido utilizada dada la portabilidad de JDBC. De hecho, también se ha probado ART con datos almacenados en Oracle 8i e IBM DB2 UDB.

La tabla 3.5 recoge los conjuntos de datos utilizados en nuestros experimentos. Dichos conjuntos de datos se pueden conseguir gratuitamente accediendo al servidor web de la Universidad de California en Irvine. En concreto, se obtuvieron del UCI Machine Learning Repository, al cual se puede acceder a través de la siguiente URL:

http://www.ics.uci.edu/∼mlearn/MLRepository.html.

Todos los resultados presentados en esta sección se han obtenido utilizando validación cruzada con 10 particiones sobre cada uno de los conjuntos de datos utilizados.

3.6.1. Precisión

La tabla 3.6 compara el rendimiento de ART frente a varios clasificadores ampliamente utilizados. En esta tabla, que recoge los porcentajes de clasificación obtenidos por distintos clasificadores, se agrupan los conjuntos de datos en función de su tamaño. Además, en la tabla se compara ART con cada uno de los demás clasificadores utilizados en los experimentos[*].

[*] Las ternas G-P-E indican el número de veces en que el porcentaje de clasificación obtenido con ART es mejor (G), el número de ocasiones en que ART es peor (P) y el número de conjuntos de datos en que la precisión de ART es similar (E). El comportamiento de dos clasificadores se considera similar cuando la diferencia entre ambos no llega al 1 %.

Conjunto de datos	*Tuplas*	*Atributos*	*Clases*
AUDIOLOGY	226	70	24
CAR	1728	7	4
CHESS	3196	36	2
HAYES-ROTH	160	5	3
LENSES	24	6	3
LUNG CANCER	32	57	3
MUSHROOM	8124	23	2
NURSERY	12960	9	5
SOYBEAN	683	36	19
SPLICE	3175	61	3
TICTACTOE	958	10	2
TITANIC	2201	4	2
VOTE	435	17	2

Tabla 3.5: Conjuntos de datos utilizados en los experimentos.

En estos experimentos se utilizó ART con selección automática del umbral de confianza (tal como se describe en la sección 3.1.3), un umbral de soporte del 5 % para las reglas de asociación (*MinSupp=5 %*) y una cota superior para el tamaño de los antecedentes de las reglas igual a 3 (*MaxSize=3*).

También se muestran en la tabla los resultados conseguidos por C4.5 [131], utilizando el criterio de proporción de ganancia como regla de división y poda pesimista (con CF=0.25). Las siguientes tres columnas incluyen resultados obtenidos con algoritmos basados en la metodología STAR de Michalski y sus colaboradores: AQR (una variante del conocido algoritmo AQ) y dos versiones del algoritmo CN2. La primera de ellas, CN2-STAR [34], obtiene un conjunto de reglas (como AQR), mientras que la segunda versión, CN2-DL [35], obtiene una lista de decisión, como los algoritmos IREP [64] y RIPPER [38] que aparecen a continuación. La tabla de resultados se completa con el clasificador Naive Bayes y el clasificador por defecto que resulta de asignar siempre la clase más común en el conjunto de entrenamiento, el cual se incluye a título meramente informativo.

Tamaño	*Conjunto de datos*	*ART*	*C4.5*	*AQR*	*CN2 (STAR)*	*CN2 (DL)*	*IREP*	*RIPPER k=2*	*Naive Bayes*	*Clasificador por defecto*
>1000	NURSERY	99.11 %	96.17 %	91.70 %	98.10 %	98.99 %	89.40 %	33.33 %	88.05 %	33.33 %
	MUSHROOM	98.52 %	100.00 %	100.00 %	100.00 %	100.00 %	100.00 %	99.95 %	94.31 %	51.80 %
	SPLICE	89.32 %	94.08 %	79.21 %	92.32 %	90.21 %	90.99 %	67.21 %	51.91 %	51.91 %
	CHESS	97.69 %	99.22 %	97.06 %	99.37 %	99.27 %	99.22 %	98.59 %	62.48 %	52.22 %
	TITANIC	78.56 %	79.05 %	67.70 %	75.78 %	79.05 %	78.33 %	70.42 %	68.01 %	67.70 %
	CAR	98.55 %	92.88 %	85.07 %	93.92 %	95.78 %	74.89 %	70.02 %	70.02 %	70.02 %
	Media	93.63 %	93.57 %	86.79 %	93.25 %	93.88 %	89.47 %	73.25 %	72.46 %	54.50 %
	Resultados		2-3-1	4-1-1	3-3-0	1-2-3	2-3-1	4-1-1	6-0-0	6-0-0
<1000	TICTACTOE	81.62 %	83.82 %	82.26 %	98.02 %	92.37 %	97.60 %	93.64 %	65.34 %	65.34 %
	SOYBEAN	91.52 %	93.70 %	83.17 %	92.09 %	93.41 %	88.15 %	14.35 %	58.71 %	13.18 %
	VOTE	95.86 %	95.86 %	93.32 %	94.71 %	92.19 %	94.02 %	91.71 %	89.19 %	61.36 %
	AUDIOLOGY	65.21 %	81.36 %	66.36 %	75.57 %	80.06 %	65.37 %	25.14 %	23.40 %	25.14 %
	HAYES-ROTH	84.38 %	73.75 %	65.00 %	76.25 %	76.25 %	68.75 %	48.75 %	61.88 %	29.38 %
	LUNG CANCER	40.83 %	43.33 %	25.83 %	39.17 %	35.83 %	35.00 %	31.67 %	43.33 %	40.00 %
	LENSES	70.00 %	81.67 %	65.00 %	76.67 %	73.33 %	56.67 %	63.33 %	63.33 %	63.33 %
	Media	75.63 %	79.07 %	68.71 %	78.93 %	77.63 %	72.22 %	52.66 %	57.88 %	42.53 %
	Resultados		1-5-1	5-1-1	3-3-1	3-4-0	5-1-1	6-1-0	6-1-0	6-0-1
Media		83.94 %	85.76 %	77.05 %	85.54 %	85.13 %	80.18 %	62.16 %	64.61 %	48.05 %
Mejor-Peor-Igual (1 %)			3-8-2	9-2-2	6-6-1	4-6-3	7-4-2	10-2-1	12-1-0	12-0-1

Tabla 3.6: Precisión del clasificador ART y de otros conocidos métodos de clasificación.

Aunque no aparecen recogidas en la tabla 3.6, también se realizaron pruebas con otros clasificadores TDIDT (utilizando distintas reglas de división) y algunas variantes del clasificador Naive Bayes (usando las leyes de Laplace o de Lidstone para estimar probabilidades). Dichos experimentos obtuvieron resultados similares a los ya mostrados, por lo cual se omiten en la tabla 3.6.

En los experimentos mostrados en la tabla 3.6 se utilizó una heurística de selección automática del umbral de confianza *MinConf* porque se ha observado experimentalmente que dicha heurística obtiene resultados cercanos al mejor resultado obtenido cuando se establece dicho umbral realizando una batería de pruebas (tal como se verá en la sección 3.6.4). De hecho, cuando se emplea un umbral de confianza preestablecido, aumentar su valor no implica necesariamente mejorar la precisión del clasificador.

Tal como se recoge en la tabla 3.6 y se muestra en la figura 3.7 de la página 97, la precisión del clasificador ART es comparable a la que obtienen otros clasificadores, especialmente en conjuntos de datos de cierto tamaño (como NURSERY o CAR). Sin embargo, el rendimiento de ART empeora ligeramente cuando los conjuntos de datos son pequeños (AUDIOLOGY o LENSES) debido a la propia naturaleza del proceso de extracción de reglas de asociación, el cual está basado en la obtención eficiente de patrones frecuentes en grandes conjuntos de datos.

En líneas generales, no obstante, ART mantiene los porcentajes de clasificación obtenidos por otros clasificadores como C4.5 o CN2, mejorando significativamente los resultados conseguidos con otros métodos (AQR, RIPPER, Naive Bayes y clasificador por defecto), como se recoge en la tabla 3.7.

La tabla 3.7 recoge los resultados obtenidos al realizar tests estadísticos para comparar la precisión obtenida por ART con los resultados logrados por los demás clasificadores analizados. Dado que el test t de Student, habitualmente empleado para comparar clasificadores, exhibe una probabilidad elevada de error de tipo I (esto es, detecta diferencias significativas no existentes en realidad) y asume que las diferencias existentes están normalmente distribuidas (algo que no podemos asegurar en nuestros experimentos), la tabla 3.7 incluye también los resultados obtenidos al realizar un test de Wilcoxon sobre los resultados recogidos en la tabla 3.6. Dicho test no presupone la normalidad de la

	Test t de Student	*Test de Wilcoxon*
ART vs. C4.5	$p \leq 0{,}3507$	$p \leq 0{,}3475$
ART vs. AQR	$p \leq 0{,}0032$ ++	$p \leq 0{,}0100$ ++
ART vs. CN2-STAR	$p \leq 0{,}3901$	$p \leq 0{,}5405$
ART vs. CN2-DL	$p \leq 0{,}3942$	$p \leq 0{,}5860$
ART vs. IREP	$p \leq 0{,}1645$	$p \leq 0{,}1548$
ART vs. RIPPER	$p \leq 0{,}0128$ +	$p \leq 0{,}0100$ ++
ART vs. Naive Bayes	$p \leq 0{,}0004$ ++	$p \leq 0{,}0100$ ++
ART vs. por defecto	$p \leq 0{,}0001$ ++	$p \leq 0{,}0100$ ++

Tabla 3.7: Tests estadísticos realizados para comprobar si las diferencias apreciadas en la precisión de los distintos clasificadores son significativas (+, $p \leq 0{,}05$) o estadísticamente significativas (++, $p \leq 0{,}01$).

población (técnica no paramétrica).

Como es lógico, ART es significativamente mejor que el clasificador por defecto. Además, ART es consistentemente mejor que AQR y Naive Bayes. Así mismo, ART mejora los resultados obtenidos por RIPPER porque este último no se comporta adecuadamente con algunos de los conjuntos de datos utilizados en los experimentos. Respecto a los demás métodos, ART consigue resultados similares en cuanto a la precisión de los modelos de clasificación construidos, si bien a continuación se verá cómo el modelo de clasificación ART propuesto en esta memoria es mejor que otros modelos en aspectos tales como la eficiencia del proceso de construcción del clasificador (apartado 3.6.2) o la complejidad de los modelos de clasificación construidos (apartado 3.6.3).

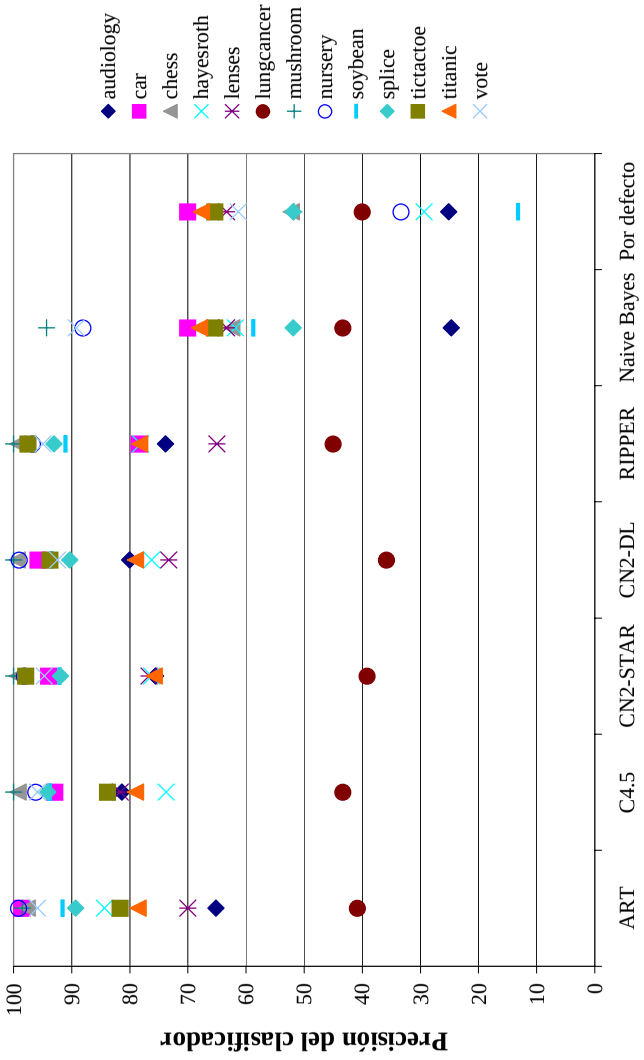

Figura 3.7: Precisión del clasificador ART frente a otros clasificadores comunes.

3.6.2. Eficiencia

La tabla 3.8 muestra el tiempo empleado por cada uno de los métodos de construcción de clasificadores utilizados en el apartado anterior. La tabla recoge los tiempos de entrenamiento para algunos de los conjuntos de datos de la tabla 3.5. En concreto, se muestran los resultados obtenidos con los seis conjuntos de datos de la tabla 3.5 que contienen más de mil tuplas.

Con el objetivo de hacer más notorias las diferencias existentes entre los distintos algoritmos de construcción de clasificadores, también se ha realizado una serie de experimentos en los cuales los clasificadores se construyen trabajando directamente sobre ficheros (sin almacenar los datos en memoria principal), de forma que se fuerza un acceso a disco cada vez que se accede al conjunto de datos de entrenamiento y se refleja de una forma más fidedigna el comportamiento de los distintos algoritmos en la resolución de problemas de *Data Mining* cuando los conjuntos de datos son demasiado grandes para caber en memoria principal. Los resultados obtenidos de este modo se resumen en la tabla 3.9.

De hecho, cuando los conjuntos de datos no caben en memoria principal, los algoritmos de inducción de reglas necesitan utilizar alguna técnica de muestreo para seguir siendo competitivos con ART y con la implementación de C4.5 que se ha utilizado en esta memoria.

La figura 3.8 muestra gráficamente los tiempos de entrenamiento correspondientes a todos los conjuntos de datos de la tabla 3.5. El clasificador Naive Bayes se incluye en esta figura para que sirva de referencia a los demás métodos de clasificación. Dicho clasificador se construye recorriendo secuencialmente el conjunto de datos una sola vez, por lo que su eficiencia se puede considerar óptima.

Nuestra implementación de C4.5, similar a RainForest [69], resulta especialmente adecuada para trabajar con grandes conjuntos de datos. Dicha implementación sólo requiere recorrer el conjunto de datos de entrenamiento dos veces en cada nodo interno del árbol y una sola vez en sus nodos hoja. El primer recorrido sirve para obtener la frecuencia de cada valor de los distintos atributos de los casos del conjunto de entrenamiento y la frecuencia de cada

Algoritmo	*NURSERY*	*MUSHROOM*	*SPLICE*	*CHESS*	*TITANIC*	*CAR*
C4.5	0.8s	0.8s	0.9s	0.9s	0.1s	0.2s
CN2-DL	15.9s	1.7s	23.7s	1.4s	0.2s	0.3s
IREP	61s	4.2s	19.7s	2.6s	0.2s	0.5s
AQR	63s	1.8s	64s	3.3s	0.1s	1.1s
ART	6.2s	1.6s	188s	6.8s	1.1s	4.8s
RIPPER	236s	7.0s	39.5s	4.6s	0.3s	1.5s
CN2-STAR	310s	8.0s	217s	14.1s	0.2s	4.2s

Tabla 3.8: Tiempo de entrenamiento requerido por distintos algoritmos de construcción de clasificadores.

Algoritmo	*NURSERY*	*MUSHROOM*	*SPLICE*	*CHESS*	*TITANIC*	*CAR*
C4.5	17s	4s	9s	7s	1.2s	4s
ART	101s	13s	605s	57s	4.7s	13s
RIPPER	45s	719s	3062s	819s	6.8s	9s
IREP	5634s	415s	4389s	505s	12.0s	101s
CN2-DL	1743s	129s	4710s	155s	12.7s	51s
AQR	5906s	112s	12297s	403s	0.6s	223s
CN2-STAR	29552s	836s	29257s	4528s	21.5s	1023s

Tabla 3.9: Tiempo de entrenamiento requerido por los distintos clasificadores cuando se trabaja directamente sobre disco.

clase para cada uno de esos valores (esto es, los conjuntos atributo-valor-clase utilizados en RainForest). Ésta es la única información que requiere el algoritmo TDIDT para ramificar el árbol de decisión. Cuando se decide no seguir ramificando el árbol (llegamos a un nodo hoja) no es necesario acceder más al conjunto de datos de entrenamiento, ya que simplemente se etiqueta la hoja con la clase más común en los datos de entrenamiento. Si, por el contrario, hay que ramificar el árbol de decisión, es necesario recorrer de nuevo el conjunto de entrenamiento para distribuir los casos de entrenamiento entre los distintos subárboles generados.

En cuanto al clasificador ART, hay que resaltar que también es muy eficiente cuando se trabaja sobre grandes conjuntos de datos, ya que ART sólo requiere recorrer, en el peor de los casos, *MaxSize*+1 veces el conjunto de datos de entrenamiento para decidir qué reglas se utilizan en la ramificación de cada nivel del árbol. El número de veces que se recorren secuencialmente los datos de entrenamiento viene dado por la implementación del proceso de extracción de reglas de asociación empleado para obtener hipótesis candidatas. Este proceso será estudiado con detalle en el capítulo siguiente de esta memoria.

Cuando el volumen de datos del conjunto de entrenamiento es elevado, la estrategia de búsqueda de ART es mucho más adecuada que la empleada por algoritmos anteriores de inducción de reglas para resolver problemas de extracción de conocimiento en bases de datos. Estos algoritmos, por lo general, suelen requerir un recorrido completo del conjunto de datos cada vez que se formula una hipótesis. Aunque la tabla 3.8 no refleja este hecho en toda su magnitud porque los conjuntos de datos utilizados caben perfectamente en memoria principal, hay que tener en cuenta que algoritmos como CN2 o RIPPER realizan un recorrido secuencial del conjunto de entrenamiento cada vez que se evalúa una regla para ver si resulta adecuado añadirla al modelo de clasificación correspondiente. Por su parte, ART realiza un máximo de *MaxSize*+1 recorridos sobre el conjunto de entrenamiento, durante los cuales obtiene en paralelo todas las reglas candidatas que serán consideradas para formar parte del modelo de clasificación.

Cuando el conjunto de datos de entrenamiento no se puede cargar en memoria principal, por su tamaño o por restricciones de espacio en un servidor

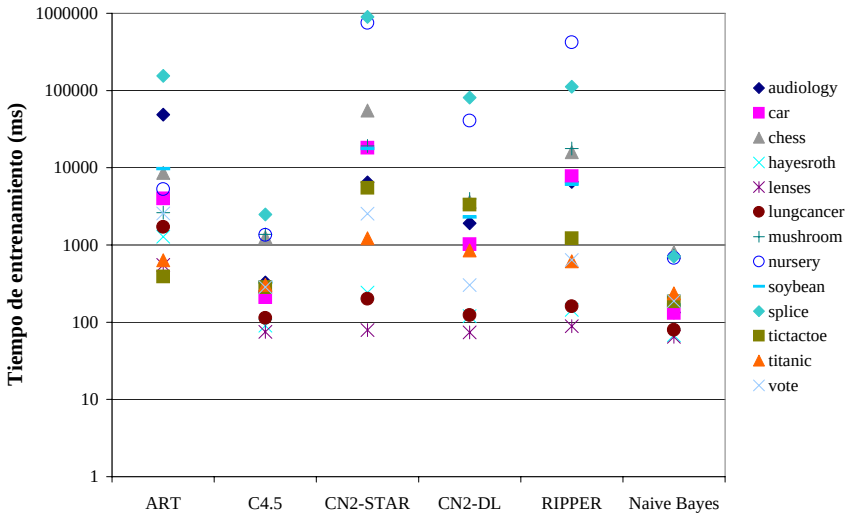

Figura 3.8: Tiempo de entrenamiento requerido para la construcción de los distintos clasificadores.

que ha de dar servicio a muchos clientes diferentes, ART puede llegar a ser uno o varios órdenes de magnitud más eficiente que los algoritmos AQR, CN2, IREP y RIPPER. En tales situaciones, estos últimos requieren la utilización de alguna técnica de muestreo porque, a diferencia de ART, no son escalables.

Las diferencias existentes entre las distintas técnicas se deben principalmente al ancho de banda requerido por las operaciones de E/S, el verdadero cuello de botella de los algoritmos de aprendizaje en KDD. En la figura 3.9 aparece el número de operaciones de entrada/salida necesarias para la construcción de cada tipo de clasificador[**].

[**]En realidad, las operaciones de E/S son únicamente de entrada, pues consisten exclusivamente en recorridos secuenciales del conjunto de entrenamiento. La gráfica de la parte superior muestra el número de veces que se recorre secuencialmente el conjunto de entrenamiento (o una parte de él cuando avanzamos en la construcción del modelo de clasificación). En la gráfica de debajo aparece el número de veces que se accede a una tupla del conjunto de entrenamiento.

Las dos gráficas que aparecen en la figura 3.9 ilustran claramente a lo que nos referimos cuando decimos que el algoritmo ART destaca por su escalabilidad.

3.6.3. Complejidad

Si bien el tiempo de entrenamiento requerido por ART es superior al necesario para construir un árbol de decisión utilizando un algoritmo TDIDT clásico (al realizar ART una búsqueda en un espacio se soluciones mayor), los modelos de clasificación obtenidos con ART tienden a ser más compactos que los frondosos árboles de decisión construidos por algoritmos como C4.5, incluso después de podarlos. La búsqueda adicional que realiza ART es precisamente la que le permite obtener modelos de clasificación más simples, que son potencialmente más fáciles de entender para el usuario.

La complejidad de los clasificadores obtenidos utilizando los distintos métodos de clasificación vistos se muestra en la figura 3.10. En esta gráfica aparece representado, en escala logarítmica, el número de reglas de cada clasificador como medida de su complejidad. En el caso de los árboles de decisión construidos por C4.5, se muestra su número hojas, ya que de cada hoja de un árbol de decisión se deriva una regla (como se vio en el apartado 2.1.5).

Los clasificadores más complejos son los generados por los algoritmos de inducción de reglas basados en la metodología STAR de Michalski; es decir, AQR y CN2-STAR. En el extremo opuesto se hallan los algoritmos de inducción de listas de decisión, CN2-DL y RIPPER, si bien hay que tener en cuenta que el significado de las reglas en una lista de decisión depende de su posición, lo que puede dificultar la interpretación del modelo construido. En el caso de los algoritmos TDIDT de construcción de árboles de decisión (C4.5), su complejidad en número de hojas se halla a medio camino entre las listas de decisión y los algoritmos STAR.

Tal como se aprecia en la figura 3.10, el algoritmo ART, a medio camino entre las listas y los árboles de decisión, destaca por conseguir modelos de clasificación de complejidad comparable o incluso menor que la de cualquier otro de los métodos analizados.

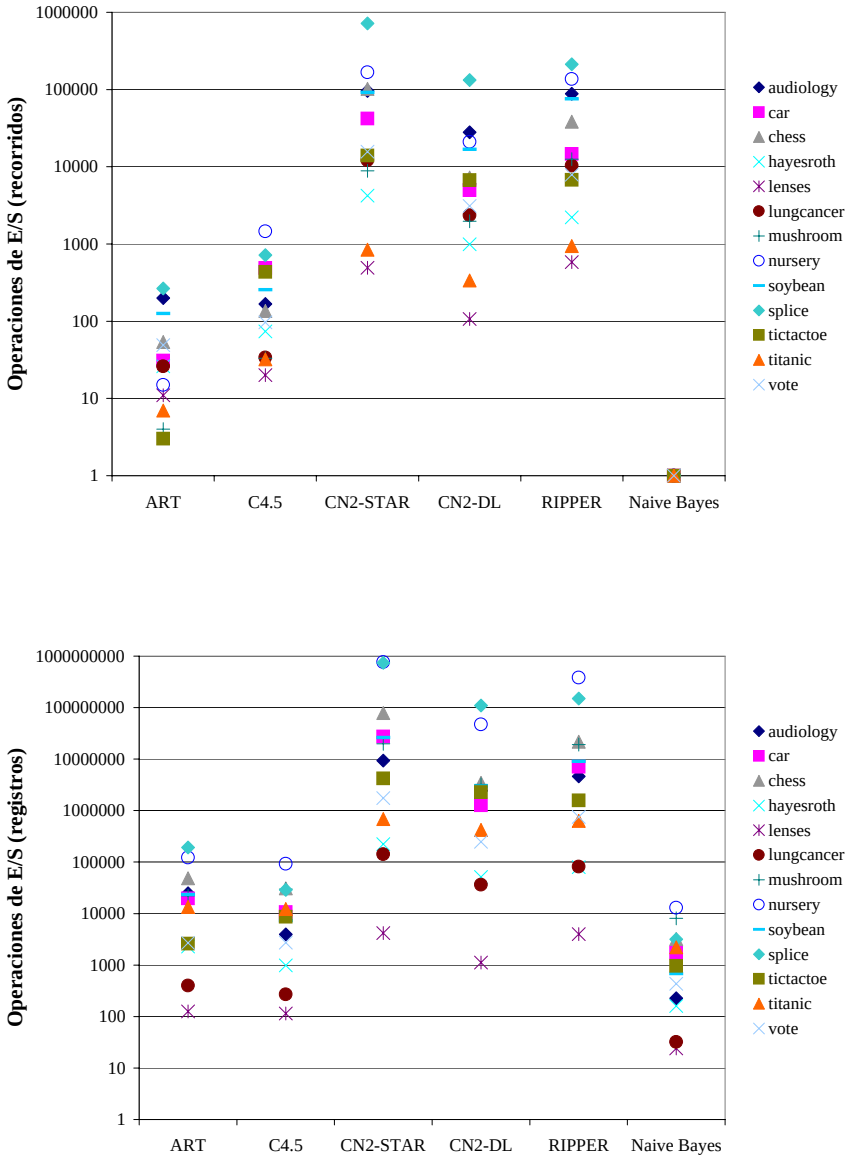

Figura 3.9: Operaciones de E/S requeridas para la construcción de los distintos clasificadores.

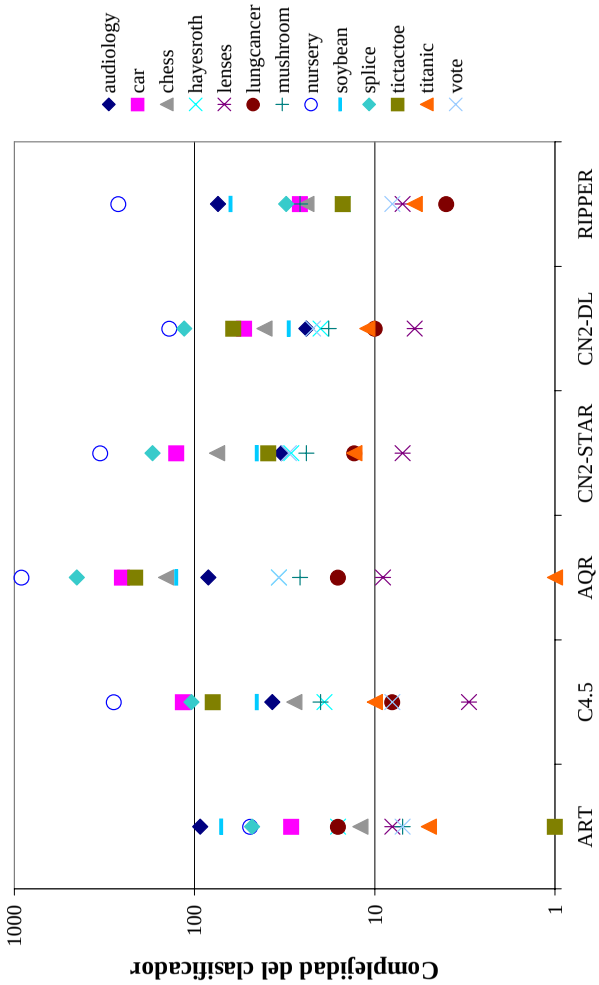

Figura 3.10: Complejidad media de los distintos clasificadores.

3.6.4. El umbral de confianza

En este apartado se resumen los resultados obtenidos en una serie de experimentos realizados con la intención de justificar el uso de una heurística que permita seleccionar automáticamente el umbral de confianza involucrado en el proceso de extracción de reglas de asociación utilizado por ART para construir hipótesis candidatas. Al emplear esta heurística, el usuario no ha de preocuparse en establecer el valor de este parámetro, salvo que lo estime oportuno para resolver su problema de clasificación adecuadamente (tal como se discutió en el apartado 3.1.3).

La tabla 3.10 resume los resultados obtenidos por ART sobre algunos de los conjuntos de datos empleados en los experimentos de las secciones anteriores. En la tabla se muestra el porcentaje de clasificación obtenido para cada conjunto de datos variando los umbrales de soporte y confianza utilizados para obtener las reglas de asociación que sirven como base del modelo de clasificación ART.

Dicha tabla muestra, además, cómo incrementar el umbral de confianza mínimo no implica necesariamente la obtención de un mejor porcentaje de clasificación. Si bien es cierto que este umbral puede utilizarse para establecer una precisión mínima deseada para las reglas extraídas del conjunto de entrenamiento, esta característica no es aplicable al conjunto de prueba con el que se mide la precisión del clasificador.

Los resultados obtenidos con el conjunto de datos NURSERY muestran que las variaciones en los parámetros de ART no siempre modifican los resultados obtenidos. Al trabajar con este conjunto de datos, llega un momento en el cual la precisión del clasificador queda estancada en el 66.25 %, cuando se logran clasificar perfectamente dos de las cinco clases del problema. Las otras tres ni tan siquiera aparecen en los modelos de clasificación construidos por ART, ya que no son lo suficientemente frecuentes como para aparecer en el consecuente de ninguna regla de asociación.

Los resultados obtenidos con la heurística de selección automática del umbral, utilizada en los experimentos mostrados en la tabla 3.6, suelen encontrarse cercanos al mejor valor obtenido realizando una serie completa de experi-

MinSupp	*MinConf*	*VOTE*	*SOYBEAN*	*NURSERY*
0.0	*0.7*	95.62 %	81.56 %	86.93 %
	0.8	95.39 %	84.78 %	**91.04 %**
	0.9	95.61 %	90.20 %	73.10 %
0.1	*0.7*	95.62 %	85.07 %	66.25 %
	0.8	95.39 %	85.37 %	66.25 %
	0.9	95.61 %	90.64 %	66.25 %
0.2	*0.7*	95.62 %	**91.37 %**	66.25 %
	0.8	95.62 %	**91.37 %**	66.25 %
	0.9	**95.85 %**	**91.37 %**	66.25 %
0.3	*0.7*	88.50 %	57.07 %	66.25 %
	0.8	88.74 %	57.07 %	66.25 %
	0.9	88.74 %	57.07 %	66.25 %

Tabla 3.10: Resultados obtenidos con ART variando los umbrales de soporte y confianza. Cuando aparece *0.0* como umbral de soporte mínimo, este umbral indica que se tendrán en cuenta, al construir el árbol de decisión, todas las reglas de asociación posibles (incluso aunque su soporte se limite a una única tupla en el conjunto de entrenamiento).

mentos con distintos valores para el umbral de confianza, lo que confirma que la heurística utilizada de selección automática del umbral resulta adecuada. En la práctica, resulta incluso preferible de cara al usuario, pues al utilizar dicho mecanismo automático se obtienen buenos resultados y se ahorra bastante tiempo (el tiempo necesario para ajustar los parámetros del clasificador[***]).

3.6.5. El umbral de soporte

Respecto al umbral de soporte mínimo, la tabla 3.10 también nos muestra que disminuir el valor del umbral de confianza no implica que ART consiga mejores porcentajes de clasificación, tal como indican los resultados obtenidos con los conjuntos de datos VOTE y SOYBEAN.

[***] Esta experimentación puede convertirse una ardua tarea no sólo para ART, sino para cualquier otro modelo de clasificación.

MinSupp	1 %	2.5 %	5 %	7.5 %	10 %	20 %
Precisión (10-CV)	77.18 %	79.04 %	79.22 %	80.86 %	81.08 %	80.92 %
Tiempo de entrenamiento	17.9s	21.8s	18.5s	13.9s	8.0s	5.6s
Topología del árbol						
- Hojas del árbol	36.8	44.1	37.0	35.9	27.8	19.6
- Nodos internos	19.0	20.4	18.2	17.5	14.2	11.2
- Profundidad media	8.08	7.88	7.41	6.88	6.28	5.46
Operaciones de E/S						
- Registros	38900	41400	36400	34800	28700	18200
- Recorridos	65.7	70.9	62.8	59.8	47.4	35.4

Tabla 3.11: Resumen de los experimentos realizados para distintos valores del umbral de soporte.

El parámetro *MinSupp* nos ayuda a establecer el nivel de granularidad que mejor se adapte a nuestro problema y, además, permite acotar el tiempo necesario para construir el clasificador, como se puede deducir de los resultados obtenidos empíricamente que se resumen en la tabla 3.11. En la figura 3.11 se muestra gráficamente la evolución de la precisión del clasificador ART conforme se varía el umbral de soporte mínimo *MinSupp*, mientras que la figura 3.12 recoge la complejidad de los árboles construidos y los gráficos de la figura 3.13 hacen referencia al coste del proceso de construcción del clasificador.

Un valor bajo del umbral de soporte mínimo tiende a favorecer el sobreaprendizaje y la construcción de clasificadores más complejos de lo necesario. Además, cuanto más bajo sea el umbral de soporte mínimo, la construcción del clasificador será más costosa en términos computacionales. Un umbral excesivamente elevado, en cambio, puede conducir a la construcción de un modelo de clasificación cuya precisión no alcance niveles aceptables al no considerarse lo suficientemente frecuentes algunos patrones potencialmente interesantes, especialmente si se utiliza la heurística de selección automática del umbral de confianza descrita en la sección 3.1.3.

Es cierto, no obstante, que, además de restringir el tiempo necesario para la construcción del clasificador, el umbral de soporte mínimo *MinSupp* nos ayuda a establecer el nivel deseado de granularidad de nuestro clasificador.

Para concluir esta sección, sólo falta mencionar el papel desempeñado por el tercer parámetro de ART, *MaxSize*, que también influye en el tiempo de entrenamiento necesario para construir el clasificador. Si utilizásemos *MaxSize=1* conseguiríamos un clasificador ART tan eficiente como cualquier otro clasificador TDIDT, si bien eliminaríamos de ART la capacidad de emplear combinaciones de atributos para ramificar el árbol de decisión. Por otro lado, cuando el valor *MaxSize* es mayor que 3, el rendimiento de ART podría deteriorarse ya que el número de patrones frecuentes presentes en un conjunto de datos puede aumentar exponencialmente con el tamaño de los patrones. Este hecho, especialmente preocupante en conjuntos de datos densos como los utilizados habitualmente para construir clasificadores, dio razón de ser al algoritmo TBAR, que se describirá detalladamente en el capítulo siguiente de esta memoria.

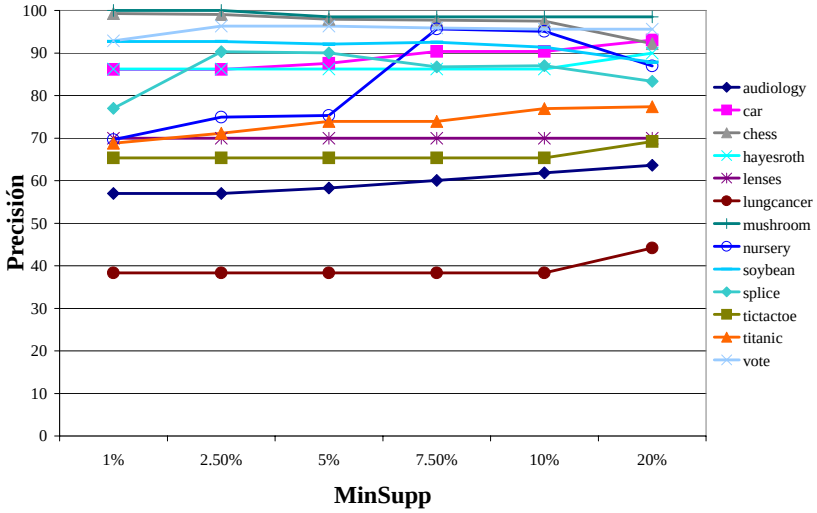

Figura 3.11: Variación de la precisión del clasificador ART en función del umbral de soporte mínimo.

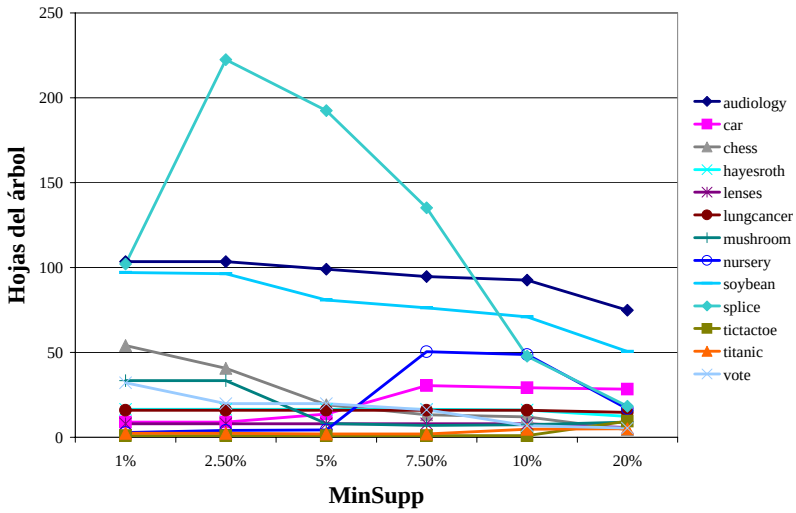

Figura 3.12: Variación del número de hojas del árbol ART en función del umbral de soporte mínimo.

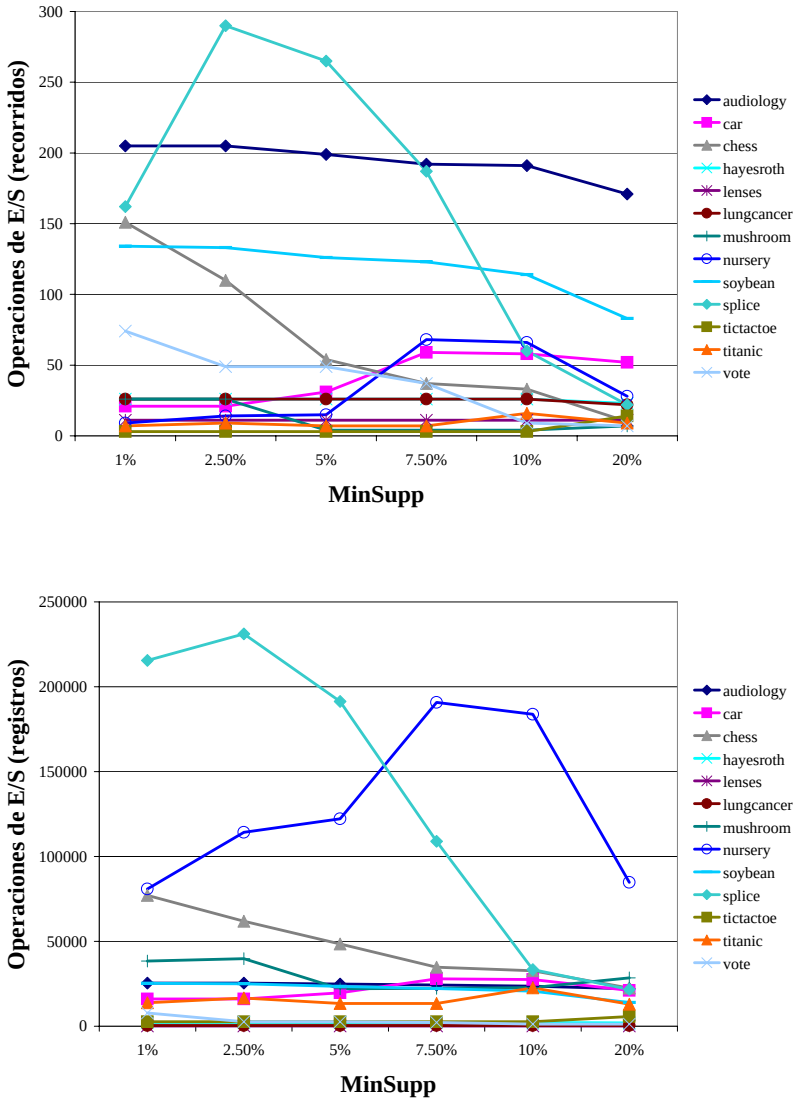

Figura 3.13: Variación del número de operaciones de E/S realizadas en la construcción de ART en función del umbral de soporte mínimo.

En resumen, las restricciones impuestas por los parámetros *MaxSupp* y *MaxSize* sobre el proceso de búsqueda realizado por ART permiten que éste pueda utilizarse eficientemente con grandes conjuntos de datos. Ambos parámetros acotan el espacio de búsqueda explorado por ART, haciéndolo competitivo con otros algoritmos de inducción de reglas. Resulta recomendable, pues, llegar a un compromiso a través del valor de estos umbrales entre el espacio de búsqueda explorado y el tiempo de entrenamiento requerido para construir el modelo de clasificación.

3.6.6. Otros experimentos

Aparte del mecanismo de selección automática del umbral de confianza y del método de selección de reglas descritos en las secciones anteriores de este capítulo, también se han realizado otros experimentos en los que se evalúan algunas heurísticas alternativas.

Entre las opciones analizadas relativas al ajuste automático de los parámetros empleados por ART, se realizaron algunas pruebas en las que se utilizaba un umbral de confianza adaptativo independiente para cada clase del problema. Esta técnica se ideó para intentar evitar que ART ignorase clases muy poco frecuentes en el conjunto de entrenamiento, aunque no se consiguieron mejoras notables a nivel global. Obviamente, si las clases son verdaderamente poco frecuentes, la precisión del clasificador no se verá afectada.

Tampoco se consiguieron mejoras destacables al utilizar métodos heurísticos más complejos para evaluar las reglas candidatas durante el proceso de construcción del clasificador (recuérdese el proceso de selección de reglas realizado a la hora de ramificar el árbol de decisión, sección 3.1.3). Una alternativa posible consiste, por ejemplo, en medir la divergencia Hellinger [98] asociada a las reglas candidatas en vez de tomar simplemente el número de ejemplos del conjunto de datos de entrada correctamente clasificados.

A pesar de no haberse conseguido resultados positivos hasta ahora con heurísticas alternativas, la búsqueda de heurísticas que mejoren la versión actual de ART sigue siendo una línea de investigación abierta que puede proporcionar resultados prometedores en el futuro.

3.6.7. Comentarios finales

Como se ha visto en las secciones anteriores, ART destaca por conseguir un buen compromiso entre la precisión de los clasificadores que construye, el tiempo necesario para construirlos y la complejidad de los modelos generados:

- ART consigue obtener clasificadores con una precisión aceptable, similar a la obtenida con métodos más complejos como C4.5 o RIPPER.
- ART es altamente escalable, característica que lo hace idóneo en la resolución de problemas de *Data Mining*. Aun siendo ligeramente más ineficiente que las versiones más recientes de algoritmos TDIDT (como la implementación de C4.5 similar a RainForest utilizada en los experimentos), es mucho más eficiente que otros algoritmos de inducción de reglas y listas de decisión.
- ART no sólo es más eficiente que dichos algoritmos, sino que, además, consigue modelos de clasificación tan compactos como los obtenidos por otros métodos, modelos que son especialmente simples si se comparan con los árboles de decisión obtenidos por C4.5.
- ART permite tratar con facilidad la presencia de valores desconocidos en los datos al utilizar ramas 'else', sin tener que recurrir a mecanismos complejos ni restringir la topología del árbol de decisión.
- ART realiza un proceso de búsqueda, guiado por los parámetros *MinSupp* y *MaxSize*, gracias al cual es capaz de aprovechar las relaciones existentes entre distintos atributos y ramificar el árbol utilizando los valores de varios atributos simultáneamente.
- ART, finalmente, dispone de un mecanismo automático de selección del umbral de confianza *MinConf* que simplifica su utilización por parte de usuarios no expertos sin suponer perjuicio alguno para las características del clasificador obtenido.