Capítulo 5

Manejo de atributos continuos

No todo lo que puede contarse cuenta, y no todo lo que cuenta puede contarse

ALBERT EINSTEIN

En conjuntos de datos reales, no todos los datos con los que nos encontramos son de tipo nominal o categórico, sino que también suele haber datos de tipo numérico. Costes, precios de venta, tamaños o pesos son algunos ejemplos de cantidades que pueden aparecer como atributos numéricos en cualquier base de datos. Es más, en el ámbito de las bases multidimensionales y de las aplicaciones OLAP, todas las medidas con las que se trabaja son cantidades numéricas que resumen la información de una empresa respecto a una serie de dimensiones de tipo categórico, las cuales se pueden tratar a distintos niveles de granularidad definiendo jerarquías de conceptos.

Resulta necesario, por tanto, disponer de métodos que nos permitan trabajar con atributos continuos a la hora de construir un modelo de clasificación como el propuesto en el capítulo 3. Las técnicas de discretización comentadas en la sección 5.1 nos permitirán ampliar el ámbito de aplicación de ART y de otros muchos algoritmos de aprendizaje. La sección 5.2 presenta un método alternativo de discretización que puede ser aplicado de una forma eficiente para construir árboles de decisión n-arios con atributos continuos. El método

propuesto se añade así al repertorio de técnicas de discretización empleadas usualmente para construir árboles de decisión, que son objeto de estudio en la sección 5.3. Finalmente, en el apartado 5.4, se describen algunos resultados obtenidos experimentalmente al emplear distintas técnicas de discretización para construir árboles de clasificación con un algoritmo clásico como C4.5 y con el modelo de clasificación ART.

5.1. La discretización de espacios continuos

Cuando trabajamos con modelos simbólicos como ART y nos encontramos con patrones definidos sobre un espacio continuo (esto es, ejemplos de entrenamiento cuyos atributos toman valores de tipo numérico), no nos queda más remedio que agrupar los datos de los que disponemos de la forma que resulte más útil para su procesamiento posterior.

5.1.1. El caso general: Métodos de agrupamiento

El *agrupamiento* de un conjunto de patrones mediante la utilización de alguna métrica de similitud que resulte adecuada es un problema bastante estudiado en Inteligencia Artificial, también conocido como aprendizaje no supervisado en contraste con el aprendizaje supervisado realizado al construir modelos de clasificación a partir de ejemplos previamente etiquetados.

En su caso general, los métodos de agrupamiento intentan, dado un conjunto de patrones definidos sobre un espacio multidimensional, agrupar dichos patrones en un pequeño número de agrupamientos, de forma que cada agrupamiento contenga únicamente aquellos patrones que sean similares en cierta medida. Este proceso puede verse también como una clasificación no supervisada de los patrones del conjunto de entrenamiento: el proceso de generar automáticamente clases que no están predefinidas. Tanto en el capítulo 8 del libro de Han y Kamber [75] como en los apuntes de clase de Ullman [153] se pueden encontrar excelentes introducciones a este tema.

Igual que los algoritmos TDIDT objeto de esta memoria, los métodos de agrupamiento son métodos heurísticos, si bien los primeros suelen emplear

medidas de pureza en sus reglas de división (sección 2.1.1) mientras los segundos utilizan medidas de similitud como las descritas en el anexo 5.5 para decidir cómo se agrupan los datos.

En su incansable búsqueda de mejores métodos de agrupamiento, los investigadores en el tema han propuesto numerosos algoritmos. Muchos de estos algoritmos son iterativos y su grado de complejidad varía desde la sencillez de métodos de agrupamiento como el algoritmo de las K Medias hasta algoritmos altamente parametrizados como ISODATA, acrónimo de *Iterative Self-Organizing Data Analysis Techniques* con la A final añadida para hacer pronunciable su nombre. Algoritmos iterativos de este tipo aparecen descritos con detalle en cualquier libro de texto de Reconocimiento de Formas [54] [151] y suelen caracterizarse porque los agrupamientos obtenidos dependen del orden de presentación de los patrones del conjunto de entrenamiento. Esta dependencia se puede disminuir si se emplean estrategias modernas de búsqueda como GRASP [45], acrónimo de *Greedy Randomized Adaptive Search Procedure*. También existen métodos de agrupamiento basados en modelos teóricos, como las técnicas basadas grafos, si bien suelen resultar poco prácticos para resolver problemas reales por motivos de eficiencia computacional (cualquier técnica de complejidad superior a $O(n \log n)$ es de poca utilidad en la resolución de problemas de *Data Mining*).

El gran número de métodos heurísticos de agrupamiento existentes puede clasificarse, a grandes rasgos, en dos grandes grupos: los métodos basados en centroides y los métodos jerárquicos de agrupamiento.

- Los métodos de agrupamiento basados en centroides utilizan prototipos (puntos centrales o centroides) para caracterizar los agrupamientos y asignan cada patrón al agrupamiento cuyo centroide está más cercano. El conocido algoritmo de las K Medias pertenece a esta categoría y es uno de los más sencillos.

- Los métodos de agrupamiento jerárquico son incrementales y pueden ser aglomerativos o divisivos. Aunque en español quizá sería más correcto denominarlos aglomerantes y divisores, aquí se emplea la terminología usual en Reconocimiento de Formas:

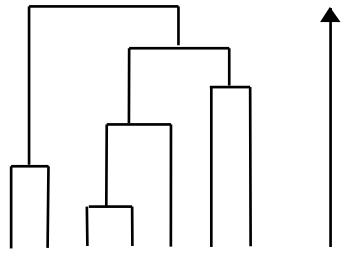

Figura 5.1: Dendrograma que ilustra el funcionamiento de un método de agrupamiento jerárquico.

- Los métodos jerárquicos aglomerativos son métodos de tipo ascendente. Partiendo de un agrupamiento para cada patrón del conjunto de entrenamiento, van combinando los agrupamientos más cercanos hasta que se cumpla algún criterio de parada. El proceso de agrupamiento, cuando se realiza en una única dimensión, se puede dibujar mediante un dendrograma como el de la figura 5.1, donde la flecha indica la evolución del método de agrupamiento aglomerativo.

- A diferencia de los métodos aglomerativos, los métodos jerárquicos divisivos son de tipo descendente. Comenzando con un único agrupamiento que contiene todos los patrones del conjunto de entrenamiento, se va dividiendo este agrupamiento hasta que se verifique el criterio de parada del algoritmo. En este caso, el dendrograma de la figura 5.1 se construiría de arriba hacia abajo.

Los métodos de agrupamiento jerárquico se han utilizando ampliamente en distintos ámbitos. En [133], por ejemplo, puede encontrarse una descripción detallada del uso de este tipo de algoritmos en la resolución de problemas de recuperación de información.

5.1.2. El caso unidimensional: Métodos de discretización

Dividir los valores de un atributo continuo en un conjunto de intervalos adyacentes corresponde al caso unidimensional de los métodos de agrupamiento. Este caso, conocido como **discretización**, es de especial importancia en Inteligencia Artificial, pues permite que muchos algoritmos de aprendizaje ideados para funcionar con atributos nominales o categóricos puedan también utilizarse con conjuntos de datos que incluyen valores numéricos [85], algo esencial en la resolución de problemas reales.

La discretización de los valores no sólo permite construir modelos de clasificación más compactos y sencillos, que resultan más fáciles de comprender, comparar, utilizar y explicar, sino que permite mejorar la precisión del clasificador y hace más rápido el aprendizaje [51]. En otras palabras, "la discretización amplía las fronteras de muchos algoritmos de aprendizaje" [85].

En problemas de clasificación, la discretización suele realizarse para cada atributo continuo por separado por cuestiones de eficiencia. Al considerar cada atributo numérico por separado, sólo hay que agrupar los patrones unidimensionales definidos por los valores del atributo continuo. Cualquier método de agrupamiento de los mencionados en el apartado 5.1.1 puede emplearse para resolver este problema, incluso los métodos de agrupamiento basados en grafos. En el caso unidimensional, la implementación de estos métodos es bastante sencilla y además resulta óptima desde un punto de vista teórico, ya que se maximiza la distancia entre agrupamientos (v.g. construyendo un árbol generador minimal [20]). Sin embargo, estos modelos clásicos de aprendizaje no supervisado no tienen en cuenta el contexto en el que se aplican (el problema de clasificación), algo que sí hacen otros métodos existentes, como veremos a continuación.

5.1.2.1. Clasificación

Los métodos de discretización empleados como herramienta auxiliar para resolver problemas de aprendizaje supervisado pueden clasificarse atendiendo a varios criterios:

- **Métodos de discretización supervisada vs. Métodos de discretización no supervisada** (dependiendo de si se utiliza o no información sobre las clases para agrupar los datos).

  Los métodos supervisados emplean información sobre la clase a la que pertenece cada caso del conjunto de entrenamiento a la hora de evaluar y escoger los puntos de corte que definen los intervalos en que quedan agrupados los valores numéricos; los métodos no supervisados no tienen en cuenta esta información.

  Los métodos de discretización no supervisada utilizan la distribución de valores de un atributo continuo como única fuente de información, algo que no parece ser adecuado a la hora de agrupar valores numéricos en intervalos adyacentes si tenemos más información disponible, como sucede al intentar construir un modelo de clasificación. En cambio, las técnicas de discretización supervisada, como el algoritmo propuesto en la sección 5.2, sí tienen en cuenta la información adicional de la que disponemos acerca de las clases cuando intentamos resolver un problema de clasificación.

- **Métodos de discretización local (dinámicos) vs. Métodos de discretización global (estáticos)**.

  En un método global, la discretización se realiza a priori. Los métodos locales, en cambio, discretizan los atributos continuos en una región localizada del espacio (p.ej. la correspondiente a un nodo de un árbol de decisión) y, por lo tanto, la discretización que realizan de un atributo dado puede no ser única para todo el espacio definido por los patrones del conjunto de entrenamiento.

  Hay que tener en cuenta, no obstante, que la distinción entre métodos locales y métodos globales de discretización es una cuestión de uso independiente del algoritmo de discretización concreto que se utilice.

5.1.2.2. Algoritmos de discretización

Los métodos de discretización más simples crean un número preestablecido de intervalos. *Equiwidth* y *Equidepth* [75, p.110] son ejemplos no supervisados de este tipo: *Equiwidth* crea intervalos de la misma amplitud, mientras que *Equidepth* define intervalos de la misma frecuencia en el conjunto de entrenamiento. El discretizador 1R de Holte [83] es una variante supervisada de este tipo de métodos, que ajusta los límites de los intervalos en función de la clase a la que pertenezcan los casos del conjunto de entrenamiento. Existen incluso otros métodos, como el no supervisado basado en la regla 3-4-5 [75, p.135], que intentan conseguir intervalos que nos resulten más fáciles de leer e interpretar (como el intervalo $[1000, 2000]$), ya que los intervalos generados por otros métodos suelen resultar poco intuitivos ($[978, 2163]$, por ejemplo).

La Teoría de la Información también ha originado la aparición de algunas técnicas de discretización, como el uso de la ganancia de información o del criterio de proporción de ganancia en la construcción de árboles de decisión binarios con atributos continuos (sección 5.3.1.1). El método de discretización supervisada de Fayyad e Irani [58], basado en el principio MDL de Rissanen, es un ejemplo destacado de este tipo de métodos. La distancia de Màntaras [105] puede emplearse como alternativa a la propuesta de Fayyad e Irani, al igual que la medida de contraste presentada por Van de Merckt [154].

Zeta [82] pertenece a otro tipo de técnicas de discretización. Zeta mide el grado de asociación entre los valores nominales de dos atributos categóricos: el atributo continuo discretizado y la clase en el problema de clasificación. Zeta selecciona los intervalos que consiguen la máxima precisión del clasificador que resulta de intentar predecir la clase en función única y exclusivamente del atributo discretizado. El estadístico χ^2 también puede utilizarse en métodos de discretización alternativos (como ChiMerge, Chi2 o ConMerge).

En el informe [85] puede encontrarse un amplio repaso de los métodos citados así como una taxonomía que permite categorizarlos. La referencia [51] también puede ser un buen punto de partida para todo aquél que esté interesado en el tema.

5.2. Discretización contextual: Un enfoque alternativo

En la sección anterior se presentaron algunas nociones de aprendizaje no supervisado y se comentaron varios métodos clásicos de discretización. En ésta se introduce una técnica que nos permite discretizar los valores de un atributo continuo atendiendo a la similitud existente entre las distribuciones de las clases para cada valor o conjunto de valores adyacentes del atributo continuo que se desea discretizar. En secciones posteriores se verá cómo puede emplearse este método de discretización para construir árboles de clasificación n-arios con atributos continuos.

El método de discretización propuesto en esta sección es un método de agrupamiento jerárquico y un método de discretización supervisada atendiendo a los criterios de clasificación expuestos en la sección anterior de esta memoria.

5.2.1. La discretización contextual como método de discretización supervisada

Al construir un árbol de decisión, un método no supervisado no tiene en cuenta si la partición generada del conjunto de entrenamiento conduce a un árbol de decisión mejor o no. Es decir, no considera el contexto en el que se ha de realizar la discretización. Por lo tanto, desde un punto de vista conceptual, el uso exclusivo de la distribución de valores de un atributo continuo a la hora de discretizarlo no resulta del todo adecuado para resolver problemas de clasificación. Sin embargo, éste es el enfoque utilizado por muchos de los métodos de discretización: los métodos no supervisados (Equidepth, Equiwidth, 3-4-5, k-Means...).

A pesar de que los métodos de discretización existentes se suelen basar en la utilización de alguna medida de pureza asociada a cada intervalo resultante del proceso de discretización, el método de discretización contextual mide la similitud entre valores adyacentes para seleccionar los puntos de corte que determinan los intervalos en que se discretiza un atributo continuo. La primera estrategia es la usual en la construcción de árboles de decisión con algoritmos TDIDT, mientras que la opción escogida es la empleada por la mayor parte de métodos de agrupamiento estándar (apartado 5.1.1).

A la hora de realizar la discretización, obviamente, sólo tienen significado los intervalos resultantes de agrupar valores adyacentes en el sentido tradicional, aquellos valores más cercanos entre sí utilizando cualquier métrica de distancia (ya sea la euclídea, la de Manhattan o la de Mahalanobis), pues en el caso unidimensional se mantiene la relación de adyacencia entre los distintos valores numéricos independientemente de la métrica empleada. En realidad, no será necesario calcular ninguna distancia si ordenamos los valores del atributo continuo que aparecen en el conjunto de entrenamiento de forma que los valores adyacentes numéricamente estén en posiciones consecutivas.

Si tenemos en cuenta lo anterior, nos basta medir la similitud entre pares de valores adyacentes para decidir cómo han de agruparse los valores de un atributo continuo. La similitud existente entre pares de valores adyacentes se puede medir utilizando cualquiera de los modelos descritos en el apartado 5.5, teniendo en cuenta que la discretización contextual emplea la distribución de las clases para caracterizar cada valor del atributo (como método de discretización supervisada que es) en vez de utilizar directamente los valores del atributo, como harían los métodos de agrupamiento tradicionales (no supervisados).

Combinando las ideas expuestas en los dos párrafos anteriores se consigue un algoritmo de discretización dependiente del contexto (supervisado) que utiliza las técnicas habituales de los métodos de agrupamiento (no supervisados).

De hecho, el método de discretización contextual puede verse como un algoritmo de agrupamiento estándar si se redefine la noción de patrón empleada usualmente por estos métodos. En vez de considerar el patrón constituido por el vector característico de un ejemplo de entrenamiento (esto es, el vector dado por los valores del ejemplo para cada uno de los atributos de nuestro problema), se utiliza la distribución de clases para un valor de un atributo continuo como patrón que caracteriza el valor del atributo.

Con el fin de que los resultados obtenidos sean interpretables y tengan sentido, se establece un orden entre los valores del atributo como restricción adicional: dos valores x e y de un atributo, con $x < y$, estarán en un mismo agrupamiento (intervalo) sólo si todos los valores z presentes en el conjunto de entrenamiento tales que $x < z < y$ también pertenecen a dicho agrupamiento.

5.2.2. La discretización contextual como método de discretización jerárquica

El método general de discretización contextual, que primero ordena los valores de un atributo y después emplea las diferencias existentes entre las distribuciones de clases para los distintos valores del atributo, se ha de plantear como un método de agrupamiento jerárquico.

Una vez que tenemos el conjunto ordenado de valores de un atributo continuo, podemos utilizar cualquiera de los criterios descritos en la sección 5.5 para decidir qué pares de valores o intervalos adyacentes combinar (si implementamos nuestro proceso de discretización como un algoritmo de agrupamiento jerárquico aglomerativo) o cómo seleccionar el punto de corte por el cual dividir un intervalo dado (si empleamos un enfoque divisivo).

5.2.2.1. Discretización contextual aglomerativa

Los pasos correspondientes a la implementación aglomerativa del método de discretización contextual se especifican en la figura 5.2. Partiendo de un agrupamiento (intervalo en nuestro caso) para cada valor del atributo continuo que queremos discretizar, se selecciona en cada iteración el par de intervalos adyacentes cuyas distribuciones de clases sean más similares según el criterio de similitud seleccionado por el usuario, que puede ser cualquiera de los que se describen en el apartado 5.5. Los intervalos así seleccionados se funden en un único intervalo que los reemplaza.

A la hora de implementar el algoritmo, se ha de establecer de antemano un criterio de parada que se verifique cuando la discretización del atributo continuo alcance las propiedades que el usuario desee. A continuación se describen algunos de los criterios de parada más usuales:

- El criterio de parada más elemental consiste en establecer de antemano el número de agrupamientos o intervalos que deseamos generar (de forma idéntica a como funcionan algoritmos no supervisados como el de las K Medias, Equiwidth o Equidepth, e incluso algunos métodos supervisados como Zeta).

1. Crear un intervalo para cada valor del atributo continuo.
2. Identificar los dos intervalos adyacentes más similares.
3. Combinar los dos intervalos identificados en el punto anterior.
4. Mientras queden más de dos intervalos y no se verifique ningún criterio de parada establecido por el usuario, volver al paso 2.

Figura 5.2: Versión aglomerativa del método de discretización contextual.

- Algunos métodos de discretización establecen algún tipo de umbral que determina hasta dónde ha de seguir el proceso de discretización jerárquica (como los basados en el estadístico χ^2).
- También se puede emplear algún criterio de parada automático como los propuestos por los métodos que utilizan conceptos de Teoría de la Información (vg: el principio MDL de Rissanen).

Independientemente del criterio que se emplee, su idoneidad dependerá de las características del conjunto de datos concreto y de los objetivos que el usuario tenga en mente al realizar la discretización.

5.2.2.2. Discretización contextual divisiva

El algoritmo correspondiente a la implementación divisiva del método de discretización contextual se describe en la figura 5.3. En este caso, se parte de un único intervalo que abarca el dominio completo del atributo continuo en el conjunto de entrenamiento. En cada iteración, se escoge un punto de corte por el cual dividir el intervalo actual en dos subintervalos, de forma que se maximice la disimilitud existente entre las distribuciones de clases de los intervalos resultantes (utilizando cualquier criterio de los que aparecen en la sección 5.5).

Como al dividir un intervalo en dos se modifica la similitud existente entre dichos intervalos y los intervalos adyacentes al intervalo original, se pueden

1. Comenzar con un único intervalo que incluya todos los valores del atributo continuo.
2. Identificar el punto de corte que da lugar a la configuración de intervalos más disimilares entre sí.
3. Utilizar dicho punto de corte para dividir en dos uno de los intervalos de la configuración actual de intervalos.
4. Mientras haya menos intervalos que valores diferentes y no se verifique ningún criterio de parada establecido por el usuario, volver al paso 2.

Figura 5.3: Implementación divisiva del método de discretización contextual.

tener en cuenta estos intervalos al seleccionar el punto de corte óptimo.

Sean $[a, v_{min-1}]$, $[v_{min}, v_{max}]$ y $[v_{max+1}, b]$ tres intervalos consecutivos que se utilizan para discretizar los valores de un atributo continuo cualquiera. Al evaluar un punto de corte x correspondiente al intervalo $[v_{min}, v_{max}]$ para dividir éste en dos subintervalos $[v_{min}, x]$ y $(x, v_{max}]$, se ha de minimizar la siguiente función de energía:

$$\begin{aligned} \Delta Energía &= s([a, v_{min-1}], [v_{min}, x]) \\ &+ s([v_{min}, x], (x, v_{max}]) \\ &+ s([x, v_{max}], [v_{max+1}, b]) \\ &- s([a, v_{min-1}], [v_{min}, v_{max}]) \\ &- s([v_{min}, v_{max}], [v_{max+1}, b]) \end{aligned}$$

donde s representa la medida de similitud utilizada para evaluar la similitud entre intervalos adyacentes, los tres primeros sumandos corresponden a la configuración final en que quedan los intervalos y los dos últimos términos corresponden al estado inicial del conjunto de intervalos.

Podría haberse aplicado un razonamiento similar al caso anterior, cuando los intervalos se construyen empleando un método aglomerativo. Sin embar-

go, en la versión aglomerativa no es tan importante incluir el entorno de los intervalos que se funden. Dada una serie de intervalos consecutivos I_1, I_2, I_3 e I_4, cuando se decide fusionar un intervalo I_2 con su vecino I_3 empleando el método aglomerativo, ya sabemos que ni I_2 se parece demasiado a I_1 ni I_3 se asemeja a I_4 más de lo que lo hacen I_2 e I_3 (pues, en tal caso, se fusionarían esos intervalos en vez de I_2 e I_3).

Al emplear el método divisivo, no obstante, no sabemos qué relación guardan los intervalos recién creados con su entorno, por lo que se emplea la función de energía de la página anterior para no crear parejas de intervalos que difieran entre sí pero sean similares a los intervalos ya existentes a su alrededor.

5.2.2.3. Eficiencia de la discretización contextual

En la discretización contextual, cualquier modificación que se realice sobre un conjunto de intervalos será una modificación local que no afectará globalmente a todos los intervalos, al existir un orden lineal preestablecido para los intervalos en que se discretiza un atributo continuo.

En la versión aglomerativa del método, al combinar dos intervalos adyacentes en uno nuevo, sólo se ve modificada la similitud existente entre los intervalos adyacentes al intervalo recién creado y dicho intervalo. Por tanto, ya que todas las demás medidas de similitud no se ven afectadas por el cambio en la configuración de los intervalos, no es necesario recalcularlas en cada iteración.

De igual forma, en la implementación divisiva del método de discretización contextual, sólo es necesario recalcular la función de similitud para los intervalos modificados en cada iteración del algoritmo, independientemente de si se usa o no una función de energía para guiar el método de agrupamiento.

Esta interesante propiedad del método de discretización contextual implica que su implementación puede realizarse eficientemente. Dado que sólo hay que actualizar localmente el conjunto de intervalos actual, cada iteración del algoritmo puede realizarse en $O(\log N)$ pasos si mantenemos indexados los intervalos de la configuración actual, siendo N el número de valores diferentes del atributo continuo. Obviamente, como mucho se tendrán que efectuar $N-1$

iteraciones. Por tanto, la discretización contextual es de orden $O(NlogN)$, a diferencia de los métodos generales de agrupamiento jerárquico, los cuales suelen ser de orden $O(N^2)$.

5.2.3. Uso de la discretización contextual como método de discretización local

Por su carácter iterativo, cualquier método de discretización jerárquico nos ofrece la posibilidad de ir evaluando las distintas configuraciones de intervalos que se generan en cada iteración. Esta propiedad hace que el método de discretización contextual propuesto en esta memoria resulte especialmente interesante como método de discretización local durante la construcción de árboles de decisión con algoritmos TDIDT, tal como se verá en la sección 5.3.2.

Cuando se construye un árbol de decisión y se emplea el método de discretización aquí propuesto, la topología del árbol de decisión construido dependerá del conjunto de datos particular y no será necesario establecer de antemano su factor de ramificación cuando se emplean atributos continuos. Además, al ser un método de discretización eficiente ($O(NlogN)$), la complejidad algorítmica del proceso de construcción global del árbol de decisión no se verá afectada, lo que nos permite aplicar esta técnica en problemas de *Data Mining* para ampliar la flexibilidad de cualquier algoritmo TDIDT, desde C4.5 [131] hasta RainForest [69].

La construcción de árboles de decisión con atributos continuos será objeto de estudio en la sección 5.3, pero antes veremos un ejemplo de funcionamiento del método de discretización propuesto.

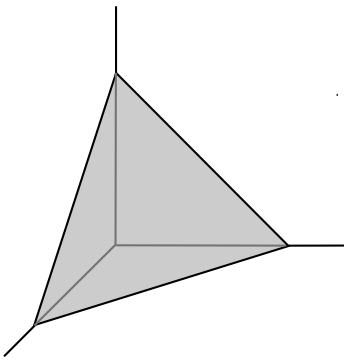

Figura 5.4: Plano que representa los valores posibles para las probabilidades de las distintas clases cuando $J = 3$.

5.2.4. Un pequeño ejemplo

Dado un conjunto de intervalos adyacentesI_1, I_2.. I_n para el atributo continuo A, caracterizamos cada uno de esos intervalos con la distribución de clases presente en los ejemplos de entrenamiento correspondientes a cada intervalo. Si existen J clases diferentes, cada intervalo I tendrá un vector característico asociado $V_I = (p_1, p_2, .., p_J)$, en donde p_j es la probabilidad de la j-ésima clase en el conjunto de ejemplos de entrenamiento cuyo valor de A está incluido en el intervalo I.

Supongamos que nuestro problema de clasificación fuese binario (esto es, $J = 2$). Para un subconjunto dado del conjunto de entrenamiento, la probabilidad de que el ejemplo pertenezca a la clase j se denota p_j. Como la suma de las probabilidades ha de ser igual a 1, en nuestro problema de clasificación binario tenemos que $p_2 = 1 - p_1$. Esto es, los valores posibles para las probabilidades en nuestro problema de decisión binario pueden representarse como una recta en $\mathbb{R}_2$.

Análogamente, los valores posibles para las probabilidades de las clases cuando tenemos tres de ellas ($J = 3$) puede representarse como el plano $p_1 + p_2 + p_3 = 1$ mostrado en la figura 5.4. Esta figura nos permite observar, de un modo gráfico, que la distancia euclídea definida sobre los vectores

característicos $(p_1, p_2, .., p_J)$ puede servir de candidata para medir la similitud entre dos distribuciones de clases distintas. En realidad, la distancia euclídea mide su disimilitud, pues la distancia disminuye cuanto más similares son las distribuciones. En la sección 5.5 se describirán otras medidas alternativas a la distancia euclídea.

Consideremos un problema de clasificación con tres clases. Sea A un atributo continuo con cuatro valores distintos presentes en el conjunto de entrenamiento (v_1, v_2, v_3 y v_4). En esta situación, cada valor v_i del atributo continuo A viene caracterizado por un vector tridimensional V_i que contiene, en su j-ésima componente, la frecuencia relativa de la clase j para los ejemplos del conjunto de entrenamiento cuyo valor de A es v_i. De modo alternativo, también se podría emplear una estimación laplaciana de estas probabilidades:

$$V_i(j) = \frac{n_j + 1}{N + J}$$

donde n_j es el número de ejemplos de la clase j correspondientes a v_i, N es el tamaño del conjunto de entrenamiento y J el número de clases en nuestro problema.

Al utilizar la versión aglomerativa del método contextual, se parte inicialmente de un intervalo para cada valor del atributo A. Supongamos que nuestros vectores característicos quedan de la siguiente manera:

$$V_1 = (0{,}3, 0{,}4, 0{,}3)$$

$$V_2 = (0{,}2, 0{,}6, 0{,}2)$$

$$V_3 = (0{,}8, 0{,}1, 0{,}1)$$

$$V_4 = (0{,}6, 0{,}4, 0{,}0)$$

Si estuviésemos empleando el método de discretización como herramienta auxiliar durante la construcción de un árbol de decisión (sección 5.3.2), evaluaríamos el árbol cuaternario resultante de utilizar cada uno de los cuatro valores para construir un subárbol.

A continuación, se combinan los dos intervalos adyacentes cuyas distribuciones de clases son más similares. Si decidimos emplear la distancia euclídea

$d_2^2(V_i, V_j)$ para medir la disimilitud existente entre vectores característicos, obtenemos lo siguiente:

$$d_2^2(V_1, V_2) = 0{,}06$$

$$d_2^2(V_2, V_3) = 0{,}62$$

$$d_2^2(V_3, V_4) = 0{,}14$$

Al ser la distancia euclídea una medida de disimilitud, (V_1, V_2) es el par de vectores más similar. Por tanto, combinamos los valores v_1 y v_2 para obtener $V_{12} = (0{,}25, 0{,}5, 0{,}25)$ suponiendo que tanto v_1 como v_2 representan al mismo número de ejemplos de entrenamiento.

Entonces podríamos evaluar la calidad de la partición actual del conjunto de valores continuos del atributo A, lo cual nos conduciría, en el contexto de un algoritmo TDIDT, a la construcción de un árbol de decisión ternario: el resultante de emplear $\{v_1, v_2\}$, $\{v_3\}$ y $\{v_4\}$ como conjuntos discretos de valores.

De nuevo, volvemos a calcular la distancia entre los intervalos adyacentes para obtener:

$$d_2^2(V_{12}, V_3) = 0{,}3475$$

$$d_2^2(V_3, V_4) = 0{,}14$$

Nótese que la segunda distancia $d_2^2(V_3, V_4)$ sigue siendo la misma que antes, por lo que no sería necesario calcularla nuevamente.

Ahora decidimos combinar $\{v_3\}$ y $\{v_4\}$ pues son los intervalos adyacentes más similares. Nos quedan, de este modo, dos agrupamientos: $\{v_1, v_2\}$ y $\{v_3, v_4\}$.

Si estuviésemos construyendo un árbol de decisión, evaluaríamos el árbol binario correspondiente y finalizaríamos la ejecución del algoritmo ya que no tiene sentido continuar combinando intervalos adyacentes.

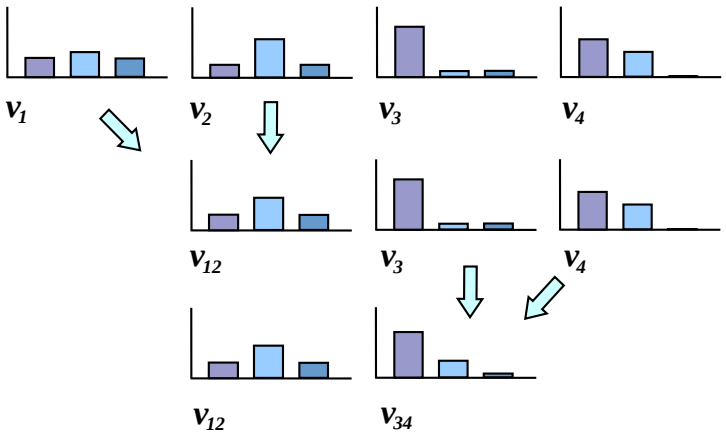

Figura 5.5: Agrupamiento contextual de intervalos adyacentes.

El proceso completo que hemos seguido para discretizar los valores del atributo continuo A aparece dibujado esquemáticamente en la figura 5.5.

Como ya se comentó en la sección anterior, hay que resaltar que sólo hay que volver a calcular dos medidas de similitud/disimilitud en cada iteración de este algoritmo. Cuando v_i se combina con v_{i+1}, hay que obtener la distribución de clases resultante $V_{i,i+1}$. Una vez que disponemos de ella, sólo tenemos que evaluar la similitud entre $V_{i,i+1}$ y sus nuevos vecinos (los intervalos adyacentes que corresponden a V_{i-1} y V_{i+2}). Las demás distribuciones de clases y medidas de similitud permanecen inalterables, por lo que no es necesario calcularlas nuevamente.

Aunque en este ejemplo se ha empleado la distancia euclídea, también se podría haber utilizado otra medida para evaluar la similitud entre las distribuciones de clases. La tabla 5.1 recoge algunas de las medidas de similitud que pueden emplearse en la discretización contextual (y, por extensión, en cualquier método de agrupamiento jerárquico tradicional). En el anexo 5.5 se puede encontrar una descripción detallada de las distintas medidas que figuran el la tabla 5.1.

Modelos basados en medidas de distancia

Distancia de Minkowski	$d_r(x,y) = \left(\sum_{j=1}^{J} \lvert x_j - y_j \rvert^r\right)^{\frac{1}{r}}, \quad r \geq 1$
- Distancia euclídea	$d_2(x,y) = \sqrt{\sum_{j=1}^{J} (x_j - y_j)^2}$
- Manhattan	$d_1(x,y) = \sum_{j=1}^{J} \lvert x_j - y_j \rvert$
- Dominio	$d_\infty(x,y) = \text{máx}_{j=1..J} \lvert x_j - y_j \rvert$
Bhattacharyya	$R(x,y) = \sqrt{1 - \sum_{j=1}^{J} \sqrt{x_j, y_j}}$

Modelos basados en medidas de correlación

Producto escalar	$S.(x,y) = x \cdot y = \sum_{j=1}^{J} x_j y_j$
Índice de correlación	$\rho(x,y) = 1 - \left(\frac{4}{N_x + N_y}\right) d_2^2$
	donde $N_v = \sum_{j=1}^{J} (2v_j - 1)^2$

Modelos basados en Teoría de Conjuntos

Modelo de Tversky	$s(a,b) = \theta f(A \cap B) - \alpha f(A - B) - \beta f(B - A)$,
	donde $\theta, \alpha, \beta \geq 0$
- Restle	$-S_{Restle}(A,B) = \lvert A \square B \rvert$
	$-S_\square(A,B) = \sup_x \mu_{A \square B}(x)$
- Intersección	$S_{MinSum}(A,B) = \lvert A \cap B \rvert$
	$-S_{Enta}(A,B) = 1 - \sup_x \mu_{A \cap B}(x)$
Modelo proporcional	$s(a,b) = \dfrac{f(A \cap B)}{f(A \cap B) + \alpha f(A - B) + \beta f(B - A)}$
	donde $\alpha, \beta \geq 0$
- Gregson	$S_{Gregson}(A,B) = \dfrac{\lvert A \cap B \rvert}{\lvert A \cup B \rvert}$

Notas:

$\lvert A \rvert = \sum_x \mu_A(x)$
$\mu_{A \cap B}(x) = \text{mín}\{\mu_A(x), \mu_B(x)\}$
$\mu_{A \cup B}(x) = \text{máx}\{\mu_A(x), \mu_B(x)\}$
$\mu_{A \square B}(x) = \text{máx}\{\text{mín}\{\mu_A(x), 1 - \mu_B(x)\}, \text{mín}\{1 - \mu_A(x), \mu_B(x)\}\}$

Tabla 5.1: Algunas medidas de similitud

5.3. Atributos continuos en árboles de decisión

En esta sección se comentan las alternativas disponibles a la hora de construir árboles de decisión con atributos continuos (sección 5.3.1), tras lo cual se analizan las aplicaciones de los métodos de discretización presentados en las secciones anteriores de este capítulo en la construcción de clasificadores que utilizan árboles de decisión como modelo de representación del conocimiento obtenido.

Posteriormente, se propone entrelazar un método jerárquico de discretización con el proceso de evaluación de hipótesis alternativas realizado para ramificar el árbol de decisión (sección 5.3.2). Este método nos permitirá construir árboles n-arios arbitrarios sin degradar la precisión del clasificador obtenido. Además, gracias a la eficiencia de métodos de discretización como el propuesto en la sección 5.2, se consigue un mayor grado de libertad sin que la complejidad computacional del proceso de construcción del árbol se vea afectada.

5.3.1. Árboles binarios vs. árboles n-arios

Como se vio en el capítulo 2, algunos algoritmos de inducción de árboles de decisión, como CART, construyen árboles binarios, mientras que los algoritmos pertenecientes a la familia de ID3 prefieren construir árboles n-arios. Algoritmos del tipo de C4.5, que construyen árboles de decisión n-arios para atributos discretos, añaden una rama al árbol para cada uno de los valores posibles del atributo. En ocasiones se permiten tests más complejos en los cuales se agrupan valores para reducir el factor de ramificación del árbol, algo similar en cierto sentido a las ramas ‘else’ del árbol de decisión construido por ART. CART, por su parte, lleva esta situación al extremo: todos los valores del atributo se agrupan en dos conjuntos para construir un árbol de decisión binario, incluso aunque el atributo sea de tipo categórico.

5.3.1.1. Árboles binarios con atributos continuos

Los algoritmos TDIDT, tanto los pertenecientes a la familia de ID3 como CART, suelen construir árboles binarios cuando trabajan con atributos continuos. Cuando el conjunto de entrenamiento incluye atributos de tipo numérico, el árbol de decisión se ramifica usualmente empleando tests binarios de la forma $atributo \leq umbral$, donde el umbral se escoge de forma que la partición del conjunto de entrenamiento generada por el test minimice la medida de impureza utilizada como regla de división al construir el árbol.

El formato de los tests utilizados se restringe para poder examinar todos los tests candidatos de una forma eficiente. Tales tests sólo involucran a un único atributo de tipo numérico para que los árboles resultantes sean más fáciles de comprender y se evite la explosión combinatoria que resultaría si se permitiese la aparición de múltiples atributos en un único test [131], como sucede en el caso de las combinaciones lineales en CART [23].

Para escoger el umbral utilizado en el árbol de decisión, se comprueban todos los valores posibles de dicho umbral teniendo en cuenta que, si utilizamos una medida de impureza como criterio de división, basta con evaluar aquellos puntos de corte que se encuentran en el límite entre dos clases [58]. El proceso de evaluación de los puntos de corte candidatos se puede efectuar de una manera eficiente si ordenamos el conjunto de entrenamiento utilizando los valores del atributo. Dado que un atributo sólo puede tomar un conjunto finito de valores $\{v_1, v_2..v_n\}$ en el conjunto de entrenamiento (ya que éste es también finito), cualquier umbral t_i existente entre v_i and v_{i+1} tendrá el mismo efecto al dividir el conjunto de entrenamiento. Por tanto, sólo hay que comprobar un máximo de $n - 1$ posibles umbrales para cada atributo numérico presente en el conjunto de entrenamiento. Aunque pueda parecer un proceso computacionalmente costoso, la evaluación de los distintos umbrales posibles puede realizarse con un único recorrido secuencial de los datos una vez que éstos han sido ordenados. Además, el rendimiento del algoritmo de inducción puede mejorarse si se emplean conjuntos AVC (RainForest [69]) o se implementan técnicas escalables como el middleware descrito en [30].

Una vez que se ha establecido que el mejor umbral posible debe encontrar-

se entre v_i y v_{i+1}, se ha de seleccionar un valor concreto para el umbral que aparecerá en el árbol de decisión. La mayor parte de los algoritmos escogen el punto medio del intervalo $[v_i, v_{i+1}]$:

$$t_i = \frac{v_i + v_{i+1}}{2}$$

C4.5, no obstante, prefiere escoger el mayor valor de A presente en el conjunto de entrenamiento que no excede del punto medio del intervalo $[v_i, v_{i+1}]$, es decir:

$$t_i = max \left\{ v \mid v \leq \frac{v_i + v_{i+1}}{2} \right\}$$

De esta forma, C4.5 se asegura de que cualquier valor utilizado como umbral en el árbol de decisión tiene sentido, pues aparece en el conjunto de entrenamiento, si bien es cierto que el umbral seleccionado puede resultar engañoso si la muestra del conjunto de valores del atributo presente en el conjunto de entrenamiento no es representativa.

Para evitar problemas de este tipo, aquí se propone utilizar un enfoque ligeramente distinto consistente en seleccionar el umbral entre v_i y v_{i+1} en función del número de casos de entrenamiento que queden por encima y por debajo del umbral. Si tenemos L ejemplos de entrenamiento con valor $v \leq v_i$ y R casos con $v \geq v_{i+1}$, el umbral t_i se define de la siguiente forma:

$$t_i = \frac{R * v_i + L * v_{i+1}}{L + R}$$

El número R de ejemplos cuyo valor de A es mayor que el umbral multiplica a la cota inferior v_i, mientras que la cota superior v_{i+1} se multiplica por el número de ejemplos que quedan por debajo del umbral (L). De esta forma, el umbral estará más cerca de v_i que de v_{i+1} cuando haya menos casos de entrenamiento con $v \leq v_i$. Igualmente, el umbral estará más cerca de v_{i+1} cuando la mayor parte de los ejemplos del conjunto de entrenamiento tengan un valor de A menor o igual a v_i.

Aunque esta ligera modificación del algoritmo usual no supondrá, por lo general, ninguna mejora sobre la precisión final del clasificador, los árboles de decisión resultantes parecen ser más adecuados. Esto es así especialmente

cuando los árboles no están balanceados, una situación bastante común cuando hay atributos continuos en el conjunto de entrenamiento. De hecho, a pesar de que normalmente no se mencione, C4.5 incluye un parámetro que, por defecto, establece que ninguna de las dos ramas resultantes al ramificar un árbol utilizando un atributo continuo puede tener menos del 20 % de los casos de entrenamiento [131].

5.3.1.2. Árboles n-arios con atributos continuos

La utilización de técnicas que permitan construir árboles n-arios puede, en principio, conducir a una reducción en la complejidad del modelo construido sin deteriorar la precisión del clasificador. Además, los árboles n-arios tienden a ser menos profundos y suelen resultar más fáciles de interpretar para los humanos.

Cuando sólo empleamos atributos de tipo categórico o atributos continuos previamente discretizados, algoritmos como C4.5 son capaces de construir árboles n-arios directamente. Sin embargo, cuando los atributos son continuos, los árboles de decisión resultantes son exclusivamente binarios, puesto que los nodos internos del árbol de decisión construido sólo incluirán tests binarios de la forma $atributo \leq valor$. La utilización de esos tests binarios implica que un atributo continuo puede aparecer varias veces en un camino desde la raíz hasta una hoja del árbol. Si bien es verdad que tales repeticiones de un atributo podrían simplificarse al convertir el árbol de decisión en un conjunto de reglas, no es menos cierto que el árbol construido tendrá más hojas, será innecesariamente más profundo y resultará más difícil de interpretar que si empleásemos algún mecanismo que nos permitiese construir árboles n-arios con atributos continuos. Tal como se comenta en [132], “divisiones no binarias sobre atributos continuos hacen que los árboles sean más fáciles de comprender y también parece conducir a árboles más precisos en algunos dominios”. Este hecho sugiere la posibilidad de utilizar métodos de discretización como los comentados en la sección 5.1 al construir un árbol de decisión cuando existen atributos continuos en el conjunto de entrenamiento.

Si utilizásemos tests n-arios, en vez de tests exclusivamente binarios, y pudiésemos determinar con una certeza suficiente que la división realizada del

conjunto de entrenamiento nos permite discriminar bien las distintas clases de nuestro problema de clasificación, entonces podríamos descartar el atributo numérico tras su utilización. De esta forma no podrían aparecer atributos repetidos en un camino del árbol desde la raíz hasta una hoja, de modo que el árbol resultante sería potencialmente más pequeño y más fácil de comprender para nosotros. Además, su construcción sería más rápida y su precisión no se vería afectada significativamente.

En la práctica, se pueden construir árboles n-arios directamente si agrupamos los valores numéricos de un atributo continuo en una serie de intervalos antes de construir el árbol y consideramos los intervalos seleccionados como si fuesen valores de un atributo categórico. Esta solución corresponde a la utilización de cualquier método de *discretización global*.

También existe la posibilidad de agrupar los valores numéricos de un atributo continuo dado en función de la situación particular en cada nodo del árbol. De esta forma, los intervalos utilizados para ramificar el árbol de decisión podrán variar ajustándose mejor a los datos del conjunto de entrenamiento que correspondan a cada nodo del árbol. Esta estrategia se conoce con el nombre de *discretización local*.

En el apartado siguiente se propone un método que permite utilizar discretización local de tal forma que no es necesario establecer de antemano el factor de ramificación del árbol de decisión.

5.3.2. Discretización local jerárquica en árboles n-arios

Aplicado localmente al construir cada nodo del árbol, cualquier método de discretización jerárquica, como el método de discretización contextual presentado en la sección 5.2, permite la construcción de árboles de decisión n-arios sin establecer a priori los intervalos en los que se han de agrupar los valores de un atributo continuo.

Si bien se han empleado métodos de discretización, usualmente jerárquica, para obtener un conjunto de intervalos con los cuales ramificar un árbol n-ario (p.ej. [58]), el conjunto de intervalos obtenido por el método de discretización fija de antemano el factor de ramificación del árbol de decisión. Incluso cuando se emplea discretización local, los métodos de discretización jerárquica se

aplican a priori sobre el conjunto de datos de entrenamiento y el conjunto de intervalos que generan sólo se evalúa como alternativa para ramificar el árbol tras finalizar del proceso de discretización (cuando el algoritmo de discretización termina su ejecución al verificarse el criterio de parada que utilice, que puede ser cualquiera de los mencionados en el apartado 5.2.2).

Por lo general, no se ha tenido en cuenta la estructura iterativa de los métodos jerárquicos de discretización para dotar de mayor flexibilidad al proceso de construcción del árbol de decisión. Si en vez de esperar la terminación del proceso de discretización, evaluamos en cada iteración del algoritmo jerárquico el conjunto de intervalos actual, podemos conseguir un proceso más flexible en el que el factor de ramificación del árbol no lo determina de antemano ninguna regla de parada del método de discretización, sino que lo escoge la propia regla de división que se utiliza en los algoritmos TDIDT de construcción de árboles de decisión.

En consecuencia, la técnica de discretización utilizada es completamente ortogonal a la regla de división empleada para construir el árbol de decisión, ya sea ésta el criterio de proporción de ganancia de C4.5, el índice de diversidad de Gini o cualquier otra medida de impureza. De este modo, se obtiene un método de discretización que no impone ninguna restricción sobre los demás parámetros del algoritmo TDIDT (algo que sí hacen algoritmos como el discretizador MDLP de Fayyad e Irani [58]).

5.3.2.1. Versión aglomerativa

Si empleamos como técnica de discretización un método jerárquico aglomerativo, disponemos inicialmente de un intervalo para cada valor de un atributo continuo y, en cada iteración, se combinan los dos intervalos adyacentes más similares para reducir el número de intervalos en que se discretiza el atributo continuo. Este proceso se puede repetir hasta que sólo queden dos intervalos, que son los que servirían para construir un árbol de decisión binario si se seleccionasen como mejor partición posible del conjunto de entrenamiento. Ahora bien, cada vez que combinamos dos intervalos adyacentes, podemos comprobar la medida de impureza asociada al árbol de decisión que construiríamos si empleásemos el conjunto actual de intervalos para ramificar

1. Crear un intervalo para cada valor del atributo continuo.
2. Identificar los dos intervalos adyacentes más similares.
3. Combinar los dos intervalos identificados en el punto anterior.
4. Evaluar el árbol de decisión que resultaría de utilizar el conjunto actual de intervalos para ramificar el árbol de decisión.
5. Si quedan más de dos intervalos, volver al paso 2.

Figura 5.6: Versión aglomerativa de la discretización jerárquica entrelazada con el proceso de evaluación de hipótesis candidatas que realiza cualquier algoritmo TDIDT.

el árbol. Conforme vamos evaluando alternativas, se registra la mejor partición obtenida hasta el momento, de forma que al final se utilizará el conjunto de intervalos que mejor se adapte a nuestro problema independientemente de su cardinalidad, algo que no permiten las propuestas previas de aplicación de métodos de discretización a la construcción de árboles de decisión.

El algoritmo resultante de entrelazar el método jerárquico aglomerativo de discretización con el proceso de construcción del árbol de decisión puede expresarse como aparece en la figura 5.6. Como se puede apreciar en dicha figura, el entrelazamiento entre el método de discretización y el proceso de construcción de árboles de decisión es lo que nos permite emplear tests n-arios sobre un atributo continuo sin tener que establecer de antemano el número de intervalos.

Por otra parte, cuando se emplea un método aglomerativo, no tiene sentido evaluar las particiones generadas durante las primeras iteraciones (cuando cada valor del atributo continuo en el conjunto de entrenamiento constituye un intervalo por sí mismo) y puede ser útil emplear otro método de discretización para generar la configuración inicial de los intervalos (por ejemplo, un método sencillo como *Equiwidth* o, especialmente, *Equidepth*). De esta forma se puede reducir la influencia que pueda tener sobre el proceso de discretización la

1. Discretizar los valores del atributo utilizando un método sencillo (por ejemplo, *Equidepth*).
2. Evaluar el árbol de decisión que resultaría de utilizar el conjunto actual de intervalos para ramificar el árbol de decisión.
3. Identificar los dos intervalos adyacentes más similares.
4. Combinar los dos intervalos identificados en el punto anterior.
5. Evaluar el árbol de decisión que resultaría de utilizar el conjunto actual de intervalos para ramificar el árbol de decisión.
6. Si quedan más de dos intervalos, volver al paso 3.

Figura 5.7: Versión aglomerativa con pre-discretización del método propuesto.

presencia de ruido en el conjunto de entrenamiento (y se disminuye el número de veces que se ha de medir la similitud entre intervalos adyacentes cuando empleamos el discretizador contextual de la sección 5.2). El método propuesto quedaría entonces como se muestra en la figura 5.7

5.3.2.2. Variante divisiva

De forma análoga a como se obtiene el algoritmo de la figura 5.6, se puede formular la versión divisiva del método jerárquico propuesto. Si deseamos emplear un método jerárquico divisivo para ramificar un árbol de decisión n-ario podemos emplear el algoritmo de la figura 5.8.

En este caso, resulta aconsejable establecer un factor de ramificación máximo para reducir el número de árboles alternativos que se evalúan. Este valor máximo puede determinarse automáticamente utilizando las reglas de prepoda: cuando los subconjuntos resultantes del conjunto de entrenamiento son demasiado pequeños, no es necesario seguir ampliando el conjunto de intervalos actual.

Si empleamos un método jerárquico divisivo para discretizar los valores

1. Comenzar con un único intervalo que incluya todos los valores del atributo continuo.
2. Identificar el punto de corte que da lugar a la configuración de intervalos más disimilares entre sí.
3. Utilizar dicho punto de corte para dividir en dos uno de los intervalos de la configuración actual de intervalos.
4. Evaluar el árbol de decisión que resultaría de utilizar el conjunto actual de intervalos para ramificar el árbol de decisión.
5. Mientras el número de intervalos sea inferior al factor de ramificación máximo deseado para el árbol de decisión, volver al paso 2.

Figura 5.8: Implementación divisiva del método propuesto.

de un atributo continuo, la presencia de ruido en el conjunto de entrenamiento afectará menos a los intervalos resultantes de la discretización. El algoritmo divisivo es menos sensible a la presencia de datos erróneos pues comienza con un único intervalo que cubre al conjunto de entrenamiento completo y lo va dividiendo de una forma más global que un método aglomerativo, que parte de las distribuciones de clases particulares para cada valor del atributo continuo.

5.3.2.3. Eficiencia

La utilización, en general, de cualquier método de discretización jerárquico nos permite, pues, construir árboles n-arios arbitrarios. Además, un algoritmo de discretización como el propuesto en la sección 5.2, cuya complejidad computacional es de orden $O(N \log N)$, se puede utilizar para generar una serie de conjuntos de intervalos sin afectar a la complejidad computacional del proceso de inducción del árbol de decisión. C4.5, por ejemplo, requiere realizar una ordenación de los valores de los atributos continuos en cada nodo, por lo que también es de orden $O(N \log N)$.

En algoritmos como el discretizador contextual de la sección 5.2, al existir una relación de adyacencia preestablecida entre los intervalos empleados en la discretización, sólo hay que recalcular dos medidas de similitud en cada iteración cuando se emplea un método aglometativo y tres cuando se utiliza un método divisivo. Como se puede emplear una estructura de datos similar a un árbol parcialmente ordenado, cada iteración se puede realizar en $O(\log N)$ operaciones. Como se realizan un máximo de $O(N)$ iteraciones, la complejidad resultante del algoritmo es $O(N \log N)$. Además, tanto la versión aglomerativa con pre-discretización como la divisiva acotan el número de iteraciones necesarias, con lo cual la complejidad del proceso se reduce a la complejidad de la etapa de preprocesamiento, $O(N)$ por lo general.

Por tanto, el método propuesto se puede considerar adecuado para realizar tareas de *Data Mining*, en las cuales los conjuntos de datos utilizados suelen tener un volumen considerable y es esencial disponer de algoritmos eficientes.

En la siguiente sección se presentan los resultados experimentales que se han conseguido utilizando el método de discretización descrito en la sección 5.2 para construir árboles de decisión n-arios con atributos continuos, tal como se describe en este apartado, y se comparan estos resultados con los obtenidos por las versiones básicas de C4.5 y ART, así como con la utilización de otras técnicas de discretización propuestas en la literatura.

5.4. Resultados experimentales

Se han realizado una serie de experimentos con los dieciséis conjuntos de datos de tamaño pequeño y mediano que aparecen descritos en la tabla 5.2. La mayor parte de esos conjuntos de datos se pueden obtener gratuitamente del *Machine Learning Repository* de la Universidad de California en Irvine, al cual se puede acceder a través de la siguiente dirección web:

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

Como en capítulos anteriores de esta memoria, todos los resultados comentados en esta sección se obtuvieron realizando validación cruzada con 10 particiones de cada conjunto de datos (10-CV).

Conjunto de datos	#Ejemplos	#Atributos	#Clases
ADULT	48842	14	2
AUSTRALIAN	690	14	2
BREAST	699	9	2
BUPA	245	7	2
CAR	1728	6	4
GLASS	214	10	6
HAYESROTH	160	4	3
HEART	270	13	2
IONOSPHERE	351	34	2
IRIS	150	4	3
PIMA	768	8	2
SPAMBASE	4601	57	2
THYROID	2800	29	2
WAVEFORM	5000	21	3
WINE	178	13	3
YEAST	1484	8	10

Tabla 5.2: Conjuntos de datos utilizados en los experimentos.

En los siguientes apartados de esta sección se comparan los resultados obtenidos por las versiones estándar de los algoritmos C4.5 y ART con los logrados al utilizar métodos de discretización en la construcción de los árboles de decisión. En concreto, se han implementado nueve métodos de discretización:

- Las tres variantes del discretizador contextual propuesto en la sección 5.2 de la presente memoria: su versión aglomerativa (Contextual A, figura 5.2 de la página 179), una variante de ésta con pre-discretización que utiliza *Equidepth* (Contextual B) y la versión divisiva del discretizador contextual (Contextual C, figura 5.3, página 180). Al emplear estos métodos de discretización de forma local en la construcción del árbol de decisión se utiliza la técnica entrelazada descrita en la sección 5.3.2 (figuras 5.6 a 5.8, páginas 194 a 196).

- Tres métodos de discretización supervisada ya existentes: el discretizador 1R (*One Rule*) de Holte [83], el método de Fayyad e Irani (MDLP) basado en el principio MDL de Rissanen [58] y Zeta, el método ideado por Ho y Scott [82].

- Otros tres métodos no supervisados estándar: el omnipresente algoritmo de la K Medias y dos sencillos métodos de discretización muy utilizados en KDD (*Equiwidth* y *Equidepth*).

Los resultados obtenidos por los métodos de discretización citados se comparan con la versión habitual de C4.5, que realiza tests binarios sobre los atributos continuos, en el apartado 5.4.1 y con la versión básica de ART, que considera categóricos todos los atributos, en el apartado 5.4.2.

5.4.1. Discretización en algoritmos TDIDT

En los experimentos se ha implementado el algoritmo C4.5, paradigma de los algoritmos TDIDT, que utiliza el criterio de proporción de ganancia como regla de división y emplea la poda pesimista descrita por Quinlan [131] con $CF = 0{,}25$. Las figuras 5.9 y 5.10 muestran los resultados medidos experimentalmente relativos a la precisión y complejidad de los árboles obtenidos.

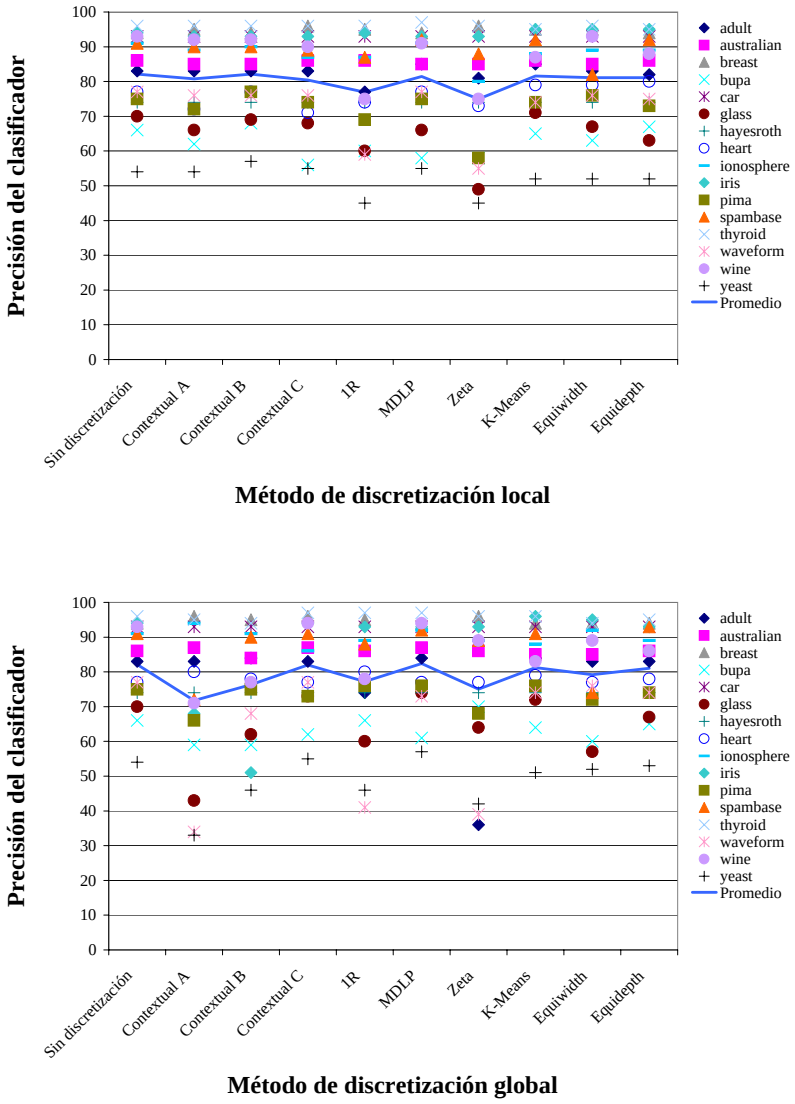

Figura 5.9: Precisión del clasificador TDIDT cuando se utilizan distintas técnicas de discretización.

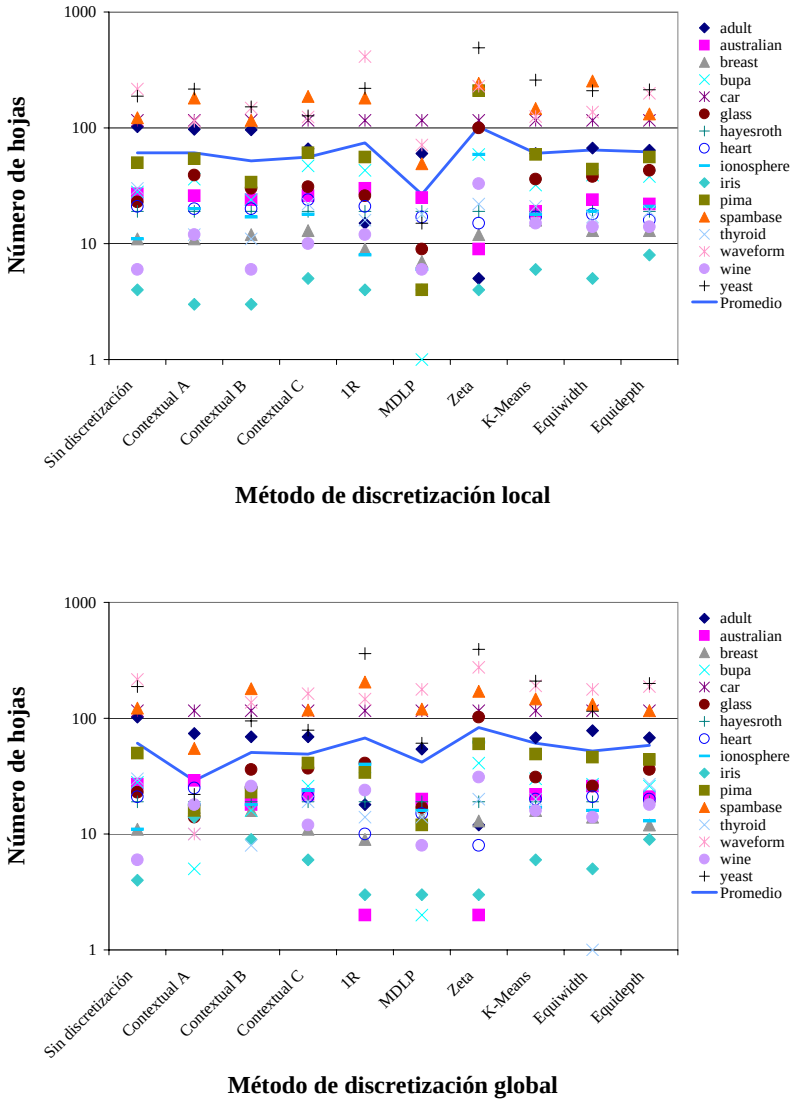

Figura 5.10: Número de hojas del árbol TDIDT cuando se emplean distintas técnicas de discretización.

5.4.1.1. Discretización local

La tabla 5.3 resume los resultados que se han obtenido al construir árboles de decisión utilizando distintos métodos de discretización localmente en cada nodo del árbol.

Con objeto de mejorar la robustez del discretizador contextual en presencia de ruido, los experimentos que figuran en las tablas de este apartado correspondientes al discretizador contextual se han realizado utilizando la variante aglomerativa precedida de un proceso de discretización previo con Equidepth, tal como se describe en la figura 5.7. En concreto, en los experimentos se utilizan 25 intervalos como punto de partida de la versión aglomerativa del método de discretización contextual. Igual que en los demás experimentos de esta sección, el discretizador contextual emplea la distancia euclídea para medir la disimilitud entre las distribuciones de clases de los intervalos adyacentes.

La tabla 5.3 muestra que el método de discretización contextual propuesto en esta memoria es comparable a otros métodos existentes en términos de la precisión del clasificador obtenido y, además, tiende a construir árboles de decisión pequeños.

Es digno de mención destacar que el método de discretización contextual mejora el error estimado del árbol de clasificación respecto a todas las demás técnicas de discretización. Dicha estimación del error es la utilizada por la poda pesimista de Quinlan. Sin embargo, no se aprecian diferencias notables en el error real medido al utilizar validación cruzada. Los resultados concretos de precisión del clasificador para cada conjunto de datos y método de discretización utilizando validación cruzada se muestran en la tabla 5.4.

Respecto al tamaño del árbol, el método de discretización contextual planteado en esta memoria obtiene árboles que, en media, contienen sólo el 84 % de los nodos empleados por el árbol binario construido por C4.5 y únicamente se ve mejorado por el método de discretización MDLP de Fayyad e Irani (véase la tabla 5.5).

	C4.5	Contextual	1R	MDLP	Zeta	KMeans	Equiwidth	Equidepth
Precisión (10-CV)	82.00 %	82.04 %	76.90 %	81.44 %	74.87 %	81.53 %	81.06 %	81.01 %
- Error medido	18.00 %	17.96 %	23.10 %	18.56 %	25.13 %	18.47 %	18.94 %	18.99 %
- Error estimado	12.74 %	14.34 %	16.88 %	16.44 %	19.99 %	16.36 %	17.27 %	17.20 %
Tiempo (segundos)	6.93	12.17	7.30	7.92	15.53	20.82	8.61	8.26
Tamaño del árbol	110.9	83.8	96.1	42.5	116.8	84.3	93.5	89.0
- Nodos internos	49.9	32.1	21.8	15.8	15.3	24.0	29.0	27.0
- Nodos hoja	60.9	51.7	74.3	26.7	101.6	60.3	64.5	62.0
Profundidad media	3.91	3.40	2.37	3.02	1.84	2.80	4.43	2.70

Tabla 5.3: Resumen de los experimentos realizados con discretización local.

	C4.5	Contextual	1R	MDLP	Zeta	KMeans	Equiwidth	Equidepth
ADULT	82.60 %	82.58 %	77.10 %	85.27 %	80.56 %	84.73 %	82.92 %	82.42 %
AUSTRALIAN	85.65 %	84.93 %	85.80 %	85.07 %	84.78 %	85.51 %	84.64 %	85.65 %
BREAST	93.99 %	93.84 %	94.85 %	94.28 %	95.57 %	95.28 %	95.28 %	94.27 %
BUPA	66.10 %	67.83 %	59.76 %	57.71 %	57.71 %	64.71 %	62.96 %	66.72 %
CAR	92.88 %	92.88 %	92.88 %	92.88 %	92.88 %	92.88 %	92.88 %	92.88 %
GLASS	70.06 %	68.77 %	59.70 %	66.34 %	49.46 %	70.58 %	67.38 %	63.46 %
HAYESROTH	73.75 %	73.75 %	73.75 %	73.75 %	73.75 %	73.75 %	73.75 %	73.75 %
HEART	77.04 %	76.67 %	74.07 %	76.67 %	73.33 %	78.89 %	78.52 %	79.63 %
IONOSPHERE	90.60 %	90.32 %	87.21 %	91.16 %	79.76 %	88.31 %	89.17 %	88.60 %
IRIS	94.00 %	93.33 %	94.00 %	93.33 %	93.33 %	95.33 %	94.67 %	94.67 %
PIMA	74.85 %	76.81 %	68.74 %	74.72 %	58.19 %	74.20 %	76.03 %	72.76 %
SPAMBASE	90.52 %	90.24 %	87.09 %	92.00 %	87.83 %	91.68 %	82.22 %	92.11 %
THYROID	96.18 %	95.61 %	96.46 %	97.29 %	96.04 %	95.32 %	96.04 %	94.54 %
WAVEFORM	76.80 %	76.50 %	58.86 %	76.80 %	55.36 %	74.38 %	75.74 %	74.58 %
WINE	92.71 %	91.57 %	75.07 %	90.92 %	74.51 %	87.09 %	92.68 %	88.20 %
YEAST	54.25 %	56.94 %	45.15 %	54.79 %	44.81 %	51.76 %	52.09 %	51.89 %
Promedio	82.00 %	82.04 %	76.90 %	81.44 %	74.87 %	81.53 %	81.06 %	81.01 %
Mejor-Peor-Igual (1 %)		3-2-11	0-10-2	3-3-10	1-10-5	5-5-6	3-7-6	2-7-7

Tabla 5.4: Precisión del clasificador obtenido con discretización local.

	C4.5	Contextual	1R	MDLP	Zeta	KMeans	Equiwidth	Equidepth
ADULT	145.8	134.5	18.0	80.2	5.6	79.8	95.2	94.4
AUSTRALIAN	39.9	36.2	40.5	36.9	11.6	28.3	33.9	30.9
BREAST	20.8	17.2	15.5	11.4	19.8	22.2	18.4	18.4
BUPA	55.0	38.5	55.1	1.8	67.3	42.1	53.8	48.1
CAR	162.0	162.0	162.0	162.0	162.0	162.0	162.0	162.0
GLASS	44.8	48.8	31.2	15.0	101.3	47.3	51.2	57.1
HAYESROTH	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8
HEART	35.2	30.9	33.8	28.2	19.7	25.6	26.1	24.4
IONOSPHERE	21.0	28.0	11.2	10.4	61.0	24.4	27.8	29.5
IRIS	7.4	5.0	5.8	5.2	5.5	7.0	6.5	10.4
PIMA	99.6	56.2	72.5	6.9	218.5	76.2	58.4	73.6
SPAMBASE	244.0	203.8	229.6	94.9	288.1	225.2	400.1	225.0
THYROID	55.7	16.8	24.5	29.5	32.4	31.7	25.8	28.8
WAVEFORM	431.6	295.4	517.7	134.5	257.9	187.0	204.4	295.4
WINE	11.0	11.3	13.4	11.4	35.2	21.9	20.3	20.0
YEAST	375.2	231.7	281.8	26.2	558.4	343.0	286.8	281.1
Complejidad relativa	100 %	84 %	82 %	55 %	126 %	91 %	95 %	96 %

Tabla 5.5: Tamaño del árbol resultante (en número de nodos) al utilizar distintos métodos de discretización local.

El uso de árboles n-arios con atributos numéricos permite mejorar el porcentaje de clasificación del algoritmo TDIDT estándar en algunos de los experimentos.

En concreto, el método de discretización contextual consiguió mejoras importantes en el porcentaje de clasificación de tres de los conjuntos de datos (BUPA, PIMA y YEAST). De hecho, este método de discretización sólo empeora de modo apreciable los resultados obtenidos por C4.5 en dos de los dieciséis conjuntos de datos (GLASS y WINE).

Otros métodos de discretización consiguen resultados similares, si bien es verdad que el método de discretización contextual propuesto en la sección 5.2 tiende a ser algo mejor que ellos. Aunque es cierto que el discretizador contextual no siempre mejora el porcentaje de clasificación obtenido por otros métodos, también es cierto que en los conjuntos de datos en que peor se comporta la pérdida de precisión es mínima. En cualquier caso, ha de mencionarse que la poda pesimista que se realiza sobre el árbol influye en algunos de los resultados (como en el conjunto de datos BUPA cuando se utiliza el discretizador MDLP de Fayyad e Irani y se poda el árbol de decisión de una forma excesivamente agresiva).

Los resultados comentados relativos a la precisión del clasificador son más relevantes si tenemos en cuenta que la complejidad del árbol de decisión suele disminuir de forma apreciable cuando se utilizan árboles n-arios con atributos continuos. El tamaño del árbol, dado por el número total de nodos del árbol (tanto nodos internos como hojas), suele ser menor al emplear técnicas de discretización local, tal como muestra la tabla 5.5. La profundidad media del árbol de decisión también tiende a disminuir cuando se utilizan métodos de discretización local para los atributos numéricos del conjunto de datos de entrenamiento.

Aunque tradicionalmente se asume que los árboles de decisión binarios tienen menos hojas y son más profundos que los árboles n-arios [108], en los experimentos realizados nos encontramos con que los árboles binarios no tienen menos hojas necesariamente (al menos, tras el proceso de poda). Como muestra la tabla 5.3, el árbol binario podado, que aparece en la columna etiquetada C4.5, puede tener bastantes más hojas que los árboles obtenidos utilizando

métodos de discretización como el propuesto en la sección 5.2 (Contextual) o el ideado por Fayyad e Irani (MDLP). Por tanto, podemos afirmar que se pueden lograr árboles de decisión más pequeños si se emplean tests n-arios para los atributos continuos del conjunto de entrenamiento.

5.4.1.2. Discretización global

También se han realizado pruebas utilizando distintos métodos de discretización global con los mismos conjuntos de datos utilizados en la sección anterior. Se ha evaluado el comportamiento de los mismos métodos de discretización que se emplearon localmente en cada nodo del árbol, aunque esta vez la discretización se realizó de forma global antes de comenzar a construir el árbol de decisión. Una vez discretizados globalmente, los atributos continuos pueden considerarse como si fueran atributos categóricos, de forma que algoritmos TDIDT como C4.5 pueden construir árboles n-arios con ellos sin necesitar ningún mecanismo adicional para su procesamiento.

La tabla 5.6 resume los resultados que se han obtenido utilizando la variante divisiva del discretizador contextual (más robusta ante la presencia de ruido) y los otros seis discretizadores empleados en los experimentos de la sección anterior. Las tablas 5.7 y 5.8 muestran los resultados obtenidos por cada discretizador para cada uno de los conjuntos de datos empleados en los experimentos. La primera muestra los porcentajes de clasificación obtenidos utilizando validación cruzada y en la segunda se recoge el tamaño de los árboles obtenidos.

Tal como se sugiere en estudios anteriores realizados por otros autores [51] [85], el empleo de métodos de discretización global mejora la eficiencia de los algoritmos TDIDT, reduce la complejidad de los árboles de decisión resultantes y mantiene la precisión de los modelos de clasificación construidos.

A pesar de que el método de discretización contextual expuesto en la sección 5.2 se ideó inicialmente como método de discretización local para su empleo en la construcción de árboles n-arios (tal como se describe en el apartado 5.3.2), se ha comprobado experimentalmente que, igual que otros métodos de discretización, se comporta adecuadamente cuando se utiliza globalmente.

	C4.5	Contextual	1R	MDLP	Zeta	KMeans	Equiwidth	Equidepth
Precisión (10-CV)	82.00 %	81.92 %	77.24 %	82.40 %	74.91 %	81.25 %	79.24 %	81.16 %
- Error	18.00 %	18.08 %	22.76 %	17.60 %	25.09 %	18.75 %	20.76 %	18.84 %
- Error estimado	12.74 %	16.60 %	20.53 %	15.94 %	22.12 %	16.96 %	19.47 %	17.24 %
Tiempo (segundos)	6.93	1.62	6.30	1.02	10.54	1.03	1.05	1.02
Tamaño del árbol	110.9	70.7	77.6	62.7	92.1	85.5	75.4	82.4
- Nodos internos	49.9	21.8	10.1	20.8	9.0	24.4	23.2	24.1
- Nodos hoja	60.9	48.9	67.4	41.8	83.0	61.1	52.2	58.2
Profundidad media	3.91	2.76	1.70	3.03	1.55	2.80	4.09	2.60

Tabla 5.6: Resumen de los experimentos realizados con discretización global.

	C4.5	Contextual	1R	MDLP	Zeta	KMeans	Equiwidth	Equidepth
ADULT	82.60 %	82.67 %	74.37 %	84.32 %	36.20 %	84.45 %	83.11 %	82.93 %
AUSTRALIAN	85.65 %	86.67 %	85.51 %	86.67 %	85.51 %	84.78 %	84.93 %	86.23 %
BREAST	93.99 %	95.71 %	95.13 %	94.14 %	95.71 %	93.99 %	94.13 %	94.42 %
BUPA	66.10 %	62.13 %	65.54 %	60.61 %	69.61 %	64.43 %	59.76 %	65.30 %
CAR	92.88 %	92.88 %	92.88 %	92.88 %	92.88 %	92.88 %	92.88 %	92.88 %
GLASS	70.06 %	73.31 %	59.83 %	74.29 %	63.53 %	71.90 %	56.56 %	67.25 %
HAYESROTH	73.75 %	73.75 %	73.75 %	73.75 %	73.75 %	73.75 %	73.75 %	73.75 %
HEART	77.04 %	77.41 %	80.37 %	77.41 %	76.67 %	78.52 %	77.41 %	77.78 %
IONOSPHERE	90.60 %	86.33 %	89.18 %	91.72 %	88.88 %	88.32 %	91.75 %	89.47 %
IRIS	94.00 %	94.00 %	93.33 %	93.33 %	93.33 %	96.00 %	95.33 %	93.33 %
PIMA	74.85 %	73.02 %	75.76 %	75.50 %	67.82 %	75.89 %	72.50 %	74.19 %
SPAMBASE	90.52 %	90.74 %	88.15 %	92.50 %	89.11 %	90.55 %	74.40 %	92.65 %
THYROID	96.18 %	96.64 %	97.04 %	96.96 %	96.46 %	95.82 %	93.89 %	94.89 %
WAVEFORM	76.80 %	76.56 %	41.40 %	73.44 %	38.62 %	74.40 %	76.18 %	73.80 %
WINE	92.71 %	94.38 %	77.97 %	93.79 %	88.73 %	83.10 %	89.31 %	86.50 %
YEAST	54.25 %	54.58 %	45.62 %	57.01 %	41.71 %	51.15 %	51.96 %	53.11 %
Promedio	82.00 %	81.92 %	77.24 %	82.40 %	74.91 %	81.25 %	79.24 %	81.16 %
Mejor-Peor-Igual (1 %)		4-3-9	2-7-7	7-2-7	2-8-6	5-5-6	2-7-7	1-6-9

Tabla 5.7: Precisión del clasificador cuando se emplea discretización global.

	C4.5	Contextual	1R	MDLP	Zeta	KMeans	Equiwidth	Equidepth
ADULT	145.8	93.7	21.3	66.9	12.6	90.5	108.7	94.5
AUSTRALIAN	39.9	32.4	3.0	31.2	3.0	32.6	37.5	31.4
BREAST	20.8	15.0	17.6	18.4	21.0	19.8	18.5	16.0
BUPA	55.0	35.5	40.8	3.0	44.4	38.7	37.5	32.6
CAR	162.0	162.0	162.0	162.0	162.0	162.0	162.0	162.0
GLASS	44.8	51.3	47.3	29.1	103.5	43.1	36.3	50.0
HAYESROTH	24.8	24.8	24.8	24.8	24.8	24.8	24.8	24.8
HEART	35.2	30.0	14.2	24.7	11.2	29.7	30.0	28.0
IONOSPHERE	21.0	33.5	42.0	27.3	63.5	23.3	22.8	17.2
IRIS	7.4	8.1	4.4	4.0	4.0	8.1	6.3	11.7
PIMA	99.6	54.4	35.8	20.2	60.5	64.6	64.9	56.7
SPAMBASE	244.0	170.3	232.8	216.2	196.0	222.9	193.4	206.2
THYROID	55.7	28.2	21.4	23.5	31.4	28.8	1.0	37.9
WAVEFORM	431.6	263.2	147.8	238.7	276.1	267.6	274.8	258.5
WINE	11.0	14.5	26.4	13.8	31.9	20.9	18.2	21.8
YEAST	375.2	114.8	399.8	98.8	427.2	290.3	169.5	269.0
Complejidad relativa	100 %	84 %	84 %	68 %	105 %	90 %	95 %	91 %

Tabla 5.8: Complejidad del clasificador obtenido cuando se emplea discretización global.

5.4.2. ART con discretización

Del mismo modo que se ha evaluado la obtención de árboles de decisión n-arios con algoritmos TDIDT (esto es, C4.5 con discretización), en este apartado se presenta un estudio empírico del uso de los distintos métodos de discretización en el modelo de clasificación ART.

En el caso de ART, la discretización es fundamental para permitir la utilización de este modelo en la construcción de clasificadores a partir de conjuntos de datos que incluyan atributos de tipo numérico.

5.4.2.1. Precisión

En cuanto a los resultados relativos a la precisión del clasificador ART, resumidos en la tabla 5.9, se puede apreciar que los métodos de discretización utilizados globalmente suelen conseguir resultados mejores que localmente. Aunque este resultado pueda parecer poco intuitivo, se consiguen mejores resultados si discretizamos los atributos continuos antes de construir el clasificador ART en vez de discretizarlos localmente en cada nivel del árbol.

Respecto a los resultados obtenidos con las distintas variantes del discretizador contextual de la sección 5.2, se puede apreciar que la versión divisiva del mismo consigue mejores resultados que las variantes aglomerativas, tal como cabría esperar, porque es menos sensible a la existencia de ruido en los datos del conjunto de entrenamiento.

Los resultados concretos de los que se extrajeron los porcentajes medios que figuran en la tabla 5.9 aparecen en la figura 5.11 de la página 214.

5.4.2.2. Complejidad

Por otra parte, los resultados experimentales relativos a la complejidad del árbol ART se resumen en la tabla 5.10 y aparecen reflejados en la figura 5.12 de la página 215. Igual que antes, los clasificadores globales tienden a conseguir mejores resultados al lograr árboles más pequeños de menor profundidad (figura 5.13, página 216) y, por tanto, de construcción más eficiente (figura 5.14, página 217)

Método de discretización	Precisión
Global - Contextual C (divisivo)	78.03 %
Global - MDLP	77.70 %
Global - K Means	77.27 %
Global - Zeta	76.47 %
Global - Equidepth	75.93 %
Local - Equidepth	75.42 %
Local - K Means	74.54 %
Local - MDLP	74.16 %
Local - Contextual B (con Equidepth)	74.04 %
Local - Contextual C (divisivo)	74.03 %
Global - 1R (Holte)	73.92 %
Local - Contextual A (aglomerativo)	73.21 %
Global - Equiwidth	73.15 %
Local - 1R (Holte)	72.02 %
Local - Equiwidth	71.68 %
Global - Contextual B (con Equidepth)	71.44 %
Global - Contextual A (aglomerativo)	69.75 %
Local - Zeta	69.33 %
Sin discretización	55.50 %

Tabla 5.9: Porcentajes de clasificación obtenidos con ART empleando distintos métodos de discretización (ordenados de mejor a peor).

Método de discretización	Hojas
Local - MDLP	11.0
Global - MDLP	15.8
Global - Contextual A (aglomerativo)	15.9
Sin discretización	20.3
Global - Equiwidth	24.3
Global - K Means	31.5
Global - Contextual C (divisivo)	33.2
Global - Contextual B (con Equidepth)	33.7
Local - Equiwidth	35.2
Local - Contextual A (aglomerativo)	36.9
Global - Equidepth	38.6
Local - K Means	39.9
Local - Equidepth	40.9
Local - Contextual B (con Equidepth)	71.9
Local - Contextual C (divisivo)	88.4
Local - 1R (Holte)	93.9
Global - 1R (Holte)	104.7
Local - Zeta	109.1
Global - Zeta	111.8

Tabla 5.10: Número medio de hojas del árbol construido por ART al utilizar distintos métodos de discretización (ordenados de mejor a peor).

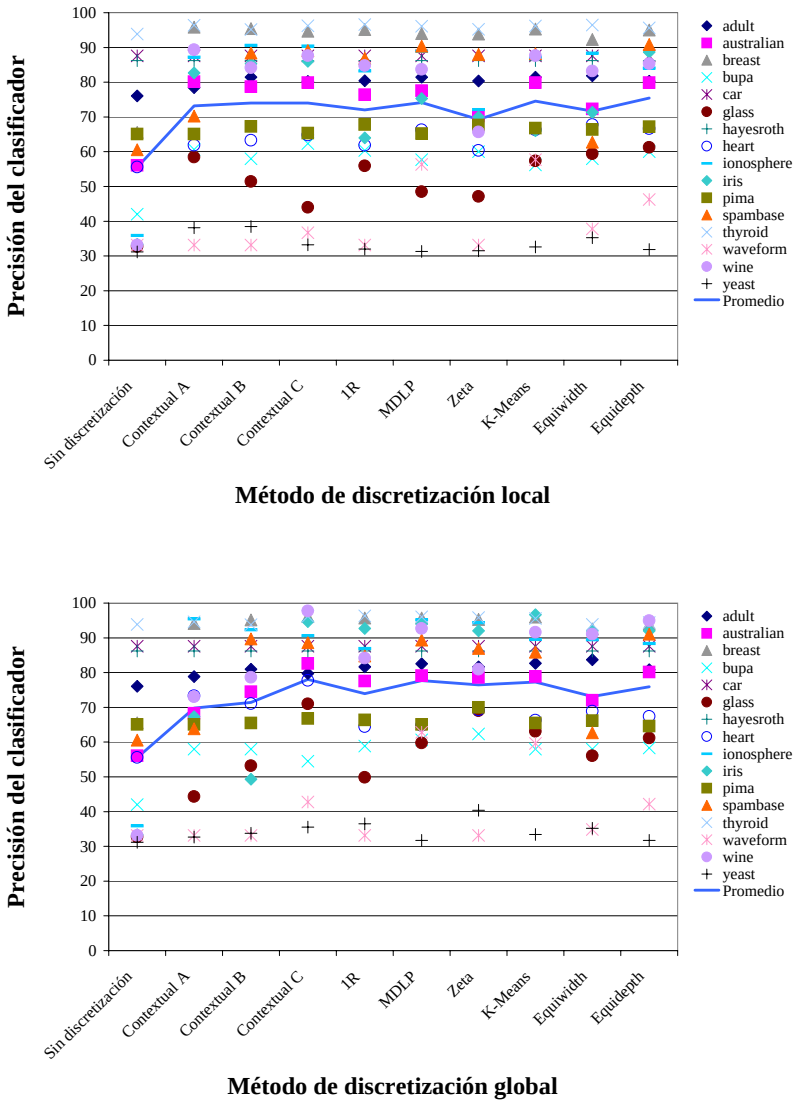

Figura 5.11: Precisión del clasificador ART al utilizar distintos discretizadores.

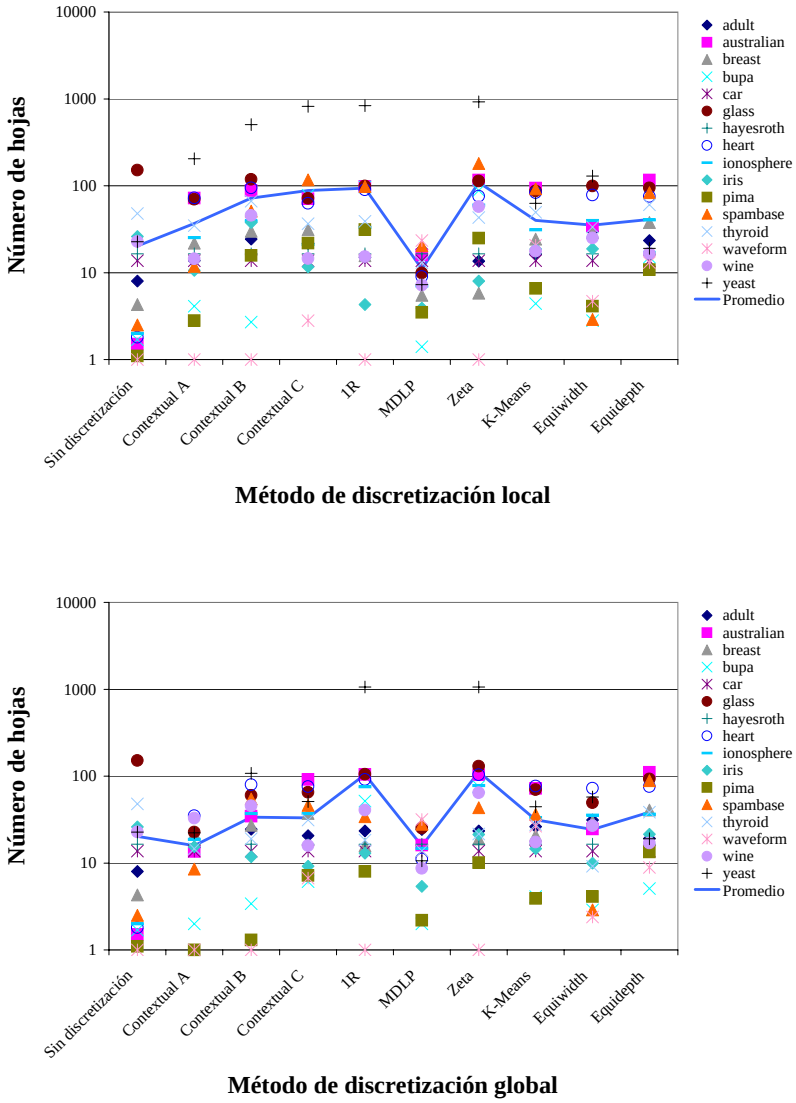

Figura 5.12: Número de hojas del clasificador ART cuando se utilizan distintas técnicas de discretización.

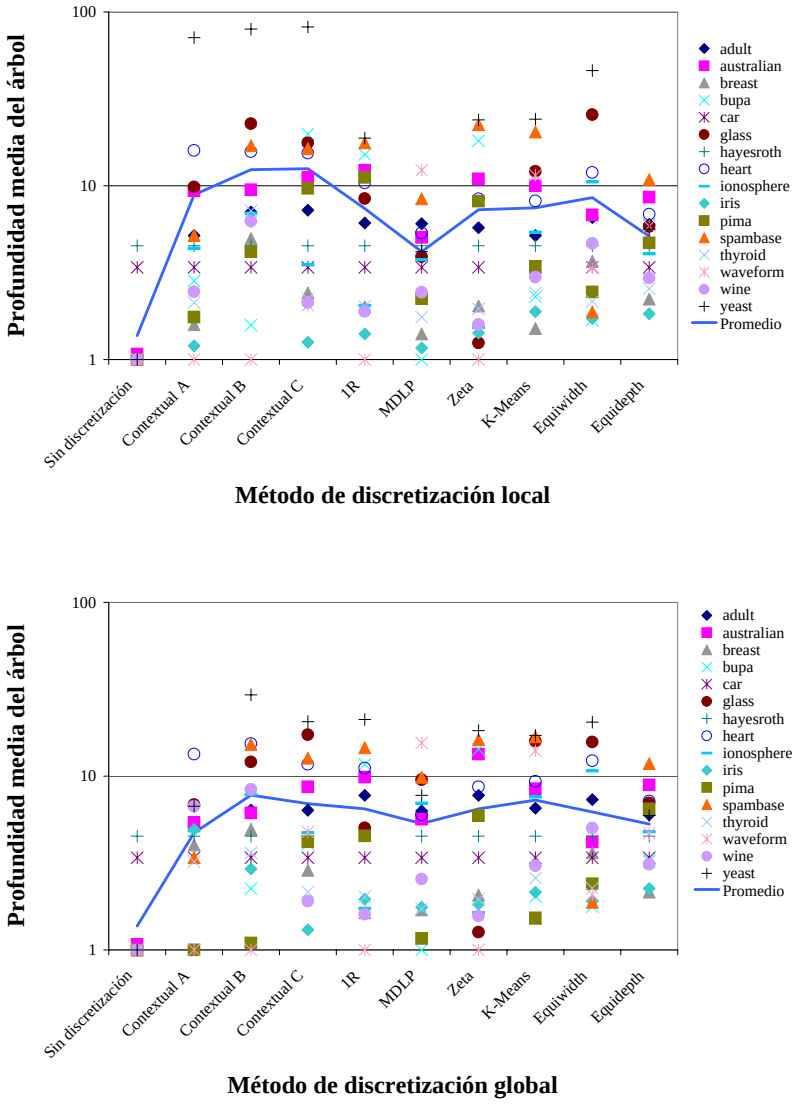

Figura 5.13: Profundidad media del árbol ART cuando se utilizan distintas técnicas de discretización.

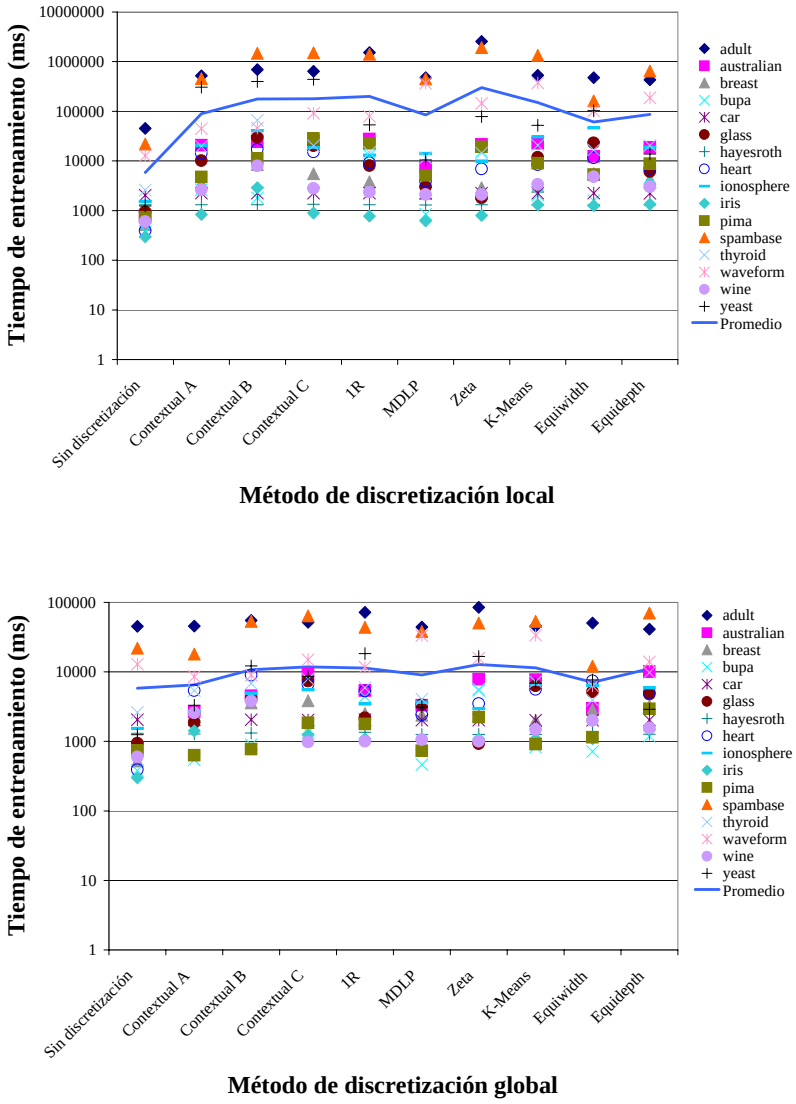

Figura 5.14: Tiempo de entrenamiento necesario para construir el clasificador ART cuando se utilizan distintas técnicas de discretización.

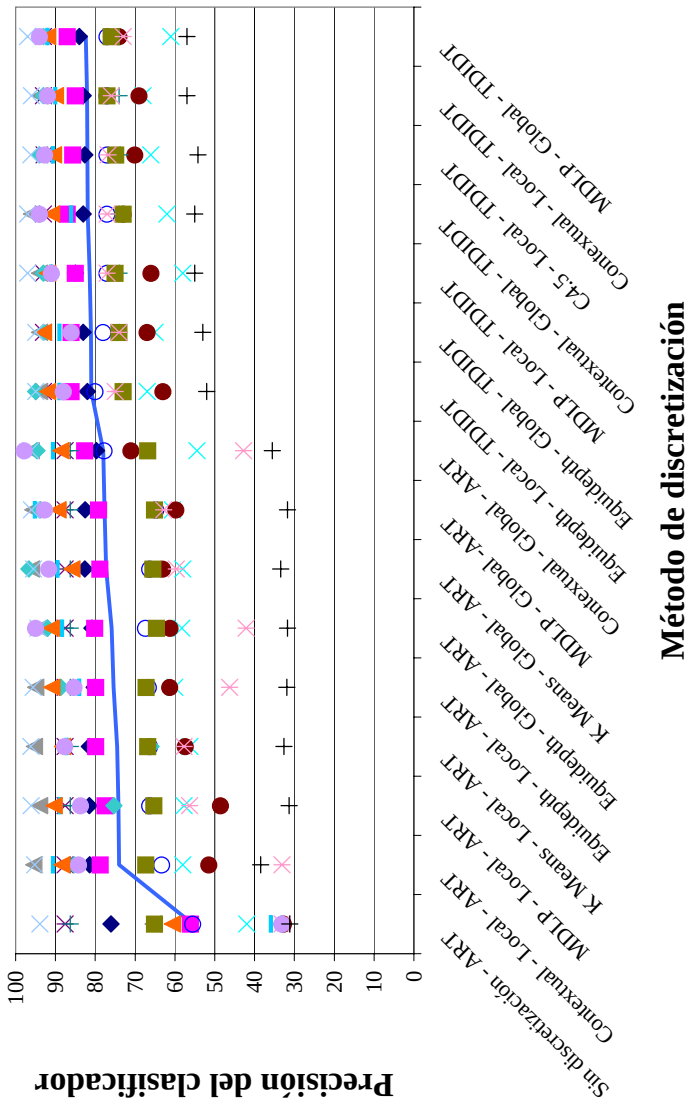

Figura 5.15: Efectos de la discretización en la precisión de los clasificadores.

5.4.3. Observaciones finales

Teniendo en cuenta los resultados anteriores, se recomienda el uso de ART con métodos de discretización global porque con ellos se suelen obtener clasificadores más precisos y compactos en menos tiempo.

Hay que tener en cuenta que los resultados obtenidos por ART cuando se trabaja con atributos continuos son ligeramente peores que los que se logran al emplear un algoritmo TDIDT tradicional como C4.5, tal como se aprecia en la figura 5.15, al menos en lo que respecta al porcentaje de clasificación obtenido.

Sin embargo, si incluimos en nuestra comparación el tamaño de los árboles obtenidos, se observa que ART consigue árboles de decisión más pequeños y, por tanto, más fáciles de interpretar cuando se utiliza el clasificador contextual propuesto en la sección 5.2 o el discretizador MDLP de Fayyad e Irani [58]. El compromiso entre precisión del clasificador y complejidad del modelo obtenido al que se llega se puede apreciar mejor en las figuras 5.16 y 5.17 que aparecen en las páginas siguientes. En dichas figuras se ordenan de mejor a peor algunos de los métodos de discretización utilizados, tanto local como globalmente, para construir clasificadores ART y TDIDT. En el caso de ART, la discretización es esencial para que se puedan conseguir buenos clasificadores.

En cuanto al método de discretización propuesto en este capítulo, hay que destacar que, utilizado como método de discretización global consigue resultados comparables a los obtenidos por los mejores métodos existentes (MDLP). Además, utilizando el enfoque propuesto en el apartado 5.3.2 se dota de mayor flexibilidad al proceso de construcción del árbol de decisión.

En cualquier caso, gracias a su eficiencia, el método de discretización contextual se añade a las técnicas de discretización que se pueden emplear en aplicaciones *Data Mining*, con la peculiaridad de que utiliza la estructura tradicional de los algoritmos de agrupamiento en la resolución de problemas de discretización. A diferencia de otros métodos de discretización supervisada (como MDLP, 1R o Zeta), los cuales emplean medidas de pureza para evaluar la calidad de un intervalo sin tener en cuenta su entorno, el discretizador propuesto utiliza medidas de similitud para evaluar conjuntos de intervalos adyacentes.

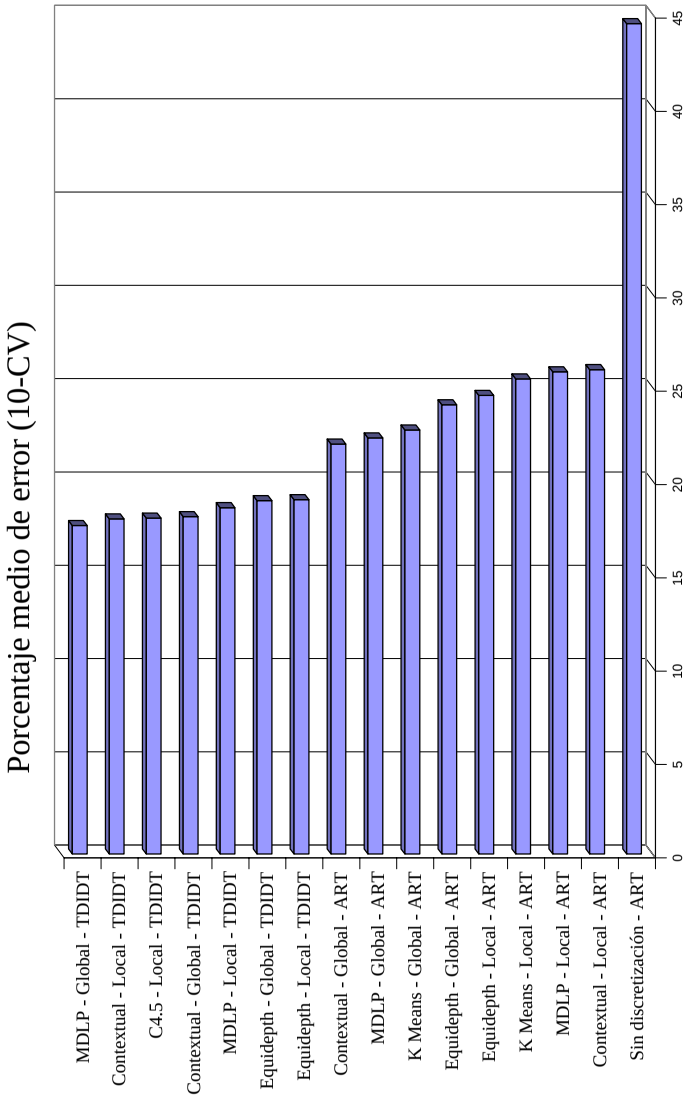

Figura 5.16: Efectos de la discretización en el error cometido por los clasificadores.

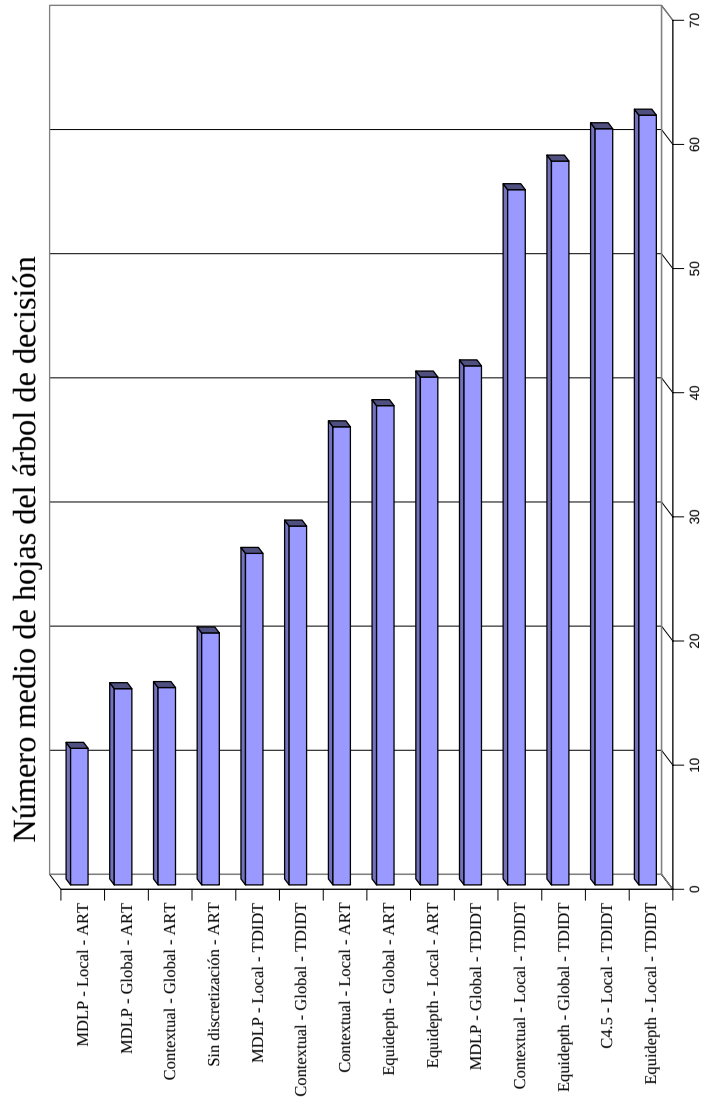

Figura 5.17: Efectos de la discretización en la complejidad de los clasificadores.

5.5. Anexo: Medidas de similitud

En esta sección se comentan algunas de las medidas que nos permiten comparar objetos y evaluar su similitud. Estas medidas son de gran utilidad en los métodos de agrupamiento y se pueden emplear con el método de discretización contextual propuesto en la sección 5.2.

Hablamos de medidas de similitud cuando el valor de la medida utilizada aumenta conforme son más similares los objetos que se comparan (distribuciones de clases en el método de discretización contextual). De forma complementaria, denominamos medidas de disimilitud a aquellas medidas cuyo valor disminuye cuanto más aumenta el parecido entre los objetos que se comparan, como es el caso de la distancia euclídea.

En el caso que nos ocupa, cualquier medida de similitud debe fomentar la combinación de intervalos adyacentes que compartan su clase más común, si bien este criterio es insuficiente en problemas donde las poblaciones de las distintas clases están muy desequilibradas. Si un 95 % de los ejemplos de entrenamiento pertenecen a una misma clase, es probable que la clase más común sea siempre la misma para todos los valores del atributo, por lo que fijarnos sólo en la clase más común no resulta práctico a la hora de discretizar un atributo continuo con un método supervisado como la discretización contextual. Cuanta más información aporte, más útil será la medida de similitud a la hora de establecer el orden en que se van combinando o dividiendo los intervalos. Por tanto, hay que utilizar siempre medidas de similitud que tengan en cuenta la probabilidad de todas las clases del problema, no sólo la más común, en clara oposición a lo que sucede con las reglas de división utilizadas al construir árboles de decisión (página 23).

En los apartados siguientes se describen brevemente los distintos modelos en que se basan las medidas de similitud que figuran en la tabla 5.1 de la página 187, las cuales pueden emplearse en métodos de agrupamiento jerárquico como la discretización contextual. En [164] se puede encontrar un análisis más riguroso de dichas medidas desde un punto de vista psicológico.

5.5.1. Modelos basados en medidas de distancia

Las medidas de similitud empleadas habitualmente por los métodos de aprendizaje no supervisado (esto es, los métodos de agrupamiento o *clustering*) suelen formularse como medidas de disimilitud utilizando métricas de distancia.

Cualquier función d que mida distancias debe verificar las siguientes tres propiedades:

- Propiedad reflexiva:

$$d(x, x) = 0$$

- Propiedad simétrica:

$$d(x, y) = d(y, x)$$

- Desigualdad triangular:

$$d(x, y) \leq d(x, z) + d(z, y)$$

Existe una clase particular de medidas de distancia que se ha estudiado exhaustivamente. Es la métrica r de Minkowski:

$$d_r(x, y) = \left(\sum_{j=1}^{J} |x_j - y_j|^r \right)^{\frac{1}{r}}, \qquad r \geq 1$$

Como una medida de distancia es, en realidad, una medida de disimilitud, podemos formular la distancia de Minkowsky como una medida de similitud de la siguiente forma:

$$S_{d_r}(x, y) = 1 - d_r(x, y)$$

Existen tres casos particulares de la clase de funciones definida por la distancia de Minkowsky que se utilizan habitualmente en la resolución de problemas de agrupamiento:

- La distancia euclídea ($r = 2$):

$$d_2(x, y) = \sqrt{\sum_{j=1}^{J} (x_j - y_j)^2}$$

- La distancia de Manhattan ($r = 1$):

$$d_1(x, y) = \sum_{j=1}^{J} |x_j - y_j|$$

- La "métrica de dominio" (conforme r tiende a ∞, la métrica de Minkowski viene determinada por la diferencia entre las coordenadas correspondientes a la dimensión para la que $|x_j - y_j|$ es mayor):

$$d_\infty(x, y) = \max_{j=1..J} |x_j - y_j|$$

En el método de discretización contextual, la desigualdad triangular no es estrictamente necesaria porque no utilizamos la medida de similitud para establecer la adyacencia entre patrones (ésta viene dada por el valor numérico o intervalo al que representa el vector característico). Por tanto, se podrían utilizar medidas de similitud que no verificasen la desigualdad triangular. De hecho, existen otros tipos de medidas de similitud:

5.5.2. Modelos basados en medidas de correlación

El producto escalar de dos vectores es la medida más simple de este tipo. Puede emplearse para medir la similitud existente entre vectores que representan distribuciones de clases igual que se emplea en muchos sistemas de recuperación de información para emparejar los términos de una consulta con los términos que aparecen en un documento. Matemáticamente, la medida de similitud dada por el producto escalar se define como:

$$S.(x, y) = x \cdot y = \sum_{j=1}^{J} x_j y_j$$

También se puede medir de forma alternativa la correlación entre dos vectores para reflejar la similitud entre dos conjuntos difusos a partir de la distancia euclídea d_2 al cuadrado:

$$\rho(x,y) = 1 - \left(\frac{4}{N_x + N_y}\right) d_2^2$$

donde

$$N_v = \sum_{j=1}^{J} (2v_j - 1)^2$$

Esta distancia estandarizada nos sirve de enlace con el tercer grupo de modelos que nos permiten medir la similitud entre dos objetos:

5.5.3. Modelos basados en Teoría de Conjuntos

Desde un punto de vista psicológico, Tversky describió la similitud como un proceso de emparejamiento de características. La similitud entre objetos puede expresarse como una combinación lineal de sus características comunes y de las características exclusivas de cada uno de ellos:

$$s(a,b) = \theta f(A \cap B) - \alpha f(A - B) - \beta f(B - A), \quad \theta, \alpha, \beta \geq 0$$

donde a y b son los objetos que se comparan mientras que A y B representan los conjuntos de características asociados a los objetos a y b respectivamente.

Cuando $\alpha = \beta = 1$ y $\theta = 0$, entonces $-s(a,b) = f(A - B) + f(B - A)$. Éste es el modelo propuesto por Restle para medir la disimilitud entre objetos fijándose únicamente en las características exclusivas de cada uno de ellos.

También podemos establecer $\alpha = \beta = 0$ y $\theta = 1$ para obtener el modelo más simple: $s(a,b) = f(A \cap B)$. En este caso, nos fijamos exclusivamente en las características que a y b tienen en común.

Otro modelo de interés es el modelo proporcional en el cual se normaliza la medida de similitud para que quede entre 0 y 1:

$$s(a,b) = \frac{f(A \cap B)}{f(A \cap B) + \alpha f(A - B) + \beta f(B - A)}, \quad \alpha, \beta \geq 0$$

Cuando la función f es aditiva (esto es, $f(A \cup B) = f(A) + f(B)$ cuando $A \cap B = \varnothing$), el modelo proporcional generaliza las propuestas de otros autores:

	T-normas ($\cap$)	T-conormas ($\cup$)
F	$min\{x, y\}$	$max\{x, y\}$
T_1	xy	$x + y - xy$

Tabla 5.11: Algunos pares de operadores T

- Gregson ($\alpha = \beta = 1$):

$$s(a, b) = \frac{f(A \cap B)}{f(A \cup B)}$$

- Eisler & Ekman ($\alpha = \beta = \frac{1}{2}$):

$$s(a, b) = \frac{2f(A \cap B)}{f(A) + f(B)}$$

- Bush & Mosteller ($\alpha = 1$, $\beta = 0$):

$$s(a, b) = \frac{2f(A \cap B)}{f(A)}$$

En todas las funciones de similitud presentadas en este apartado, la función f suele ser la cardinalidad del conjunto que recibe como argumento, si bien no se descartan otras posibilidades.

La Teoría de los Conjuntos Difusos nos permite utilizar todos estos modelos en conjuntos de objetos en los cuales la función de pertenencia puede variar entre 0 y 1. Así mismo, también define nuevos operadores para realizar las operaciones clásicas sobre conjuntos, como la unión, la intersección o la negación. Por ejemplo, los operadores T (T-normas y T-conormas) pueden emplearse para calcular intersecciones y uniones de conjuntos difusos, como también podrían utilizarse otros operadores, como los operadores promedio utilizados por algunos modelos difusos de recuperación de información [97].

Aunque también existen distintos cardinales difusos para medir la cardinalidad de un conjunto, aquí emplearemos la cardinalidad escalar, que se define como

$$|A| = \sum_x \mu_A(x)$$

La propuesta que Restle definió para conjuntos clásicos puede generalizarse de la siguiente forma haciendo uso de los operadores de la Teoría de Conjuntos Difusos:

$$-S_{Restle}(A,B) = |A \square B|$$

donde $A \square B$ es el conjunto difuso de los elementos que pertenecen a A pero no a B y viceversa. Utilizando el par F de operadores T de la tabla 5.11, dicho conjunto puede definirse como

$$\mu_{A\square B}(x) = \text{máx}\left\{\text{mín}\{\mu_A(x), 1-\mu_B(x)\}, \text{mín}\{1-\mu_A(x), \mu_B(x)\}\right\}$$

Si empleamos una función f distinta, obtenemos otra variación del modelo de Restle:

$$-S_{\square}(A,B) = \sup_x \mu_{A\square B}(x)$$

Cuando, a partir del modelo general de Tversky, fijamos $\alpha = \beta = 0$ y $\theta = 1$ y volvemos a utilizar los operadores mínimo y máximo, se obtiene otra medida de similitud alternativa:

$$S_{MinSum}(A,B) = |A \cap B| = \sum_x min\left\{\mu_A(x), \mu_B(x)\right\}$$

Nótese que, si hubiésemos utilizado la T-norma dada por T_1 en la tabla 5.11 en vez del mínimo, habríamos obtenido una medida de similitud equivalente al producto escalar que vimos en el apartado 5.5.2 de esta memoria.

De forma análoga a como sucede con el modelo de Restle, podemos obtener una variación de la medida de similitud data por S_{MinSum}. Dicha variación ha sido formulada por Enta como un índice de inconsistencia o "grado de separación":

$$-S_{Enta}(A,B) = 1 - \sup_x \mu_{A\cap B}(x)$$

La siguiente medida normaliza la medida de similitud dada por S_{MinSum} y es análoga a la propuesta de Gregson para conjuntos clásicos:

$$S_{Gregson}(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{\sum_x min\left\{\mu_A(x), \mu_B(x)\right\}}{\sum_x max\left\{\mu_A(x), \mu_B(x)\right\}}$$

A pesar de que todas las medidas de similitud definidas en los párrafos anteriores se definen sobre conjuntos difusos, también pueden aplicarse a los vectores característicos empleados por el método de discretización contextual propuesto en esta sección. Dichos vectores contienen probabilidades a partir de las cuales puede definirse un conjunto difuso C con función de pertenencia

$$\mu_C(j) = p_j$$

Es obligatorio mencionar que, aunque hemos empleado probabilidades (o, mejor dicho, estimaciones de probabilidades) para definir la función de pertenencia de un conjunto difuso, los modelos difusos y probabilísticos representan distintos tipos de información: los grados de pertenencia a un conjunto difuso nos permiten cuantificar la similitud entre objetos con propiedades definidas de un modo impreciso, mientras que las probabilidades nos dan información acerca de lo que podríamos esperar si realizásemos un número elevado de experimentos (si interpretamos las probabilidades desde un punto de vista estadístico).

No obstante, se puede interpretar la función de pertenencia de un conjunto difuso desde un punto de vista diferente. Podemos verla como una función de posibilidad que nos permite interpretar las conectivas lógicas (T-normas y T-conormas) en el marco de la Teoría de la Probabilidad [52]. De hecho, ésta es la base que nos ofrece la posibilidad de construir funciones de pertenencia experimentalmente. Dada una población de individuos y un concepto difuso C, a cada individuo se le pregunta si un elemento dado j puede considerarse C o no. La función de posibilidad $P(C|j)$ representa la proporción de individuos que respondieron afirmativamente a la pregunta. De esta forma resulta natural permitir que $\mu_C(j)$ sea igual a $P(C|j)$, p_j en nuestro problema de discretización.

Este razonamiento también puede aplicarse para emplear en nuestro problema la distancia de Bhattacharyya, también conocida como distancia de Jeffreis-Matusita. En términos de conjuntos difusos, puede expresarse como:

$$R(A,B) = \sqrt{1 - \sum_x \sqrt{\mu_A^*(x)\mu_B^*(x)}}$$

donde las funciones de pertenencia se normalizan:

$$\mu_A^*(x) = \frac{\mu_A^*(x)}{|A|}$$

Tanto el modelo propuesto por Eisler y Ekman como el de Bush y Mosteller pueden reducirse a la función de similitud S_{MinSum} si asumimos que la función $f(X)$ representa la cardinalidad de X y observamos que su valor se mantiene constante en nuestro problema para cualquier conjunto X, al ser $\sum \mu_C(j) = \sum p_j = 1$.

Con esto se da por cerrado el estudio de las medidas de similitud que pueden emplearse en nuestro método de discretización contextual (y en cualquier otro método de agrupamiento).

5.5.4. Resultados experimentales

Los experimentos realizados han servido para comprobar que el método de discretización propuesto en la sección 5.2, aplicado a la construcción de árboles de decisión tal como se describe en el apartado 5.3.2, nos permite obtener resultados interesantes, independientemente de la medida de similitud empleada.

Las tablas 5.12 a 5.14 resumen los resultados obtenidos para algunos de los conjuntos de datos empleados con anterioridad en este capítulo para evaluar el comportamiento de distintos métodos de discretización. Los experimentos que recogen las tablas se realizaron utilizando validación cruzada (10-CV), el criterio de proporción de ganancia de Quinlan y poda pesimista (con $CF = 0{,}25$), tal como se describe en [131].

Las tablas comparan los resultados obtenidos por el algoritmo TDIDT clásico (que construye árboles de decisión binarios cuando aparecen atributos continuos) con el método propuesto en el apartado 5.3.2, para el cual se han utilizado las distintas medidas de similitud que aparecen en la tabla 5.1.

	Algoritmo	Medidas de similitud basadas en distancias			
	clásico	Euclídea	Manhattan	Dominio	Bhattacharyya
ADULT	**84.59 %**	84.36 %	84.36 %	84.36 %	84.33 %
BUPA	64.36 %	**67.03 %**	**67.03 %**	**66.10 %**	**65.54 %**
GLASS	68.70 %	63.14 %	64.96 %	61.62 %	61.21 %
HEART	75.93 %	**77.41 %**	**77.41 %**	**77.04 %**	**78.15 %**
IONOSPHERE	**91.45 %**	88.62 %	87.76 %	88.33 %	88.04 %
PIMA	72.38 %	**74.59 %**	**74.59 %**	**74.46 %**	**72.88 %**
WAVEFORM	76.94 %	**77.32 %**	**77.84 %**	**77.58 %**	**77.38 %**
WINE	88.63 %	**94.97 %**	**94.97 %**	**94.97 %**	**94.38 %**

	Otras medidas de similitud					
	P.Escalar	Correlación	Restle	MinSum	Enta	Gregson
ADULT	84.54 %	84.20 %	84.22 %	84.37 %	84.30 %	84.37 %
BUPA	58.87 %	**68.16 %**	**65.25 %**	**65.87 %**	**67.03 %**	**65.87 %**
GLASS	**70.09 %**	66.86 %	66.86 %	67.31 %	**70.15 %**	64.98 %
HEART	**76.67 %**	**77.04 %**	**76.30 %**	**78.15 %**	**77.78 %**	**78.15 %**
IONOSPHERE	90.34 %	87.48 %	89.45 %	86.61 %	86.61 %	86.61 %
PIMA	**72.50 %**	**74.46 %**	**75.64 %**	**74.59 %**	**76.15 %**	**74.59 %**
WAVEFORM	76.08 %	**77.48 %**	75.94 %	76.68 %	**77.12 %**	76.32 %
WINE	**91.07 %**	**94.97 %**	**94.97 %**	**92.75 %**	**92.19 %**	**92.19 %**

Tabla 5.12: Precisión del clasificador TDIDT al utilizar distintas medidas de similitud para construir árboles n-arios con atributos continuos.

	Algoritmo	Medidas de similitud basadas en distancias			
	clásico	Euclídea	Manhattan	Dominio	Bhattacharyya
ADULT	3.42	**3.18**	**3.18**	**3.18**	**3.20**
BUPA	6.56	**2.75**	**2.75**	**2.87**	**3.38**
GLASS	5.08	**4.01**	**3.64**	**4.05**	**4.42**
HEART	3.19	**2.61**	**2.61**	**2.61**	**2.48**
IONOSPHERE	4.91	**4.81**	**4.88**	**4.81**	**4.70**
PIMA	6.81	**2.97**	**2.97**	**2.93**	**3.36**
WAVEFORM	9.20	**6.53**	**6.59**	**6.54**	**6.57**
WINE	2.41	2.41	2.41	2.41	2.44

	Otras medidas de similitud					
	P.Escalar	Correlación	Restle	MinSum	Enta	Gregson
ADULT	4.10	**3.26**	**3.23**	3.99	**3.21**	3.99
BUPA	**4.54**	**3.49**	**4.59**	**4.76**	**4.39**	**4.76**
GLASS	**3.94**	**3.84**	**3.32**	**4.07**	**3.67**	**3.93**
HEART	**3.07**	**2.50**	**2.59**	**2.75**	**2.59**	**2.75**
IONOSPHERE	**3.29**	**4.62**	6.51	**4.70**	**4.73**	**4.71**
PIMA	**4.68**	**3.69**	**4.01**	**5.04**	**4.65**	**5.04**
WAVEFORM	**7.15**	**6.63**	**6.99**	**7.10**	**6.75**	**7.03**
WINE	2.65	2.41	2.41	**2.35**	**2.35**	**2.35**

Tabla 5.13: Profundidad media de los árboles obtenidos.

	Algoritmo	Medidas de similitud basadas en distancias			
	clásico	Euclídea	Manhattan	Dominio	Bhattacharyya
ADULT	88.8	**78.6**	**78.6**	**78.7**	**77.5**
BUPA	32.2	**8.6**	**8.6**	**9.5**	**16.5**
GLASS	22.7	**22.4**	**18.7**	23.3	23.4
HEART	21.5	**20.1**	**20.1**	**20.2**	**19.1**
IONOSPHERE	14.7	15.6	15.1	15.5	18.5
PIMA	58.9	**14.6**	**14.6**	**13.7**	**20.4**
WAVEFORM	291.9	**108.7**	**112.8**	**110.3**	**117.8**
WINE	5.3	**5.2**	**5.2**	**5.2**	5.3

	Otras medidas de similitud					
	P.Escalar	Correlación	Restle	MinSum	Enta	Gregson
ADULT	92.1	**79.4**	**81.7**	90.0	**79.9**	90.0
BUPA	44.8	**10.0**	**15.8**	**19.7**	**18.7**	**19.7**
GLASS	34.3	**20.7**	**13.0**	**22.5**	**20.3**	**22.2**
HEART	22.6	**16.6**	**16.8**	**19.4**	**18.1**	**19.4**
IONOSPHERE	19.0	**11.7**	**12.2**	**12.7**	**13.9**	**12.7**
PIMA	**56.1**	**18.8**	**19.1**	**32.1**	**36.1**	**32.1**
WAVEFORM	**137.1**	**102.5**	**105.6**	**109.6**	**102.5**	**114.5**
WINE	8.1	**5.2**	**5.2**	5.3	5.3	5.3

Tabla 5.14: Número medio de hojas de los árboles obtenidos.

Recapitulación

En las tablas anteriores, los resultados que mejoran el rendimiento del algoritmo C4.5 tradicional aparecen en negrita.

En líneas generales, las medidas de similitud basadas en medidas de distancia tienden a obtener resultados uniformes, mientras que se observa una mayor variabilidad en los experimentos realizados con criterios de similitud basados en medidas de correlación. De las medidas basadas en conjuntos, el modelo de Enta destaca ligeramente. No obstante, en la práctica no se aprecian diferencias notables entre las distintas medidas evaluadas, independientemente del modelo de similitud en que tengan su origen.

Como resumen de nuestros experimentos, la tabla 5.15 muestra los resultados relativos que se obtienen al utilizar la distancia euclídea como medida de similitud frente al algoritmo C4.5 clásico. A pesar de los recursos computacionales adicionales que el método del apartado 5.3.2 requiere, el método jerárquico aglomerativo propuesto para construir árboles n-arios arbitrarios consigue buenos resultados al mantener o incluso mejorar el porcentaje de clasificación obtenido a la vez que reduce notablemente la complejidad del árbol de decisión construido.

	Precisión	Profundidad	Hojas	Tiempo
ADULT	-0.23 %	-7.02 %	-11.49 %	+16.8 %
BUPA	+2.67 %	-58.08 %	-73.29 %	+90.8 %
GLASS	-3.23 %	-21.06 %	-1.32 %	+114.6 %
HEART	+1.48 %	-18.18 %	-6.51 %	+58.7 %
IONOSPHERE	-2.83 %	-2.04 %	+6.12 %	+129.4 %
PIMA	+2.21 %	-56.39 %	-75.21 %	+93.3 %
WAVEFORM	+0.38 %	-29.13 %	-62.76 %	+60.8 %
WINE	+6.34 %	0.00 %	-1.89 %	+39.3 %
Promedio	+0.85 %	-23.99 %	-28.29 %	+75.5 %

Tabla 5.15: Mejoras obtenidas al utilizar la distancia euclídea para construir árboles de decisión n-arios con atributos continuos.